NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

THE CONCEPTUAL DESIGN RELIABILITY PREDICTION METHOD: ESTABLISHING FUNCTIONAL-PHYSICAL RELIABILITY RELATIONSHIPS FOR SYSTEM RELIABILITY PREDICTIONS DURING CONCEPTUAL DESIGN

by

Elizabeth T. Rajchel

September 2020

Thesis Advisor: Bryan M. O'Halloran
Co-Advisor: Jefferson Huang

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2020	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE
THE CONCEPTUAL DESIGN RELIABILITY PREDICTION METHOD: ESTABLISHING FUNCTIONAL-PHYSICAL RELIABILITY RELATIONSHIPS FOR SYSTEM RELIABILITY PREDICTIONS DURING CONCEPTUAL DESIGN

5. FUNDING NUMBERS

6. AUTHOR(S) Elizabeth T. Rajchel

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)

This paper presents a system reliability prediction method suitable for use during conceptual design, called the conceptual design reliability prediction method (CDRPM). The CDRPM extends the early design reliability prediction method (EDRPM) by facilitating parameter characterizations that follow non-normal distributions. Functional-physical reliability relationships are established through a hierarchical Bayesian model solved by Markov Chain Monte Carlo (MCMC) sampling and aggregated using the system's reliability block diagram (RBD) to assess the likelihood of candidate architectures meeting a reliability requirement. Reliability predictions based on different types of failure data, specifically success ratios and failure rates, are compared herein in a case study of a generic launcher system assessed by the CDRPM. This research shows the effects of failure data type selection and distribution assumptions on architecture down-selection, leading to enhanced insight during conceptual design analysis.

14. SUBJECT TERMS	**15. NUMBER OF PAGES**
early design reliability prediction method, EDRPM, conceptual design reliability prediction method, CDRPM, reliability prediction, hierarchical Bayesian, Markov Chain Monte Carlo, MCMC, Monte Carlo Simulation, MCS, reliability block diagram, RBD, failure rate, success ratio	199
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

THE CONCEPTUAL DESIGN RELIABILITY PREDICTION METHOD: ESTABLISHING FUNCTIONAL-PHYSICAL RELIABILITY RELATIONSHIPS FOR SYSTEM RELIABILITY PREDICTIONS DURING CONCEPTUAL DESIGN

Elizabeth T. Rajchel
Lieutenant Commander, United States Navy

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2020

Approved by: Bryan M. O'Halloran
Advisor

Jefferson Huang
Co-Advisor

Ronald E. Giachetti
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This paper presents a system reliability prediction method suitable for use during conceptual design, called the conceptual design reliability prediction method (CDRPM). The CDRPM extends the early design reliability prediction method (EDRPM) by facilitating parameter characterizations that follow non-normal distributions. Functional-physical reliability relationships are established through a hierarchical Bayesian model solved by Markov Chain Monte Carlo (MCMC) sampling and aggregated using the system's reliability block diagram (RBD) to assess the likelihood of candidate architectures meeting a reliability requirement. Reliability predictions based on different types of failure data, specifically success ratios and failure rates, are compared herein in a case study of a generic launcher system assessed by the CDRPM. This research shows the effects of failure data type selection and distribution assumptions on architecture down-selection, leading to enhanced insight during conceptual design analysis.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

AoA	analysis of alternatives
cdf	cumulative distribution function
CDRPM	Conceptual Design Reliability Prediction Method
CI	confidence interval
DOD	Department of Defense
EDRPM	Early Design Reliability Prediction Method
EDO	Engineering Duty Officer
EPRD	Electronic Parts Reliability Data
FMEA	Failure Mode and Effects Analysis
FMECA	Failure Mode, Effects, and Criticality Analysis
FTA	Fault Tree Analysis
HB	hierarchical Bayesian
HDD	hard disk drive
iid	independent and identically distributed
MCMC	Markov Chain Monte Carlo
MCS	Monte Carlo Simulation
MS	Microsoft
MTBF	mean time between failure
MTTF	mean time to failure

NPRD	Non-electronic Parts Reliability Data
NPS	Naval Postgraduate School
PRA	Probabilistic Risk Assessment
pdf	probability density function
RBD	reliability block diagram
SE	systems engineering
SSD	solid state drive
SNR	signal-to-noise ratio
TLCSM	Total Life Cycle Systems Management
USN	U.S. Navy

Executive Summary

The purpose of this research was to develop a reliability prediction methodology suitable for application during conceptual design to aid in identifying and eliminating unreliable system configurations from further consideration as the system concept matures. The Conceptual Design Reliability Prediction Method (CDRPM), developed herein, provides a means to establish functional-physical reliability relationships which are combined according to a system's functional architecture to make a system-level reliability prediction before a physical architecture is selected. The cumulative distribution function (cdf) of the system-level prediction distribution is compared to a reliability requirement to identify functions and component types that are unlikely to meet said requirement. By focusing on elimination of unreliable solutions, rather than selection of the most reliable ones, the CDRPM preserves the decision space for trade-offs in other performance metrics evaluated during an analysis of alternatives (AoA).

Incorporating system reliability as early as possible in the design life cycle mitigates requisite design changes that are more costly the later they are made [1], [2], [3]. Many reliability prediction methods currently exist but are ill-suited for application during conceptual design because they fail to meet four tenets identified for a method applied in this phase of interest. Most reliability prediction methods (1) rely on a physical architecture, which is not yet established in the design life cycle, and/or (2) fail to leverage the information that is known, the functional architecture. Furthermore, failure to (3) capture data uncertainty and failure to (4) explicitly facilitate a requirement comparison both lead to misinterpretation of results. One methodology reviewed, the Early Design Reliability Prediction Method (EDRPM) [3], met all four tenets but relies upon an assumption that the supporting failure data are normally-distributed to facilitate assessment.

The CDRPM extends the EDRPM by facilitating data characterizations that follow non-normal distributions. Functional-physical relationships are established in both methods through a hierarchical Bayesian (HB) model. The CDRPM uses Markov Chain Monte Carlo (MCMC) sampling to evaluate the joint posterior distribution of the HB model for each function, enabling the data distribution independence not afforded by the closed-form solution of the EDRPM. Monte Carlo Simulations (MCSs) from each function's posterior

distribution are combined according to the system's reliability block diagram (RBD) to generate a system-level reliability prediction distribution. The cumulative probability of the fitted system distribution is evaluated at the requirement to determine the likelihood that the system will meet or exceed its reliability requirement.

The CDRPM was illustrated through its development using success ratio data for an abstract system with means that follow beta distributions, demonstrating methodology proof of concept. The CDRPM was subsequently applied to the same abstract system used in [4], which required failure rate data that was normally distributed. The accuracy loss presumed from a simulated approach, the CDRPM, opposed to a deterministic one, the EDRPM, was evaluated and assessed as insignificant, specifically less than 2%, under most credible scenarios. A case study of a generic launcher system provided an opportunity to assess a purposeful system with real-world data, as well as investigate the consequences of data selection on subsequent predictions. Use of success ratio data and failure rate data for the exact same parts, from the same publications, yielded different reliability outcomes. For the assumptions in the case study, success ratio data indicated the likelihood of meeting the system reliability requirement could be improved by eliminating some component types from consideration, while the failure rate data indicated all solutions were equally likely to be successful. More significant than the exact conclusion, the case study demonstrated the CDRPM is flexible and versatile enough to handle a variety of data types to inform a system reliability prediction based on its functions.

References

[1] B. S. Blanchard and W. J. Fabrycky, *Systems Engineering and Analysis*, 5th ed. Edinburgh Gate, Essex, England: Pearson, 2014.

[2] G. Molcho, A. Cristal, and M. Shpitalni, "Part cost estimation at early design phase," *CIRP Annals - Manufacturing Technology*, vol. 63, pp. 153–156, Jan 2014. Available: https://doi.org/10.1016/j.cirp.2014.03.107

[3] M. E. Saravi, L. B. Newnes, A. R. Mileham, and Y. M. Goh, "Estimating cost at the conceptual design stage to optimize design in terms of performance and cost," in Collaborative Product and Service Life Cycle Management for a Sustainable World. London, England: Springer, 2008, pp. 123–130. Available: https://doi.org/10.1007/978-1-84800-972-1_11

[4] B. M. O'Halloran, C. Hoyle, I. Y. Tumer, and R. B. Stone, "The early design reliability prediction method," *Research in Engineering Design*, vol. 30, pp. 489–508, 2019. Available: https://doi.org/10.1007/s00163-019-00314-8

Acknowledgments

To Sam Buttrey, who delivered "Christmas in July" with his model writing script assistance, thanks for lending so much of your time to a student you'd never met and, frankly, have no affiliation with. I hold you in the highest regard for your selflessness and eagerness in your profession.

To my advisors and the many professionals who helped edit this paper into a cogent and coherent thought, thanks for your patience in the process. That you regularly pour this much effort into dozens of students is astounding to me.

To my husband, thanks for sticking it out with me! Two parents working on their master's degrees full time with an infant and toddler at home could have been a recipe for disaster, even before "shelter in place," but we're still standing. I'm proud of us! And I'm extremely grateful for all the sacrifices you made along the way to help make it happen.

Most importantly, to my sons, Teddy and Jackie, researching and writing the bulk of this work while at home with you during a pandemic has made it abundantly clear that you two are my heart and soul. Anything that takes time away from you should only be of the utmost importance and deserving of my best effort. I am otherwise doing us all a disservice for squandering precious moments that could be spent together. If you ever read this, I hope you consider this a worthy pursuit. I love you both the biggest "much" in the world.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

The cost and effort required to change a system's design increase significantly the later the changes are made in the system's life cycle. The earliest opportunity to influence a system's performance is during conceptual design. Reliability is the specific performance attribute addressed by this thesis. Reaching a reasonable reliability prediction as soon as possible mitigates the likelihood of discovering inadequate performance that necessitates costly design changes later. However, existing methodologies for reliability prediction are ill-suited for the conceptual design phase. This chapter establishes the context for the significance of a reliability prediction during conceptual design and includes a broad assessment of the capability gap in reliability methodologies when applied to this specific phase. How the engineering community can benefit from addressing the gap with the methodology derived herein is presented, as well as the author's personal motivation. The chapter concludes with the research design, concisely presenting the purpose of this work, five objectives to meet in order to fulfill that purpose, which serve as the framework for subsequent chapters, and the research methodology used to achieve those objectives.

1.1 Context

Once a problem is identified, the engineering design process for a new system or product to rectify that problem commences with conceptual design [1]. From a systems engineering (SE) perspective, high-level activities during this phase should address life cycle considerations such as producibility and supportability, not just the performance necessary to satisfy the immediate stakeholder need, even if the parameters characterizing those considerations are loosely defined [1], [2]. The physical design, also referred to as physical architecture, is not defined during the conceptual design phase; conceptual design focuses on the functions the system must perform to meet its life cycle requirements (performance and otherwise). It is widely accepted that 70% to 80% of life cycle costs are determined in the early design phases [1], [3]–[5], especially during conceptual design, often depicted in a graphic such as Figure 1.1.

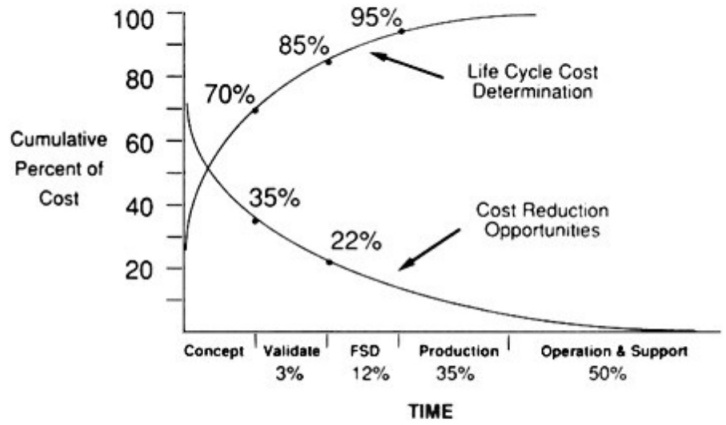


Figure 1.1. Committed Life Cycle Costs. This specific example was selected due to the "Cost Reduction Opportunities" overlaid on the same graph. Source: [1].

Therefore, it is paramount to incorporate system reliability, one form of a supportability consideration, in the design as early as possible, consistent with the Department of Defense (DOD) initiative for Total Life Cycle Systems Management (TLCSM) [6]. If reliability is unaccounted for in conceptual design, there is a high likelihood of costly design modifications to subsequently achieve requisite reliability performance.

1.2 Methodology Gap

The classical approach to determine system reliability is to utilize a reliability block diagram (RBD) that traces the logical dependence of sub-level operations (whether represented as components or as functions) necessary to achieve a higher-level (system) operation. Often, the likelihood of successful sub-level operation is given by the failure rates of constituent components. The logical dependence of operations is informed by one or more failure and risk assessments such as Failure Mode and Effects Analysis (FMEA), Failure Mode, Effects, and Criticality Analysis (FMECA), Fault Tree Analysis (FTA), and Probabilistic Risk Assessment (PRA). Further information on these types of assessments can be found in [2], [7]–[9].

There are a few challenges with using these methods to generate reliability predictions during the conceptual design phase. The primary challenge with applying an RBD approach is that

the analysis commonly requires a physical design to support the evaluation; the specific components in the prescribed architecture dictate which failure rate data to use. FMEA, FMECA, FTA, and PRA also rely on a physical architecture to facilitate application.

