2019-09

UTILIZING ADAPTIVE DESIGN OF EXPERIMENTS WITH LOGISTICS BATTLE COMMAND TO DETERMINE THE MOST IMPORTANT FACTORS FOR A DECISION MAKER

Streams, James E., Jr.

Monterey, CA; Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

UTILIZING ADAPTIVE DESIGN OF EXPERIMENTS WITH LOGISTICS BATTLE COMMAND TO DETERMINE THE MOST IMPORTANT FACTORS FOR A DECISION MAKER

by

James E. Streams Jr.

September 2019

Thesis Advisor:	Jeffrey A. Appleget
Co-Advisor:	Jonathan K. Alt
Second Reader:	Brian M. Wade

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.

1. AGENCY USE ONLY ***(Leave blank)***

2. REPORT DATE September 2019

3. REPORT TYPE AND DATES COVERED Master's thesis

4. TITLE AND SUBTITLE
UTILIZING ADAPTIVE DESIGN OF EXPERIMENTS WITH LOGISTICS BATTLE COMMAND TO DETERMINE THE MOST IMPORTANT FACTORS FOR A DECISION MAKER

5. FUNDING NUMBERS

6. AUTHOR(S) James E. Streams Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

The Army makes extensive use of simulation models to obtain data on the operational effectiveness of future capabilities to inform acquisition decisions. A key step in this process is understanding how the key performance parameters of a new system impact the system's performance in the operational environment. This enables decision makers to set threshold and objective values of these parameters to provide to industry as system requirements. Another challenge faced with new technology is understanding how tactics are potentially impacted by the new capability. This requires the exploration of new methods of employment. Efficiently exploring the interaction between key performance parameters and tactics within its simulation models remains a challenge for the analytic community.

The analytic community makes use of some methods from the design of experiments literature but continues to require more efficient methods of searching the design space. Methods from the adaptive sequential design of experiments community or the field of machine learning could potentially be leveraged to address this need. These methods make use of the results of previous experimentation to inform the subsequent trials, potentially resulting in fewer costly simulation experiments. TRAC has identified the Logistics Battle Command (LBC) model as a candidate for proof-of-principle work in this area.

The end result is a metamodel of LBC for the selected factors with significant factors highlighted.

14. SUBJECT TERMS
Design of Experiments, TRAC, LBC, simulation, models

15. NUMBER OF PAGES
87

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
UU

THIS PAGE INTENTIONALLY LEFT BLANK

UTILIZING ADAPTIVE DESIGN OF EXPERIMENTS WITH LOGISTICS BATTLE COMMAND TO DETERMINE THE MOST IMPORTANT FACTORS FOR A DECISION MAKER

James E. Streams Jr.
Major, United States Army
BS, U.S. Military Academy, 2008

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 2019

Approved by: Jeffrey A. Appleget
Advisor

Jonathan K. Alt
Co-Advisor

Brian M. Wade
Second Reader

W. Matthew Carlyle
Chair, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Army makes extensive use of simulation models to obtain data on the operational effectiveness of future capabilities to inform acquisition decisions. A key step in this process is understanding how the key performance parameters of a new system impact the system's performance in the operational environment. This enables decision makers to set threshold and objective values of these parameters to provide to industry as system requirements. Another challenge faced with new technology is understanding how tactics are potentially impacted by the new capability. This requires the exploration of new methods of employment. Efficiently exploring the interaction between key performance parameters and tactics within its simulation models remains a challenge for the analytic community.

The analytic community makes use of some methods from the design of experiments literature but continues to require more efficient methods of searching the design space. Methods from the adaptive sequential design of experiments community or the field of machine learning could potentially be leveraged to address this need. These methods make use of the results of previous experimentation to inform the subsequent trials, potentially resulting in fewer costly simulation experiments. TRAC has identified the Logistics Battle Command (LBC) model as a candidate for proof-of-principle work in this area.

The end result is a metamodel of LBC for the selected factors with significant factors highlighted.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ABCT	Armored Brigade Combat Team
ADA BTRY	Air Defense Artillery Battery
ADOE	Adaptive Design of Experiment
AFC	Army Future Command
AR	Army Regulation
AR BDE	Armored Battalion
ASB	Aviation Support Battalion
BDE	Brigade
BN	Battalion
BSB	Brigade Support Battalion
CAB	Combat Aviation Brigade
CAV	Calvary
CONOPS	Concept of Operations
CSV	Comma Separated Values
DOE	Design of Experiments
DIVARTY	Division Artillery
DP	Design Points
DPG	Defense Planning Guidance
EGO	Efficient Global Optimization
EI	Expected Improvement
ENG BN	Engineering Battalion
FA	Field Artillery
FSC	Forward Support Company
GSAB	General Support Aviation Battalion
HPC	High-performance Computer
ILW	Institute of Land Warfare
INF	Infantry
KPP	Key Performance Parameters
LBC	Logistics Battle Command
LEE	Fort Lee

LH	Latin Hypercube
MEB	Maneuver Enhancement Battalion
MOE	Measure of Effectiveness
MTRY	Monterey
NIST	National Institute of Standards and Technology
NOLH	Nearly Orthogonal Latin Hypercube
SBCT	Stryker Brigade Combat Team
SUS BDE	Sustainment Brigade
TRAC	The Research Analysis Center
TRADOC	Training and Doctrine Command
TS	Tabu Search
XML	eXtensbile Markup Language

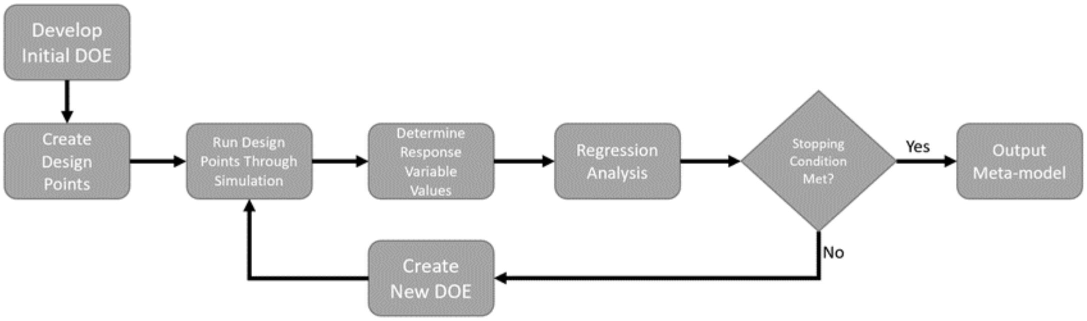
EXECUTIVE SUMMARY

Military planners and strategists have a difficult change in planning for the future combat environment. They need to ensure the military modernizes in a way that will provide the U.S. with a decisive advantage in tactics and equipment against a near-peer adversary in any location in the world. Computer simulations provide a glimpse of what the potential battle space will look like and allows analysts to modify certain aspects to different outcomes. The complexity of the simulations continues to increase, which increases the computational cost for a simulation. This can increase time for analysis as well as operation costs. Utilizing design of experiments (DOE) methodologies, allows analysts to examine thousands of parameters in a complex simulation to determine the important parameters for a simulation. Analysts iterating creating DOEs to find the optimal meta-model of a simulation. We propose using DOE within a batch adaptive sequential design process to develop meta-models with consistent prediction error across the range of factors explored.

The adaptive design of experiments (ADOE) process in this thesis builds off the current DOE and ADOE methodologies, multivariate linear regression and tabu search method, and utilizes high-performance computers (HPCs). The process creates a meta-model, based on selected factors, of the Logistics Battle Command (LBC) simulation. The analyst identifies the analysis questions that must be addressed as well as the simulation factors that relate to it and creates the initial DOE, using a nearly orthogonal Latin Hypercube (NOLH) design. The analyst establishes stopping conditions which inform the process to end iterating and display the final meta-model. The primary stopping condition is to limit the percentage of residuals outside of two standard deviations of the residual mean. In addition, the process will stop if it reaches too many iterations or successive meta-models fail to show improvement in decreasing the number of residuals outside of the acceptable range. Each of the design points from the DOE is executed in LBC and generates an output measures.

The ADOE process searches the results finds the response variable value for each replication. This creates a data set to use for linear regression. The lm function in R takes

the data and creates a meta-model. The process evaluates the meta-model and determines if it meets a stopping condition. If the meta-model meets a stopping conditions, the process will display the final meta-model. If not, the process will create another DOE using the residual with the highest residual value as the base and perform another iteration.



Flowchart for ADOE Process

The ADOE process created a meta-model from the selected factors. Each iteration decreased the percentage of residuals outside of the acceptable range. As the range decreases the confidence interval range for the predicted values for the meta-model. The final meta-model showed a 21% decrease in range from the initial meta-model and a 7% reduction in the percentage of residuals outside of the acceptable range.

Using the HPC, the ADOE process takes about 3.5 to 4.5 hours to complete an iteration, depending on the HPC node available for computing. The proof of principle for this thesis took 12 iterations, which took about two days to complete.   The ADOE process generates a final meta-model but the percentage of residuals outside of the acceptable range is higher than 5%. The process stopped due to little improvement in reducing the percentage of residuals between meta-models over 11 iterations.

We created a process that can create a meta-model of LBC simulation using selected factors. An analyst can adapt several sections of the process to utilize for other simulations. The performance of the process varies depending on the simulation, number of factors selected, and the computational power available. Improving factor selection or modifying the process to search for the most significant factor could increase the explanatory power of the meta-models. An analyst could use another method for regression analysis and compare how each one performs with a given data set.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

A special thank you and much appreciation for Dr. Alt, Dr. Appleget, and LTC Wade. Their time, effort, expertise and guidance made this possible and this thesis would not have been completed without them. They are amazing and I am so grateful for them throughout this process.

Also, I am thankful for my classmates. Times got tough during this program, but we are were able to keep each other motivated and finish the program strong. Without their aid, this program would have been a lot harder, and I may not have been able to finish.

Last, but not least, to my family and friends, your belief, support and love gave me a reason to get my graduate degree. It was not the easiest of two years for me academically and personally, but I would not be here if it were not for you all. Thank you.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. INTRODUCTION

The Army must modernize to keep the advantage against peer or near peer adversaries. The Army makes use of approved Defense Planning Guidance (DPG) scenarios represented within a suite of combat simulations to gather data on the effectiveness of current and future capabilities in simulated combat. The knowledge gained from the simulation experimentation provides operational effectiveness measures that inform comparative analysis in support of the acquisition process. As U.S. Army Futures Command reorganizes the Army Modernization Enterprise, the need exists to identify methodologies and emerging technologies to gain greater insights from combat simulation output to identify challenges and opportunities for the future force through a greater exploration of the impact of potential changes to concepts of operation (CONOPs) and to identify threshold key performance parameters (KPP) for future capabilities. The identification of these threshold KPPs through objective analysis will aid in the documentation of requirements—the first step in the acquisition process.

Developing and executing simulation experiments to efficiently explore the interaction of key performance parameters and tactics within tight study timelines presents challenges for the analytic community. Analysts require methodologies to both effectively and efficiently explore the design space for future capabilities within these simulation models. By leveraging advances in computational power and experimental design, they can conduct experiments to generate data that can inform future capabilities and concepts development (Institute of Land Warfare 2018). The value of such data increases when used to construct compact meta-models of performance of operational effectiveness of a given system. These models enable sensitivity analysis and greater understanding of a capabilities impact on operational effectiveness of the overall force under a variety of conditions. Study teams can leverage these models to quickly answer questions regarding the impact of KPPs. Adaptive design of experiments (DOE) methods, used to efficiently develop meta-models in other domains, show great potential to enable this use case.

The increased capabilities of adversaries and other threats require the military to find ways to maintain the advantage. The Army Futures Command (AFC) provides a "unified, responsive, efficient modernization enterprise" (Institute of Land Warfare [ILW] 2018). The AFC will change the way the Army modernizes the forces and attempt to regain overmatch over near-peer threats in the future (ILW 2018).

B. PROBLEM STATEMENT

Combat simulations allow analysts to explore future combat environments and provide important insight on future needs of the military. These simulations attempt to model as many aspects of the battlefield as possible, however that comes with a cost. These simulations often require from 3–6 months to instantiate a new scenario. Given the sunk cost of developing these simulation scenarios, the Army could benefit from leveraging the output data to the greatest extent possible and the quality and timeliness of subsequent analysis could benefit from the use of this data to develop meta-models capable of supporting questions from decision makers regarding the potential trade space between alternative capabilities or CONOPS.

We develop and apply a batch adaptive design of experiments process to generate data to develop meta-models of simulation output with consistent prediction error across the range of factors explored. We develop a proof of principle use case applying these ideas to a scenario within the U.S. Army's Logistics Battle Command (LBC) simulation.

C. THESIS ORGANIZATION

Chapter II provides background information about Logistics Battle Command (LBC) and the structure of the data used for analysis and a brief review of the relevant literature. Chapter III provides the methodology developed to address the problem statement above. It discusses the plan to execute the adaptive experimental design based on the studies found in the literature review. Chapter IV contains the operational results and lessons learned from the development of the adaptive sequential process. Chapter V summarizes the research findings and provides potential topics for future research.

II. BACKGROUND

This section provides background information on the Logistics Battle Command simulation used in the proof of principle application as well as a brief review of the methods and techniques used in this research.

A. LITERATURE REVIEW

1. Design of Experiments (DOE)

In an experiment, the analyst changes one or more factors to see the effect on the outcome or response variable: "A DOE is an efficient procedure for planning experiments so that the data obtained can be analyzed to yield valid and objective conclusions" (National Institute of Standards and Technology [NIST] 2012). For any DOE the analyst must choose both the number of factors and the values for each of the selected factors to access. Choosing to examine all factors at all possible values of each factor is called a full factorial design.

Using a full factorial design can be computational taxing as well as take longer than humanly possible to evaluate. For instance, would it be possible to evaluate a full factorial design of 100 factors each with two levels, 2^{100}? Given a petaflop computer and a simulation that runs in seconds, the process would take over 40 million years to run one replication of the experiment (Sanchez 2012). In LBC, there is over 1500 factors, so an analyst must create an efficient design to explore the design space and provide an answer using a meta-model. "A meta-model approximates the relationship between the simulation input and output parameters by computationally efficient mathematical models" (Syberfeldt, A., Grimm, H., Ng, A. 2008, p. 3). A meta-model takes a simulation and represents the simulation as a formula (Syberfeldt, A., Grimm, H., Ng, A. 2008).

