2022

# Advancing the Application of Design of Experiments (DOE) to Synthetic Theater Operations Research Model (STORM) Data

Sanchez, Susan M.; Lucas, Thomas W.; Upton, Stephen C.; McDonald, Mary L.; Hernandez, Alejandro S.; Morgan, Brian L.; Barreto, Jane F.

Monterey, California: Naval Postgraduate School

https://hdl.handle.net/10945/71899

NPS-OR-22-007

# NAVAL POSTGRADUATE SCHOOL

## MONTEREY, CALIFORNIA

### ADVANCING THE APPLICATION OF DESIGN OF EXPERIMENTS TO SYNTHETIC THEATER OPERATIONS RESEARCH MODEL DATA

by

Dr. Susan M. Sanchez
Ms. Mary L. McDonald
Mr. Stephen C. Upton

March 2022

**Distribution Statement A: Approved for public release. Distribution is unlimited.**

Prepared for: OPNAV N81. This research is supported by funding from the Naval Postgraduate School, Naval Research Program (PE 0605853N/2098).
NRP Project ID NPS-N224-A.

THIS PAGE INTENTIONALLY LEFT BLANK

# REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

| | | | |
|---|---|---|---|
| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED | |
| | | START DATE | END DATE |
| 11 Mar 22 | Technical Report | 24 Oct 21 | 11 Mar 22 |

4. TITLE AND SUBTITLE
Advancing the Application of Design of Experiments to Synthetic Theater Operations Research Model Data

| | | |
|---|---|---|
| 5a. CONTRACT NUMBER | 5b. GRANT NUMBER | 5c. PROGRAM ELEMENT NUMBER<br>0605853N/2098 |
| 5d. PROJECT NUMBER<br>NPS-22-N224-A | 5e. TASK NUMBER | 5f. WORK UNIT NUMBER |

6. AUTHOR(S)
Dr. Susan M. Sanchez, Ms. Mary L. McDonald, Mr. Stephen C. Upton

| | |
|---|---|
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Operations Research Department<br>Naval Postgraduate School<br>Monterey, CA 93943 | 8. PERFORMING ORGANIZATION REPORT NUMBER<br>NPS-OR-22-007 |

| | | |
|---|---|---|
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>Naval Postgraduate School, Naval Research Program; OPNAV N81 | 10. SPONSOR/MONITOR'S ACRONYM(S) | 11. SPONSOR/MONITOR'S REPORT NUMBER(S)<br>NPS-OR-22-007;<br>NPS-22-N224-A |

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Statement A: Approved for public release. Distribution is unlimited.

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

14. ABSTRACT

The Navy uses simulation-based campaign analysis to help measure risk for investment options for equipping, supplying, maintaining, and employing naval forces. The Synthetic Theater Operations Research Model (STORM) is a stochastic simulation model used to support campaign analysis by the U.S. Navy. Building, testing, running, and analyzing campaign scenarios in STORM can be a complex time-consuming process. The goal of this research is to apply Design of Experiment methods in the selection and creation of design points to minimize the number of modeling runs required for meaningful comparisons. Another objective is to understand how best these methods can complement traditional baseline and excursion modeling.

15. SUBJECT TERMS
Campaign analysis, data science, design of experiments, modeling, simulation, synthetic theater operations research model, STORM

| | | | | |
|---|---|---|---|---|
| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES |
| a. REPORT | b. ABSTRACT | C. THIS PAGE | | |
| U | U | U | SAR | 37 |

| | |
|---|---|
| 19a. NAME OF RESPONSIBLE PERSON<br>Dr. Susan M. Sanchez | 19b. PHONE NUMBER *(Include area code)*<br>831-656-2780 |

THIS PAGE INTENTIONALLY LEFT BLANK

**NAVAL POSTGRADUATE SCHOOL**
**Monterey, California 93943-5000**

Ann E. Rondeau
President

Scott Gartner
Provost

The report entitled "Advancing the Application of Design of Experiments to Synthetic Operations Research Model Data" was prepared for OPNAV N81 and funded by the Naval Postgraduate School, Naval Research Program (PE 0605853N/2098).

**Distribution Statement A: Approved for public release. Distribution is unlimited.**

**This report was prepared by:**

Susan M. Sanchez
Distinguished Professor
Operations Research Department

Mary L. McDonald
Faculty Associate – Research
Operations Research Department

Stephen C. Upton
Faculty Associate – Research
Operations Research Department

**Reviewed by:**

W. Matthew Carlyle
Chair, Operations Research Department

**Released by:**

Kevin B. Smith
Vice Provost for Research

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

# I. BACKGROUND

## A. PURPOSE

Navy leadership is interested in initiatives that can potentially increase the responsiveness of campaign analysis. Simulation-based campaign analysis is used to measure risk for investment options in how best to equip, organize, supply, maintain, train, and employ our naval forces. The Synthetic Theater Operations Research Model (STORM) is a stochastic simulation model used to support campaign analysis by the U.S. Navy, Marine Corps, and Air Force. Building, testing, running, and analyzing campaign scenarios in STORM is a complex, time-consuming process. A simulated campaign may span months, involve scores of ships and battalions, hundreds of aircraft and installations, all executing thousands of interconnected missions involving numerous events in time and space. Creating, testing, and approving the inputs for a single design point (DP) requires a significant investment in analysts' time and computing resources. Consequently, there are limits on the number of DPs that can be produced, executed, and analyzed during a study's timeframe.

The purpose of this research is to assess state-of-the-art methods in computational experimental design and other technologies with a goal of improving the timeliness, breadth, and robustness of future Navy studies using STORM. The long-term objectives are to apply cutting-edge sequential and adaptive design of experiment (DOE) methods in the selection of DPs to minimize the number of modeling runs required for meaningful comparisons, and to develop an understanding of the conditions in which these sophisticated designs are useful in comparison to traditional baseline and excursion modeling. The DOE methods should ensure control over variation so that insights gained through analysis are meaningful, timely, and defensible. In this initial phase, we present three approaches (sequential, comparative, and focused) and describe opportunities for their use on STORM scenarios that are either unclassified, mature classified scenarios, or working scenarios.  We recommend applying a mix of all three methodologies to a classified scenario in the future.

## B. STORM CAMPAIGN MODEL

The Synthetic Theater Operations Research Model (STORM) is a data-driven, multi-sided, stochastic computer simulation of military operations covering the air, space, land and maritime domains, including full logistics, energy, maintenance, intelligence, surveillance, and reconnaissance (ISR), and weather impacts (Group W, 2021). STORM is designed to provide campaign analysts with the ability to examine issues involving the utility and effectiveness of combat power in a theater-level joint warfighting context. STORM is sponsored by Headquarters US Air Force Studies, Analyses and Assessments.