Assessing a system in terms of its functions, rather than its components, is a second challenge with reliability prediction during conceptual design; this challenge is related to, but distinctly different from, the first challenge. A number of methods have been developed in recognition of the shortcomings of the aforementioned "design phase agnostic" methods for early design application. Unfortunately, "early design" does not mean the same thing to all authors, likely because "early" is a non-specific relational term that could be interpreted as anything prior to "final"; a wide range of system definition fidelity and maturity may then be encompassed as *early design*. However, *conceptual design*, which occurs relatively *early* in a design life cycle, is explicitly affiliated with a certain degree of system definition fidelity and maturity. In the conceptual design phase, as previously defined in this report, only the system functions are defined, so the early design reliability approaches that could be applied to this phase must frame the system in terms of functions and not components. Assuming the new system will look similar to a related existing one, even if the exact architecture is unknown, is insufficient, especially for emerging technology, to include novel materials and manufacturing techniques. In conceptual design, the system functions are known well before a design alternative is selected, so it is most advantageous to capture the component-failure to function-failure relationship.

A third challenge is that component failure rate data is typically gathered during tests with conditions that may or may not reflect the operational conditions of the new system. The difference in environments could affect the type and frequency of observed failures, so the uncertainty in the failure estimate data must be captured to meaningfully inform the prediction. The *Challenger* disaster in 1986 is a notorious example of how operational conditions, such as temperature, can drastically affect component functionality [10]. An assessment activity must capture variability and uncertainty when applying results from one testing activity in the context of another employment.

A fourth challenge addressed in this research is how to assess a confidence in the reliability prediction, especially as compared to a reliability requirement for the system. If a reliability requirement is presented as a percentage, which is typically the case though occasionally it

is in terms of failures per unit time, and the prediction methodology assesses failure by some other measure, there may be an inconsistent application of findings, ultimately indicating an unclear methodology intent. Intent and rationale are two aspects that distinguish a method from a methodology. While the other works may strive to present a method, specifically, the comparison step enables the prediction approach to directly serve a dual purpose of design selection. Without facilitating a reliability comparison to a requirement, adding a performance characterization that does not meaningfully inform selection of a design solution is not a value-added activity in the conceptual design phase.

The aforementioned challenges have been converted into four tenets for a reliability prediction method suitable for conceptual design. The first is restrictive, while the others are prescriptive.

1. Avoid reliance on physical architecture.
2. Evaluate the system in the context of its functions.
3. Capture data uncertainty.
4. Facilitate reliability comparison.

The Early Design Reliability Prediction Method (EDRPM), developed over time with the most recent publication in 2019 [11]–[13], addresses the four challenges identified above. It (1) makes no assumptions about the specific physical architecture of the system, (2) identifies component-to-function relationships leveraging the functional architecture for the prediction, (3) quantifies data uncertainty, and (4) uses a preset reliability requirement as a baseline comparison for identification of unreliable designs. However, the method assumes that component and function failure rate data are normally distributed in order to provide a closed-form solution to its reliability prediction model. It then leverages the resultant normal distribution to facilitate the prediction comparison to a requirement. These assumptions were reasonable to facilitate the initial development of the EDRPM [13] as a methodology, but reconsideration is necessary for further evolution to expand its applicability. While the normal distribution is not typically affiliated with modeling reliability since it is unbounded and a negative reliability is nonsensical, it is possible there are situations when this consequence is insignificant, and a normal distribution sufficiently represents the data. However, other distributions that are bounded or partially bounded (greater than zero) are more likely to appropriately represent reliability-affiliated data such as success

ratios or times to failure, which are naturally bounded or partially bounded. Shifting from the normality assumption is a significant task because it necessitates changing how the reliability prediction is mathematically determined, as well as how the prediction is compared to a system reliability requirement for making quantitatively informed design decisions. The EDRPM [13] and its future work recommendations were treated as the primary inspiration for this methodology.

1.3 Motivation

The motivation for addressing this methodology gap is founded in the trends of new system development. Systems are becoming much more complex; they have more software, more parts, unique materials, and specialized technology applications. Additionally, the fiscal and political environments in which they are developed necessitate that they are completed on shorter timelines with smaller budgets while adhering to more restrictive information security, disposal, and material-use requirements (among other requirements). Therefore, the pressure to avoid the consequences of late-in-design changes shown in Figure 1.1 is as present as ever for all disciplines in the engineering community.

Technology advancements are not limited to the physical systems. Developments in computing technology have enabled more mathematically intensive and computationally sophisticated methods to be available to a broader population of engineers.

As an Engineering Duty Officer (EDO), reviewing and refining ways to improve reliability predictions for enhanced decision making during weapon system development is of personal interest and professional significance to the author. Beyond the responsibility to follow published directives such as TLCSM, it is her opinion that EDOs should always strive to be conscientious stewards of the resources entrusted to them for the advancement of national defense capabilities. Minimizing time spent considering unreliable design solutions, thus maximizing the opportunity to optimize other performance characteristics, is part of that stewardship.

1.4 Research Design

The fundamental intent of this research was to rectify the methodology gap identified in Section 1.2 for the reasons given in Section 1.3. The way in which that intent was executed is

succinctly captured as this research's purpose, achieved through five contributing objectives utilizing a mixed research methodology, all of which are presented in the subsections below. Assumptions and limitations made during the execution of the research design follow.

1.4.1 Research Purpose

The purpose of this research was to develop a reliability prediction methodology, without assuming a particular data distribution, suitable for application during conceptual design to aid in identifying and eliminating unreliable system configurations from further consideration as potential design solutions.

1.4.2 Research Objectives and Methods

The purpose of this research was fulfilled through meeting the five objectives presented in this section. Qualitative and quantitative research methods were used to address these objectives, following a mixed empirical and experimental approach. Each objective is presented with its affiliated research method, as well as the chapter in which it is addressed.

Objective 1

The first objective was to review existing reliability prediction methods specifically intended for application in early design. Relative strengths and shortcomings were qualitatively assessed for consideration during the development of the methodology presented in this work. As primarily a literature review, the approach for this objective was not strictly classified as either empirical or experimental. Chapter 2 addresses this objective.

Objective 2

The second objective was to develop a methodology for reliability prediction that adheres to the tenets derived in Section 1.2, incorporates strengths of existing methods where applicable, and is not data-distribution dependent. An experimental approach was used to meet this objective through the qualitative aggregation of concepts to develop suitable methodology. Chapter 3 addresses this objective.

Objective 3

The third objective was to apply the new methodology to a non-specific, abstract example for illustration of execution. The representative reliability prediction was quantitatively determined as a continuation of the experimental approach for Objective 2. Chapter 3 addresses this objective.

Objective 4

The fourth objective was to compare the reliability prediction of this method to that determined by the EDRPM as presented in [13] to serve as a verification for methodology application. Calculated percent errors in parameter estimates supported a quantitative comparison and results were qualitatively interpreted as part of an empirical approach. Chapter 4 addresses this objective.

Objective 5

The fifth objective was to apply the new methodology as a case study for a generic launcher system. That is, the case study does not represent an existing real-world system but is a generalized representation of a launcher that may be used for rockets, aerial vehicles, or missiles. The reliability prediction was assessed using quantitative methods delineated in the methodology derived in Chapter 3 as part of an empirical approach to addressing the research's purpose. Furthermore, the case study is assessed twice using different types of failure data to qualitatively illustrate how data selection influences system reliability predictions. Chapter 5 addresses this objective.

1.4.3 Assumptions and Limitations

Many reliability prediction methods intended for early design could feasibly be considered "early design reliability prediction methods" or "EDRPMs" but, for this research, all subsequent references to the EDRPM explicitly and exclusively refer to the collective body of work developed by O'Halloran et al. in [11]–[13], unless otherwise cited.

For the purpose of this research, the reliability of a system is the probability it will perform a function without failure. The following terms may be used interchangeably throughout this work:

- Reliability *goal* and *requirement* both convey a higher-level objective the system is expected to achieve with equal desirability; a *goal* should not be interpreted as greater or "better" than a *requirement*.
- The terms *component*, *candidate solution*, and *product* all represent a unique item that can satisfy a function.[1]
- *Design solution*, *design configuration*, *design alternative* all refer to a selection of components to satisfy all the functions of a system. Feasibility and physical configuration should not be inferred.[2]

This research assumes that component performance data can be captured by parametric distributions. Some of the mathematical concepts applied in this methodology, such as hierarchical Bayesian (HB) models discussed in Chapter 2, may be appropriate for non-parametric data, but not as presented in this work.

This methodology is intended for use during conceptual design when only functions are known. If greater detailed information is available, this approach may not be the best suited for as accurate a prediction as possible. Related, this methodology is intended for use without assuming a particular data distribution, namely without assuming a normal distribution. If normality is assumed, use of this methodology may result in a loss of prediction accuracy. This particular consequence of application is explored in Chapter 4.

Any similarities between the system used for the case study in Chapter 5 and a real-world operational system, particularly one utilized by the DOD, is completely unintentional. All conclusions are for academic illustrative purposes, only.

1.5 Thesis Overview

The engineering community needs a reliability prediction method suitable for use during conceptual design in order to identify design solutions with unacceptable reliability performance as early as possible, saving valuable resources. Based on the literature review of Chapter 2, the author determined only one methodology adheres to the four tenets for reliability prediction during the phase of interest: the EDRPM. Modifying the mathematical

[1] Interchangeability of *component* and *candidate solution* is consistent with the definitions provided in the EDRPM.

[2] Consistent with the definitions provided in the EDRPM.

execution of the EDRPM alleviates some of the method's limitations and simultaneously provides an opportunity to incorporate some of the strengths of other methods reviewed. A suitable, versatile methodology to address the prediction capability gap by modifying the EDRPM to permit application to non-normal data, is developed in Chapter 3. The generalized form of the EDRPM derived in this work is henceforth referred to as the Conceptual Design Reliability Prediction Method (CDRPM). The new name was not intended to suggest original work but was advantageous for two reasons. Firstly, there are many portions of this paper that compare the differences in application and execution between the two methods, so a unique identifier to refer to each version aided clarity and brevity. Secondly, while considering a new moniker, it seemed appropriate to explicitly express "conceptual design" since the broad interpretation of "early" was a critique of other methods. Chapter 4 illustrates differences in reliability conclusions drawn using the CDRPM, a simulation-based method, opposed to using the EDRPM, a deterministic method, by assessing the same example in [13] by both methods. This second demonstration of the CDRPM additionally confirms the versatility and data-distribution independence of the CDRPM. Application to a case study in Chapter 5 validates the method's utility for assessing reliability during the conceptual design of a launcher and explores the implications of failure data selection on prediction outcome. Chapter 6 addresses conclusions and limitations of the CDRPM. The most notable conclusion is that the methodology works as intended, mirroring the EDRPM but without underlying data distribution assumptions, while the greatest limitation is the volume of data required to make a system reliability prediction distribution.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Background and Related Work

This chapter reviews existing methods for early design reliability prediction and assesses the extent to which they adhere to the early reliability prediction tenets presented in Chapter 1. A more in-depth review of the EDRPM follows the survey of other methods as it is the only method that addresses all four tenets and its future work recommendations served as the primary inspiration for this work. Hierarchical Bayesian models provide a framework for mathematically relating data and are commonly used in many types of predictions, including the reliability prediction of the EDRPM. Markov Chain Monte Carlo (MCMC) is a method or tool to determine the relationships in the hierarchical model by simulation. An HB model solved by MCMC sampling is utilized by the CDRPM. The use of MCMC was necessary in this work to alleviate assumptions about failure data distribution that otherwise enable a closed-form solution, like the normality assumption affords the EDRPM. The mathematical theory of HB models and MCMC sampling are briefly introduced at the conclusion of this chapter as related work, while their specific implementation in this work is addressed in Chapter 3.

2.1 Existing Methods for Reliability Prediction

As alluded to in Chapter 1, some of the most commonly used methods to assess reliability are not explicitly prohibited from use in early design, but their application is infeasible due to the lack of system data available in this nascent design phase. Several methodologies purportedly developed to specifically handle reliability prediction in early design already exist in the engineering community. Most of these address some of the tenets presented in Section 1.2, but none addresses all.

The tenets are summarized here:

1. Avoid reliance on physical architecture.
2. Evaluate the system in the context of its functions.
3. Capture data uncertainty.
4. Facilitate reliability comparison.

The selection of methodologies presented in this review as related research were systematically evaluated against the identified tenets; if a method failed to uphold the first tenet, it was not evaluated for meeting the second. The related works are organized by the primary tenet they did not meet.

Cheng and Du [14] present an approach to reduce the uncertainty of system reliability through optimization models but fail to meet tenet (1). The approach challenges the assumption of independent component failures in a system. This assumption is made when there is a lack of information to suggest otherwise, which is often the case in early design. Using a physics-based approach, implicitly for mechanical systems based on failure modes related to strength, Cheng and Du postulate dependent component failures to identify narrower bounds of predicted system reliability. While the authors consider their work to be applicable to "early design" [14], the design phase of concern for this effort does not have a physical architecture yet; component selection is unknown in conceptual design. This work is likely more applicable during preliminary design as described in [2].

Bergman and Ringi [15] utilize Bayesian estimation to apply knowledge from similar systems in non-identical operating environments to influence reliability predictions based on independent, exponentially distributed component failure rates. Tenet (1) is arguably met but tenet (2) is not. While different operating environments provide different stressors, affecting the absolute failure rates of system components, Bergman and Ringi contend that components maintain the same relative failure rate and use this relative relationship to update the reliability prediction for the system operating in the conditions of concern [15]. While the concept of using analogous systems does not explicitly require a physical architecture, there is an inherent assumption about the design solution in order to identify which systems to use for comparison. Notwithstanding, the authors do not assess component-to-function relationships so are not aligned with SE principles for this design stage.