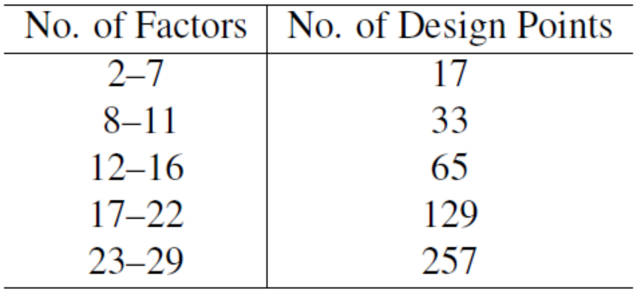
There are several methods to develop an optimal space filling design (Sanchez 2012). Latin Hypercubes (LH) are one of the most common space-filling designs (Sanchez 2012). LH are good all-purpose designs because they offer the following:

1. *“Design Flexibility*: One can readily generate an LH for a combination of continuous *k* factors and a selected number of design points” (Sanchez 2012).

2. *“Space-filling*: LH sample over the entire experimental domain” (Sanchez 2012).

3. *“Analysis Flexibility*: The output data allows for a fit of different models to several performance measures” (Sanchez 2012).

Due to randomness in the construction, LHs can create substantial multicollinearity among the factors (Hernandez 2008). Multicollinearity affects the output for regression modeling and partition trees. Researchers look for design that mitigate or remove correlations among the factors (Hernandez 2008). “Multicollinearity generally occurs when there are high correlations between two or more predictor variables. In other words, one predictor variable can be used to predict the other. This creates redundant information, skewing the results in a regression model” (Statistics How To 2019). An orthogonal Latin hypercube is a design in which the correlation among all the factors is zero (Hernandez 2008). A nearly orthogonal latin hypercube (NOLH) design has a maximum absolute pairwise correlation less than 0.05 (Hernandez 2008). The NOLH design does not allow the regression analysis to suffer greatly from multicollinearity. NOLH designs allow the analyst to explore the full experimental design space with a fewer design points, if the number of factors is low (Sanchez and Sanchez, 2018). Figure 1 outlines the number of design points needed based on the number of factors for evaluation.



No. of Factors	No. of Design Points
2–7	17
8–11	33
12–16	65
17–22	129
23–29	257

Figure 1. Data Requirements for Nearly Orthogonal Latin Hypercubes. Source: Sanchez and Sanchez (2018).

These designs will help ensure that the design space for a given scenario context is explored in an efficient manner to facilitate creation of the meta-model of the original simulation (Cioppa and Lucas 2007, p. 46). “Latin hypercube sampling is a statistical method for generating a sample of plausible collections of parameter values from a multidimensional distribution” (Jenkins 2015). According to Cioppa and Lucas, A good meta-model keeps the errors between the predicted and actual observations small. An example of a linear meta-model is $y = mx + b$, where y is the response variable, m is the coefficient, x is factor value and b is the *y-intercept*. A design matrix with orthogonal or nearly orthogonal values and with good space filling properties is used for fractional factorial analysis (Sanchez 2012).

Using the NOLH design to create a DOE is one method to create a set of data with good space filling properties. The DOE provides a set of design points (DPs) of different values for each of the factors that will be used to construct the meta-model. The analyst then runs the actual simulation for each design point. Once complete, the analyst can then take the results of the DOE and the simulation output to create a meta-model. A meta-model is a simpler approximation of a model or simulation, a model of a model (Barton and Meckesheimer 2006, p. 538). Meta-models allow the analyst to explore a simulation or a model without using the same computational requirements as the parent simulation. A meta-model takes less time, money and computational power for analysis because it is a subset of the parent simulation or model.

High costs, time constraints, and computational demands are some reasons why an efficient DOE is important to gain understanding and insight from combat models. A good experimental design allows the analyst to gain insights, and systematically and efficiently explore outcomes within a model. Analysts need to continue to advance this work to make the most of the methods available in order to increase the effectiveness of meta-models to represent the real world. One method is to develop a way for the computer to conduct the analysis for the analyst using adaptive sequential designs.

2. Adaptive Sequential Design of Experiments

Adaptive Sequential Design of Experiments goes by several names in analytic community; adaptive recursive approach, adaptive design of experiments (ADOE) or active learning (Blondet, Boudaoud, and Duigou 2015, p. 5). Figure 2 is a diagram of the meta-modeling process. An ADOE iteratively creates a DOE for a specific problem, model or simulation in order to maximize the DOE efficiency (Blondet, Boudaoud, and Duigou 2015, p. 5). An efficient DOE gives the analyst a set of data to build a meta-model that represents the simulation output based on the selected factors. It takes two steps to build an accurate as possible global meta-model for a deterministic simulation-based model with a reasonable computational budget (Liu et al. 2017). The first step is sampling, by building a set of points within a specified domain and the second is meta-modeling, where a function is fitted to observed points (Liu et al. 2017). This process repeats until it reaches a stopping condition. Figure 3 is one method or strategy to create an ADOE.

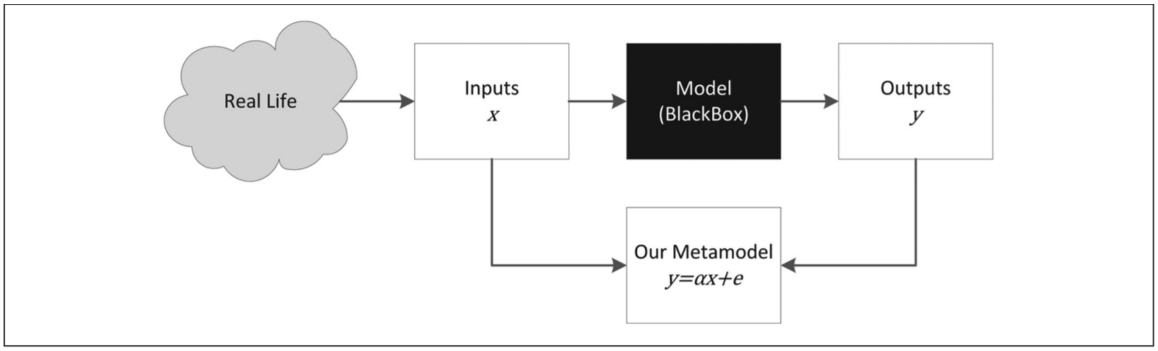


Figure 2. Meta-Modeling Fits in Simulation Analysis. Source: Jalal, Dowd, Sainfort, and Kuntz (2013).

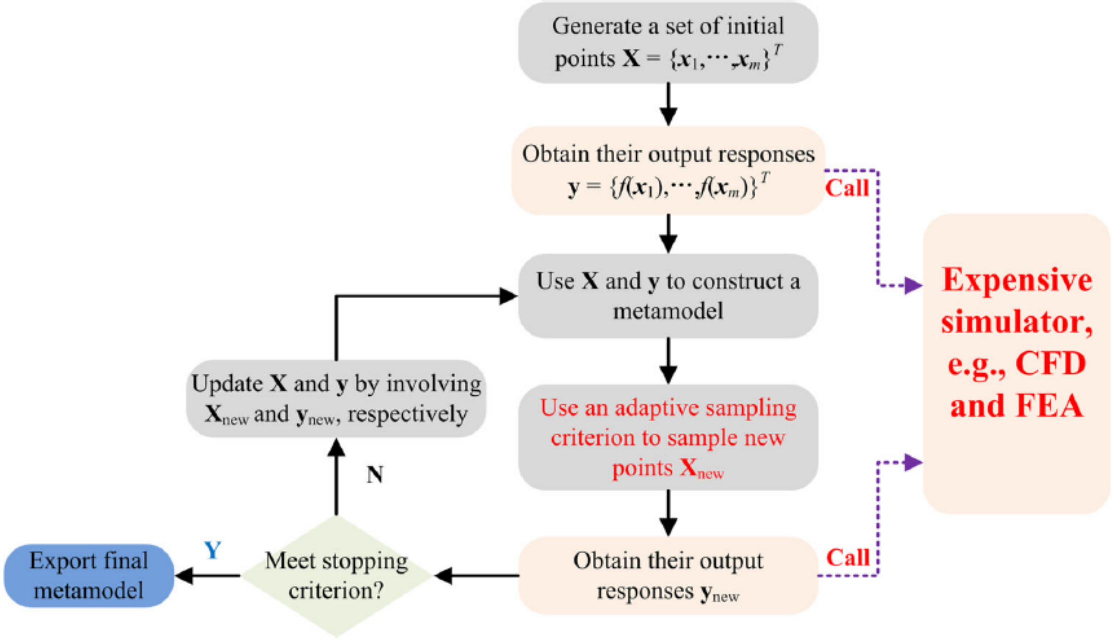

Generate a set of initial points $\mathbf{X} = \{x_1, \cdots, x_m\}^T$
Obtain their output responses $\mathbf{y} = \{f(x_1), \cdots, f(x_m)\}^T$
Call
Expensive simulator, e.g., CFD and FEA
Use X and y to construct a metamodel
Update X and y by involving $\mathbf{X}_{new}$ and $\mathbf{y}_{new}$, respectively
Use an adaptive sampling criterion to sample new points $\mathbf{X}_{new}$
N
Y
Export final metamodel
Meet stopping criterion?
Obtain their output responses $\mathbf{y}_{new}$
Call

Figure 3. “General Flowchart of an Adaptive Sampling Approach for Global Meta-Modeling.” Source: Liu et al. (2017).

An adaptive design sequentially builds off previous DOEs by choosing a new set of DPs as the function of the DPs used to create the current meta-model (Picheny et al. 2010, p. 3). A primary issue with developing dynamic ADOE is choice of infill criterion or stopping conditions (Blondet, Boudaoud, and Duigou 2015, p. 5). The infill criterion is value or formula from a meta-model that determines if another iteration is required. A widely used infill criterion is the Expected Improvement (EI) criterion, which involves a measure of possible improvement (Blondet, Boudaoud, and Duigou 2015, p. 6). Other methods are Efficient Global Optimization (EGO) and Particle Swarm Optimization Intelligent Sampling (PSIOS) (Blondet, Boudaoud, and Duigou 2015, p. 6). We will assess the quality of the meta-model by measuring the error between observed and predicted values. Each time the meta-model is created, the accuracy should improve. A meta-model improves as the error gets smaller.

A successful ADOE process seeks to create an accurate meta-model by efficiently exploring the design space. The infill criterion is important in selecting the most efficient

DOE and adding new DOEs sequentially to the initial DOE. The DOE efficiency of an ADOE relies heavily on the meta-model and the infill criterion (Blondet, Boudaoud, and Duigou 2015, p. 7). There is still more research needed to improve selecting the correct meta-model or infill criterion. Improving these would reduce computational cost and increase the speed of the process. This thesis will build off the concepts seen in the flowchart in Figure 5.

3. Multiple Linear Regression

Multiple linear regression is one of the most widely used approaches to approximate a model or simulation (Jalal et al. 2013, p. 889). "Linear regression metamodeling involves defining a model outcome as a linear function of the model input parameters. For example, in a simple model with 2 input parameters, we can define the outcome as" (Jalal et al. 2013, p. 881):

$$y = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \ldots + \alpha_n x_n + \varepsilon$$

where y is the response variable or the outcome, α_0 is the y-intercept, $\alpha_1 \ldots \alpha_n$ are the coefficients, $x_1 \ldots x_n$ are the values of factors, and ε accounts for the error between the observed and expected values. The magnitude of the coefficient represents how much of an effect the factor has on the outcome of a model. A meta-model with lower error is closer to modeling the simulation. An analyst attempts to develop the best meta-model without underfitting, overfitting, having high error, or a meta-model that works for the training data and no other case.

To use linear regression there must be a linear relationship between the factors and the predicted value. Plotting the data gives a quick assessment of the pattern in the relationship. Other assumptions for linear regression are the errors are independent, constant variance and normally distributed (Faraway 2005, p. 58). These can be observed through the behaviors of the residuals. In addition, one needs to assess data points that are potential outliers or have a high leverage, since they can influence the regression model (Faraway 2005, p. 69).

4. Tabu Search

"Tabu Search (TS) is a meta-heuristic that guides a local heuristic search procedure to explore the solution space beyond local optimality" (Glover et al. 2008, p. 1). According to Glover, TS revolves around problem solving, and incorporates adaptive memory and responsive exploration. He continues that adaptive memory feature permits the execution of procedures to search a solution space efficiently and effectively. Glover states the responsive exploration feature comes from the idea even bad choices can provide valuable information. Any optimization problem, whether linear, nonlinear or stochastic may utilize TS (Glover et al. 2008).

TS begins with a search for a local minimum solution and continues through iterations from one solution to another until termination criterion is satisfied (Glover et al. 2008). A local minimum solution is a solution that is in the feasible area to solve a problem, but it is not the best optimal solution. A global solution is the optimal solution for the problem. The TS attempts to find new solutions by exploring new regions or neighbors in the solution space in order to find a global minimum, according to Glover. TS keeps a memory of already found solutions and stores them as tabu-active attributes, he continues, and the search should not revisit tabu-active attributes satisfied. When TS stops iterations, it produces a systematically obtained global solution satisfied (Glover et al. 2008). TS methods help find the ideal meta-model for the process in this thesis.

B. LOGISTICS BATTLE COMMAND

LBC is a simulation for studying alternative logistics systems and concepts and developed in a partnership between TRAC-LEE, TRAC-MTRY, the Naval Postgraduate School, and the Argonne National Library (Akers 2008). The model considers logistics assets as entities and tracks events as these entities interact over a logistics network (Akers 2008). LBC models consumption, supply requests, resupply decisions, convoys and other logistics operations through a series of networked nodes connected by arcs (Akers 2008).

Input and scenario data for LBC is stored as Excel spreadsheets that contain the information required to model logistics and maintenance in a given real-world scenario. The simulation monitors logistics operations over a network of roads and unit locations

(Akers 2008). LBC is a discrete-event simulation model (Akers 2008). “A discrete-event simulation concerns the modeling of a system as it evolves over time by representation in which the state variables change instantaneously at separate points in time” (Law and Kelton 1991, p. 7). LBC records any change in ammunition, fuel, cargo and water levels as an event. The output of the simulation is log describing the nature and time of these state changes organized by the time of the event. An example is shown in Appendix A. These changes in the simulation provide data to assess various measures of effectiveness, such as number of convoys during the operation or percent of supplies on hand, by supply class.

This thesis used a corps level LBC scenario starting in phase 3 (major combat operations). The scenario includes four divisions with division 4 being the main effort. Division 4 consists of two Armored Brigade Combat Teams (ABCT), Combat Aviation Brigade (CAB), Division Artillery Battery (DIVARTY), Engineering Battalion (ENG BN), Maneuver Enhancement Battalion (MEB), Stryker Brigade Combat Team (SBCT), and a Sustainment Brigade (SUS BDE) to provide the necessary Brigade Support Battalions (BSBs) and Forward Support Companies (FSCs) (TRAC-LEE 2018). Figure 4 is an example of the logistics operations in support of an ABCT or SBCT.