STORM is a closed-form analytical campaign simulation, meaning that an analyst does not operate in the loop during run time. The model runs orders of magnitude faster than real time and has been used for force structure, force employment, and system trades studies (Group W, 2021). Being stochastic in nature, a set of replications is typically performed for a given case, and therefore, for any metric or data element, a range of outcomes is expected over the replications.

Having been designed to capture warfare at the campaign level, this means: (1) physical entity representation is often aggregated, (2) geography and time are represented on a large scale, for example, at the theater or multi-theater level and spanning weeks or months vice days, and (3) the model is capable of representing a wide breadth of missions, capturing the outcome of complex interactions for the duration of large-scale, joint force operations.

# II. DESIGN OF EXPERIMENT METHODOLOGY

## A. INTRODUCTION

The Simulation Experiments & Efficient Designs (SEED) Center uses the metaphor of “data farming' to describe iterative design and analysis of computer experiments. Just like a farmer cultivates their plot of land to maximize yield, a data farmer intentionally and effectively manipulates simulation inputs using sound techniques from the design of experiments (DOE) literature, in order to maximize information yield. The idea is that the data farmer “grows” data needed for their analysis, according to their carefully designed experimental plan. If the plan for conducting model runs is not well-designed, several pitfalls can occur, which would severely limit the information that can be gained. Among these pitfalls are (1) basing analytic recommendations upon the output of only a few runs of a model; (2) confounding experiment variables (called factors), meaning that the effects of factors on the response cannot be untangled; and (3) failing to detect an interaction effect, meaning that the effect that a factor has on the response depends on the value of the factor(s) it interacts with.

Experimental design has been used for decades, within a wide variety of fields, but technological breakthroughs in computing power and efficient designs have better positioned decision makers to conduct experiments on their (increasingly complex) simulation models in a more timely matter. Quite simply, data farming makes possible the completion of experiments that otherwise would not have been feasible due to the amount of time required. It enables the simultaneous variation of a large number of inputs, such as force sizes, concept of operations or employment, weapon capabilities, environmental conditions, etc., in an efficient manner. This makes possible the estimation of models relating multiple outputs (e.g., casualties, number of platforms out of action, achievement of conditions necessary to advance the campaign, etc.) to the inputs that were varied. By applying sound design of experiment techniques to simulations, analysts drastically improve their ability to extract valuable information, gain a better understanding of the solution space, and therefore better support decision making.

For a relatively concise, yet thorough and recently updated, introduction to designing computer simulation experiments, see (Sanchez et al., 2021). We will present a few main points, along with a brief discussion of selected design types, in the next section.

## B. DISCUSSION OF SELECTED DESIGNS

There are many design types; we select a few to describe in this section that we have found particularly useful for simulation experiments.

### 1. Full factorial

Perhaps the most commonly known design is a gridded, factorial design called the full factorial. The full factorial tests every possible combination of a set of factors. From an information standpoint, this design is ideal, but the issue is when a model has a large number of input variables the number of required runs increases exponentially. This exponential increase is known as the curse of dimensionality. As we will discuss, efficient designs can break the curse.

### 2. Fractional factorial and central composite designs

A fractional factorial design is a reduced version of the $2^k$ full factorial design, testing k factors at two levels each, at notional "low" and "high" values, requiring only a fraction of the number of design points that would have been required for the full factorial. These designs come in various sizes, and resolutions. The basic idea is that in return for increased efficiency (fewer design points), one must trade off some ability to estimate interactions. This may be acceptable, particularly if the goal is to perform a screening experiment, in order to separate the vital few from the trivially many.

It is beyond the scope here to delve into further detail, but we frequently use the resolution V fractional factorials (R5-FFs) because they allow the linear main effects and two-way interactions of many factors to be investigated simultaneously, without confounding. Expanding these R5-FFs to central composite designs provides some information about nonlinear behavior in simulation response surfaces and permit orthogonal estimation of full second-order metamodels.

### 3. Very large resolution V fractional factorial designs

These so called very large resolution V fractional factorial designs are able to simultaneously estimate all main effects and two-way interactions without confounding for up to 120 factors, many more than are available in the classic literature. These designs can be used for screening on both main-effects and two-way interactions, as well as for exploring binary factors.

Design generators for these are available for download at the SEED Center for Data Farming web site (https://harvest.nps.edu) as portable cross-platform Ruby (https://www.ruby-lang.org/) scripts. The method has also been implemented in the R package FrF2Large (https://rdrr.io/cran/FrF2/man/FrF2Large.html).

### 4. Nearly orthogonal Latin hypercube

An alternative to the full factorial design, ideal for a set of continuous-valued factors, is the nearly orthogonal Latin hypercube (NOLH). Developed by Cioppa and Lucas in 2007, the NOLH design is an efficient, flexible, and space-filling design that minimizes correlations among variables (Cioppa and Lucas, 2007). Correlations among variables can mask or confound effects. The NOLH designs have space-filling properties similar to those of full factorial designs, but require fewer runs to achieve comparable space-filling results. A convenient Excel workbook NOLH template can be downloaded from https://harvest.nps.edu. It contains worksheets for designs of various sizes. The largest design can accommodate up to 29 factors using 257 DPs.

### 5. Nearly orthogonal and balanced (NOB) designs

The nearly orthogonal and balanced (NOB) design is similar in principle to the NOLH, in that it is space-filling, efficient, and flexible with respect to analyses that can be performed on the resulting data, but unlike the NOLH, it was explicitly designed to handle a mix of factor types (nominal, discrete, continuous). Though rounding is possible with the NOLH design, it works best with a set of continuous factors, as the rounding can significantly increase the pairwise correlations.

A design generator developed by (Vieira Jr et al. 2013) allows for the creation of designs of various sizes. An Excel workbook NOB template, available on the SEED web site, can accommodate 10 blocks of 20 k-level factors (k=2,3,…11) plus 100 continuous factors, requiring only 512 design points.

### 6. Sequential designs

In contrast to a single-stage design, sequential design methodology works by first testing a set of DPs, using information gained to determine which DPs to try next, and repeating until some stopping criterion is met.

Methods for adaptive experiments can generally be classified under two separate categories: objective seeking and global fitting (Erickson et al., 2021). Objective seeking experiments aim to find a single point that optimizes a function, while global fitting methods aim to fit a model that estimates a surface over its entire domain. An example of a strategy to determine the next DPs to be tested, given the set already collected, is to add points where the metamodel uncertainty is highest. Sampling in a sequential manner, vice all at once in a single-stage design, can be particularly valuable when each design point requires significant time and effort to create and/or have lengthy run time.

There also exist hybrid algorithms that combine global fitting and optimization. Selection of next points may also be based on a gradient descent algorithm and/or use value criteria (Erickson et al., 2021).