Johnson, Moosman, and Cotter [16] also use analogous systems to inform performance predictions through an HB model, ultimately failing to meet tenet (2) like the work in [15]. Hierarchical models allow for pooling of data across a variety of sources and formally account for the similarity or dissimilarity of the data. An HB model is one of the most commonly used models to do so, discussed in greater detail in Section 2.3. The model in [16] accounts for manufacturer experience in a "quality index," applying

greater weight to "experienced" developers than to "inexperienced" developers, as defined by their own classification scheme. The HB approach in [16] is slightly preferable to the non-hierarchical Bayesian estimation of [15] because the hierarchical nature avoids over-fitting the data, protecting for slightly more flexibility in what the design solution is. However, it is not reasonable to extend reliability performance from any manufacturer, experienced or inexperienced, to performance predictions for new technology. For example, the Maneuvering Characteristics Augmentation System (MCAS) on the Boeing 737 Max built by Boeing Commercial Airlines [17] and missile tube welds by Electric Boat on the USS *Columbia* (SSBN 826) [18] are two contemporary examples of seasoned manufacturers encountering reliability challenges with new technology, or even just new manufacturing techniques. Had this methodology been used, Boeing and Electric Boat would have been classified as experienced manufacturers with higher quality indices, resulting in greater reliability predictions, but the performance of their new technology and technique would not have been commensurate with a high prediction.

Similarly, Baun [19] used previous "ancestor" designs as reliability benchmarks, failing tenet (2) as the other comparative methods did. He proposes an evaluation of the changes from the ancestor designs to the new proposed one, including a quantification of the impact of those changes on the system reliability, but this evaluation is intended to be executed at the component level, not the function level, facing the same challenges for implementation as [16]. While quantifying reliability implications due to differences in operating environments, manufacturer experience, and design modifications can all enlighten a reliability prediction, without explicit component-to-function relationships, it becomes unclear which aspects of a related system are of interest to a truly new complex system when only functions are known.

An alternative framework presented by Mayda and Choi [20] utilizes random sampling, specifically Monte Carlo Simulation (MCS), from component-failure to function-failure relationships to frame system-level reliability requirements as Gaussian probability distributions. [20] meet tenets (1) and (2), but the implementation of tenet (3) creates a different related challenge and tenet (4) is plainly omitted. The authors capture system performance uncertainty through the use of distributions rather than point estimates, which meets the intent of accounting for data uncertainty, tenet (3). However, doing so in the performance requirement does not aid in facilitating a quantitatively informed decision

since there is not a discrete objective, introducing a new design challenge: consideration of a variable performance threshold. Additionally, variable requirements are contractually challenging for bidding and delivery.

In "A new Bayesian Network Approach to Reliability modelling" by Marquez et al. [21], the authors consider Weibull distributions for an HB model, while also addressing the challenge of discretization of continuous parameters such as component time to failure. This methodology satisfies all tenets with the exception of tenet (4). The intent behind Marquez et al.'s algorithm is to alleviate reliance on MCMC to solve HB models for posterior distributions. MCMC is reviewed in Section 2.3.3, but the crux of their concern is that MCMC is computationally intensive so, for large complex systems, storing the transition matrix and state space vectors can be problematic for some computing systems. One issue with using the algorithm in [21] is that it is not widely published or adopted, whereas MCMC algorithms have been reviewed and utilized for over a decade, and open-source software executing MCMC sampling is readily available. Notwithstanding, the resultant predictions are not meaningfully compared to a system requirement, which could lead to misinterpretation or misapplication of results as discussed during the initial presentation of tenet (4) in Section 1.2.

The only methodology reviewed that intentionally included a step to facilitate a reliability requirement comparison was the EDRPM. This may seem like a trivial step that could be incorporated into any method, but inclusion focuses the intent of the application. In a classical progression of systems engineering, a trade-off analysis or analysis of alternatives (AoA) is conducted to evaluate different design alternatives, ultimately selecting one to progress to preliminary design [2]. If the way in which the prediction should be compared to a baseline requirement is not codified, it will likely manifest in a variety of ways that may, or may not, adhere to any assumptions or limitations in the prediction development. For example, the EDRPM calculates a system-level design failure rate by summing the means and variances of the function-level failure distributions, which is valid because this is a property of normal distributions. Subsequently, the design distribution is compared to a failure rate goal through a calculation similar to the conversion to a standard normal or modified signal-to-noise ratio (SNR), where the goal is treated as a cut-off, and design alternatives with predictions that exceed the failure goal are eliminated. However, if another method yielded hyperparameters for a different superpopulation, such as a beta,

the conversion of hyperparameter vectors to a standard normal would be nonsensical. It is also of note that this step affords the opportunity to clarify application intent. The EDRPM is not meant to identify the most reliable design—the purpose of the EDRPM is to identify and eliminate design solutions that are unlikely to meet reliability requirements; other performance parameters will collectively lead to design selection. Therefore, only retaining the design alternative that "least exceeded" the failure requirement would be contrary to the EDRPM's objective of retaining "decision-making freedom" with respect to reliability so that the system can be optimized for other cost, schedule, and performance measures. Including the comparison step in the methodology ensures a consistent, self-contained approach.

2.2 Early Design Reliability Prediction Method

The EDRPM was developed in direct response to the issues identified in Section 1.2. It posits that, consistent with SE principles, components can be categorized by the function they are used to execute. Therefore, an HB model can be constructed of function-specific reliability (superpopulation, governed by ϕ) that encompasses component reliability (population, governed by θ), supported by failure rate data (evidence, y). Using failure rate data from multiple sources creates component-level distributions that are industry-agnostic, as well as function-level distributions that are industry-agnostic and, to an extent, component-agnostic.

A concept coined "frequency weighting" is used to intentionally influence the extent to which component-level information impacts the function-level posterior, and subsequently the component-level update. Essentially, frequency weighting is a method to mitigate the quantity of data over-influencing hierarchical dependencies. The weight reflects the number of times a component has historically been used to solve a function, influencing the mode(s) of the function-level distribution. This prevents the distribution from being determined by data volume. That is, just because there is a lot of data for a certain component type does not mean it is more likely to be used; alternatively, less data for another component type does not imply it is less likely to be used. For illustration, commercial use of hard disk drives (HDDs) began in 1957 [22] while solid state drives (SSDs) were introduced for some military devices in 1991 but were not available to commercial notebooks as HDD alternatives until 2006 [23]. Thus, it is reasonable to assume there is significantly more failure rate data available for a HDD than SSD as mass storage, but it is not reasonable

to assume that a new system is significantly more likely to use a HDD than SSD, so it is desirable to increase the relative influence of a SSD on a mass storage function.

In practice, the frequency weight of a particular component is simply the proportion of times that component has been used for a particular function. For clarity and consistency of terms, calculation of frequency weights is presented in equations 2.1–2.3. For J components that could potentially solve a function, Equation 2.1 shows the function-specific frequency vector, $\overline{C}$, which captures all the function-to-component occurrences for candidate components, c_j, where $j = 1, ..., J$. Of note, the resource used for finding function-to-component occurrences for the EDRPM demonstrated in [13] was the Design Engineering Laboratory at Oregon State University (https://design.engr.oregonstate.edu). The total number of solved function occurrences, W, is the sum of function-to-component occurrences, shown by Equation 2.2. Equation 2.3 shows how the frequency weight, w_j is calculated for each component.

$$\overline{C} = (c_1, c_2, ..., c_j) \tag{2.1}$$

$$\sum_{j=1}^{J} c_j = W \tag{2.2}$$

$$w_j = \frac{c_j}{W} \tag{2.3}$$

A limitation of EDRPM is that it assumes component and function failure rate data are normally distributed, in order to enable closed-form solutions for the posterior distributions of the HB model. That is, the relationships in the model can be solved by a set of algebraic equations. The normality assumptions alleviated the need to solve for the joint posterior density through simulation. Expressions for the posterior distribution of hyperparameters in a normal HB model, μ and τ, found in [24], were modified by O'Halloran et al.'s concept of frequency weighting to yield three equations necessary to determine the hyperposterior distribution.

Equation 2.4 calculates the posterior update to a function's updated standard deviation, τ, as a function of evidence $y_{.j}$ for a quantity J of independent (exchangeable) component

groupings, each with a frequency weight of w_j.

$$p(\tau \mid y) \propto p(\tau)\sqrt{V_\mu}\left(\prod_{j=1}^{J}\left(\frac{e^{-\left(\frac{(\overline{y}_{.j}-\hat{\mu})^2}{2(\sigma_j^2+\tau^2)}\right)}}{\sqrt{\sigma_j^2+\tau^2}}\right)^{w_j}\right)^{J} \tag{2.4}$$

The hyperprior is represented by $p(\tau)$ and assigned a non-informative uniform distribution. The error variance, σ^2 is assumed to be known, with each j group having a sampling variance of $\sigma_j^2 = \sigma^2/J$. When μ is set to the predicted value, $\hat{\mu}$, all factors of function mean cancel in the simplified expression, such that the identity for τ holds for any value of μ. Equations 2.5 and 2.6 calculate the predicted function mean and mean error, respectively.

$$\hat{\mu} = \frac{\sum_{j=1}^{J}\frac{1}{\sigma_j^2+\tau^2}(w_j)\overline{y}_{.j}}{\sum_{j=1}^{J}\frac{1}{\sigma_j^2+\tau^2}(w_j)} \tag{2.5}$$

$$V_\mu^{-1} = (J)\sum_{j=1}^{J}\frac{w_j}{\sigma_j^2+\tau^2} \tag{2.6}$$

Since frequency weighting was only applied at the function level, though still influenced the component-level update due to shrinkage in the hierarchical arrangement, component posterior distributions were determined according to equations found in [24]; they were not modified for use in the EDRPM. Calculations for the updated component parameters are not reviewed in this work because their presentation is not germane to appreciating what the EDRPM developed, or how this work extends principles therein.

The function-level distribution was always normal by model construction so characteristics of a normal distribution were directly leveraged for the comparison of the prediction to a baseline requirement. The system-level failure rate for a design alternative, which was calculated as the sum of the contributing function distributions, was compared to the failure goal in a modified SNR approach similar to the conversion to a standard normal to compare a design failure rate to a failure rate goal, shown in equation 2.7.

$$z = \frac{(F_G - F_D)}{\sigma} \tag{2.7}$$

For equation 2.7, F_G is the failure goal, F_D is the mean of the system-level failure distribution, and σ is its standard deviation. For reliability requirements framed as a specific percent likelihood of operation at a specific time, a constant failure rate was assumed. The reliability for a component at a given time with a constant failure rate is shown in Equation 2.8, where λ is the failure rate in failures per unit time and t is the evaluation period.

$$R(t) = e^{-\lambda t} \tag{2.8}$$

The result of the ratio, z, indicates the likelihood of a design failure rate exceeding the failure rate goal, that is, the design not meeting a reliability cut off, using standard lookup tables for small z. Therefore, the normality assumption is fundamentally embedded in how the EDRPM is executed, driving the equations for determining posterior distributions and well as the method in which unreliable design alternatives are identified. If component or function failure data are not known or expected to follow a non-normal distribution, the validity of applying the EDRPM becomes suspect.

2.3 Hierarchical Bayesian Estimation

The growing prevalence with which HB models are used across a wide range of fields, including but not limited to engineering, economic forecasting, and meteorological forecasting, warrants a dedicated discussion to their mathematical foundation and what makes them powerful tools. The robustness of an HB approach, particularly when applied to reliability analyses, is not a new revelation. Alex Papadopoulus heralded the strength of an HB model in 1989 [25]. His demonstration based on exponential failures was entirely theoretical though, only seeking to quantitatively show that incorporation of prior information could lead to a more accurate prediction. Widespread adoption was delayed, however, until modern computers could reasonably handle the potentially computationally intensive operations affiliated with HB estimation.

2.3.1 Bayesian Estimation

The major benefit of any Bayesian estimation, sometimes called Bayesian inference, is that it allows incorporation of different types of information, such as behavior data, expert opinion, and engineering experience, into one model prediction [16]. Bayesian estimation is predicated on Bayes theorem, which establishes a relationship between the initial knowledge (the prior), the knowledge gleaned from observations (the likelihood), and the updated knowledge (the posterior). For data y generated from a population distribution characterized by parameter θ, the Bayesian relationship between the prior ($p(\theta)$), likelihood ($p(y|\theta)$), and posterior ($p(\theta|y)$) distributions is as shown in Equation 2.9. Information such as expert opinion and engineering experience manifest in the assumptions and distributions selected for the probabilities involved. The denominator in Equation 2.9 is a normalization factor most commonly referred to as the "evidence." Sometimes this distribution is referred to as the "marginal likelihood." This factor provides another means for incorporating beliefs about the population of data without respect to a given parameter.

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)} \tag{2.9}$$

The next most significant benefit of Bayesian inference is that it does not require assumptions about sampling intentions or sample sizes; intended comparisons do not require multiple tests to compute p-values [26]. The posterior distribution produced by Bayesian estimation is fixed based on the observations, but the nature of the distribution characterizes uncertainty. For example, rather than constructing multiple comparisons between a null hypothesis and various alternate hypotheses for hypothesis testing, an equivalent insight can be gleaned directly from the posterior distribution as-is, without additional manipulation. The "increased emphasis on interval estimation rather than hypothesis testing" shown by the applied statistics community [24] indicates that common-sense Bayesian interpretation of uncertainty is inherently valuable.