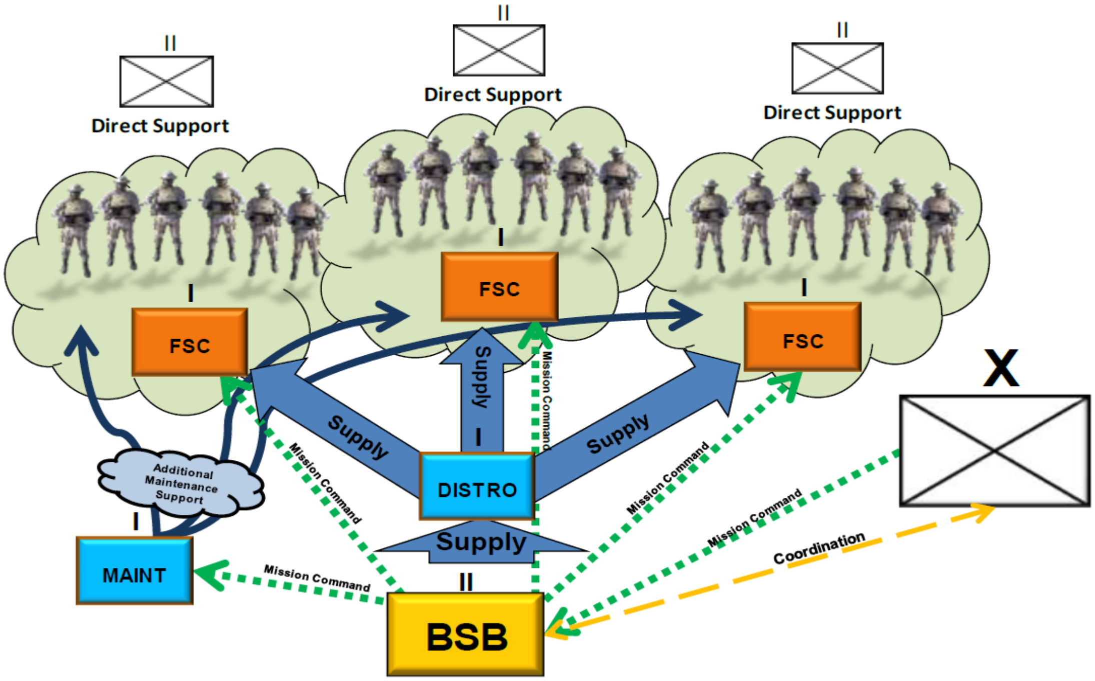


Figure 4. Example of Logistics Operations for a BSB. Source: Burgdorf (2014).

THIS PAGE INTENTIONALLY LEFT BLANK

III. METHODOLOGY PROCESS

This section outlines the tools and functions the ADOE process uses to create the final meta-model of the simulation. Figure 5 depicts the process the thesis will use to find a meta-model of a simulation.

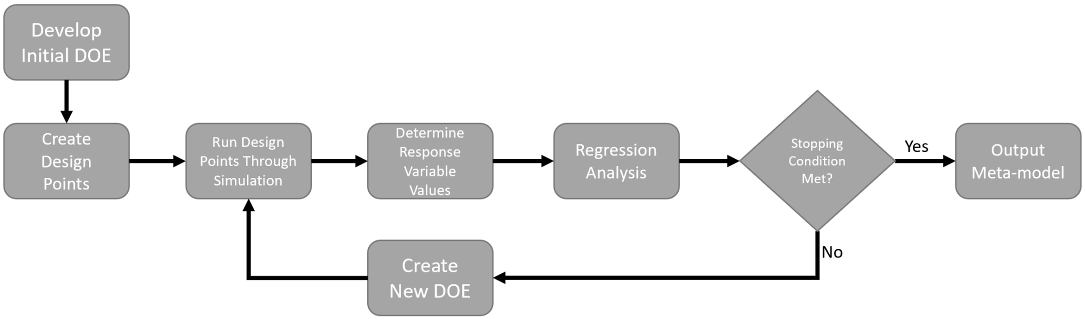


Figure 5. Flowchart for ADOE Process

A. DEVELOP INITIAL DOE

The analyst begins by identifying the measure of effectiveness from the simulation required to address their issue for analysis. For this thesis, the measure of effectiveness (MOE) is the percentage of units in the division that fell below 60% of supplies on hand in ammunition, fuel, cargo, or water during the simulation. The ADOE process creates the initial DOE after the user sets the initial operating values for the ADOE process, stopping conditions, and the acceptable threshold for the residual range. The analyst or the decision maker establishes the importance level or alpha, which varies the results or the significant factors that display.

A function gathers and collects all possible factor values from the scenario file into a data frame. The analyst explores these factors for possible use in the designed experiments. From this set, the analyst selects the factors to be used in the meta-model. For the proof of principle, we limit the number of factors to 29 and make use of an appropriate NOLH design.

For this thesis, the focus is on a single division, Division 4 since they are the main effort in Phase 3 operations. In Phase 3, the forces dominate "by breaking the enemy's will for organized resistance" (TRADOC Pamphlet 525–5-600, p. 32). This phase of the operation involves heavy offensive operations, so ammunition is important. The reorder point for combat units is a priority and effects resupply operations. We choose the ammunition reorder points for the 26 units of division 4 as the factors to include in our the ADOE process with a goal of developing a meta-model that helps the analyst understand how changes in these re-order points will impact Division 4's supply levels.

With the selected factors, the process extracts their starting values from the initial scenario file and finds a minimum and maximum range of +/- 15% of the base case values. This range reflects the region identified by the author, a former logistics officer and planner at the Brigade and Company level, as realistic for this scenario. The range is adjustable by the analyst depending on the size of the design space they want to explore.

The process creates an initial DOE with 257 design points based on a NOLH design. Recalling Figure 1, a NOLH design of 23–29 factors needs at least 257 levels to adequately fill a design space (Sanchez and Sanchez 2018). Using this space filling design method, the function creates the design points for processing through the simulation. Each design point is a different set of values for each selected factor. Figure 6 gives more detail on the steps of the process.

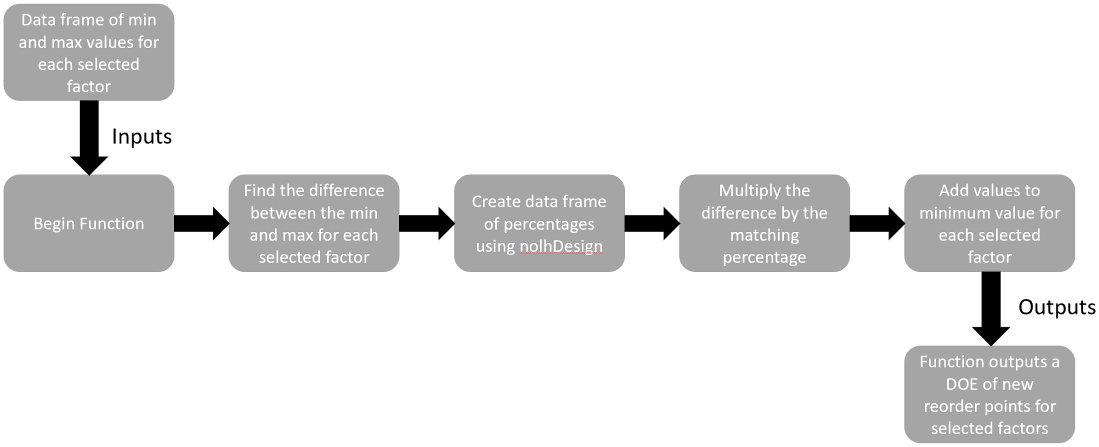

Data frame of min and max values for each selected factor
Inputs
Begin Function
Find the difference between the min and max for each selected factor
Create data frame of percentages using nolhDesign
Multiply the difference by the matching percentage
Add values to minimum value for each selected factor
Outputs
Function outputs a DOE of new reorder points for selected factors

Figure 6. Flowchart for DOE Generation Function

B. CREATE DESIGN POINTS

Taking the output, the initial DOE, the write scenario function writes the eXtensible markup language (XML) input files representing different design points. The function rewrites the values of the selected factors in the xml files along with the number of replications. The function stores the created input files on the hard drive. There are 257 design points, each replicated 30 times for a total of 7,710 replications.

C. RUN DESIGN POINTS THROUGH LBC

The next function processes each of the 7,710 replications through LBC. LBC produces comma separated values (CSV) files storing the events of all replications in one file. Any time there is a change in the supply level in any commodity type for any unit an event is logged. The function stores the csv files to the user's hard drive.

D. DETERMINE RESPONSE VARIABLE VALUES

We calculate the response variable for each replication. The response variable of interest to this analysis is the percentage of units that fell below 60% at any time during the simulation. According to Army Regulation (AR) 700–138, a unit reaches black status when supply levels fall below 60%. Black status means the unit is at grave risk and may not be able to continue with the mission (AR 700-138, p. 30). The process attaches the response variable to the corresponding design point, for regression analysis. An analyst may change the response variable to support analysis of additional measures of effectiveness.

E. REGRESSION ANALYSIS AND STOPPING CONDITIONS

The *getModel* function fits a regression model of the response variable. There are several modeling methods available for future use through the R statistical language (multiple linear regression, random forests, neural networks, etc), but for this proof of principle we make use of multiple linear regression. The process uses the *lm* function in R to create a linear model from the data. The goal of our ADOE process is to build a meta-model of the simulation that provides the user with an accurate predictive model over the region explored by the design of experiments. In order to achieve this goal, following the construction of each linear model, we seek to identify the region that the model displays the worst accuracy, as measured by the difference between the predicted and actual values, the residuals and target that area for more detailed exploration with the next design of experiments (Glover et al. 2008). These steps are repeated until stopping conditions related to model accuracy are achieved. For this proof of principle work, the user specifies the maximum percentage of residuals above a given threshold allowable. Once the percentage of residuals from the developed model above this threshold falls below the maximum allowable stopping conditions are met.

For this work, we set the threshold for maximum allowable residual values at two standard deviations from the mean residual value over all design points. For the ADOE process, *z* is the threshold value that is one of the stopping conditions for the ADOE, which is the allowable percentage of residuals outside of the acceptable range. If percentage of residuals outside two standards deviations is greater than *z,* the process does another iteration.

From the regression analysis, the ADOE process finds the design point with the highest mean residual value (after excluding extreme outliers), which becomes the base values for the next DOE. The design point with the largest mean residual allows analysts to explore more of the solution space in an effort to improve overall model performance, and effectively avoiding a local minimum. This is a variation on the tabu search method. Choosing the design point with the largest mean residual will allow the algorithm to search another set of values to see if those values will be able to improve the meta-model as the process explores more of the design space (Glover et al. 2008). The next iteration begins by generating a new DOE, using the selected design point from previous iteration as the base values. The new DOE uses a smaller range for min and max values then the initial DOE in order to target the region of interest. The initial DOE used a range of plus or minus 15% of the base values, the follow-on iterations' ranges are 10%. Keeping the range smaller for follow-on iterations allows the subsequent design to generate additional samples from the target region.

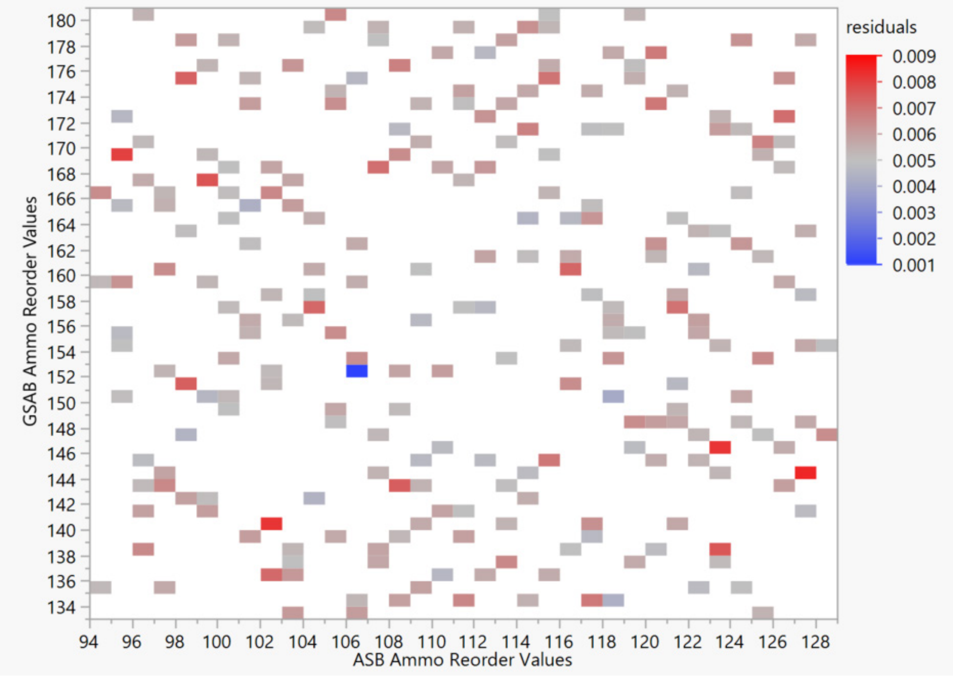


Figure 7. Heatmap of the Initial Residuals Values between the General Support Aviation Battalion (GSAB) and Aviation Support Battalion (ASB)

The heatmap in Figure 7 shows the residual for each design point in the initial DOE. Each color block is residual value of the design point containing the pair of ammo reorder values for the GSAB and the ASB, with blue being a low residual value and red a high residual value. The white space is unexplored value pairs for the GSAB and ASB. The ADOE process continues exploring the space until we end with a heatmap like in Figure 8. Figure 8 covers more values as the process moves around the design space generating additional data to inform the meta-model.