### 7. Nested designs

In a nested design, the levels of one factor depend on the value of the factor it is nested within. For example, say there may be two main variants of each of two surface vessel platforms, but the two platforms are significantly different from one another, as are each of its variants. In this case, each surface vessel may have two variants, named 'A' and 'B', but each vessel's variants bear no similarity to one another, even though they are similarly named with the labels of A and B. In this case, the Variant factor would be nested within the Surface Vessel factor, and in this case, the analysis would not consider a full two-way interaction between Variant and Surface Vessel. One would instead analyze the effect of Variant only within each category of Surface Vessel.

### 8. Combining designs

Designs can be combined in interesting and useful ways. For instance, if factor ranges are kept the same (or symmetric about a central baseline design point), two or more orthogonal designs can be concatenated and still maintain orthogonality—an example would be adding an R5-FF design to a NOB design to provide more sampling near the corners of the input factor space.

If we classify and separate the factors into decision factors (controllable in the real world) and noise factors (uncontrollable in the real world), we can either construct a crossed design from separate designs for the two factor classes, or construct a single combined design. Crossing a decision factor design (say with m design points) with a noise factor design (say with n design points) means that the n noise design points will be repeated for each of the m decision design points, resulting in m x n total design points. Crossing can be computational expensive, but allows a full "apples to apples" comparison of results, since each of the decision design variations is run over the exact same noise variations.

It is more important to use any good design than to use a particular design. While the NOB designs (and other designs based on Latin hypercube (LH) designs) we commonly use were developed with an explicit interest in reducing the maximum pairwise correlation among main effect estimates, other designs are available in various software packages, including the DOE menu in statistical software JMP (https://www.jmp.com/), several R packages for creating maximin or spacefilling LH designs (http://CRAN.R-project.org/), and custom design software (http://www.statease.com/dx10.html).

# III. APPLYING DOE TO STORM

## A. TWO TYPES OF DESIGN POINTS

The overarching research objectives are to apply design of experiment (DOE) methods in the selection and creation of design points to minimize the number of STORM runs required for meaningful comparisons and to determine how best DOE methods can complement traditional baseline and excursion modeling. The following questions will guide the research:

- How much variation in design points (DPs) can modelers explore before the level of effort exceeds the benefit?
- What variations in capabilities or force structure provide the best information to be prepared to answer future questions from decision makers?
- How should we include existing STORM results in the selection process of new DPs to sequentially and efficiently explore the most important variations?

For our purpose in this research, we describe two types of DPs:

- Those that require a significant investment in analysts' time and computing resources, e.g., that reflect qualitatively different operational policies or C2 plans. There are limits on the number of these "major" DPs that can be produced, executed, and analyzed during a study's timeframe.
- Those that involve changes to quantitative inputs that are more straightforward to articulate and implement, such as quantitative model inputs that can vary over specified ranges. Running experiments involving these types of DPs may still require time, but they generally do not have such long lead times.

The "major" DPs are of most interest to N81 because of the extensive development time and cost required. The potential utility of our research will be to better understand the relationships between STORM input variables and STORM outputs, and to determine the effectiveness of sequential DOE for achieving this understanding.

## B. THREE APPROACHES

The application of DOE methods to the selection and creation of new DPs may help to streamline the study process and reduce the number of modeling runs required for meaningful comparisons.  Three types of approaches are possible. One or more of these can be used, as determined by the team.

The first approach, a sequential approach, could be used to suggest future "major" DPs involving the difficult-to-change inputs. This would involve qualitative inputs that are not as amenable to automatic exploration, but require substantial time and effort to instantiate new major DPs. Determination of specific DPs to instantiate would be driven by needs/expertise/questions of N81 following deep dives into analysis and interpretation of previous major DPs. The DOE comprised of these new DPs will be very efficient, i.e., a limited number of new DPs will be created. The sequential DOE approach means that a deep dive into analysis and interpretation (by the team) will occur after each new major DP is created. If the results indicate that changes to the DOE are needed, or that further DP exploration is unlikely to be worth the effort, the initial DOE plan will be altered or halted as appropriate.

The second approach, a comparative approach could be used to aid in verification and validation efforts and help identify reasonable factor ranges or settings. This involves structured parameter variation for (some of) the easier-to-change, quantitative factors and thresholds. It can provide guidance on factor ranges and combinations for which STORM does or does not produce credible output. If a working scenario is selected, comparisons can be used within the sequential approach: iterating over smaller DOEs can assist in verification/validation efforts during the (longer process of) creation of a new major DP.

The third approach, a focused approach involving one or more existing DPs could provide guidance regarding appropriate metrics, factor ranges or levels, or sensitivity levels. This does not involve creation of new major DPs. Analysis of experiments involving an existing major DP may help reveal how much variation in other DPs is worth exploring. For example, DOE could be used to efficiently identify ranges for selected inputs for which the STORM output is relatively stable, or ranges beyond which the STORM output is not credible. Other features or components to explore would be guided by general needs that N81 has identified. For example, one challenge in a long-term study is the need to wait for certain types of data (perhaps coming from higher classification levels) before doing analysis. With a focused approach, we could explore what type(s) of experiments might provide useful intermediate information regarding STORM's sensitivity to components where data are uncertain or at a classification level above SECRET. These methods or lessons learned might transfer over to working scenarios, so N81 could gain insights from experiments conducted while waiting for input from others on certain components of the STORM database.

## C. SELECTING FACTOR TYPES AND RANGES

The first step in designing an experiment is determining the set of factors to be varied and their respective levels or ranges. There are many strategies for deciding on an input space to be explored. For example, a sensitivity analysis that varies factors plus or minus 10-20% of their baseline values may be selected to determine if any key measures are highly sensitive to these changes. Alternatively, a broader "what if" type of experiment that varies factors more broadly, beyond what is considered possible today, might be performed to determine where the so called 'knees in the curve' occur.

Another approach would be to conduct a structured search for solutions (campaign strategies and tactics) that are robust to disruptions or other sources of uncertainty. Such a study might also provide insight about how new technologies (e.g., unmanned vehicles or new weapons systems), perhaps combined with an alternative force structure and tactics, might simultaneously lead to mission success.

These approaches might also be combined, for example, by varying some factors over broad ranges, other factors over more narrow ranges, and combining into a single or crossed design.

## D. APPROACHES FOR HANDLING CONSTRAINTS IN INPUT SPACE

### 1. Functions of inputs as factors

Factors need not correspond directly to simulation inputs. Taking an example from (Sanchez et al., 2022), suppose two inputs are the mean times $\mu_1$ and $\mu_2$ required for a specific agent to process messages from class 1 and class 2, respectively, where message class 2 is considered more complex than message class 1. Varying $\mu_1$ and $\mu_2$ independently may either result in unrealistic situations where $\mu_1 > \mu_2$, or require the analyst to select narrow factor ranges. Instead, we could use $\mu_1$ as one factor to represent the capabilities of the agent, and vary the ratio $\mu_2/\mu_1$ over a range of interesting values (say, 1.1 to 2.0) to represent the relative difference in message complexity.