2.3.2 Hierarchical Bayesian Models

If the data can be meaningfully grouped to collectively enhance informing individual predictions, it is better to treat the data hierarchically rather than independently [26]. That is, instead of applying Bayesian estimation as described in Section 2.3.1 to each parameter

observation (θ) individually, all of the θ's with their associated data (y) are simultaneously evaluated. When the parameters are grouped, they are represented as a distribution, opposed to independent points, where each parameter observation is considered a random sample from the common parent distribution or "superpopulation" [24]. The parent distribution forms a higher level in the hierarchy, and the parameters that characterize the parent distribution are known as "hyperparameters." For the example presented thus far, this would be to say that the parameter θ is drawn from a distribution characterized by parameter ϕ. In this arrangement, there is now a formalized relationship established between the evidence observations and the parameter (as exists in all Bayesian estimation), as well as a relationship between the parameter observations and the hyperparameter. The hierarchical relationship between evidence y, parameter θ, and hyperparameter ϕ is depicted in Figure 2.1. Of note, θ and ϕ in all discussion thus far, and as intimated in Figure 2.1, are representative of parameter vectors. For example, ϕ may represent the parameter vector containing (α, β) for a Beta distribution, such that the hierarchy conveys $\theta_j \sim \text{beta}(\alpha, \beta)$.

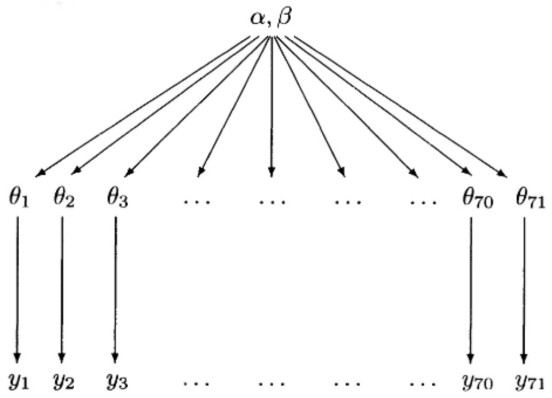


Figure 2.1. Illustration of Hierarchical Bayesian Model. Source: [24].

In this case, the joint posterior density is the "updated" probability for the observance of parameters θ and ϕ, given the observance of data y. Arranging the data in a HB model is different from pooling all the data into one population, where each y is considered an identical sample replication from the super population, because of the ability to retain information at the "intermediate" parameter level. Retaining the group parameter influence enables insight or "updated knowledge" to be applied back to that level since θ and ϕ are estimated simultaneously when determining the joint posterior distribution.

Furthermore, the hierarchical arrangement avoids over-fitting observable outcomes because parameter dependence is accounted for in the population distribution's shape, thus fewer parameters are required to capture the collective characteristics [24]. The interested reader can review Section 5.6 of [24] for an explicit comparison of predictions generated when a collection of data is treated in three different ways: (1) as one pooled population, (2) totally independent of one another, and (3) in a HB model.

The concept of "exchangeability" is critical to the relationship between parameters and hyperparameters, and is similar to the independent and identically distributed (iid) assumption often encountered in random sampling. Parameters (θ_1,...,θ_J for J observations of ϕ) are exchangeable if the probability of selection from the joint distribution is invariant to permutations in index (1,...,J) [24]. Essentially, there is no further ordering, grouping, or means to distinguish between the observations from the distribution. Models for data observations (y_1,...,y_n for n observations of θ_j) must also be exchangeable for the given parameter. When exchangeability is maintained, the hierarchical model interprets $y_{.j}$'s as random samples from θ_j's, and θ_j's as random samples from ϕ.

Although the parameters are considered random samples, the hierarchical relationship causes the updated superpopulation characterization to influence the update of the parameters in the middle tier. The nature of this influence is referred to as "shrinkage," and is considered a "rational, mathematical consequence of the hierarchical model structure" [26]. Shrinkage refers to the shifting of a posterior toward the closest mode of its parent's distribution; it is not necessarily toward the middle of the distribution if the distribution is multi-modal. Therefore, it is possible to have "outward" shrinkage. Shrinkage is observed at all levels where there is a parent: parameter θ will shrink toward hyperparameter ϕ, and any subsequent samples of y will shrink toward the respective θ parameter. Shrinkage reinforces why a concept like frequency weighting in the EDRPM, discussed in Section 2.2, is necessary—so that the modes of a superpopulation for a function are reflective of how the functions are historically achieved (by a population of components).

The equation for the unnormalized joint posterior density is provided in Equation 2.10, where $p(\phi)$ is the prior distribution of the hyperparameter, the hyperprior; $p(\theta|\phi)$ is the

population distribution, and the likelihood is as previously presented.

$$p(\theta, \phi|y) = p(\phi)p(\theta|\phi)p(y|\theta) \tag{2.10}$$

The challenge in evaluating the normalized joint posterior is in evaluating the marginal posterior distribution, $p(\phi|y)$. The marginal posterior is determined by the integral shown in Equation 2.11:

$$p(\phi|y) = \int p(\theta, \phi|y)d(\theta) \tag{2.11}$$

There are fundamentally three ways to evaluate the marginal posterior distribution, and thus the joint posterior distribution: algebraic analysis, integration, and simulation. Analysis by algebraic computation of the conditional probability can be used when the hierarchy is constructed of conjugate priors, that is, the posterior and prior distribution are the same "type," and ϕ is known. Examples of conjugate priors include the beta distribution for binomial, Bernoulli, and geometric likelihoods. The normal distribution is particularly unique because "normal begets normal" [27]; a normal prior and posterior yields a normal likelihood. Often, ϕ is unknown, so integration can be used for conjugate models provided there is a self-consistent normalizing factor. Otherwise, for non-conjugate HB models and/or situations when the hierarchy is so complex that determination is analytically intractable, approximation by simulation is required. MCMC is one method for simulation, discussed in the following subsection.

2.3.3 Markov Chain Monte Carlo Sampling

MCMC sampling methods are a class of algorithms that provide a way to sample from any distribution, even if that distribution is only defined up to a factor [28]. In the case of an HB, the joint posterior distribution is the distribution of interest, defined up to its normalization factor, the marginal posterior. Two of the most commonly used algorithms are the Metropolis-Hastings algorithm and the Gibbs Sampler algorithm. Evaluating the particular implementations and use-cases for these algorithms is beyond the scope of this work, but at a high-level, both operate similarly.

The idea of MCMC is to approximate sampling from the posterior distribution by simulating an appropriately constructed Markov chain. A Markov chain can be thought of as a model

of a system with a number of states, and the likelihood of progressing from one state to the next is only dependent on the current state. A number of steps are incremented, moving from state to state based on the transition probabilities of the Markov chain (a "random walk" [29]). If the walk is sufficiently long, the end state can be treated as a sample from the stationary distribution from the Markov chain. This is repeated a number of times, varying the starting point of the "walk" in a random manner (Monte Carlo). The end states of all the walks are then evaluated in aggregate to determine the stationary distribution, which conveys the probability of all states at any point in time. An overview of this process is depicted in Figure 2.2.

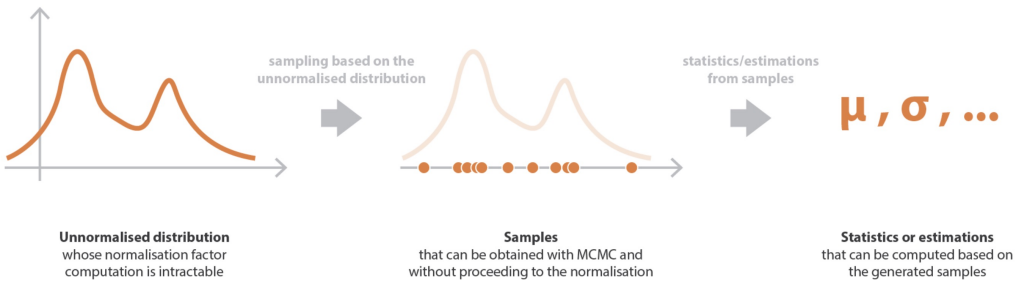


Figure 2.2. Illustration of MCMC Overview. Source: [28].

Therefore, the intent for an HB application is to sample from a Markov Chain with a stationary distribution that reflects the joint posterior distribution, such that when the samples have converged to the stationary distribution, it estimates the joint posterior distribution. Sample quality can be influenced by general techniques, including, but not limited to, slicing and burn-in [28]. Running multiple chains in parallel helps evaluating if any single chain got "stuck" and did not fully explore the entire posterior parameter space; if the chains overlap, it is reasonable to assume they converged to a representative solution, but if not, at least one got stuck [26]. Another indicator of a chain getting stuck is that the chain appears "clumpy" where successive steps are near each other. "Near" may be determined by visual observation or the descriptive statistic of correlation. Correlation also aids in the determination of convergence, which is subjectively assessed. Graphical evaluation of parameter samples for transient behavior is the primary means to determine convergence; if the stationary distribution was achieved, the sample behavior will be constant (no transients).

2.4 Chapter Summary

This chapter reviewed the suitability of a number of reliability predictions methods and methodologies, specifically for application in the conceptual design phase, and especially for an emerging system without a contemporary solution. While each related work had merit in addressing the challenges affiliated with early reliability prediction to various degrees, only one method, the EDRPM, addressed all. The biggest limitation of the EDRPM is its reliance on a normality assumption, which impedes widespread application. A HB is the right approach to reflect the component-function hierarchical relationship fundamental to systems engineering principles, but a complex hierarchy necessitates an alternative way to determine the joint posterior for hyperparameters at the function level. MCMC sampling permits determination of non-conjugate and analytically intractable posterior distributions. The next chapter discusses how the CDRPM mirrors the intent of the EDRPM, maintaining a component-to-function HB model, modified by using MCMC among other mathematically necessary revisions, to account for potentially non-normal component and function failure distributions.

CHAPTER 3:
Conceptual Design Reliability Prediction Method

This chapter presents the steps of the CDRPM, the specific tools used to execute the methodology, and some of the decision rationale during the development. The overall sequence of this methodology is essentially the same as that presented for the EDRPM in [13]. However, execution of the CDRPM necessitated a variety of computing environments and the transitions between environments indicated natural transitions in the process. The methodology sequence was therefore organized into steps that followed those transitions.

The first step is to identify the system, in terms of its required functions commensurate with the level of detail available during conceptual design, and determine the evaluation criteria that will be used to assess reliability. The second step is to construct an HB model to mathematically relate the evidence for candidate solutions with system functions. The third step is to evaluate the HB model to determine posterior reliability estimates. The fourth step is to combine the reliability estimates according to the system's RBD based on its functional architecture to determine a system reliability prediction. The fifth and final step is to evaluate the system prediction relative to a reliability requirement and determine if particular design solutions should be eliminated from further consideration due to likelihood of failing to meet said requirement. These steps are listed in Figure 3.1 with the primary software used for each indicated in parentheses.

1. Identify System and Evaluation Criteria (Excel/MATLAB), *Section 3.1*
2. Construct HB Model (Stan), *Section 3.2*
3. Determine Posteriors (R/RStan), *Section 3.3*
4. Determine System Prediction (MATLAB), *Section 3.4*
5. Identification of Unreliable Solutions Relative to System Requirement (Excel/MATLAB), *Section 3.5*

Figure 3.1. Steps of CDRPM Sequence

The remainder of this chapter is organized into the steps of the CDRPM. Sections 3.1–3.5 review each step in detail; the specific section affiliated with a step is shown in Figure 3.1.

To illustrate the execution of the CDRPM, a generic, abstract system is presented during the system identification portion of the first step and provides the platform for a running example through the remaining sections. In Chapter 4, the CDRPM is applied to the same abstract system used to illustrate the EDRPM in [13] for methodology verification. Application to a second system provided the opportunity to demonstrate flexibility in handling different types of data, review alternative simulation outcomes, and discuss ways to interpret those results. Therefore, this chapter focuses on the fundamental concepts of the CDRPM with implementation variations presented in subsequent chapters.

3.1 Example System Identification

The first step of the CDRPM revolves around data collection. The system's functional architecture is reviewed along with the reliability requirement. Components that could fulfill those functions are identified, the reliability criteria by which they will be evaluated are established, and the affiliated data is gathered. Any software that can serve as a database, preferably in a computer-readable format, is acceptable; Microsoft (MS) Excel was the author's software of choice. Data manipulation performed prior to model application, such as frequency weighting discussed in Section 3.3.5, was done in MATLAB as a matter of preference over manual manipulation within the original database and subsequently treated as an updated database in the same computer-readable format.

The abstract system used in this chapter to illustrate the execution of the CDRPM must perform three independent functions, referred to as *Function A*, *Function B*, and *Function C*. Each function can be performed by a number of component types. Similar to the annotation used by Gelman et al. in [24] and shown in Figure 2.1, parameters affiliated with component types are identified with a j subscript where $j = 1, ..., J$ for J total component types per function. The evidence for each component type consists of the observations of a variety of specific products or unique components of that type. Observations are identified with an ij subscript where $i = 1, ..., I_j$ for I_j total observations of a component type j.[3] The determination of product-, component-, and function-level posteriors is detailed in Sections 3.2 and 3.3 for *Function A* only. Function-level posteriors for *Function B* and *Function C*

[3]Of note, this subscript index is different from that used during the exchangeability discussion in Section 2.3.2. The letter i was selected for an evidence index, opposed to n, since n and N were both used during data collection.

are provided for system-level prediction and evaluation in Sections 3.4 and 3.5. In practice, and as demonstrated in the case study in Chapter 5, each function should be assessed in the same manner shown for *Function A* for a full CDRPM analysis.

For emphasis, *system* and *product* are not used interchangeably for the remainder of the CDRPM discussion. A *product* refers to a specific instantiation of a component type, whose reliability is represented at the lowest level of the HB model. The *system* consists of the collection of components selected to fulfill a set of requirements, whose reliability is represented at the highest level of the HB model.

While the nature of a hierarchy based on functional-physical relationships was briefly introduced in Section 2.2, the following scenario is offered for clarification before continuing with an abstract example. If *Function A* was "conjoin two physical entities," the component types could be *glue*, *Velcro*, *tape*, and *nails*, among others, but this list suffices for the analogy. The evidence for *glue* would then be observed reliability data for specific glue products such as Elmer's School Glue, Gorilla Glue, Loctite Super Glue, etc. The engineer should use her judgement as to which product test results are included as evidence; the application may reasonably preclude consideration of a product, such Elmer's School Glue, as a candidate solution so its data should not be used to influence the overall *glue* reliability or *Function A* reliability posteriors in the HB model. However, it is a best practice to be inclusive of candidate solutions so the range of potential components are represented.