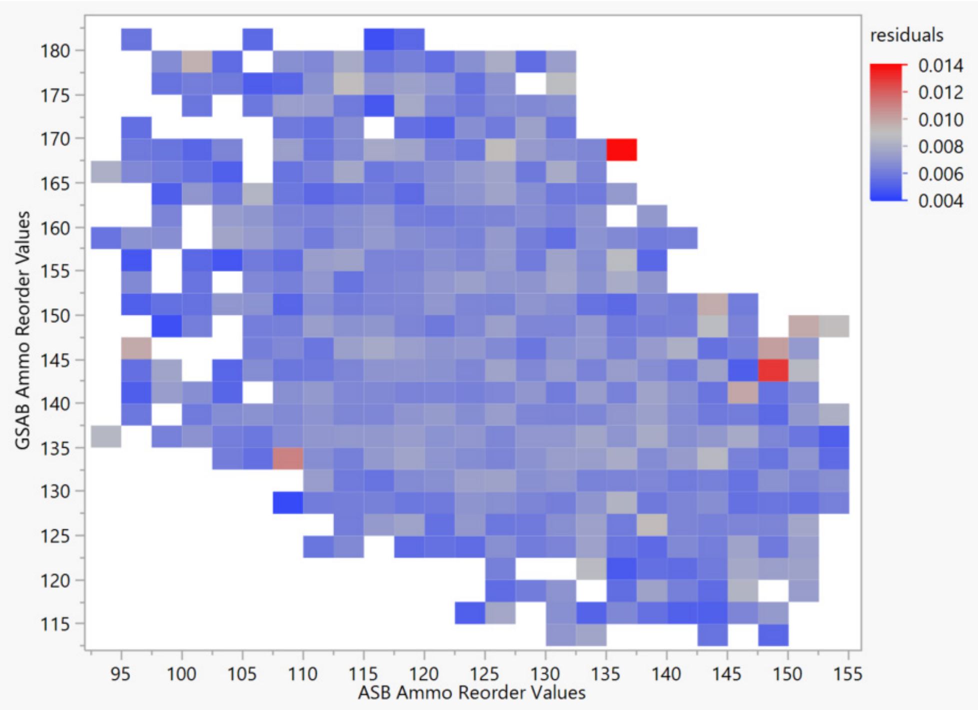


Figure 8. Heatmap of the Final Residuals Values between the General Support Aviation Battalion (GSAB) and Aviation Support Battalion (ASB)

Each follow-on iteration creates the same number of design points and the selected factors remain the same through the ADOE process. With each iteration, 30 replications of each new design point are added to the 7,710 from the original design. This increases the data set for each subsequent regression model. The process continues to iterate through DOE generation, simulating design points, and regression analysis until it reaches one of the stopping conditions.

In order to allow the analyst to conduct experimentation within their time and computational budget, the user may specify the maximum number of DOE iterations to be performed should stopping conditions not be met. This stops the model from iterating if each meta-model fails to meet the *z* threshold. Finally, the user may specify a third stopping condition to check model progress. Each iteration needs to show a minimum amount of change in the percentage of residuals above *z* - if the process is failing to demonstrate progress above a user specified threshold, the ADOE process stops. After observing different percentages of residuals outside of the acceptable range between each meta-model, the analyst determines there must be at least a 0.4% change in the percentages for this proof of principle to work.

F. FINAL META-MODEL

When the process reaches one of the stopping conditions, it stops iterating and displays information about the final meta-model. Ideally when the iterations stop, the process has generated a stable meta-model to use for predicting the MOE with consistent accuracy across the region of interest for the model's intended use. The ADOE process presents the analyst with appropriate model diagnostics such as a plot of the residuals, a QQ plot, and a list of the most significant factors.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. RESULTS

This chapter discusses the results and performance of the ADOE process during our proof of principle demonstration

We created a meta-model of the supply levels of all commodities in the units under study as a function of the reorder point of ammunition. The meta-model provides insight into the relationship between the re-order points and the percentage of units that fall below 60% of supplies in any commodity during the operation. This model will allow the decision maker to conduct what if analysis and understand how changing re-order points for one unit might impact the division as a whole.

We conducted our experiments on the high-performance computer (HPC) at the Naval Postgraduate School in order to parallelize the execution of simulation replications for each experimental design.

To find the effect the reorder point has on the percentage of units in the division that fall below 60% of supplies on hand during the operation, the analyst selects factors of interest for inclusion in the DOE. In this case, the analysis concerns the impact of ammunition reorder points for a set of units on the overall measure of effectiveness A_o. The analyst establishes the range to explore for each of the 26 factors. We specify the goal for model accuracy by setting *z* to 5% - so our process will stop when less than 5% of the residuals are above this threshold. We further specify the maximum number of iterations at 12 and that we must see at least a 0.004 change in the percentage of residuals above *z* in order to continue. Finally, we set alpha to 0.05, limiting the number of factors presented to the analyst upon completion to those with a p-value less than 0.05.

Using the NOLH space filling design, the process creates an initial DOE from the selected factors. Appendix C contains an example of the DOE generation function and appendix I is sample of the initial DOE design points the function produces.

After processing the initial data through the process, Figure 9 shows the relationship between the observed and predicted average response value for each design in the initial iteration. The graph shows the how well the meta-model predicts the percentage of units that fell below 60% of supply on hand. Ideally, if we look at an observed value of 0.11, then the corresponding predicted values are 0.11. Looking at the plot, this is not the case so there is some variance the meta-model does not account for. The goal is to find a meta-model that will provide consistent model accuracy across the range values explored. Residuals that fall outside of two standard deviations range from the average of the residual values.

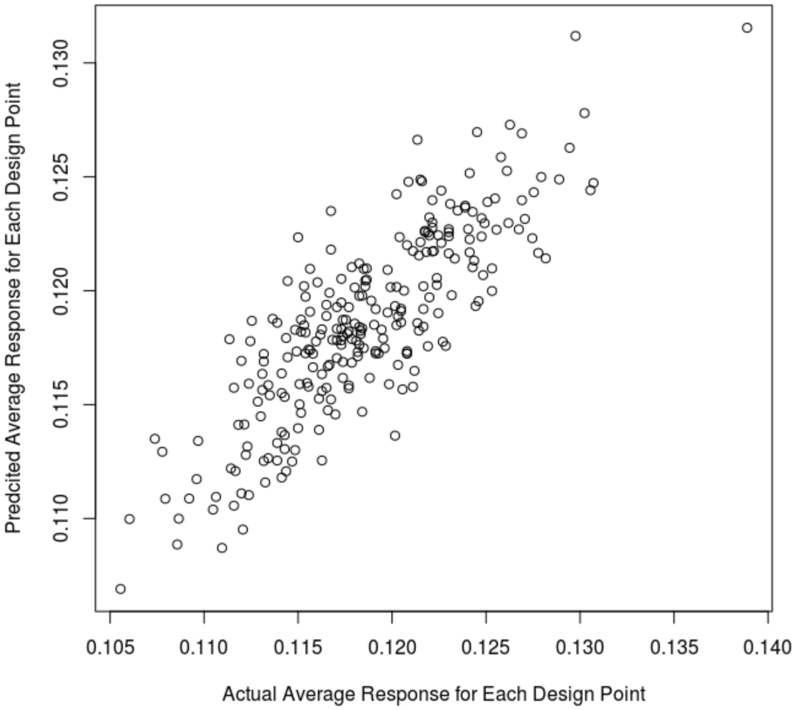


Figure 9. Observed vs. Predicted Values for Initial DOE Meta-Model.

The error in the meta-model is low, but the looking at the adjusted R squared values, at 0.233, the model only explains about 23.3% of the percent of the variance in the response. The average residual value for the meta-model is nearly zero and has about 5.42% of residuals outside of the acceptable two standard deviation range. Stopping conditions are not met, so the next design iteration begins.

The function returns the design point with the highest residual and this is the baseline for the next DOE. This time 5.49% are outside of the acceptable range with a R squared value of 25.2%.

The third iteration's meta-model's residual standard error is lower than the second meta-model, but the adjusted R squared value decreased to 18.9%. This meta-model has 5.25% of the residuals outside of the acceptable range. The improvement is less than 0.4%, which means the process stops iterating after three iterations, since it reaches a stopping condition. Looking at Figure 10, there is some improvement from the initial iteration. The predicted values seem to concentrate around the expected values.

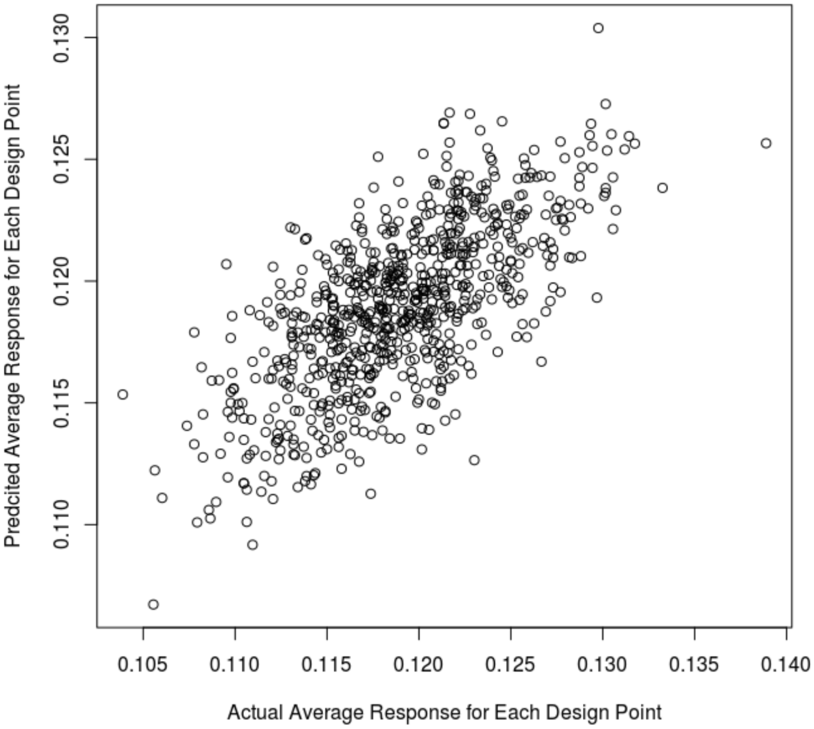
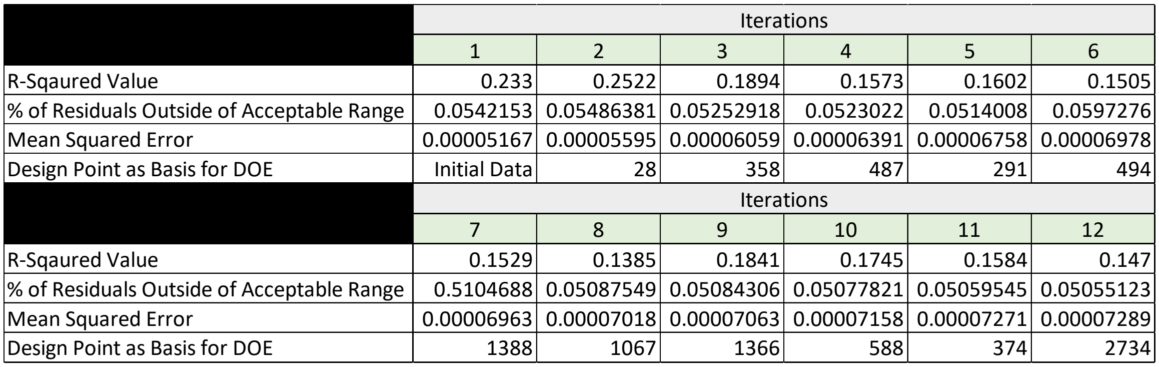


Figure 10. Observed vs. Predicted Values for 3rd DOE Meta-Model.

Table 1 is a side by side comparison on each iteration's meta-model outcome. The percentage of residuals outside of the acceptable range decreases with each iteration. In Table 1, there is steep decrease in R-squared values between iteration two and iteration three. The R-squared value measures the percentage of variance in the response accounted for by the factors in the meta-model. The R-squared value may decrease due to the increase in the number of data points that provide more information for the regression analysis to develop a meta-model. The goal is to produce a meta-model with a prescribed set of design factors and consistent predictive accuracy, rather than a model with strong explanatory power – other modeling approaches might be combined with this general methodology to develop models with greater explanatory power, but that is outside the scope of the current effort.

Table 1. Result Data for Each Meta-Model Iteration



	Iterations											
	1	2	3	4	5	6	7	8	9	10	11	12
R-Sqaured Value	0.233	0.2522	0.1894	0.1573	0.1602	0.1505	0.1529	0.1385	0.1841	0.1745	0.1584	0.147
% of Residuals Outside of Acceptable Range	0.0542153	0.05486381	0.05252918	0.0523022	0.0514008	0.0597276	0.5104688	0.05087549	0.05084306	0.05077821	0.05059545	0.05055123
Mean Squared Error	0.00005167	0.00005595	0.00006059	0.00006391	0.00006758	0.00006978	0.00006963	0.00007018	0.00007063	0.00007158	0.00007271	0.00007289
Design Point as Basis for DOE	Initial Data	28	358	487	291	494	1388	1067	1366	588	374	2734

We run the ADOE process for more iterations to see if the R squared value continues to decline after three iterations. The R squared value increases at iteration five and continues to climb for the remainder of the iterations. As the process generates more design points, each meta-model accounts for more variation and increases the explanatory power of the meta-model. However, the percentage of residuals outside of the acceptable range does not change much between the iterations. Figure 11 is a graph of the final meta-model from the ADOE process. The final meta-model has more predicted values concentrating around the expected values.

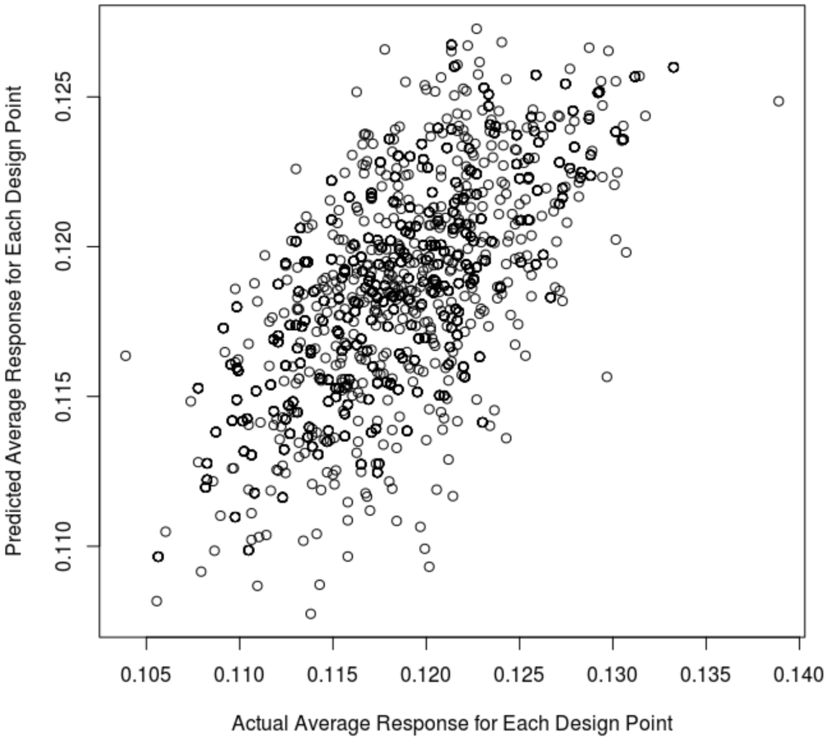


Figure 11. Observed vs. Predicted Values for Final DOE Meta-Model.