This same idea could also apply to the speeds over which to vary a slower, larger platform and an inherently smaller, faster platform. For example, the slower platform's speed might be varied over 15 to 25 knots, while the faster platform's speed is varied as a ratio that is 2 to 4 times that of the slower platform.

### 2. Mixtures

There are several options for varying factors that constitute a mix, for example, factors that represent a mix of munition types. Briefly, these include: (1) vary all but one of these independently and determine the last one's value such that the total number of munitions is fixed; (2) vary all factors independently and apply cost-benefit analysis to the results; (3) use a space-filling design that satisfies a constraint, where satisfying the constraint could just mean deciding which DPs NOT to run from, e.g., an NOLH or NOB; (4) use some ratios as factors as discussed in the previous sub-section; or (5) use a classical mixture design or other special-purpose design.

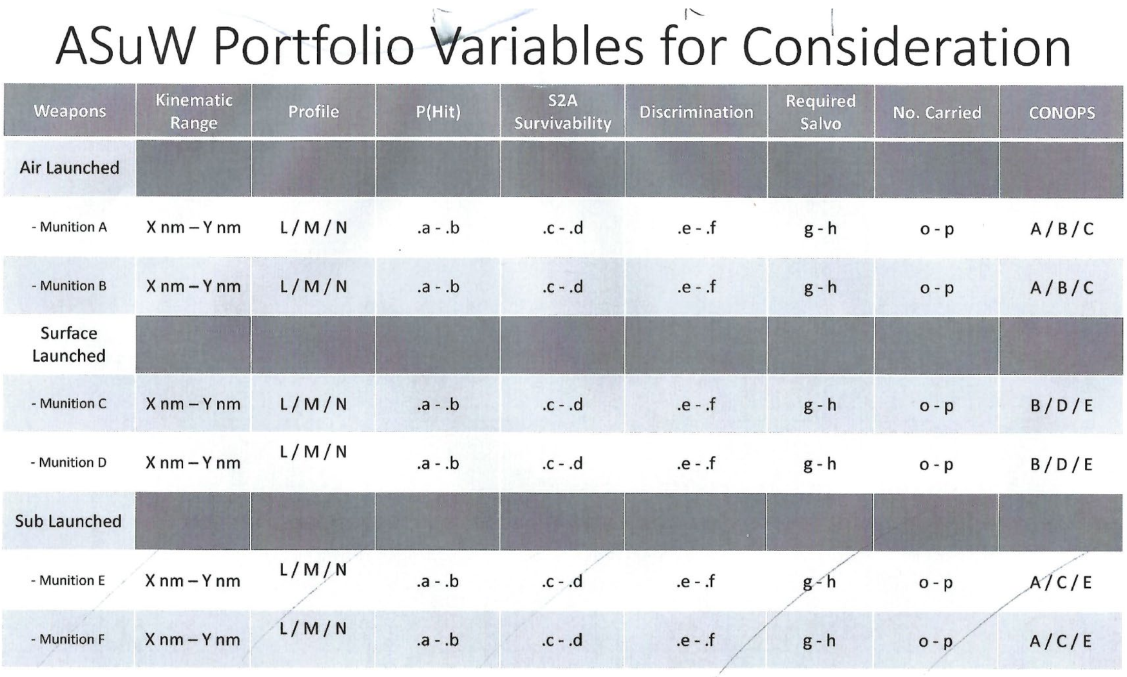
### 3. Related or correlated factors

A desired set of factors may contain some that are related to other factors, i.e., they are not independent. Some options for handling related, dependent, or correlated factors are:

- *Lock-step factors for all entities in a particular group.* For example, perhaps every unmanned surface vessel of a particular class has the same capabilities.
- *Vary distributional parameters as factors, rather than separate factors for each individual*. For example, a distributional factor might capture a range of performance values, such as phit, of a weapon. This presumes that the model is capable of receiving distributional parameters as input.
- *Lock-step some factors so they vary together*. For example, perhaps we spread out a fixed number of munitions across platforms, or, perhaps weather conditions degrade multiple capabilities.
- *Nest factors*. As discussed in the previous section, this means that the levels of a factor are tied to the level of the factor it is nested within.
- *Use qualitative factors*. This might mean implementing "big" changes that vary several things together, like a "scenario" factor.
- *Start with a full design, but remove design points that correspond to illogical or infeasible design points*. It may be useful, however, to evaluate design points that stretch initial intuition about infeasibility, however, since (1) intuition may not be correct due to complex interactions within the scenario, and (2) evaluating does not mean endorsing. Casting a wider net in exploration may be more useful than unnecessarily limiting options. If design points are removed, plots and correlations of the factors should be checked to ensure that the output will still be suitable for the analysis intended, i.e., factors do not become highly correlated with each other.
- *Generate a custom design*. Statistical and special-purpose design software can be used to generate designs of various types that take user-specified constraints into account.

## E. INITIAL DISCUSSION ABOUT EXPERIMENTAL FACTORS AND POTENTIAL APPROACHES

During the kickoff meeting held in December 2021, an initial set of possible anti-surface warfare experiment factors was discussed. The set appears in Figure 1, a scan of an unclassified meeting handout. It was quickly decided to drop the sub-launched munition factors, so those are shown lined out, and meant that we would restrict our attention to the Navy air-launched and surface-launched anti-ship munitions.



ASuW Portfolio Variables for Consideration

| Weapons | Kinematic Range | Profile | P(Hit) | S2A Survivability | Discrimination | Required Salvo | No. Carried | CONOPS |
|---|---|---|---|---|---|---|---|---|
| Air Launched | | | | | | | | |
| - Munition A | X nm – Y nm | L / M / N | .a - .b | .c - .d | .e - .f | g - h | o - p | A / B / C |
| - Munition B | X nm – Y nm | L / M / N | .a - .b | .c - .d | .e - .f | g - h | o - p | A / B / C |
| Surface Launched | | | | | | | | |
| - Munition C | X nm – Y nm | L / M / N | .a - .b | .c - .d | .e - .f | g - h | o - p | B / D / E |
| - Munition D | X nm – Y nm | L / M / N | .a - .b | .c - .d | .e - .f | g - h | o - p | B / D / E |
| Sub Launched | | | | | | | | |
| - Munition E | X nm – Y nm | L / M / N | .a - .b | .c - .d | .e - .f | g - h | o - p | A / C / E |
| - Munition F | X nm – Y nm | L / M / N | .a - .b | .c - .d | .e - .f | g - h | o - p | A / C / E |

Figure 1. Handout from kickoff meeting that described potential anti-surface experiment factors.