3.1.1 Rationale for Example Data Selection

One of the strengths of the CDRPM is that it is not reliant on a particular type of data distribution because it uses an MCMC solution for HB estimation. As shown in Section 3.2, the construction of the HB model is based on the type of data being assessed. It then follows that the CDRPM is not reliant on a particular type of reliability data because it can handle the variety of distributions different data types may necessitate.

Rather than arbitrarily selecting a type of reliability data for the abstract example system, arguments of other methods reviewed in Chapter 2 inspired an HB model based off of success ratio data. A success ratio is considered the number of successful trials out of the number of trials conducted. Therefore, a success ratio is directly representative of reliability as defined in Chapter 1 and the results can be directly compared to a reliability requirement

expressed as a percentage without further transformation like through an SNR in [13]. While Johnson, Moosman, and Carter [16] did not explicitly refer to the data for their model as "success ratios," reliability was represented in binomial terms of either success or failure, providing at least one precedent for using this type of data in an HB model. Bergman and Ringi's method in [15] was founded on the acknowledgement that non-identical operating environments influence the absolute failure rates of system components so using failure rates from components in environments dissimilar from the one of concern is fundamentally flawed. Their contention that components maintain the same relative failure rate, regardless of operating environment, is reflected in the notion that a component has an inherent likelihood to function—an inherent success ratio.

An alternative argument for using success ratios is to consider why mean time to failure (MTTF) or mean time between failure (MTBF), both generally referred to as MTBF henceforth unless otherwise noted, may not be a preferred reliability measure during conceptual design. Without knowing what the final configuration looks like, it is difficult-to-impossible to know how often or for how long each component will be required to operate. When the components' MTBFs are indiscriminately aggregated for the system-level prediction, it is assumed that all components operate continuously when the system is in operation. This is often untrue for complex systems in which subsystems periodically enter standby states for a given operation. If the types and frequency of failures for a subsystem is less in a standby state, assuming continuous operation would conservatively contribute to a failure estimate. However, if the state transition induces component cycling, entering and exiting a standby state may be a greater component stressor than continuous operation is. While common data sources such as the *Electronic Parts Reliability Data (EPRD)* [30] and *Non-electronic Parts Reliability Data (NPRD)* [31] often present reliability data with the number of trials, the specific causes of failure are rarely known, especially not on a per-trial basis, so even if the operating assumptions could be refined prior to component aggregation, appropriately refined data is generally not available to support them. Thus, the accuracy of a prediction made with this type of data is rife with caveats, but the perceived precision of the observation data may mislead the analyst into being overly confident in prediction results. Notwithstanding, MTBF is a very common type of reliability data and may be all this is available to review across multiple products and component types.

All arguments in favor of success ratios aside, this data type was ultimately selected for

the example in this chapter because normally-distributed MTTF was used in the EDRPM's example, which is replicated through the CDRPM in Chapter 4. Selecting a different data type for this chapter resulted in a different hierarchy of distributions, demonstrating the flexibility of the CDRPM.

3.1.2 Function A Data

For this example, the generic *Function A* can be performed by four types of components (J = 4). Component types 1 and 2 are supported by 10 product observations each ($I_1, I_2 = 10$), type 3 by five observations ($I_3 = 5$), and type 4 by seven observations ($I_4 = 7$). Success ratios used for product observations were captured as successes and trials rather than just the resultant ratio to protect for post-estimation analysis. A potential analysis may be evaluating a posterior result in the context of a product with three successes out of four trials (75% success ratio) or 3000 successes out of 4000 trials (also a 75% success ratio). The precise data set used for this example is shown in Table 3.1.

Table 3.1. Example Function A Data Set

$i \setminus j$	Type 1			Type 2			Type 3			Type 4		
Product	Success (n)	Trials (N)	Ratio	Success (n)	Trials (N)	Ratio	Success (n)	Trials (N)	Ratio	Success (n)	Trials (N)	Ratio
1	140	160	0.88	2	16	0.13	15	50	0.30	1723	2000	0.86
2	36	40	0.90	17	20	0.85	290	350	0.83	356	500	0.71
3	67	70	0.96	320	450	0.71	250	500	0.50	250	475	0.53
4	57	60	0.95	49	50	0.98	46	100	0.46	35	70	0.50
5	99	110	0.90	570	600	0.95	4	20	0.20	68	90	0.76
6	1780	1850	0.96	144	150	0.96				58	60	0.97
7	94	100	0.94	19	23	0.83				75	100	0.75
8	34	35	0.97	28	30	0.93						
9	13	15	0.87	4677	5000	0.94						
10	39	40	0.98	71	75	0.95						

3.2 Construction of the Hierarchy

The second step of the CDRPM establishes the relationships within the HB model and implements the model in an MCMC sampler. Stan, an open-source probabilistic programming software written in C++, is the computing environment used for MCMC in this work. Once the type of reliability data was selected, distributions were fit or selected for the hierarchy. Assumptions made in this section are specific to the type of data utilized

and the distributions selected to represent that data; the assumptions do not necessarily hold true for all hierarchical models. The rationale for the construction of the HB model used is presented "bottom-up," from the discrete evidence to the function-level characterization, in Subsections 3.2.1–3.2.3; the HB model is presented coherently and graphically in Subsection 3.2.4, with annotations to be read "top-down."

3.2.1 Individual Product Level

The success ratio, represented as the number of successes n for product i of type j, n_{ij}, out of the number for trials N for that product, N_{ij}, was assumed to come from a binomial distribution characterized by θ_{ij}, summarily shown as $n_{ij} \sim binomial(N_{ij}, \theta_{ij})$. Alternatively stated, product ij had an underlying probability of success (reliability) of θ_{ij}. This representation assumed that θ_{ij} was constant for all trials and that the outcome of any trial was independent of all other trials. Furthermore, it assumed that the specific articles of a given product tested in the evidence were representative of the population of the product considered. If the trials for Product X were performed by articles from a variety of production lots of Product X, the results were assumed to apply to all lots. However, if the trials for Product X were only performed by articles from Lot 1, the results should be applied to Lot 1 only or the engineer may elect to further assume a homogeneous population of Product X across all lots and apply the data accordingly. The product reliability, θ_{ij}, was drawn from its component type beta distribution, formally relating the individual product level to the component level of the hierarchy. This relationship is mathematically represented by $\theta_{ij} \sim beta(\alpha_j, \beta_j)$ for component type j by which product i was classified. The beta distribution describes values in the interval of 0 to 1, which all success ratios must fall within, and is commonly used for this sort of application [32].

3.2.2 Component Type Level

A component level beta distribution's shape parameters, α and β, are related to the central tendency and spread parameterization of a beta distribution according to equations 3.1 and 3.2 where μ represents the distribution mean and κ represents the spread or concentration.

$$\alpha = \mu\kappa \tag{3.1}$$

$$\beta = (1 - \mu)\kappa \tag{3.2}$$

Parameterization by mean and concentration is common in Bayesian estimation [32]. In the hierarchy for this example, the component means were drawn from the function-level distribution. A beta distribution was selected for the function level for the same reasons it was deemed appropriate for the component type level. The concentration parameters were informed by a vague prior distribution to represent positive real numbers. A gamma distribution was selected for this purpose. As later demonstrated in Section 3.3.3, posterior results were insensitive to the shape and rate parameters of the gamma prior. Therefore, while the HB model permitted a unique gamma prior for each component type, the same shape parameter, S', and rate parameter, R', were used for all component prior gamma distributions.

3.2.3 Function Level

The relationships at the function level were very similar to those at the component level. The reliability of a function was assumed to follow a beta distribution, from which the component level beta distribution means were drawn. The function level parameters were informed by a beta prior for the mean and a gamma prior for the concentration. All function prior parameters were defined to be vaguely informative priors. Sensitivity analyses for these prior parameters are shown in Section 3.3.3.

3.2.4 Hierarchical Model

Presenting the arguments for the relationships in the HB model from the lowest (product) level to the highest (function) level is rational because the presentation begins with what is known—the evidence. Once the relationships are established, however, the model is constructed in most any software from the function level down to the product level, since the upper level distributions must be defined in order to draw from them for the lower level distribution parameters. All of the parameters and mathematical relationships presented thus far in Section 3.2 are summarized in Figure 3.2.

Function

$\mu_A \sim beta(A, B)$

$\kappa_A \sim gamma(S, R)$

where A, B, S, R must be defined up front for function level priors.

$\alpha_A = \mu_A \kappa_A$ and $\beta_A = (1 - \mu_A)\kappa_A$

Component Type

$\mu_j \sim beta(\alpha_A, \beta_A)$

$\kappa_j \sim gamma(S', R')$

where S', R' must be defined up front for component prior(s).

$\alpha_j = \mu_j \kappa_j$ and $\beta_j = (1 - \mu_j)\kappa_j$

Individual Product

$\theta_{ij} \sim beta(\alpha_j, \beta_j)$

$n_{ij} \sim binomial(N_{ij}, \theta_{ij})$

where n_{ij} is the number of successes out of N_{ij} trials (the evidence) for product i classified as type j.

Figure 3.2. Mathematical Summary of Hierarchical Model. This framework was specifically developed for the type of data used to assess the reliability of *Function A*.

When Figure 3.2 is read from the top down, the parameter dependencies are readily apparent. The contents of Figure 3.2 are shown graphically in Figure 3.3. Each solid arrow can be interpreted as a random draw. For example, μ_A is a random draw from the prior function level beta distribution, defined by $beta(A, B)$. Dotted arrows show relationships to algebraic transformations (e.g., μ_A is part of the calculation to determine α_A).

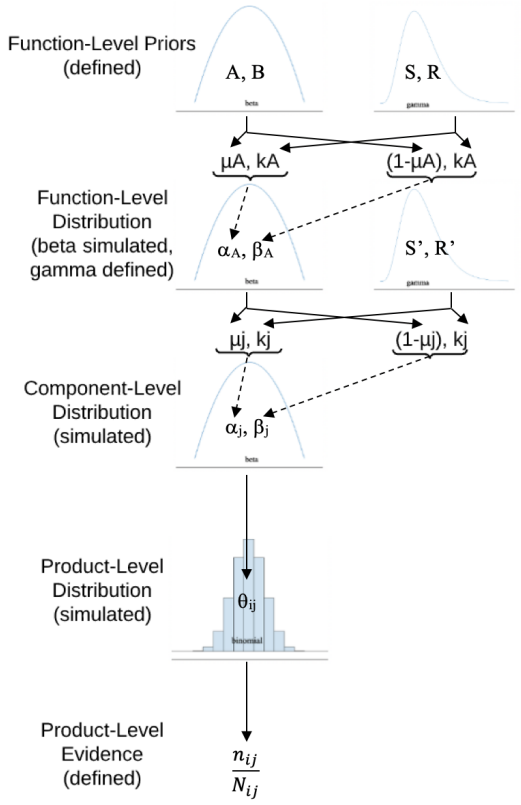


Figure 3.3. Graphical Summary of Hierarchical Model. Adapted from [26].

3.2.5 Instantiation of the Model in Code

The dual hierarchy presentations, bottom-up and top-down, illustrate the value in approaching the model from each direction, as well as the complexity involved in determining posterior distributions for a relatively simple HB model with only three levels. In practice, the execution of the HB model is more circuitous due to the simultaneous posterior estimation enabled by MCMC, and the only sequence of significance is the one specified by the computing environment.

Even for a relatively simple, three-level hierarchy whose base relationships are captured in 10 equations, those in Figure 3.2, the task of programming and assigning data to hierarchical elements can be extremely tedious. For example, it took approximately 50 lines of code in Stan to establish an HB model with a significantly reduced subset of the data in Table 3.1, specifically three success ratio observations spread over two component types. The

code for this implementation, which served as a proof of concept, is provided in Appendix A. Labor intensity and propensity for making a transcription error during this step were identified as limitations of this work, discussed in Chapter 1.5. For enhanced efficiency, the author collaborated with Samuel Buttrey, an Associate Professor in the Operations Research Department at Naval Postgraduate School (NPS), to create a script to partially automate generation of an HB model for Stan for the full *Function A* data set. The script is shown in Appendix B.

3.3 Determination of Posteriors

The data of the first step and model of the second step are combined in the third step, determining the posterior distributions. R was used to assign data to model aspects, initiate the sampling, and display the MCMC results. RStan was the program, more specifically the library, that interfaced between R and Stan. Data assignment and sampling initiation were incorporated in the script contained in Appendix B, with the Stan model captured as a string rather a separate Stan file (.stan). The comprehensive list of parameters involved in the posterior estimations based on the data in Table 3.1 applied to the HB model summarized in Figure 3.2, without frequency weighting, is provided in Appendix C.

Since the individual products' success ratios were modeled as binomial distributions, the product-level of the hierarchy also had posterior updates. These estimates, each representative of a single product, should be interpreted as what the product reliability is more likely to be based on all the information available, from the overall function as well as from the other component performances. This is one way in which operating environments, test conditions, and use-cases level out in an HB model. If a single product of a given type has a reliability significantly different from its contemporaries, it is unknown whether that difference is due to a truly significant difference in product design, or just different use conditions. For example, an electronic widget used in the desert is more likely to fail from inadequate cooling than one used in the arctic, but if the exact same product is used in both environments, it is likely to have different a failure rate for each. Without knowing under what conditions all the products were used, but assuming some relation exists based on the type of component and function for which it is used, shrinkage of the product-level posterior towards the component-level posterior is a reasonable manifestation of that relationship. A granular review of these parameters is beyond the application intent of the CDRPM which

is why their posterior updates are provided in Appendix C and not embedded in this chapter.