Figure 12 plots the predicted values for the reorder points for the Aviation Support Battalion (ASB) from the initial meta-model. The red lines are the upper and lower confidence interval while the blue line is the predicted value. The upper and lower confidence interval appear to fit evenly spaced around the predicted values. As the ADOE process creates more data, the red lines should get closer to the predicted values, which means there is less variation in the predicted values. Figure 13 is the same plot but for the final meta-model. The plot shows how the process assessed different values for the ASAB, as seen in the different scale of the x axis between Figure 12 and Figure 13. Figure 13 shows how the ADOE process decreases the variation in the predicted values. The process appears to improve over all predicted values. The upper and lower confidence values, the red lines in Figures 12 and 13 change as the ADOE process improves the meta-model. The difference between the upper and lower confidence values gets smaller with each meta-model iteration. We see the difference when looking at ASB reorder value is 115.076. In the initial meta-model, the prediction interval range for this point is 0.00282 and in the final meta-model, the range is 0.00223, which is a 21% reduction. The author used JMP to produce the graphs and the meta-model data for the graph comparison. The meta-models in JMP are very similar to the ones in R.

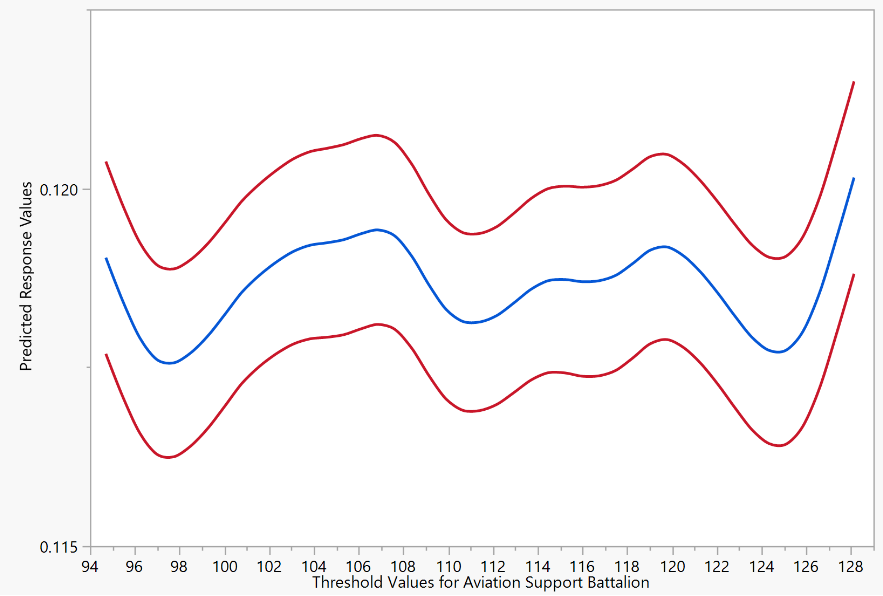


Figure 12. Graph of Initial Predicted Values for the ASB Threshold Values, with the Red Lines Representing the Confidence Interval for the Predicted Values

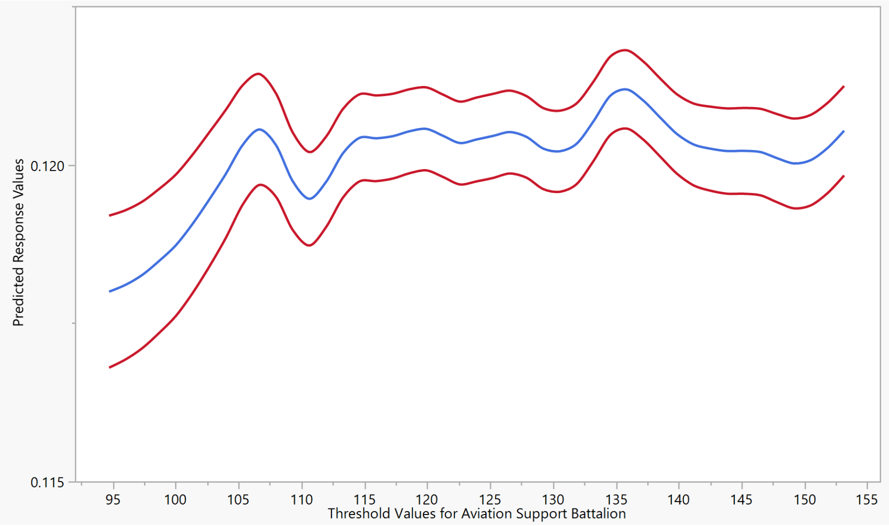


Figure 13. Graph of Final Predicted Values for the ASB Threshold Values, with the Red Lines Representing the Confidence Interval for the Predicted Values

Each iteration takes 3.5 – 4.5 hours, depending on how fast the HPC processes the simulation and the ADOE process takes about two days to complete 12 iterations when replicating each design point 30 times. Figure 14 is a chart of the first ten significant factors ordered by coefficient value. The factors returned are the reorder points the decision maker should watch since they have the greatest effect on the number of units that fall below 60% of supplies on hand for any commodity during the simulation. The analyst could use this model to make a recommendation on re-order point settings to minimize the number of units that fall below 60%.

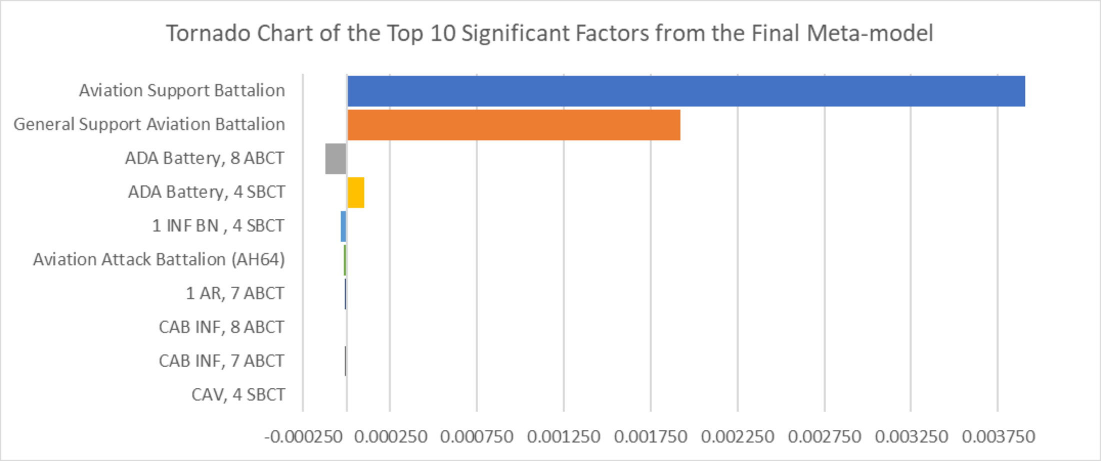


Figure 14. Chart of Top 10 Significant Factors Produced by the ADOE Process

Looking at the coefficient, out of the 26 factors and 326 two-way interactions of the factors in the linear model, these ten are the most significant. The ones with a negative value will reduce the response variable, which means increasing the reorder point value should lower the percentage of units that fall below 60% supplies on hand during the simulation and the opposite for factors with positive coefficients. For instance, each unit increase of the reorder point for General Support Aviation Battalion (GSAB) increases the response variable by that the coefficient value if all other values remained the same. The analyst verifies meta-model has a linear relationship, no multicollinearity, homoscedasticity, and residuals that are normally distributed. Once the meta-model passes all the assumptions, the analyst uses this meta-model to inform the decision maker on which units need to be a priority. Based on my analysis, the ASB and the GSAB reorder points can be lower and increase the reorder point for ADA battery for 8 ABCT. The analyst can adjust the reorder values and input those values into the meta-model and determine the percentage of units that will fall below 60%. The decision maker can use this information to determine if they can delay a supply request in favor of another unit.

The ADOE process creates a meta-model of the simulation. The meta-model does not account for all of the variance in the response, but provides a consistent level of prediction accuracy across the range of input values desired. The methodology and supporting workflow provide analysts with a set of tools to develop meta-models to support decision analysis in a variety of settings. Leveraging the data produced from extensive computational experimentation with the developed simulation model, analysts can make use of a relatively fast meta-model of the output measure of interest to support decisions in real time.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONCLUSION

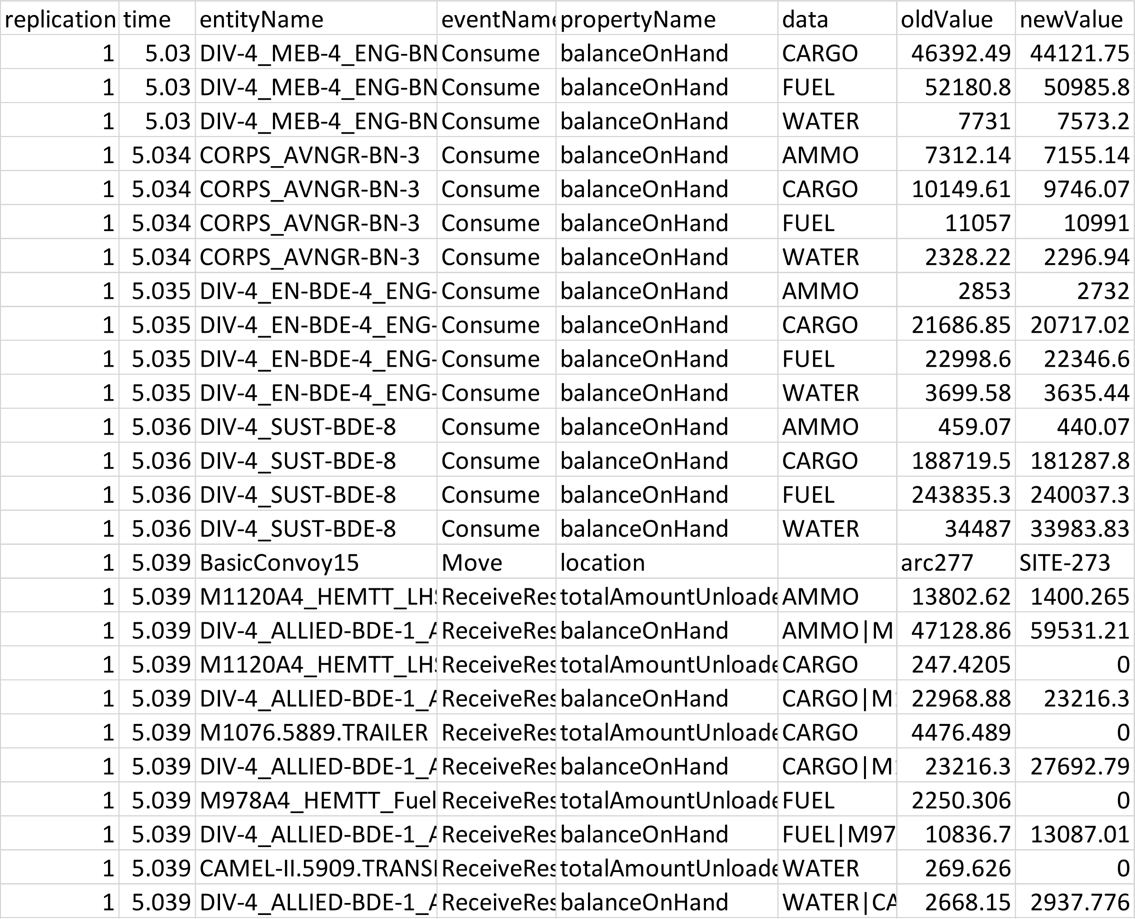
A. CONCLUSION

Simulations are important tool for military planning. The military will only increase their use of simulations for planning for future operations or conducting analysis of new capabilities designed to address operational gaps. The process of building DOEs and developing meta-models is a time intensive and iterative process. This thesis leverages techniques from the adaptive sequential design of experiments literature and ideas from tabu search to develop a simple yet innovative workflow to provide analysts with a meta-model of a complex simulation model to the desired level of predictive accuracy to support the use case (Glover et al. 2008). This thesis demonstrated a proof of principle application of this methodology to develop a meta-model intended to answer a specific analysis question. The results show the ADOE process creates a meta-model from a complex simulation that narrows the prediction interval with each iteration. As the prediction interval decreases the error in predicting the response accurately based on the selected factors decreases. The analyst informs the decision maker the reorder values for the ASB and the GSAB have the greatest effect on the response variable, using the results from the final meta-model in the proof of principle.

B. FUTURE RESEARCH

Future work could examine methods that also seek to adjust the experimental design in a manner that incorporates different factors in an effort to account for variance in the response. The current approach requires the analyst to identify a set of decision variables in the initial step and these variables are fixed throughout the adaptive design process. The proof of principle made use of multiple linear regression, but there are other modeling approaches that could be explored in future work such as random forests or neural networks. The adaptive framework could be configured to explore a number of different modeling approaches, providing recommendations to the user on the model with the best performance for a given data set. Future builds of the ADOE process can make this a product more user friendly. The development of an R Shiny application that allows the user to make the use of drop-down menus to select the factors of the ADOE process could provide access to users without a coding experience in R.