The discussion we had with N81 before the project was terminated delved into fleshing out the dependencies between the identified variables. This information is needed to determine which factors would be varied independently, which would be varied in a dependent (related) manner, and which may be dropped from initial consideration because of the complexity involved in manipulating that input.

We briefly discussed pros and cons of varying munitions with a single "type" (categorical) factor versus with a set of independent performance or characteristic factors. We decided that we might treat the air-launched munitions as categorical type but treat the surface-launched munitions with a set of more-broadly-varied factors.

With the air-launched munitions treated as a type factor, we might still vary its performance parameters over narrow ranges, or these might simply vary lock-step with the type. The purpose of varying parameters over narrow ranges would be to capture reasonable variation of these within a given type. With the surface-launched munitions more broadly defined, for example, to represent a generic future missile, we would vary performance parameters over a broader range.

The next step in the project would have been to create a table or graph specifying how changes to a given factor propagate into STORM's individual data files. In other words, we would need to specify all the parameter elements in STORM's various .dat files that need to change to accommodate a change in this factor and also understand any explicit or implicit dependencies involved. Our conjecture at the point of project conclusion was that munition types are more tightly bound to the platform carrying the munition, i.e., the missile is in effect the munition (the lethal part), so it might be more straight-forward to construct a generic missile. At least in the baseline data, there is a one-to-one correspondence with the platform/air munition.

Again, having a complete understanding of dependencies, necessary or desired, is a precursor to selecting a starting design. We next describe our limited understanding of these dependencies at project conclusion, as a starting point for future work. We discuss these by potential factor.

1. **Kinematic Range**. This factor represents the range of the munition and would likely be treated as independent within the limits of current to projected future. We would need to identify if it corresponds to one or more parameter elements or if it is derived within the model given other factors such as payload weight and fuel burn rate.

2. **Flight Profile**. Initial discussion indicated that N81 considers flight profile to be dependent on range. For example, achieving a certain speed threshold comes with trading off some range. Also may just be two levels of this. It appears that the STORM typeairmunit.dat file defines flight profile (linear or ballistic), speed, signature, and navigation system. There also may be parameters in the typeaa.dat file to modify. The profiles in the typeairmunt.dat file only look applicable to terminal flyout, i.e., after leaving the carrying platform on the way to the target.

3. **Phit**. This factor, representing the probability of hit of the munition, would be varied in an independent manner.

4. **Surface to Air (S2A) survivability**. This represents the munition's susceptibility to being intercepted. As yet undetermined, should this be treated as being related to the profile, for example, to represent speed, whether or not it is sea-skimming or can perform evasive maneuvers.

5. **Discrimination**. This represents the ability to correctly identify the highest priority target in the formation it is attacking, meaning did it correctly go after the carrier instead of a lower-priority ship type. This factor would be varied in an independent manner.

6. **Required Salvo.** This factor represents the salvo size requirement determined by the model in order to have high probability of achieving a mission kill of the intended target. It is important to note that the user can not directly manipulate salvo size. Instead, the weapon planner assigns this as a function of the munition's phit and a 'max packages' input. So for example, a missile with low phit but high max packages would likely be assigned a high required salvo. A missile with a high phit and low max packages would likely be assigned a lower required salvo. It seems this is also dependent on the vulnerability of the intended target. Our initial conclusion here was to vary the max packages as the factor and perhaps determine if, in the post-processing, we can grab the value of required salvo assigned by the planner for each run.

7. **Number Carried.** This factor represents the number of the munition carried by the individual platform, i.e., aircraft or ship. Our initial discussion determined that this factor would be varied over small ranges, as makes sense for the particular platform. So for example, a specific aircraft might carry two or four of one munition type but six or eight of another. We had not yet discussed the possibility of including mix factors.

8. **Concept of Operations (CONOPS).** It was discussed that there would only be two levels of this factor to consider at first, tied to the kinematic range of the munitions. NAVAIR conducted a study in which they varied CONOPS over two levels for the carrier, and it was discussed that this study could make use of their files. At the time of project end, N81 had requested the files from NAVAIR. It was not yet clear whether the CONOPS changes were entirely resident within the navalc2.dat file or if they involved changes to other files as well.

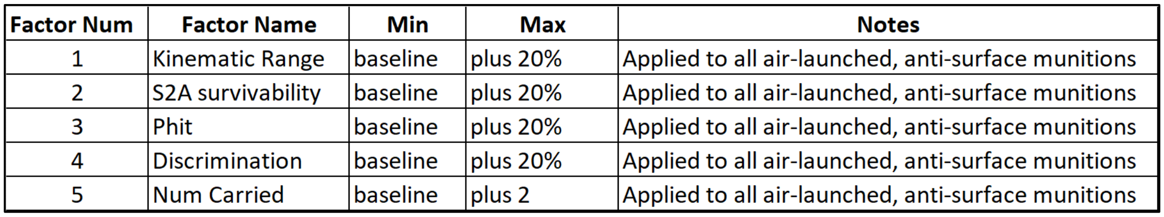
## F. OPTIONS FOR INITIAL SMALLER-SCOPED EXPERIMENT

Since the discussion about dependencies in section E were not yet fully resolved, and to potentially accommodate a smaller scope of work effort due to N81 loss of contractor support and COVID-related restrictions, we discussed two possibilities for conducting a smaller initial experiment. This initial effort would serve to both provide insight as well as to work the process involved and come to a better understanding about time required to develop and run cases.

### 1. Sensitivity analysis of a subset of the original factors

We could perform a sensitivity analysis design over the five factors listed in Table 1. These factors were chosen because they were part of the original set of identified variables and are likely straightforward to manipulate. Because of the sensitivity analysis nature of the factor ranges, it is deemed unlikely that leadership would determine that a major CONOPS change is required.

Table 1. Factors and Settings for Sensitivity Analysis DOE



| Factor Num | Factor Name | Min | Max | Notes |
|---|---|---|---|---|
| 1 | Kinematic Range | baseline | plus 20% | Applied to all air-launched, anti-surface munitions |
| 2 | S2A survivability | baseline | plus 20% | Applied to all air-launched, anti-surface munitions |
| 3 | Phit | baseline | plus 20% | Applied to all air-launched, anti-surface munitions |
| 4 | Discrimination | baseline | plus 20% | Applied to all air-launched, anti-surface munitions |
| 5 | Num Carried | baseline | plus 2 | Applied to all air-launched, anti-surface munitions |

Though one option is to vary each variable plus or minus (say) 10%, we choose here to focus on potential improvement only, hence the choice to vary from baseline as the minimum to plus 20% as the maximum. These factors would operate as multipliers and would be applied independently to all air-launched, anti-surface munitions. For example, say there are two munitions of interest, then DPs with the plus 20% setting would increase munition 1's range by 20% and also increase munition 2's range by 20%. We would not expect munition 1's range to be equal to munition 2's range.