The posterior parameters of greatest interest, as they most directly relate to the component-level and function-level beta distributions that will be used for the system reliability prediction, are the α and β parameters for each beta distribution. These parameters are summarized in Table 3.2.

Table 3.2. Posterior Distribution Parameters

Posterior Distribution Parameters		
Distribution	α	β
Function A	8.267×10^{-1}	5.823×10^{-1}
Comp Type 1	2.377×10^{0}	4.724×10^{-1}
Comp Type 2	1.445×10^{0}	4.743×10^{-1}
Comp Type 3	9.259×10^{-1}	9.829×10^{-1}
Comp Type 4	1.516×10^{0}	7.315×10^{-1}

These are the results from the MCMC evaluation of the *Function A* HB model, representing the shape parameters, α and β, for the beta distributions at the indicated level in the hierarchy. The MCMC sampling was initiated on four parallel chains, each with 6,000 iterations and a 2,000 iteration warm up.

3.3.1 Convergence Monitoring

Evaluation of the MCMC sampling for convergence is arguably of greater significance than the estimated posteriors themselves. Without convergence, the stationary distribution has not been achieved by the sampling process, thus the resultant distribution does not reflect the joint posterior distribution and the estimated parameters are relatively meaningless even if they appear reasonable. Indications of convergence were introduced in Section 2.3.3. There are a number of ways to monitor for convergence and most are not entirely redundant, so it is useful to use a variety of methods to build confidence that a stationary distribution was achieved. The specific methods used for the *Function A* example in this work are discussed in the following subsections, but they should not be interpreted as exhaustive or restrictive. From each perspective reviewed, the MCMC sampling converged to a stationary

distribution, confirming that it is possible to mathematically relate product, component type, and function reliability through an HB model.

Pairwise Scatter Plot

One of the graphing options in RStudio is the pairs() plot, which produces a pairwise scatter plot such as the one in Figure 3.4. Each dot represents a sample during the MCMC execution. Red dots indicate divergent transitions where the step size between samples likely missed a small feature in the distribution resulting in a biased estimate and potentially invalid results [33]. A greater adaptation delta sampling parameter imposes a greater likelihood of acceptance between steps and can be used to mitigate divergent transitions. Yellow dots indicate transitions that exceeded the maximum tree depth where the sampler was terminated prematurely to avoid excessively long execution times [33]. A greater maximum tree depth sampling parameter permits further exertion before termination. For larger hierarchies, the fidelity of the plot may make identification of dot color infeasible; this was the case for the example system and is readily apparent by the fidelity of Figure 3.4. Plotting fewer parameters at a time in the pairs() environment is helpful, but the most immediate indication of troublesome transitions, whether divergent or exceeding maximum tree depth, is a verbal warning printed to the screen. The results for *Function A* had no such transitions or printed warnings, indicating that all transitions met tolerances specified during the sampling initiation.

Figure 3.4. Pairwise Scatter Plot. A color-coded pairwise scatter plot of specified parameters in the HB model. Acceptable steps in the Markov chain are indicated with blue dots, while red and yellow dots indicate abnormal or unacceptable transitions. All the transitions of this simulation were acceptable.

Another benefit of the pairs() output is evaluation of the overall distribution of the samples when using multiple parallel chains, that is, when executing simultaneous simulations. If the chains executed random walks through the parameter space but sufficiently overlapped, the resultant estimation will have some sort of central tendency, which is readily observed by the histograms on the diagonal. If the chains did not overlap, there would be gaps or multiple modalities in the histograms.

Trace Plots

While the histograms show whether or not parallel chains appreciably overlapped, they do not necessarily indicate whether any or all Markov chain(s) reached a stationary distribution. The chains may achieve a consensus for a central tendency, but during similar transient periods. A relatively constant mean and variance of the sample distribution in the trace plot suggest the chain achieved the stationary distribution. Alternatively stated, if a trace plot for a parameter over the duration of a sample, excluding any warm-up or burn-in period, shows transient effects, the chain likely did not resolve to an accurate parameter estimate. Traces with flat periods indicate stuck chains, which may also appear "clumpy" if the flat periods are relatively brief so as not to be noticed individually.

All of the trace plots for the beta parameters in Table 3.2 showed relatively constant mean and variance. Examples of the trace plots for parallel chains of a single parameter are shown in Figure 3.5. Trace plots for all parameters are included in Appendix C.

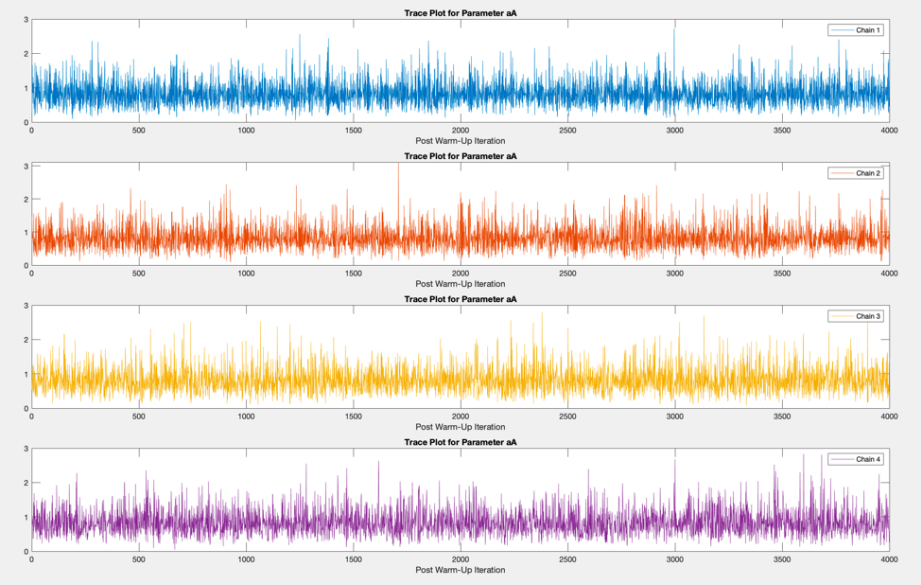


Figure 3.5. Parallel Chain Trace Plot. The plot shows the end states achieved from random sequential random walks. If the stationary distribution is achieved, there should be little variation (subjectively assessed) in the end states of each walk, which manifests as a trace plot with relatively constant mean and variance.

Correlograms

Correlograms offer another indicator of mixing, which is useful when trace plots such as those in Figure 3.5 are rather dense. The correlogram shows the autocorrelation between adjacent steps, or samples, in the sampling process, indicating how much the value of one step was influenced or related to the value of the previous step. An uncorrelated sample progression is indicative of achieving a random walk, where a chain is not stuck at a particular value. All parameters achieved zero or nearly zero correlation, thus effectively uncorrelated, such that the MCMC sampler behaved as desired. Furthermore, all estimated parameters achieved nearly zero correlation rather rapidly, suggesting appropriate mixing and arrival at the stationary distribution.

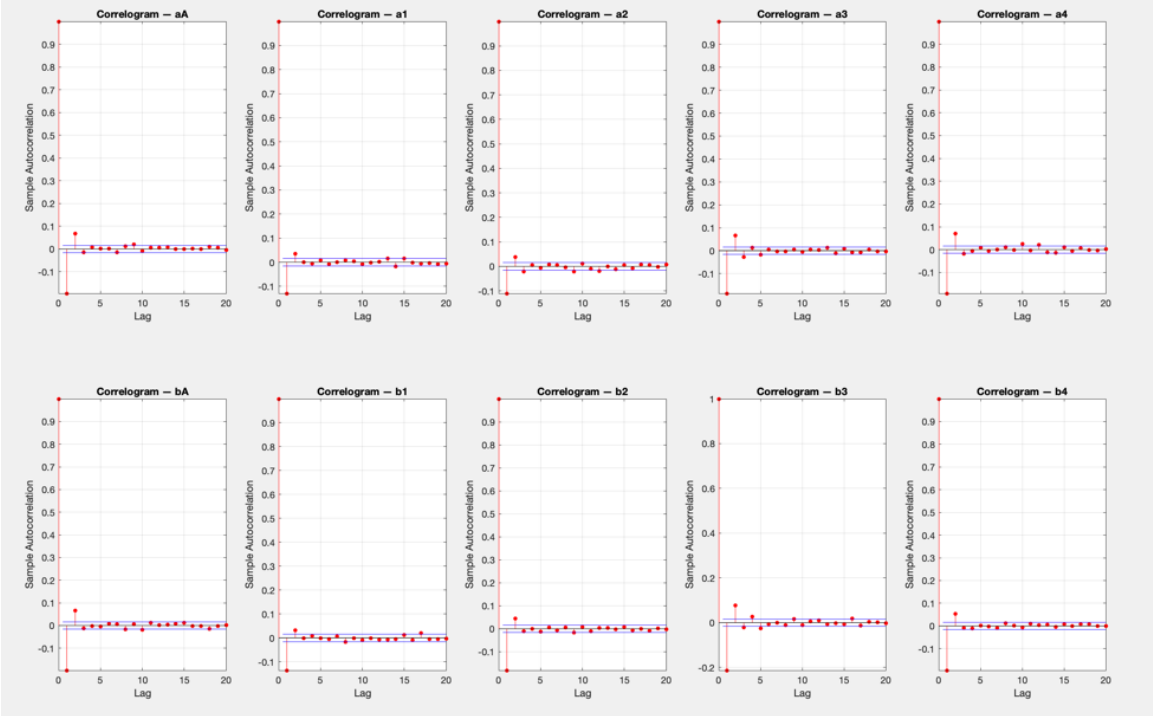


Figure 3.6. Correlogram. This plot shows the autocorrelation between steps during sampling from a Markov chain. The less correlation, the more random the walk, with zero to low correlation being most desirable.

3.3.2 Posteriors in the Context of Evidence

While not necessary to review prior to proceeding to the system prediction, it can be beneficial to graph the posterior distribution probability density function (pdf) of each

component type overlaid on a scatter plot of the respective evidence to verify reasonableness of the component-level beta distributions. This presentation is similar to that of MCMC sampling from a distribution in Figure 2.2 except the scatter plot is of evidence and not of simulation samples. In contrast with a normal distribution, it can be difficult to conceptualize the shape of a beta distribution from its shape parameters. The y-axis scale is nonsensical when data is combined in this manner, but graphs generally convey the relationship between data concentration and the posterior beta characterization.

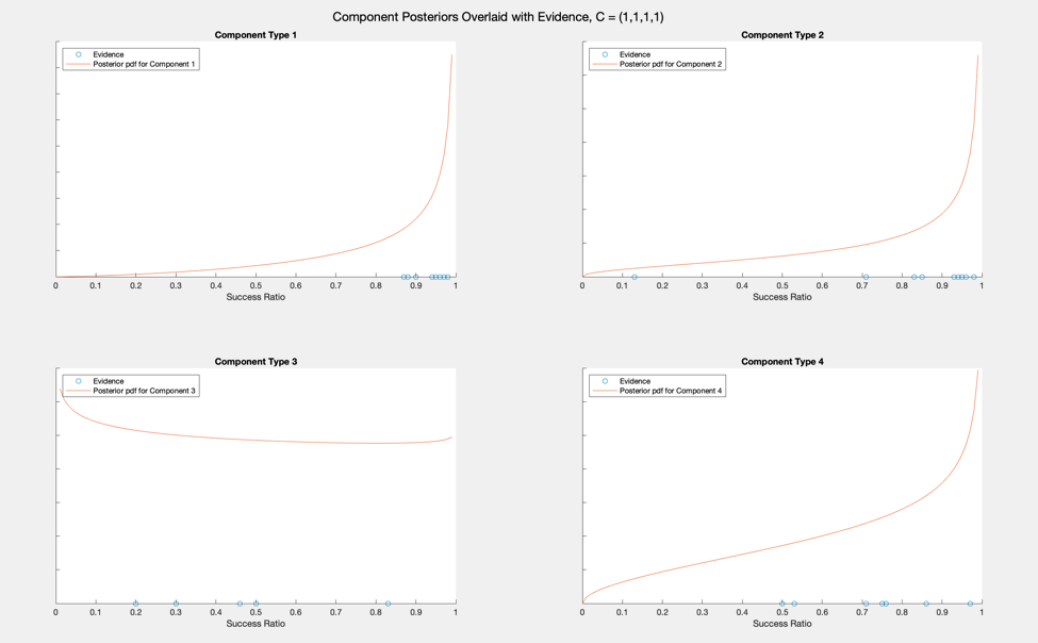


Figure 3.7. Posteriors with Evidence. The posterior distribution of each component type is shown with its original evidence to determine if the distribution generally captures the data concentration.

3.3.3 Prior Distribution Sensitivity Analysis

The HB model was initially run with a prior distribution for the function-level mean of $\mu_A \sim beta(A, B)$ where $A = 5$ and $B = 2$. The model was rerun for $\mu_A \sim beta(0.5, 0.5)$, a beta distribution with different shape parameters. The pdfs of these prior beta distributions are shown in Figure 3.8. While they appear significantly different, the primary benefit they offer to the HB model is that they confine the selection of μ_A to the interval of zero to one.