APPENDIX A. SNIPPET OF LBC RESULTS



replication	time	entityName	eventName	propertyName	data	oldValue	newValue
1	5.03	DIV-4_MEB-4_ENG-BN	Consume	balanceOnHand	CARGO	46392.49	44121.75
1	5.03	DIV-4_MEB-4_ENG-BN	Consume	balanceOnHand	FUEL	52180.8	50985.8
1	5.03	DIV-4_MEB-4_ENG-BN	Consume	balanceOnHand	WATER	7731	7573.2
1	5.034	CORPS_AVNGR-BN-3	Consume	balanceOnHand	AMMO	7312.14	7155.14
1	5.034	CORPS_AVNGR-BN-3	Consume	balanceOnHand	CARGO	10149.61	9746.07
1	5.034	CORPS_AVNGR-BN-3	Consume	balanceOnHand	FUEL	11057	10991
1	5.034	CORPS_AVNGR-BN-3	Consume	balanceOnHand	WATER	2328.22	2296.94
1	5.035	DIV-4_EN-BDE-4_ENG-	Consume	balanceOnHand	AMMO	2853	2732
1	5.035	DIV-4_EN-BDE-4_ENG-	Consume	balanceOnHand	CARGO	21686.85	20717.02
1	5.035	DIV-4_EN-BDE-4_ENG-	Consume	balanceOnHand	FUEL	22998.6	22346.6
1	5.035	DIV-4_EN-BDE-4_ENG-	Consume	balanceOnHand	WATER	3699.58	3635.44
1	5.036	DIV-4_SUST-BDE-8	Consume	balanceOnHand	AMMO	459.07	440.07
1	5.036	DIV-4_SUST-BDE-8	Consume	balanceOnHand	CARGO	188719.5	181287.8
1	5.036	DIV-4_SUST-BDE-8	Consume	balanceOnHand	FUEL	243835.3	240037.3
1	5.036	DIV-4_SUST-BDE-8	Consume	balanceOnHand	WATER	34487	33983.83
1	5.039	BasicConvoy15	Move	location		arc277	SITE-273
1	5.039	M1120A4_HEMTT_LH	ReceiveRes	totalAmountUnloade	AMMO	13802.62	1400.265
1	5.039	DIV-4_ALLIED-BDE-1_A	ReceiveRes	balanceOnHand	AMMO|M	47128.86	59531.21
1	5.039	M1120A4_HEMTT_LH	ReceiveRes	totalAmountUnloade	CARGO	247.4205	0
1	5.039	DIV-4_ALLIED-BDE-1_A	ReceiveRes	balanceOnHand	CARGO|M	22968.88	23216.3
1	5.039	M1076.5889.TRAILER	ReceiveRes	totalAmountUnloade	CARGO	4476.489	0
1	5.039	DIV-4_ALLIED-BDE-1_A	ReceiveRes	balanceOnHand	CARGO|M	23216.3	27692.79
1	5.039	M978A4_HEMTT_Fuel	ReceiveRes	totalAmountUnloade	FUEL	2250.306	0
1	5.039	DIV-4_ALLIED-BDE-1_A	ReceiveRes	balanceOnHand	FUEL|M97	10836.7	13087.01
1	5.039	CAMEL-II.5909.TRANSI	ReceiveRes	totalAmountUnloade	WATER	269.626	0
1	5.039	DIV-4_ALLIED-BDE-1_A	ReceiveRes	balanceOnHand	WATER|CA	2668.15	2937.776

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. MAIN R CODE – ADOE

```
########################################################################
##################              ADOE Code              ##########################
########################################################################

# Purpose: Aid analyst in analysis by allowing the program to determine the best range
# the factor inputs should be to obtain an objective.

########################################################################
# Packages used:

library(XML)        # Tools for parsing and generating XML within R
library(stringr)    # Common string operations
library(dplyr)      # Data manipulation
library(DiceDesign) # Generates a DOEs
library(tidyr)      # Easily tidy data with the spread() or gather() function
library(data.table) # Extension of data frame
library(bestglm)    # Best subset GLM and regression utilities
library(gtools)     # Order string experssions; mixedsort()
library(parallel)   # Allows for parallel computing
# library(DT)        # Function to print datatable
print('Libraries loaded...')

# Set initial working directory
setwd("/home/jestream/LBCv5/ADOE/")

### Load DoE Generation Functions
# Functions here include loading base case XML, creating experimental design, and
generating XML scenarios for exection
source('DOEGen.R')  # Loads DoE Generator Function (newDOE) based on 'DiceDesign'
NOLH function.
source('scenCreator.R') # Loads function to take design and base XML (makeXMLs) to
create required number of XMLs to run
source('gatherxmls.R') # Creates a list of XMLs to process through the simulation
source('runSim.R') # Loads function that runs DoE in parallel.
source('remove.R') # Remove scenarios files to clear room on hard drive
source('response4.R') # Gets the response variable from the data
source('modeling3NRAgg.R') # generates the model
source('adaptDOEGen2.R') # DOE generator for the adaptive portion of the program
source('newadptFolder3.R') # Creates new folder for storage of each iterations data
source('finalModel.R') # Displays the results and the final plot of the found meta-model
source('sigFactors.R') # Displays the found significant factors from the meta-model
print('Functions loaded...')
```

```
# Location of LBC or Simulation to use
lbcFolder <- "/home/jestream/LBCv5/"

# xml files location
setwd("/home/jestream/LBCv5/data/excel/")

# The xml file to parse for analysis
xmlfile = "SCEN7-Phase-3-5NOV2018-90.v2.xml"

# Where to store the data
StorageLoc <- "/home/jestream/LBCv5/data/"

# Name of the files for storage
namefile <- 'LR2Agg'

# identify the directory where the files are to be located
Dir = paste0(StorageLoc,namefile,"/")
Dir2 = paste0(StorageLoc,namefile,"/")

# the number of replications per design point
numReps = 30

# Sets the initial range of value for the first DOE
p = 0.15

# Sets the range of min amd max values for follow on DOEs
p2 = 0.10

# Starting value for iterations, should remain at 1 unless starting from a different iteration
k = 1

# Stopping condition for adaptive process
z = 0.05 # percent of residuals outside of acceptable range
stopVal <- 0.004 # minimum change needed to continue iteration
iterThres = 11   # number of iterations in a row the model is allowed to iterate wwithout
improvement
maxIter = 12

# Values for displaying significant factors
alpha = 0.05  # The minimum p-value a factor must have to be considered significant
n = 10       # The max number of significant factors to return

################################################################################
####################       Search Parameters in XML files
```

```
########################################################################
pstart = Sys.time() # starts the clock to time how long the program takes to complete
print('Parsing XML file...')
# Breaks the xmlFile into sections and saves a temp version for data extraction
xmldata <- xmlTreeParse(xmlfile, useInternal = TRUE)

# Locates the a number of replications value
numReplications = getNodeSet(xmldata, "//LBCAssembly[@numReplications]")

# find all ThresholdReorderLogic Nodes
ReorderNodes=getNodeSet(xmldata, "//ThresholdReorderLogic")

# Locates all the reorder point data (this section can be adjusted to look for other
#                         areas of interest such as consumption rate or truck capacities)
ReorderValueNode = getNodeSet(xmldata, "//ThresholdReorderLogic//*[@value]")
numNodes = length(ReorderValueNode)

# Create lists to collect information from xmlFile
numNodesName=character()
nodeNum = integer()
reps = character()
name = character()
Company = character()
UnitId = character()
Consumable =character()
value = character()

# this loop will return all required information from the parent nodes in reference
# to the attribute value name
for (i in 1:numNodes){
  parent = xmlParent(ReorderValueNode[[i]])
  child = xmlChildren(ReorderNodes[[1]])[i]
  grandParent = xmlParent(xmlParent(ReorderValueNode[[i]]))
  nodeNum[i] = i
  name[i] = xmlAttrs(parent) #gets the methodName of the parent node in reference to the
attribute value name
  reps[i] = xmlAttrs(numReplications[[1]])['numReplications']

  # use the name to find the Company Name
  providerNode = getNodeSet(grandParent, ".//*[@refId]")

  # extract the consumable
  Consumable[i]= xmlAttrs(providerNode[[1]])['refId']

  # extract the Company Name
  UnitId[i]= xmlAttrs(providerNode[[2]])['refId']

  # extract the value
  valueNode = getNodeSet(parent, ".//*[@value]")
  value[i]= xmlAttrs(valueNode[[1]])['value']

}
# Data frame of collected nodes from entire xmlFile
ReorderDf = data.frame(nodeNum, reps, name, UnitId, Consumable, value)
print('Parsing complete...')
##############################################################################
##################      Locating Factors of Interest
##############################################################################
# Choose the units or factors of interest through the use of dplyr filters

# This can be changed to look for any unit or factor based on the information available
# from the xml data collection

Div4 <- ReorderDf %>% select(nodeNum, reps, name, UnitId, Consumable, value) %>%
  filter(str_detect(UnitId, "DIV-4")) %>% filter(!str_detect(UnitId, paste(c("BSB", "FSC",
"DISTRO", "ALLIED", "SUST", "EN-BDE",
                                             "HHC", "HHB", "HQ",
"BEB","MEB","MI-BN"), collapse = "|"))) %>% filter(!str_detect(Consumable,
paste(c("CARGO", "WATER"),collapse = "|")))

# Enusre that MASS is not running at the same time as dplyr or else it will mask the select
function in dplyr

mainFactors <- Div4 %>% filter(str_detect(Consumable, "AMMO"))      # replace with
your choosen data set
mainFactors$value <- as.numeric(levels(mainFactors$value))[mainFactors$value] #
values to vary to create a range
# Code found at
#     https://stackoverflow.com/questions/3418128/how-to-convert-a-factor-to-integer-
numeric-without-loss-of-information

# Now set the range of min and max values

# add those values to the data frame
mainFactors['min'] <- (mainFactors$value - mainFactors$value*p)
mainFactors['max'] <- (mainFactors$value*(1+p))

# From DOEGen.R
ammoDOE <- newDOE(mainFactors)
```

```
# Creates folder for initial storage of data, it is based on the Dir set at the beginning
dir.create(Dir)

writeScenarios(ammoDOE)
files <- getXMLs(Dir)

###################################################################################
# Runs each scenario through LBC, this takes awhile since it has to shut down and restart
LBC each time
setwd(lbcFolder)

runEx(files, Dir)
removeFiles(files)

###################################################################################
################   Gather the information for the response varialbe
###################################################################################

# Finds all the csv files and then puts the names in a list
csvL <- list.files(path = Dir, pattern = '*.csv')
csvL <- mixedsort(csvL)

# Creates an empty data frame for information collection,
# this can be modified based on what desired information the user is looking for
results <- data.frame(matrix(ncol = 5))
colnames(results) <-c('DP', 'REP', 'Bad', 'Total', '%')


test <- getResponse(csvs = csvL, DOE = ammoDOE)
write.table(test, file = paste0(Dir,'test.csv'), sep = ',')
###################################################################################
# need to include code that saves each linear model and test data to the folder holding the
data

# this value is initial value to prevent repeating the same design point in consecutive
iterations
prevDPs <- list()
# prevDP = 0

par(mfrow = c(2,2))
test.mod <- getModel(test)

# takes on value of the design point with the highest residual
prevDPs <- append(prevDPs, test.mod[3])
```

```
# Save the current DOE
curDOE <- test[,1:nrow(mainFactors)]
curDOE <- as.data.frame(t(curDOE))
write.table(curDOE, file = paste0(Dir,'DOE for initial .csv'), sep = ',')

########################################################################
# Storing percent values for additional stopping criteria
percents <- list()
percents <- append(percents, test.mod[1])
c = 0

########################################################################
################   Adaptive Portion, as long as it meets conditions
########################################################################
while (test.mod[1] > z && k < maxIter && c < iterThres ){
  if(k == 1){
    print(paste('Now executing iteration',k))

    DOE <- aDOE(test, test.mod[2], mainFactors)

    drive <- newFolder(StorageLoc, k)

    Dir = paste0(drive[1],'/')
    Dir2 = paste0(drive[2],'/')

    setwd(lbcFolder)

    writeScenarios(DOE)

    files <- getXMLs(Dir)

    runEx(files, Dir)

    removeFiles(files)

    # Finds all the csv files and then puts the names in a list
    csvL <- list.files(path = Dir, pattern = '*.csv')

    # Creates an empty data frame for information collection,
    # this can be modified based on what desired information the user is looking for
    results <- data.frame(matrix(ncol = 5))
    colnames(results) <-c('DP', 'REP', 'Bad', 'Total', '%')

    newTest <- getResponse(csvs = csvL, DOE = DOE)
    test <- rbind(test,newTest)
    curDOE <- test[,1:nrow(mainFactors)]
    curDOE <- as.data.frame(t(curDOE))
    write.table(curDOE, file = paste0(Dir,'DOE_for_turn ',k,'.csv'), sep = ',')

    par(mfrow = c(2,2))
    test.mod <- getModel(test)

    prevDPs <- append(prevDPs, test.mod[3])
    # prevDP = test.mod[3]

    percents <- append(percents, test.mod[1])
    diff <- abs(percents[[k+1]] - percents[[k]])
    if(diff > stopVal){
      c = 0
    }else{
      c = c + 1
    }


    k = k + 1

  } else {

    print(paste('Now executing iteration', k))
    DOE <- aDOE(test, test.mod[2], mainFactors)
    drive <- newFolder(StorageLoc, k)

    Dir = paste0(drive[1],'/')
    Dir2 = paste0(drive[2],'/')

    setwd(lbcFolder)

    writeScenarios(DOE)

    files <- getXMLs(Dir)

    runEx(files, Dir)

    removeFiles(files)

    csvL <- list.files(path = Dir, pattern = '*.csv')

    results <- data.frame(matrix(ncol = 3, nrow = length(csvL)))
    colnames(results) <- c('All','Total', 'All%')
```

```
    newTest <- getResponse(csvs = csvL, DOE = DOE)
    test <- rbind(test,newTest)

    curDOE <- test[,1:nrow(mainFactors)]
    curDOE <- as.data.frame(t(curDOE))

    write.table(curDOE, file = paste0(Dir,'DOE_for_turn ',k,'.csv'), sep = ',')


    par(mfrow = c(2,2))
    test.mod <- getModel(test)

    prevDPs <- append(prevDPs, test.mod[3])
    # prevDP = test.mod[3]

    percents <- append(percents, test.mod[1])
    diff <- abs(percents[[k+1]] - percents[[k]])
    if(diff > stopVal){
      c = 0
    }else{
      c = c + 1
    }

    k = k + 1
  }
}

final.mod <- finalModel(test)

sigFactors(final.mod)

# End program
print('Program complete')
pend = Sys.time()
print(pend - pstart)
```