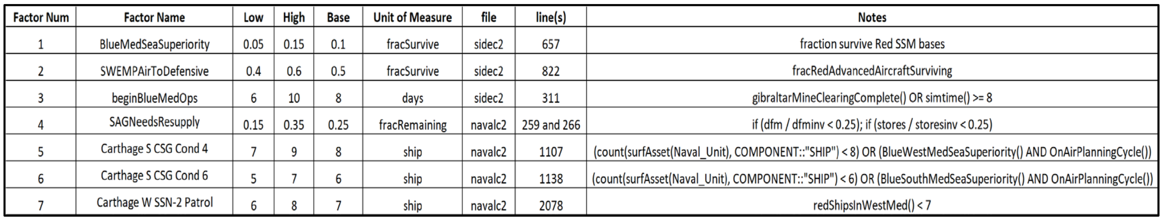
As discussed previously, there are different choices of designs available, but we would choose the two-level Resolution V fractional factorial, requiring 16 design points (DPs) to explore these 5 factors. The Resolution V nature of the design means that it requires more design points than those of lower resolution, but allows for estimation of all main effects and two-way interactions without confounding. Optionally, a single center point could be added to determine if curvature exists, though the source of the curvature could not be resolved without adding design points. Given these factors and ranges, though, it does seem that curvature would be unlikely and therefore, we could drop the center point.

To conserve run time and storage required, we could gather key measures from the dbase.out files and bypass the data warehouse loads.  We believe there are already measures that count the number of out of action red and blue surface combatants, for example, but if not, it should be straightforward to add these measures.  If we do bypass the data warehouse loads, we would not run the STORM Automated Reporting Aid (SARA), since it requires data warehouses.

### 2. Sensitivity analysis of C2 factors

LT Devon Cobbs (2018) explored robustness of results to seven factors, including thresholds in the navalc2.dat file.  Last year, as part of the unclassified work effort, we considered the possibility of performing a similar experiment on the unclassified Punic scenario.  In Table 2, we identify potentially interesting threshold values to vary, as well as their ranges and file location (sidec2 or navalc2).

Table 2. Factors and Settings for a C2 threshold experiment on Punic



| Factor Num | Factor Name | Low | High | Base | Unit of Measure | file | line(s) | Notes |
|---|---|---|---|---|---|---|---|---|
| 1 | BlueMedSeaSuperiority | 0.05 | 0.15 | 0.1 | fracSurvive | sidec2 | 657 | fraction survive Red SSM bases |
| 2 | SWEMPAirToDefensive | 0.4 | 0.6 | 0.5 | fracSurvive | sidec2 | 822 | fracRedAdvancedAircraftSurviving |
| 3 | beginBlueMedOps | 6 | 10 | 8 | days | sidec2 | 311 | gibraltarMineClearingComplete() OR simtime() >= 8 |
| 4 | SAGNeedsResupply | 0.15 | 0.35 | 0.25 | fracRemaining | navalc2 | 259 and 266 | if (dfm / dfminv < 0.25); if (stores / storesinv < 0.25) |
| 5 | Carthage S CSG Cond 4 | 7 | 9 | 8 | ship | navalc2 | 1107 | (count(surfAsset(Naval_Unit), COMPONENT::"SHIP") < 8) OR (BlueWestMedSeaSuperiority() AND OnAirPlanningCycle()) |
| 6 | Carthage S CSG Cond 6 | 5 | 7 | 6 | ship | navalc2 | 1138 | (count(surfAsset(Naval_Unit), COMPONENT::"SHIP") < 6) OR (BlueSouthMedSeaSuperiority() AND OnAirPlanningCycle()) |
| 7 | Carthage W SSN-2 Patrol | 6 | 8 | 7 | ship | navalc2 | 2078 | redShipsInWestMed() < 7 |

We could look at doing something similar on the identified classified scenario. We would require input on interesting thresholds to identify, and if changing these values required other changes outside of where they appear in the navalc2 or sidec2 files.

# IV. EXECUTION AND ANALYSIS

## A. EXECUTING THE RUNS

Prior to executing the runs, we first need to construct the excursions or design points from the agreed upon experimental design. We planned on this being in the form of a spreadsheet, or comma-separated value (CSV) file, where each column is a factor, and each row is a design point, with each cell specifying the value of that factor for that design point. This spreadsheet would have been generated using any of the DOE templates mentioned in Chapter II.

To create the individual excursions, we planned on using a template approach, similar to what STORM's Experiment Manager uses. With the STORM files from the baseline scenario, we would have worked with N81 to identify the particular location in those files where the factor value was defined, then replace that value with a templated string, e.g., "#FACTOR1#". Then, using R or python scripts we planned on developing, run that script to merge the design spreadsheet with the STORM baseline files to create a set of excursions, where each excursion corresponds to the design point row in the spreadsheet.

Depending on N81's preference, we then planned on either: (1) integrating those generated excursions/studies with the current system using git, or (2) proceed using the design artifacts (DOE spreadsheet, STORM templated files, other STORM scenario files not changed by the design) integrated with git, and then constructing a file structure similar to what STORM's Study Tool uses. The first option affords easier integration with the N81's current STORM infrastructure. The second option is potentially more flexible with respect to running multiple excursions/design points concurrently, in either a sequential or adaptive experimental design setting. As we developed a fuller understanding of N81's STORM infrastructure and preferences, we planned on discussing the advantages and disadvantages of each option, and where, if any, there are significant differences.

To execute the runs, we initially planned on manually starting each individual excursion/design point for some number of replications using N81's current STORM infrastructure. Eventually, we planned on writing R or python scripts to 'wrap around'

N81's scripts to start multiple excursions concurrently. This, of course, would depend on the computational resources available prior to starting the experimental study. Also, we planned on having a discussion regarding distribution of runs using an identified scheduler, e.g., HTCondor, OpenPBS, or any one of the other STORM-supported scheduler software. We have had good success with HTCondor (https://research.cs.wisc.edu/htcondor/) to conduct data farming experiments on our SEED Cluster. Distributing runs using a scheduler can provide a smoother workflow, with the added benefit of easily inserting other types of jobs/tasks into the workflow as needed, e.g., data warehouse loads and analysis tasks.

## B. POST-PROCESSING / KEY METRICS

Our initial plan for post-processing the output from the experimental design was to use N81's STORM Automated Reporting Aid (SARA), augmented as needed, by R or python scripts we would have written that would 'wrap around' any of the SARA scripts/code. The latter would have included, at a minimum, the ability to join/merge the data with the factors from the design.