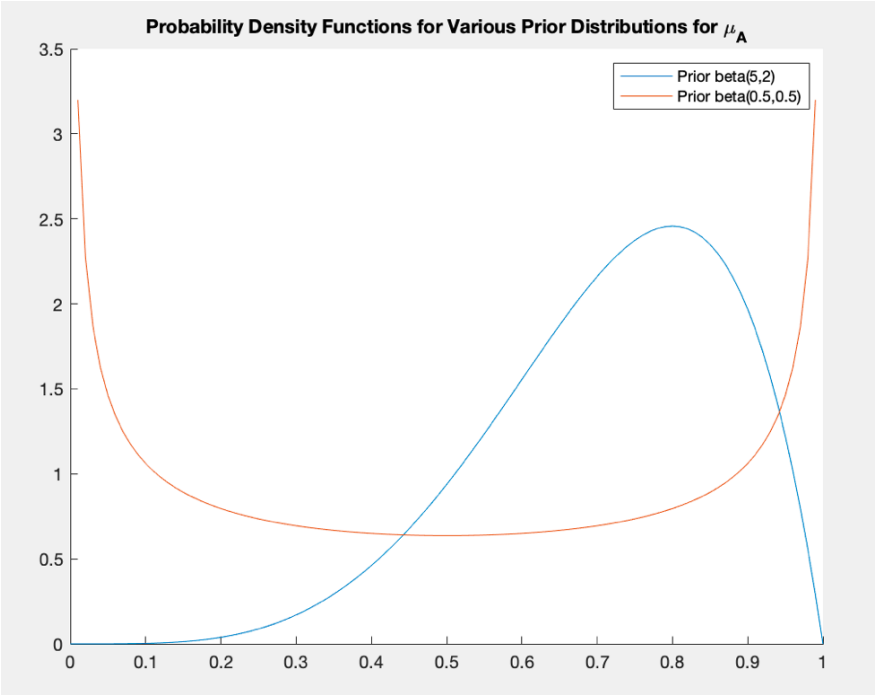


Figure 3.8. Beta Priors Considered for Sensitivity Analysis. The two illustrated beta distributions were applied as priors for the function-level mean, μ_A, to determine the sensitivity of the HB results on defined prior.

In practice, the posterior estimates were relatively insensitive to the exact characterization of the beta prior. Percent differences between the α and β estimates for each beta distribution in the HB model were calculated for the $\mu_A \sim beta(5,2)$ and $\mu_A \sim beta(0.5,0.5)$ cases. The parameter estimates and percent differences are shown in Table 3.3. The greatest percent difference observed was 0.9% for one parameter affiliated with the component type 1 beta distribution. The resultant pdfs for the component type 1 estimates are shown in Figure 3.9, illustrating that the difference in parameter estimates is essentially indistinguishable and that a nearly 1% difference is insignificant, as could reasonably be expected. The results confirmed that a beta distribution is an appropriate vague informative prior for the function-level mean, applying a natural constraint to the interval of zero to one without overly influencing the HB estimation.

Table 3.3. Sensitivity Analysis for Function-Level Beta Prior

Prior beta(A,B)	(5,2)		(0.5,0.5)		% Difference	
Beta Distribution	alpha Estimate	beta Estimate	alpha Estimate	beta Estimate	alpha Estimate	beta Estimate
Function A	0.82666	0.58232	0.82760	0.58140	0.1%	0.2%
Type 1	2.37749	0.47241	2.35562	0.47205	0.9%	0.1%
Type 2	1.44514	0.47434	1.45221	0.47382	0.5%	0.1%
Type 3	0.92592	0.98288	0.92667	0.98211	0.1%	0.1%
Type 4	1.51592	0.73148	1.52065	0.73446	0.3%	0.4%

These are the results from the MCMC evaluation of the *Function A* HB model using different beta priors defined the by the indicated parameters. The results showed the posterior parameter estimations are insensitive to the specific beta characterization.

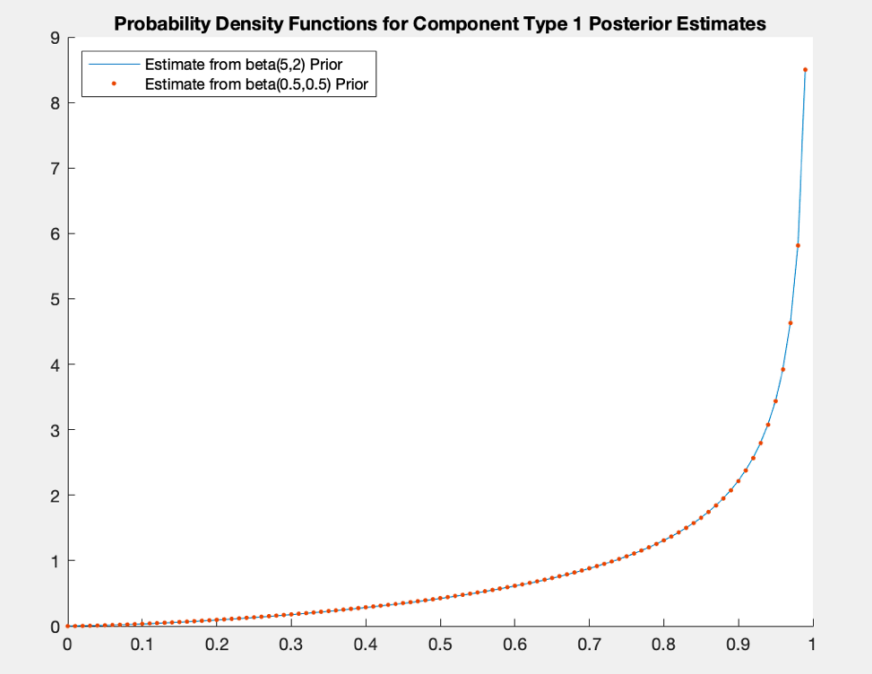


Figure 3.9. Component Type 1 Posterior Estimates for Analyzed Beta Priors. The posterior estimates for component type 1 had the greatest percent difference due to the different beta priors analyzed but were still essentially indistinguishable.

The same manner of sensitivity analysis was done for the gamma priors at the component

level as well as the function level. This analysis demonstrated more variability at the component-level than that observed for the beta prior analysis, but the function-level posterior was insensitive to its gamma prior. Therefore, engineering judgement should be applied when defining the gamma prior distributions' shape and rate parameters. Gamma priors that are more likely to yield higher concentrations for the component type distributions, κ_js, are recommended to better enable distinguishing between component type posteriors, while still maintaining a level of data uncertainty due to the probabilistic nature of the distribution characterization. The greatest difference in posterior estimates observed during the gamma prior sensitivity analysis was for component type 1. Figure 3.10 illustrates how a lower concentration prior yielded a "flatter" beta posterior. This tendency was observed for all component types (but was most exaggerated for type 1) which would make them less distinguishable from one another if a system-level analysis required granularity greater than the function-level. The negligible effect on the function-level posterior is also shown in Figure 3.10.

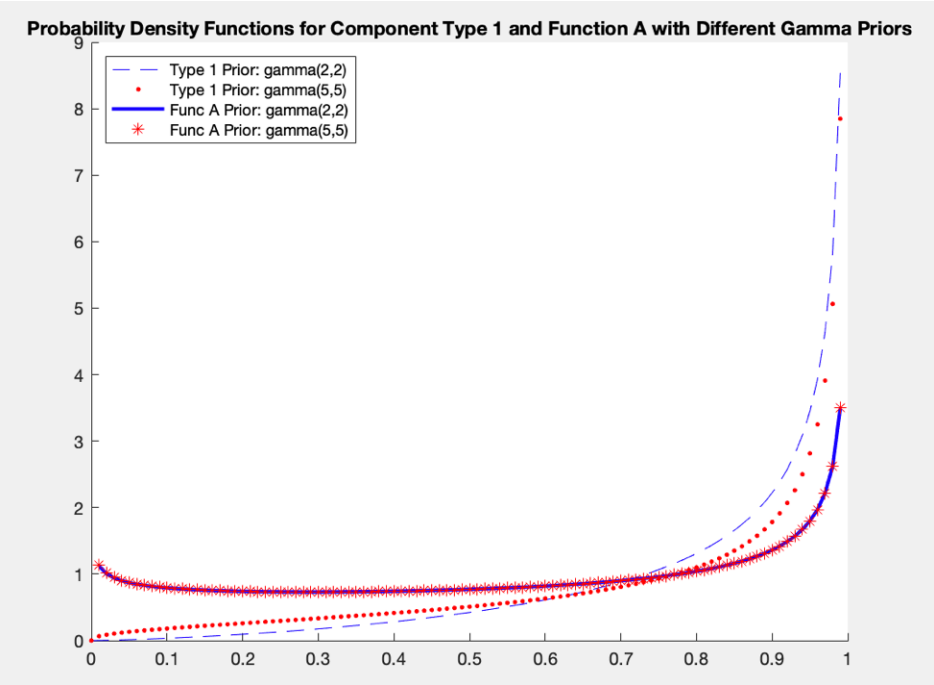


Figure 3.10. Posterior Estimates for Analyzed Gamma Priors. The posterior estimates for function level were essentially indistinguishable for the different gamma priors analyzed but the component types showed greater sensitivity. Component type 1 showed the greatest percent difference caused by the selected gamma prior.

3.3.4 Graphical Presentation of Consolidated Results

Since the MCMC sampling converged to a stationary distribution by all perspectives reviewed and the prior distribution sensitivity analyses demonstrated reasonable prior selections, the posterior distribution parameters presented in Table 3.2 were accepted as valid results for *Function A*'s reliability hierarchy. The pdfs for all component type and function posterior distributions are shown in Figure 3.11.

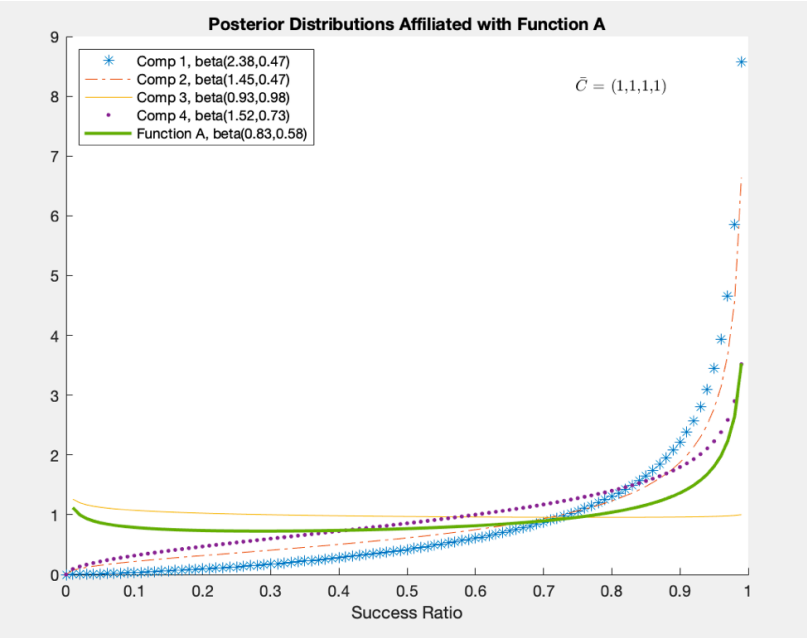


Figure 3.11. Summary Illustration of Posterior Distributions for Function A. Precise parameter estimates for the component type and function distributions are provided in the legend. These results are for an unweighted case with a frequency vector of $\bar{C} = (1, 1, 1, 1)$.

Since there was no frequency weighting in this case, meaning all types of components were historically as likely to fulfill the function, the *Function A* posterior was expected to appear similar to a median likelihood between the component type reliability distributions. Before proceeding to a system determination, a discussion of how frequency weighting was incorporated in the CDRPM is presented in Section 3.3.5.

3.3.5 Frequency Weighting Implementation

Frequency weighting, reflective of the likelihood of a particular type of component to fulfill the function, is the mathematical implementation of subjective knowledge, such as expert opinion, to influence the mode(s) of a function's posterior distribution. In [13], frequency weighting was applied in equation 2.4 as an exponent to each component type's argument in a product, which was effectively a replication of the argument, subsequently scaled back to the number of unweighted arguments. Since this modification could not be inserted within the MCMC process, the net effect was achieved by this work through data set manipulation. Prior to evaluating the posteriors, the evidence was replicated as component type groupings

according to the frequency vector, $\bar{C}$. That is, if X_j represented the full set of observations for component type j and $\bar{C} = (1, 1, 4, 2)$, evidence was passed to the MCMC software as $X_1, X_2, X_3, X_3, X_3, X_3, X_4, X_4$ rather than X_1, X_2, X_3, X_4 (the equal frequency case). As far as the MCMC software was concerned, this data represented seven independent distributions so it generated seven sets of posterior parameters at the component type level. Therefore, the posterior estimates had to be consolidated back to the appropriate number of component types post-estimation; the consolidated component-level and product-level estimates were determined by the arithmetic mean of the replications' results.

To avoid this expansion and consolidation process affiliated with replicating component groupings, replicating the observation data within each component type's data set was considered. For example, if the unweighted component type 3 had five observations ($X_3 = 5$) with an occurrence frequency of four ($c_3 = 4$), the weighted component type 3 would have 20 observations ($X_3 \times c_3$) passed to the MCMC software. The problem with this approach was twofold: (1) it artificially reduced the uncertainty in the component type performance and (2) it over-influenced the function-level posterior, significantly affecting the lesser-weighted component-level posteriors due to shrinkage towards the function-level mode. Shrinkage, as discussed in Section 2.3.2, is the manifestation of how an upper level in a hierarchical model influences a lower level posterior by causing the posterior to "shrink" towards the upper level's posterior mode.