APPENDIX C. DOEGEN FUNCTION

For example, a factor value from the xml value is 200, which means the min and max is 170 and 230, respectively. This gives a value range of 60. And the *nolhDesign()* function for this factor at a certain design point is .65, then the process takes 60 * .65 = 39. Next the process takes 39 and adds it to the min value of the factor and gives a new reorder value of 209.

```
###########################################################################
###################       Function to create new set of DOE
############################################################################

newDOE <- function(df){
  # Create a list of list of percentages to use and output a set of design points for the model
  print('Creating DOE...')
  set.seed(1234)    # sent seed for the NOLH creation
  blank <- nolhDesign(nrow(df))
  # Code found at
  # https://rdrr.io/cran/DiceDesign/man/nolhDesign.html

  # Create a data frame of ranges for each factor
  ammo <- data.frame(matrix(ncol = blank$n, nrow = nrow(df)))
  for (i in 1:nrow(df)){
    ammo[i,] <- df$max[i] - df$min[i]
  }

  # transpose NOLH percentages for matrix math operations
  blank2 <- as.data.frame(t(blank$design))

  # create a data frame of ranges * NOLH percentages values to get value to add to min of
each range of each factor
  add2min <- as.data.frame(ammo * blank2)

  # create a data frame of all the min value ranges for each factor
  ammoMin <- data.frame(matrix(ncol = blank$n, nrow = nrow(df)))
  for (i in 1:nrow(df)){
    ammoMin[i,] <- df$min[i]
  }

  # create the DOE
  ammoDOE <- as.data.frame(as.matrix(ammoMin) + as.matrix(add2min))
  # Change the Row names of the DOEs
  designPoints <- NULL
  for (i in  1:blank$n){
    designPoints[i] <- paste("DP",i, sep = " ")
  }
  colnames(ammoDOE) <- designPoints
  row.names(ammoDOE) <- df$UnitId
  print('DOE created')
  return(ammoDOE)
}
```

APPENDIX D. SCENARIO WRITER FUNCTION

```
########################################################################
##################    Function to create xml scenarios for LBC
########################################################################
writeScenarios <- function(DOE){
 # this function takes the DOE and creates new scenarios
 # majority of this code came from MAJ Jim Jablonski and TaLena Fletcher
 start_time = Sys.time()
 print('Writing scenarios...')
 for(DesignRow in seq(ncol(DOE))){

   for (i in seq(nrow(DOE))){

     # identify the node containing the value and replace it with the value contained in the
Design
     index = mainFactors$nodeNum[i]
     xmlAttrs(ReorderValueNode[[index]])[2] <- paste0(DOE[i,DesignRow])
     xmlAttrs(numReplications[[1]])['numReplications'] <- numReps

   }

   # access only the //SimEntityDataLogger that contains the file attribute name
   SaveFileNodes=getNodeSet(xmldata,"//SimEntityDataLogger[@file]")
   for (j in seq(length(SaveFileNodes))){

     loggerName=xmlAttrs(SaveFileNodes[[j]])['propertyName']    #cycles    through    to
capture the name of the csv file

     xmlAttrs(SaveFileNodes[[j]])=c(file=paste0(Dir2,      'design_output_',DesignRow,'_',
loggerName, '.csv')) #appends the correct filename to the csv file in the XML document

   }

   cat(saveXML(xmldata), file=paste0(Dir,"design_point_", DesignRow  ,".xml")) #saves
the filename based on the design point number
  }
 print('Scenarios created...')
 end_time = Sys.time()
 total = end_time-start_time
 print(total)
 return(total)
}
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E. RUNSIM FUNCTION

```
### Runs Scenarios through simulation using parallel computing

runEx <- function(xmls, Dir){
 no_cores<-detectCores()

 print(paste("detected", no_cores, "cores."))

 # sets a limit for the number of cores to be used
 no_cores= 20
 if (no_cores<1){no_cores=1}
 cl <- makeCluster(no_cores)

 print(paste("Cluster initiated, running on", no_cores, "cores."))
 Dir = Dir
 start_time = Sys.time()

 # Command Line code to execute LBC
 runSim <- function(scenarios){
  # set.seed for ech rep
  system2("java", args = c("-jar", "dist/LBCRunner.jar", paste0(Dir, scenarios)))
 }

 parLapply(cl, xmls, runSim)
 stopCluster(cl)

 end_time = Sys.time()
 total = end_time - start_time
 # print(total)
 print(paste0("Total processing time"))
 print(total)
 }
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX F. GET RESPONSE FUNCTION

```
#### One way to get a response from the data

###############################################################################
getResponse <- function(csvs,DOE){
  print('Generating Reponse for each experiment...')
  start_time4 <- Sys.time()

  for (i in 1:length(csvs)){

    outputData1 <- fread(paste0(Dir,csvs[i]))

    # print(paste0('The number of replications imported ',
length(unique(outputData1$replication)), '.'))

    ## Get Data for Supply Minimum Supply Table, must be reactive value to avoid conflicts
    supplyTable <- outputData1 %>% dplyr::filter(propertyName=="balanceOnHand")
%>% filter(str_detect(`entityName`, "DIV-4")) %>% collect()


    # Find minimum supply level reached by unit/commodity, excluding convoys
    minsupplyTable <- supplyTable %>% filter(!str_detect(`data`,"Convoy")) %>%
      group_by(replication, entityName, data) %>%
      summarise(minOnHand=min(as.numeric(newValue), na.rm=TRUE)) %>%
      ungroup()

    # Find starting supply level by unit/commodity, excluding convoys
    minPercentTable<-supplyTable%>%
      filter(!str_detect(`data`,"Convoy")) %>%
      filter(!str_detect(`entityName`,"UNLIMITED")) %>%
      dplyr::select(-eventName, -propertyName, -oldValue) %>%
      group_by(replication, entityName, data) %>%
      mutate(minOnHand=min(as.numeric(newValue), na.rm=TRUE),
minTime=min(time))  %>%
      ungroup() %>%
      filter(time==minTime) %>%
      mutate(minPercOnHand=minOnHand/as.numeric(newValue)) %>%
      dplyr::select(data,replication, entityName, minPercOnHand)


    sub <- minPercentTable %>% group_by(replication) %>% filter(minPercOnHand < .60)
%>% tally()
    things <- data.frame(matrix(ncol=5))
```

```
   colnames(things) <- c('DP ', 'REP', 'Bad', 'Total', '%')
   for (j in 1:nrow(sub)){
    things[j,1]<- paste0('DP',i)
    things[j,2] <- j
    things[j,3] <- sub[j,2]
    things[j,4] <- nrow(minPercentTable[minPercentTable$replication == j,])
    things[j,5] <- things[j,3]/things[j,4]
   }
   if (i ==1){
    results <- things
   }else{
    results <- rbind(results, things)
   }

  }
  # Applies response variables to the DOE
  inputdf <- data.frame(t(DOE))
  repdf <- inputdf[rep(seq_len(nrow(inputdf)),each = numReps),]
  test <- repdf[1:nrow(repdf),]
  test$y <- results[,5]
  print('Data complete for regression analysis')
  end_time4 <- Sys.time()
  print(total4 <- end_time4 - start_time4)
  return(test)
}
```

APPENDIX G. MODEL CREATION FUNCTION

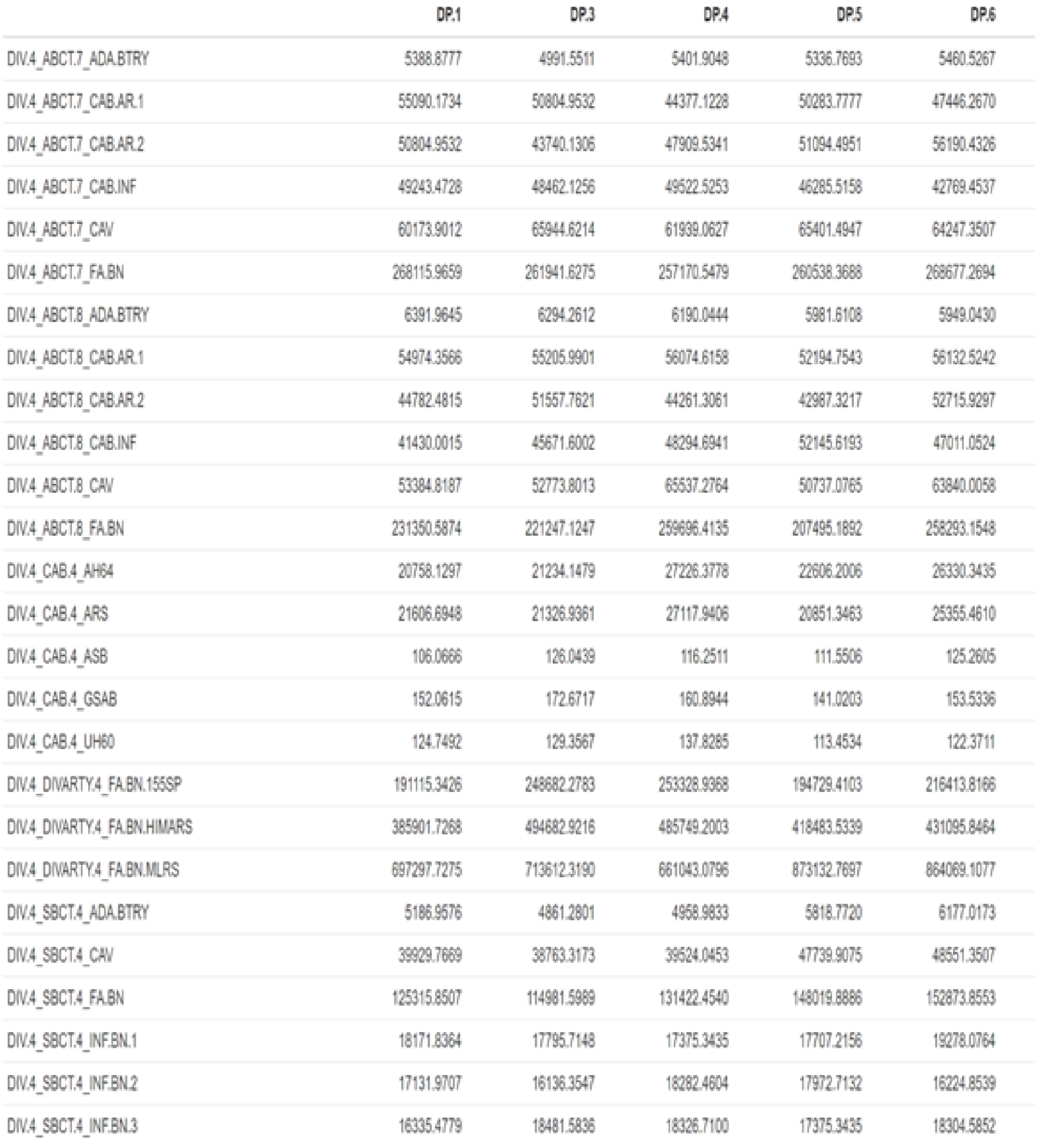
```
#######################################################################
################  Gather the information for the response varialbe
#######################################################################

getModel <- function(data){
  print('Conducting regression analysis of the data...')
  # This will find the resdiuals that fall within a certain range and return the number of
  # residuals that are outside of the range and DP with the highest residual
  lm.mod <- lm(y~(.)^2, data = data)
  print('Model Complete..')
  setwd(Dir)
  print('Saving model to folder...')
  save(lm.mod, file = 'lnmod.RData')
  # https://stackoverflow.com/questions/14761496/saving-and-loading-a-model-in-r
  print('Saving data frame to folder..')
  write.table(data, file = 'model data.csv', sep = ',')
  # https://datascienceplus.com/exporting-data-from-r/

  avg <- mean(lm.mod$residuals)
  sigma <- sd(lm.mod$residuals)

  ub <- avg + 2*sigma
  lb <- avg - 2*sigma

  resid <- sum(lm.mod$residuals>ub) + sum(lm.mod$residuals<lb)
  percent <- resid/nrow(data)
  print(paste0('The number of residuals outside of the accepted range is ',resid,'.'))
  print(paste0('The current length of the DOE is ', nrow(data),'.'))
  print(paste0('The percentage of residuals outside of the accepted range is ',
round(percent,5), '.'))

  # Section dismisses last used DP IOT avoid repeating similiar information
  resValues <- data.frame(abs(lm.mod$residuals))

  dp <- rep(1:(nrow(data)/numReps), each = 30)
  resValues$DP <- dp

  aggValues <- aggregate(resValues[,1], list(resValues[,2]), mean)

  ordAgg <- aggValues[order(-aggValues[,2]),]
  filOrdAgg <- ordAgg[!ordAgg[,1] %in% prevDPs,]
  highAgg <- which.max(resValues$DP == filOrdAgg[1,1])
  newBaseDPAgg <- filOrdAgg[1,1]


  #newCen <- which.max(abs(lm.mod$residuals))
  print(paste0('Building new DOE using design point ', newBaseDPAgg,'.'))
  print('Modeling complete...')
  return(c(percent, highAgg, newBaseDPAgg, plot(lm.mod)))
}
```