Another idea we planned on discussing that could provide faster turnaround of the run process (at the expense of more detailed analysis up front) is, for each design point/replication, to extract the Measures output from the dbase.out files as the replication completes. Not running a data warehouse load immediately after the run could save significant time in the overall process. After the experimental study concludes, or during any run/computer hiatus, the data warehouse loads could then be subsequently run for the follow-on detailed analyses. Of course, this depends on whether the Measures contain enough information about the run to help with any initial analysis. Again, assuming the Measures contain enough information, this process could be also used in a sequential or adaptive experimental design setting by extracting sufficient metrics and statistics to drive the experimental design without having to perform expensive data warehouse loads as part of that process.

The Appendix contains example python code to extract Measures data from dbase.out files.

## C. ANALYSIS OF EXPERIMENT DATA

The analysis of experiment data starts with identifying the responses (output) of interest most pertinent to the study. These responses may be of different types, for example, continuous, discrete, qualitative, transient or steady state, complete time series or aggregate measures such as frequencies, means, variances, or percentiles, etc. The types of visualizations and analysis techniques most suitable for a given response depend on response type. We expected to include both end-of-run data such as platforms out of action as well as time series measures that captured if and when key campaign progression conditions were met.

STORM has built-in capabilities for certain types of report and graphical generation. However, capabilities are typically limited to viewing certain plots for a small handful of cases side by side, and this was our understanding of the typical use case for SARA-generated plots, as well. In contrast, we planned to use complementary methods for viewing and analyzing experiment data as a whole, to identify the most influential factors and interactions, determine maxima, minima, 'knees in the curve' and broad flat regions where results are robust.

As part of the analysis, we planned to fit metamodels that relate an experiment's inputs to responses. Examples of metamodeling approaches include stepwise regression, logistic and multinomial logistic regression, partition trees, bootstrap forests, and Gaussian process models. Some of these methods are considered parametric because they come with distributional or other assumptions, and others are nonparametric in nature. We typically use a family of methods, with different strengths and limitations, to uncover interesting relationships.

The fitting of metamodels should be complemented by visual representations of the data. Even a set of replications for a single design point can yield a variety of interesting insights. A diverse set of graphs appears in Morgan et al. (2018), who provide multi-color dashboards that display, for each replication, whether or not each of many user-defined success metrics are met. They also consider correlation plots of key metrics; present a variety of heatmaps that show conditions, events, and resource levels across time and replications; use cluster analysis to identify 'good' and 'bad' clusters; and construct trees that show how often different events fire that allow for more in-depth explorations of why the clusters differ.  Part of the work would have been to determine how to adapt some of these methods to data coming from a designed experiment instead of from a single case.

As mentioned previously, we would have considered multiple objectives.  This would have been necessary, given the nature of campaign analysis. For example, one topic discussed in preliminary meetings was how to analyze a self-correcting system. As a motivating example, say that significantly increasing the inventory or capability of a certain missile surprisingly had no appreciable effect on a key response such as red ship drawdown.  Could it be that the system was self-correcting such that other USN air sorties or assets adapted to fill the gap?  Would this have been allowed per the campaign plan?  Did another service, say USAF, adapt to launch sorties to fill the gap?

STORM is a complex model, with thousands of time-space interactions occurring over a long period.  Fully understanding the 'why' behind a particular run resulting in a particularly good or bad outcome requires in-depth scenario and model understanding, and may involve doing a deep-dive into results/playbacks.  The collection of responses should include indicators such as when sorties need to be canceled, resources fall or stay below a critical threshold level, and status of key conditions that would reasonably be correlated with mission success and that would serve to drive campaign progression through its planned phases.  In short, the data collected and analyzed should provide quality leads in the right direction with respect to establishing reason and causality.

# V. SUMMARY / FUTURE WORK

Large-scale simulation models inform many important decisions within the Department of Defense. The time required to create or modify these models often require substantial time and effort from teams of developers and analysts in close consultation with subject matter experts. Recent breakthroughs in large-scale simulation experiments have allowed analysts and decision makers to gain a much broader and deeper understanding of the model behavior while avoiding the so-called ‘curse of dimensionality’ that makes brute force model exploration impossible. In a nutshell, well-designed experiments consist of carefully chosen combinations of model inputs, called design points. The Naval Postgraduate School’s Simulation Experiments & Efficient Designs (SEED) Center for Data Farming is a recognized leader in advancing the theory and application of large-scale simulation experiments. Data farming is a metaphor for growing data from computational experiments.

Further research is needed to address the needs of senior leaders who use models (such as campaign models) where some of the design points are difficult to instantiate. For example, some design points might reflect qualitatively different operational policies or command and control plans, and consequently have a long lead time and high cost. A better understanding of how designed experimentation can complement the traditional baseline and excursion modeling process merits further research.