APPENDIX H. CHANGE DISTRIBUTION FUNCTION

```
library(xlsx)

setwd('location of file to modify')
consum <- read.xlsx2('ConsumptionLogic.xlsx', sheetIndex = 1)
consum2 <- consum

str(consum)

consum2$amountN <- gsub('Constant', 'Poisson', consum2$amount)

consum2

write.xlsx(consum2,'consum2.xlsx')
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX I. SAMPLE OF INITIAL DOE



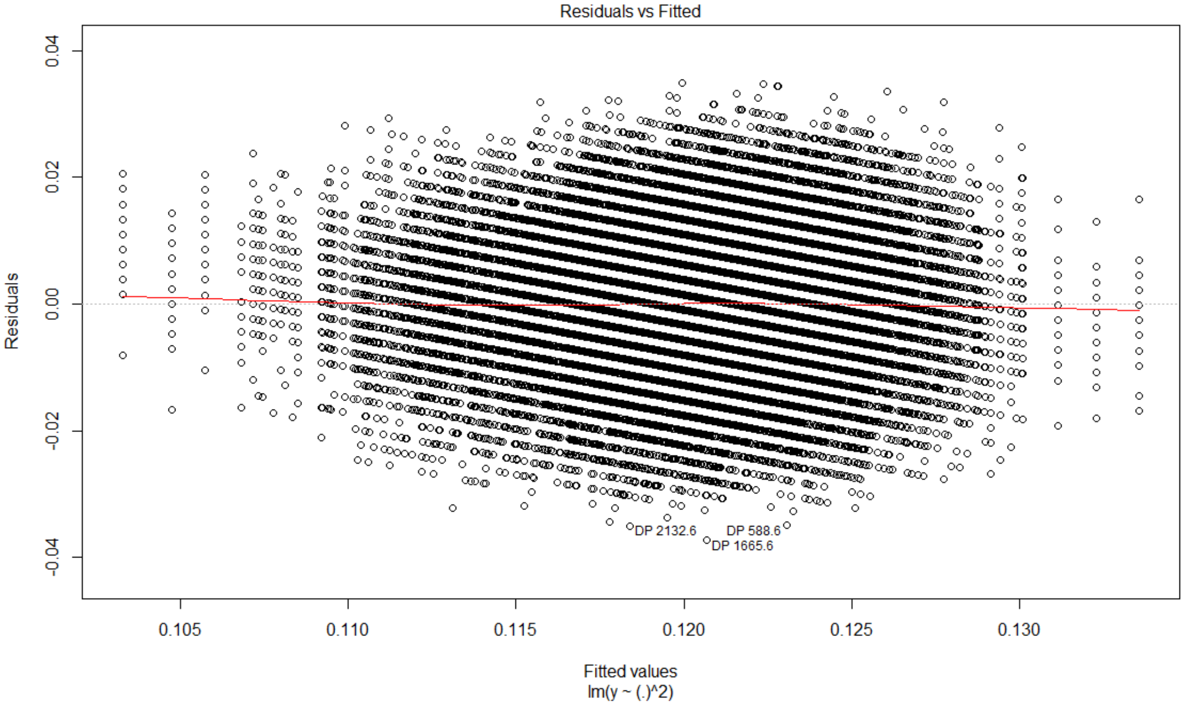
	DP.1	DP.3	DP.4	DP.5	DP.6
DIV.4_ABCT.7_ADA.BTRY	5388.8777	4991.5511	5401.9048	5336.7693	5460.5267
DIV.4_ABCT.7_CAB.AR.1	55090.1734	50804.9532	44377.1228	50283.7777	47446.2670
DIV.4_ABCT.7_CAB.AR.2	50804.9532	43740.1306	47909.5341	51094.4951	56190.4326
DIV.4_ABCT.7_CAB.INF	49243.4728	48462.1256	49522.5253	46285.5158	42769.4537
DIV.4_ABCT.7_CAV	60173.9012	65944.6214	61939.0627	65401.4947	64247.3507
DIV.4_ABCT.7_FA.BN	268115.9659	261941.6275	257170.5479	260538.3688	268677.2694
DIV.4_ABCT.8_ADA.BTRY	6391.9645	6294.2612	6190.0444	5981.6108	5949.0430
DIV.4_ABCT.8_CAB.AR.1	54974.3566	55205.9901	56074.6158	52194.7543	56132.5242
DIV.4_ABCT.8_CAB.AR.2	44782.4815	51557.7621	44261.3061	42987.3217	52715.9297
DIV.4_ABCT.8_CAB.INF	41430.0015	45671.6002	48294.6941	52145.6193	47011.0524
DIV.4_ABCT.8_CAV	53384.8187	52773.8013	65537.2764	50737.0765	63840.0058
DIV.4_ABCT.8_FA.BN	231350.5874	221247.1247	259696.4135	207495.1892	258293.1548
DIV.4_CAB.4_AH64	20758.1297	21234.1479	27226.3778	22606.2006	26330.3435
DIV.4_CAB.4_ARS	21606.6948	21326.9361	27117.9406	20851.3463	25355.4610
DIV.4_CAB.4_ASB	106.0666	126.0439	116.2511	111.5506	125.2605
DIV.4_CAB.4_GSAB	152.0615	172.6717	160.8944	141.0203	153.5336
DIV.4_CAB.4_UH60	124.7492	129.3567	137.8285	113.4534	122.3711
DIV.4_DIVARTY.4_FA.BN.155SP	191115.3426	248682.2783	253328.9368	194729.4103	216413.8166
DIV.4_DIVARTY.4_FA.BN.HIMARS	385901.7268	494682.9216	485749.2003	418483.5339	431095.8464
DIV.4_DIVARTY.4_FA.BN.MLRS	697297.7275	713612.3190	661043.0796	873132.7697	864069.1077
DIV.4_SBCT.4_ADA.BTRY	5186.9576	4861.2801	4958.9833	5818.7720	6177.0173
DIV.4_SBCT.4_CAV	39929.7669	38763.3173	39524.0453	47739.9075	48551.3507
DIV.4_SBCT.4_FA.BN	125315.8507	114981.5989	131422.4540	148019.8886	152873.8553
DIV.4_SBCT.4_INF.BN.1	18171.8364	17795.7148	17375.3435	17707.2156	19278.0764
DIV.4_SBCT.4_INF.BN.2	17131.9707	16136.3547	18282.4604	17972.7132	16224.8539
DIV.4_SBCT.4_INF.BN.3	16335.4779	18481.5836	18326.7100	17375.3435	18304.5852

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX J. TESTING THE ASSUMPTIONS OF THE FINAL META-MODEL

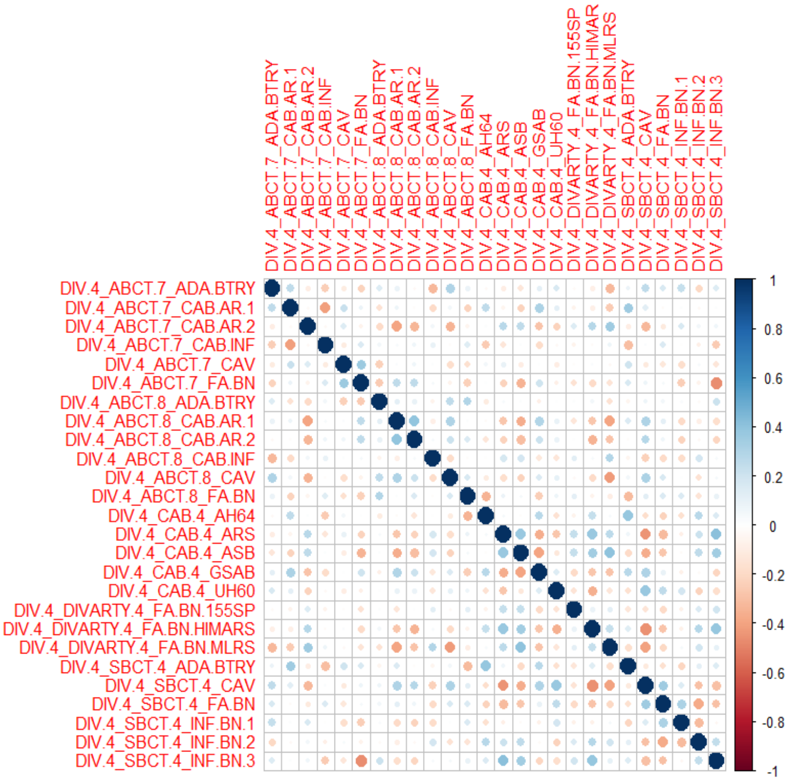
Linear Relationship?

Looking at the residuals vs fitted plot for the final meta-model we can see there is no distinct pattern and the residual value is around 0. We can assume there is a linear relationship between the dependent and independent variables.



Multicollinearity?

Looking at the corrplot from R, there does not appear to be significant collinearity among any of the factors for the final meta-models.

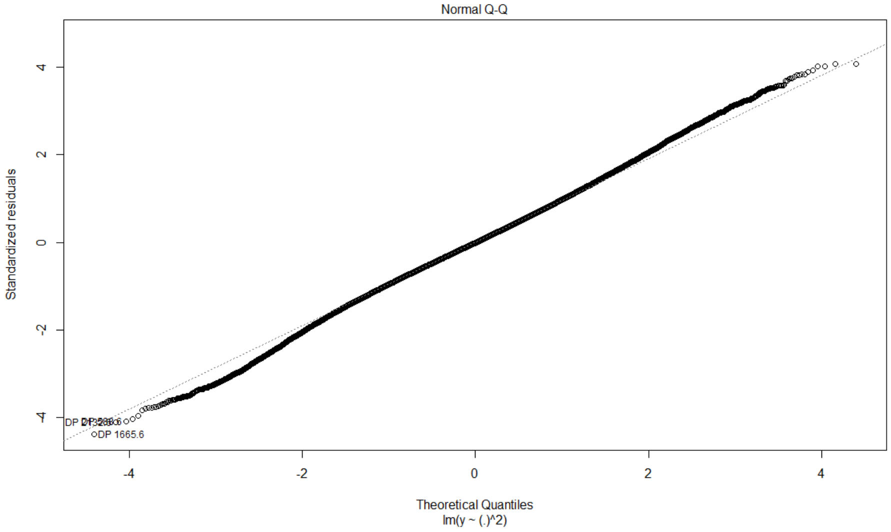


Homoscedasticity?

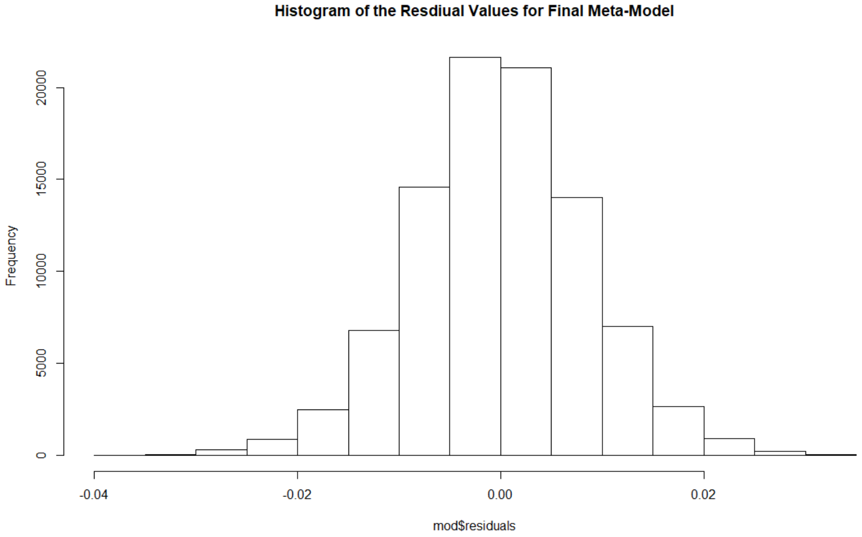
Reviewing the fitted values vs the squared standardized residuals shows no clear pattern to indicate there is heteroscedasticity.

Residuals that are normally distributed?

Looking at the QQ plot and the histogram of the residuals for the final meta-model, the residuals are normally distributed.



Normal Q-Q



Histogram of the Resdiual Values for Final Meta-Model

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Akers, Morris (2008) Logistics Battle Command (LBC) Model. Accessed May 1, 2019, https://www.trac.army.mil/20150910Lbc.pdf.

Barton R, Meckesheimer M (2006) *Meta-model-Based Simulation Optimization. Handbook in OR & MS*, Vol. 13. https://doi.org/10.1016/S0927-0507(06)13018-2.

Blondet G, Boudaound N, Le Duigou J (2015) Simulation data management for adaptive design of experiment. Literature Review, HAL Archives, Nancy, France. https://hal.archives-ouvertes.fr/hal-01149776/file/Qualita2015_Simulation_Data_Management_for_Design_of_Experiment_BLONDET_BOUDAOUD_LeDUIGOU_rev1.pdf

Burgdorf, Davie (2014) Army Logistics Quick Reference, https://www.dau.mil/cop/ammo/DAU%20Sponsored%20Documents/US%20Army%20Logistics%20Quick%20Reference%20Guide%2018Feb14.pdf.

Cioppa T, Lucas T (2012) Efficient nearly orthogonal and space-filling Latin hypercubes. *Technometrics* 49(1), https://www.tandfonline.com/doi/abs/10.1198/004017006000000453.

Department of the Army (2018) Army Logistics Readiness and Sustainability. AR 700-138, Washington, D.C., https://armypubs.army.mil/epubs/DR_pubs/DR_a/pdf/web/ARN7663_AR700-138_Web_FINAL.pdf.

Department of the Army (2008) The U.S. Army Concept of Operations for LANDWARNET 2015. TRADOC Pamphlet 525-5-600, Fort Monroe, VA, https://adminpubs.tradoc.army.mil/pamphlets/TP525-5-600.doc.

Faraway, J (2005) *Linear Models with R* (Chapman & Hall/CRC, New York).

Glover F, Laguna M, Marti R (2008) *Tabu Search*, https://www.researchgate.net/publication/228346477_Tabu_Search.

Hernandez A (2008) Breaking Barriers to Design Dimensions in Nearly Orthogonal Latin Hypercubes. Doctoral dissertation, Department of Operations Research, Naval Postgraduate School, Monterey CA. https://apps.dtic.mil/dtic/tr/fulltext/u2/a494168.pdf

Jalal H, Dowd B, Sainfort F, Kuntz K (2013) Linear regression meta-modeling as a tool to summarize and present simulation model results. *Society for Medical Decision Makin*g 33(7), https://journals.sagepub.com/doi/pdf/10.1177/0272989X13492014.

Jenkins B (2015) Latin hypercubes and all that: How DOE works. Ora Research. Accessed May 28, 2019, http://oraresearch.com/2015/02/latin-hypercubes-and-all-that/.

Law A, Kelton D (1991) *Simulation Modeling and Analysis,* 2nd ed. (McGraw-Hill Book Co., Singapore).

Liu H, Ong Y, Cai J (2017) A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design. *Structural and Multidisciplinary Optimization*, June 2017, https://www.researchgate.net/publication/317400234_A_Survey_of_Adaptive_Sampling_for_Global_Metamodeling_in_Support_of_Simulation-based_Complex_Engineering_Design.

National Institute of Standards and Technology (2012) What is experimental design? Accessed June 10, 2019, https://www.itl.nist.gov/div898/handbook/pri/section1/pri11.htm.

Picheny V, Ginsbourger D, Roustant O, Haftka R, Kim NH (2010) Adaptive designs of experiments for accurate approximation of a target region. *Journal of Mechanical Design* 132(7), http://mechanicaldesign.asmedigitalcollection.asme.org/article.aspx?articleid=1450081.

Roper D, Grassetti J (2018) Seizing the high ground – United States Army futures command. *ILW Spotlight* 18(4), https://www.ausa.org/sites/default/files/publications/SL-18-4-Seizing-the-High-Ground-United-States-Army-Futures-Command.pdf.

Sanchez S (2012) Designs for large-scale simulation experiments, with applications to defense and homeland security. Calhoun NPS Archive. Accessed January 29, 2019, https://calhoun.nps.edu/handle/10945/35261.

Sanchez S, Sanchez P (2018) OA4333 Book Draft, Winter 2018. Unpublished Textbook, Naval Postgraduate School, Monterey, CA.

Statistics How To (2019) Multicollinearity: Definition, causes, examples. Accessed June 17, 2019, https://www.statisticshowto.datasciencecentral.com/multicollinearity/.

Syberfeldt A, Grimm H, Ng A (2008) Design of experiments for training metamodels in simulation-based optimisation of manufacturing systems. *Proc. of the 18th International Conference on Flexible Automation and Intelligent Manufacturing* (University of Skovde, Skovde), http://his.diva-portal.org/smash/get/diva2:608190/FULLTEXT01.pdf.

TRAC-LEE (2018) Data provided as part of the LBC program download, November 6, TRAC-LEE, Fort Lee, VA.

TRAC-MTRY (2019) Design of experiments for simulation overview. Presentation, TRAC-MTRY, Monterey, CA.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California