# APPENDIX

### Example python code for reading dbase.out and extracting Measures data

Below is example python code to extract Measures and their values from a STORM dbase.out file. This could also be easily done in R and integrated into SARA (if functionality does not currently exist). The dbase.out file is a space-separated text file, with a single data record on each line.

```
--python snippet, with comments
# lines starting with '>>>' indicate a prompt,
# followed by a variable name or expression,
# followed by example output of that variable or expression
import gzip
import pandas as pd
import re

# dbase.out.1.gz is an example dbase.out compressed file from a Punic21 run

fp = gzip.open("dbase.out.1.gz",'rt')  # use 't' to read in as text
dd = fp.readlines()  # reads in all lines; will have the '\n' line ending
# if you provide an argument to readlines, it is the number of bytes to read in, not number
of lines! This will help if the file is large and you want to process in chunks

# Preamble of dbase.out files contain mappings and data specifications
# the first 3 characters indicate a record id; that is followed by the data specification
# for that data record
# Measures data are represented by 2 records: MeasureValue and Measure.
# Here is the mapping from the example dbase.out file:
#'+09MeasureValue  PKEY INT(8) Measure_FKEY INT(8) Time REAL MeasureValue
REAL\n',
#'+08Measure  PKEY INT(8) NAME CHAR(50)\n',

# extract all the Measures
f08 = [w for w in dd if w.startswith("-08")]

# in the Punic21 example, there are 15 Measures
>>> f08
['-08 0 "Red Ground Unit Surv"\n', '-08 1 "Red SSM surv"\n', '-08 2 "Red Legacy Short-
Range Mobile SAM surv"\n', '-08 3 "Red Legacy Long-Range SAM surv"\n', '-08 4 "Red
Medium-Range Mobile SAM surv"\n', '-08 5 "Red AC surv"\n', '-08 6 "Red Advanced
Very Long-Range SAMs surv"\n', '-08 7 "Blue Med MCM Complete"\n', '-08 8 "SWEMP
Naval Air To Def East"\n', '-08 9 "SWEMP Naval Air To Defensive"\n', '-08 10 "Blue
West Med Sea Superiority"\n', '-08 11 "Blue East Med Sea Superiority"\n', '-08 12
"SWEMP Carrier Dead"\n', '-08 13 "Blue Med Sea Superiority"\n', '-08 14 "Blue Med
Sea Withdrawl"\n']


f09 = [w for w in dd if w.startswith("-09")]
# Here is the first 10 MeasureValue entries
>>>f09[1:10]
['-09 1 4 0 1.00000000\n', '-09 2 3 0 1.00000000\n', '-09 3 2 0 1.00000000\n', '-09 4 1 0
1.00000000\n', '-09 5 0 0 1.00000000\n', '-09 6 5 0.5 1.00000000\n', '-09 7 4 0.5
1.00000000\n', '-09 8 3 0.5 1.00000000\n', '-09 9 2 0.5 1.00000000\n']

# need to sort out measures from f08, i.e., parse the dbase.out line
pp = r"-08 (\d+) (.+)"
r = re.search(pp,f08[1])
>>> r.groups()
('1', '"Red SSM surv"')
>>> r.group(0)
'-08 1 "Red SSM surv"'
>>> r.group(1)
'1'
>>> r.group(2)
'"Red SSM surv"'

f08p = [re.search(pp,f) for f in f08]
ff08 = [[r.group(1), r.group(2)] for r in f08p]

ff09 = [f.replace('\n','').split(" ") for f in f09]

## construct panda data frames for the actual MeasureValues (f09) and the Measure type
(f08)
df1 = pd.DataFrame(ff09,columns=['tag','mvid','mkey','time','value'])
df2 = pd.DataFrame(ff08,columns=['mkey','measure'])

# we then merge the 2 data frames; we can then save/post-process/analyze this data, e.g.,
# compute measure statistics, construct heatmaps, etc.
mm = pd.merge(df2,df1)
>>> mm
     mkey                  measure  tag  mvid                time       value
0       0   "Red Ground Unit Surv"  -09     5                   0  1.00000000
1       0   "Red Ground Unit Surv"  -09    11                 0.5  0.99878531
2       0   "Red Ground Unit Surv"  -09    25  0.58333333333333337  0.99878531
3       0   "Red Ground Unit Surv"  -09    31  0.66666666666666663  0.99878531
4       0   "Red Ground Unit Surv"  -09    37                0.75  0.99658430
...   ...                      ...  ...   ...                 ...         ...
1731   14  "Blue Med Sea Withdrawl"  -09  1559                  18  0.00000000
1732   14  "Blue Med Sea Withdrawl"  -09  1603                18.5  0.00000000
```

```
1733   14  "Blue Med Sea Withdrawl"  -09  1647                19  0.00000000
1734   14  "Blue Med Sea Withdrawl"  -09  1691              19.5  0.00000000
1735   14  "Blue Med Sea Withdrawl"  -09  1735                20  0.00000000

[1736 rows x 6 columns]
```
--- end python snippet

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF REFERENCES

Bickel, W. G. (2014). Improving the analysis capabilities of the Synthetic Theater Operations Research Model (STORM) [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/43878

Cioppa, T. M. & Lucas, T. W. (2007). Efficient nearly orthogonal and space-filling Latin hypercubes. Technometrics, 49(1), 45–55.

Cobbs, D. G. (2016). The influence of command and control thresholds on campaign outcomes utilizing the Synthetic Theater Operations Research Model [Master's thesis, Naval Postgraduate School]. (RESTRICTED)

Duan, W., Ankenman, B. E., Sanchez, S. M., & Sanchez, P. J. (2017). Sliced full factorial-based Latin hypercube designs as a framework for a batch sequential design algorithm. Technometrics, 59(1), 11–22.

Erickson, C. B., Ankenman, B. E., Plumlee, M., & Sanchez, S. M. (2021). Gradient based criteria for sequential experiment design. Quality and Reliability Engineering International, 37(7), 3084-3107. https://doi.org/10.1002/qre.2981

Group W. (2021). STORM: User's manual version 2.11. Fairfax, Virginia.

Hernandez, A. S., Lucas, T. W., & Carlyle, M. (2012). Enabling nearly orthogonal Latin hypercube construction for any non-saturated run-variable combination. ACM Transactions on Modeling and Computer Simulation, 22(4), 20:1-20:17.

Kleijnen J. P. (2015). Design and analysis of simulation experiments, 2nd edn. Springer, New York.

MacCalman A., Vieira H., & Lucas T. W. (2017) Second order nearly orthogonal latin hypercubes for exploring complex stochastic simulations. Journal of Simulation, 11(2), 2017, 137–150.

Marlow D. O., Sanchez S. M., & Sanchez P. J. (2015). Testing aircraft fleet management policies using designed simulation experiments. In: Proceedings of the 21st international congress on modelling and simulation, pp 917–923. Modelling and Simulation Society of Australia and New Zealand Inc (MSSANZ).

Morgan, B. L., Schramm, H. C., Smith, J. R., Lucas, T. W., McDonald, M. L., Sanchez, P. J., Sanchez, S. M., & Upton, S. C. (2018). Improving U.S. Navy campaign analysis using big data. Interfaces, 48(2), 130-146.

Renquist, J. J. (2018). An independent assessment of the energy enhancements to the Synthetic Theater Operations Research Model (STORM) [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/60453

Sanchez, P. J. & Sanchez, S. M. (2019). Orthogonal second-order space-filling designs with insights from simulation experiments to support test planning. Quality and Reliability Engineering International, 37(7), 3084-3107.

Sanchez S. M. & Sanchez P. J. (2005). Very large fractional factorial and central composite designs. ACM Trans Model Comput Simul 15(4):362–377.

Sanchez, S. M. & Sanchez, P. J. (2017). Better big data via data farming experiments. Chapter 9 in Advances in Modeling and Simulation—Seminal Research from 50 Years of Winter Simulation Conferences, eds. A. Tolk, J. Fowler, G. Shao, and E. Yücesan, 159–179. Cham, Switzerland: Springer International Publishing.

Sanchez, S. M., Sanchez P. J., & Wan H. (2021). Work smarter, not harder: a tutorial on designing and conducting simulation experiments. Proceedings of the 2021 Winter Simulation Conference. Institute of Electrical and Electronic Engineers, Piscataway, New Jersey.

SEED Center for Data Farming. (2022). https://harvest.nps.edu. Accessed 20 Feb 2022.

Seymour, C. N. (2014). Capturing the full potential of the Synthetic Operations Research Model (STORM) [Master's thesis, Naval Postgraduate School]. NPS Archive: Calhoun. https://calhoun.nps.edu/handle/10945/44000

Vieira H., Sanchez S., Kienitz K., & Belderrain M. (2013). Efficient, nearly orthogonal-and-balanced, mixed designs: an effective way to conduct trade-off analyses via simulation, *Journal of Simulation*, 7:4, 264–275.

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

3. OPNAV N81
   Office of the Chief of Naval Operations
   Washington, DC

4. Research Sponsored Programs Office, Code 41
   Naval Postgraduate School
   Monterey, California