Examples of data manipulation by replicating observations within a component type, henceforth referred to as frequency weighting "within type," and replicating full component type data sets, referred to as "by type," are shown in Appendix D. A detailed analysis of the impact on the function posterior parameter estimates, as well as the impact on the other component estimates is included. For a summary illustration, Figure 3.12 shows component-level and function-level posteriors for an arbitrary frequency vector, $\bar{C} = (1, 1, 4, 2)$, implemented in both manners discussed. The "within type" graph on the left illustrates the significant influence the component type 3 weighting had on the other component types; each component is "flatter" like the component type 3 posterior and noticeably different from the summary of unweighted results in Figure 3.11. The "by type" graph on the right demonstrates an influence on the function-level posterior to approach the component type 3 posterior, but the other component types are less influenced (because the function level is less influenced). Figure 3.13 shows a comparison of just the *Function A*

posterior under three circumstances: no frequency weighting, frequency weighting within type, and frequency weighting by type. Evaluation of the results led to the conclusion that replication by type is preferable and recommended to implement frequency weighting for Bayesian estimation by MCMC sampling.

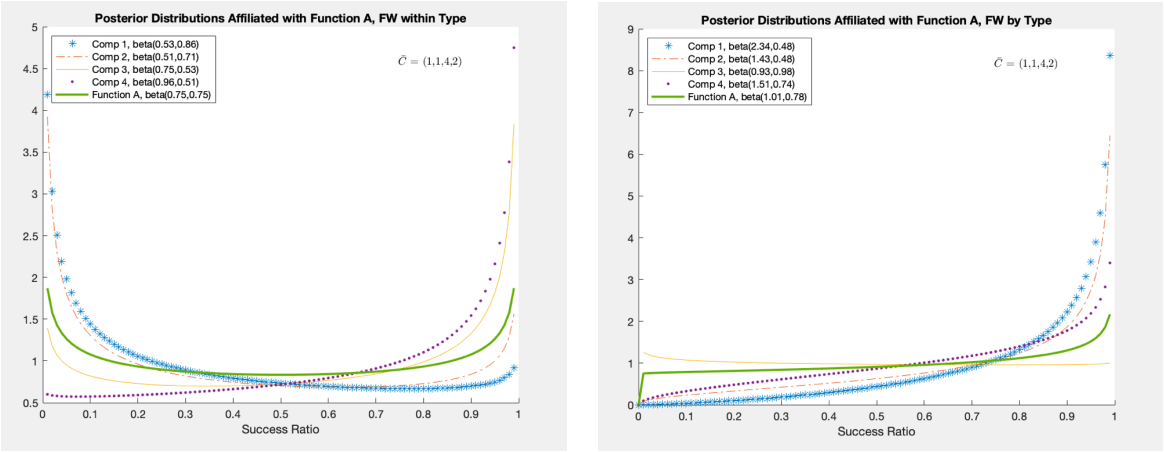


Figure 3.12. Comparison of Overall Effect of Frequency Weighting Implementation. For the same frequency vector, $\bar{C} = (1, 1, 4, 2)$, the left graph shows the effect of the within type approach while the right graph shows the effect of the by type approach.

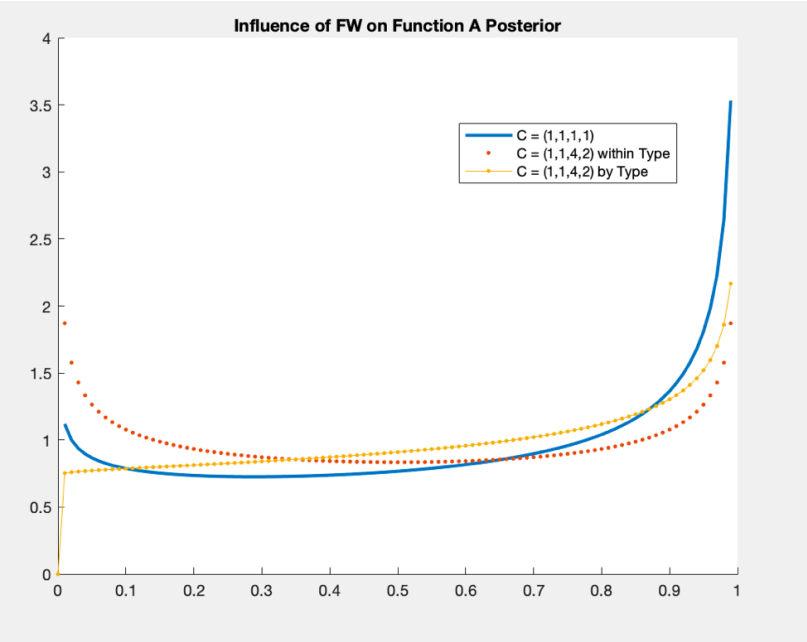


Figure 3.13. Frequency Weighting Implementation Method Influence on Function Posterior. Function-level posterior results are shown for three cases: unweighted, weighted with the within type method, and weighted with the by type method.

3.4 Determination of System Reliability Prediction

The fourth step of the CDRPM is to determine a system reliability prediction by combining the posterior function-level reliability distributions according to the system's RBD and then characterizing the distribution of the system-level prediction. This was accomplished by conducting an MCS of each function's posterior distribution and then performing element-wise operations per the mathematical relationships of the RBD. For the example system, *Function A*, *Function B*, and *Function C* operated in series. Therefore, element-wise multiplication of the samples drawn from each function yielded the system's reliability prediction distribution. There are other ways to accomplish prediction aggregation for a series arrangement of conjugate distributions at the function level. However, the MCS method is applicable for any combination of function-level posterior distributions, as well as any RBD arrangement. Any computing environment capable of MCS and distribution fitting is sufficient for this step; MATLAB was used as a matter of preference to generate the results shown in this section and Section 3.5.

For purposes of assessing a system reliability prediction, *Function B* and *Function C* were presumed to be characterized by beta distributions with the shape parameters indicated in Table 3.4. The unweighted posterior distribution parameters for *Function A*'s HB model are presented alongside to summarize all the distributions under consideration for this exercise.

Table 3.4. Parameters for All Reliability Distributions Affiliated with the System

Beta Distribution	*Function A*		*Function B*		*Function C*	
	alpha Estimate	beta Estimate	alpha Estimate	beta Estimate	alpha Estimate	beta Estimate
Function	0.83	0.58	2.5	0.4	9.0	2.0
Comp Type 1	2.38	0.47				
Comp Type 2	1.45	0.47				
Comp Type 3	0.93	0.98				
Comp Type 4	1.52	0.73				

Function A parameters determined through HB model analysis. *Function B* and *Function C* parameters provided explicitly for CDRPM demonstration efficiency.

The results of 10,000-sample MCSs of each function-level distribution and their combination to determine the system-level reliability prediction distribution are shown in Figure 3.14. Each graph within the quad chart shows a 100-bin histogram of sample results corresponding to success ratios, thus each blue column has a width equivalent to 1% of reliability performance. The overlaid red lines indication distribution fits of the sample results, whose parameters were confirmed to match those in Table 3.4 for the three functions. The red line on the system distribution was produced through a non-parametric fit. The fitted system-level non-parametric distribution is the primary information carried into the fifth step of the CDRPM.

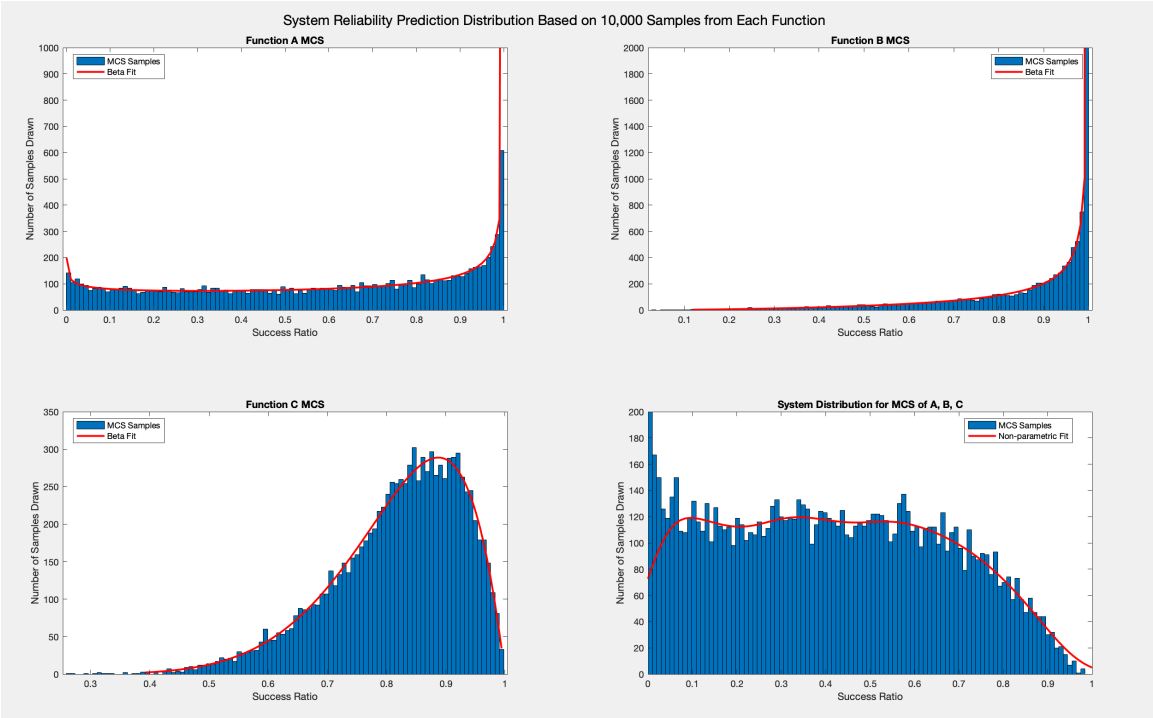


Figure 3.14. System Reliability Prediction Distribution Based on Functions A, B, and C. These particular results are for a 10,000-sample MCS of each function, binned into 100 intervals of 1% each, subsequently fitted with an appropriate distribution shown in red.

3.5 Identification of Unreliable Solutions Relative to System Requirement

The fifth and final step of the CDRPM compares the system-level reliability prediction to the reliability requirement and identifies unreliable design solutions that should be excluded from further consideration during design maturation. For this scenario, the example system has a reliability requirement of a 70% likelihood of success, that is, a success ratio of 0.7. Since the system reliability distribution is in terms of success ratios, it directly represents likelihood of failure. Conversion of failure times using Equation 2.8 to facilitate comparison with a modified SNR using Equation 2.7, as is done in [13], is not required.[4]

[4]If success ratio data was unavailable and the HB model was predicated on a different kind of reliability data, a conversion to failure likelihood is required.

Since the posterior distribution from this method directly represents the desired end-state characteristic, system reliability, the fitted system distribution is converted into a cumulative distribution function (cdf). The likelihood of meeting the reliability requirement is then the complement to the cdf of the system distribution evaluated at the requirement. Consistent with the EDRPM, component-level posterior distributions are substituted for function-level distributions in the system reliability determination for greater prediction granularity when a system is unlikely to meet a requirement.

Figure 3.15 illustrates extension of this step to the example system with pseudo-math, a statement that is representative of the calculation performed. The exact syntax to execute this step is dependent on the computing environment. The precise likelihood of the example system meeting the 70% reliability requirement based on the system-level non-parametric reliability distribution found in Section 3.4 was 15.9%.

Likelihood of System Reliability Meeting 70%

$$= 1 - cdf(fitted\_system\_distribution, 0.7)$$

$$= 15.9\%$$

Figure 3.15. Calculations for Reliability Requirement Comparison. The statements are representative of calculations performed. Exact syntax for the analysis is dependent on the computing environment.

Whether or not a 15.9% likelihood of meeting a requirement is acceptable depends on the risk posture of the engineer. To reiterate, the intent of the CDRPM is to identify and eliminate design alternatives that are unreliable, permitting other performance metrics to determine subsequent down-selection during the AoA. Therefore, relatively liberal (low) thresholds of acceptability, alternatively, greater risk tolerances, are encouraged so as not to overly constrain the design solution. If a 15.9% likelihood is acceptable, no further analysis is required and all candidate solutions progress to the next phase of design maturation. If it is not acceptable, the likelihood calculation illustrated in Figure 3.15 is repeated for the function-level distributions rather than the system-level distribution.

It is worth addressing why the function-level reliability distributions are not evaluated relative to the system requirement before this step in the CDRPM. That is, the fitted

system distribution is deliberately assessed first. The RBD arrangement determines the significance of a function's influence on the system reliability prediction. For example, if two functions operate in parallel, it may be acceptable for one individual function to be less than the system requirement depending on the performance of the second function. The finer component-level assessment would be additional work of limited value and component type down-selection would unnecessarily constrain the design solution. Similarly, requirement partitioning, otherwise known as reliability allocation, is not performed for the function-level analysis; the functions are intentionally compared to the base system-level requirement. This enables the same calculation to be applicable regardless of the particular function's role in the RBD, facilitating more efficient methodology execution and preventing over constraint of the design space by not imposing additional derived requirements.

For illustration of the CDRPM, 15.9% likelihood is deemed unacceptably low and finer assessment is desired; Table 3.5 shows the likelihood of each system function meeting the reliability requirement.

Table 3.5. Function Likelihood of Meeting Reliability Requirement

Likelihood of Meeting 70% Reliability	
Function Assessed	% Likelihood
Function A	44.7
Function B	83.8
Function C	84.6

Likelihood percentages were determined in the manner shown in Figure 3.15.

The function that is least likely to meet the system reliability requirement is then replaced by its component-type distributions in the system-level prediction determination, step 4 of the CDRPM shown in Section 3.4. Note that this is not a perfect process of elimination. A function with a higher likelihood, dominated by higher reliability component types, could mask one unreliable component type whose removal could be sufficient. However, this is unlikely since the HB model to determine posterior reliability captures uncertainty based on the preponderance of evidence more than a arithmetic mean does. In this scenario,

Function A was least likely to meet the reliability requirement, which could have been qualitatively inferred from the central tendencies of each function's MCS in Figure 3.14, though quantitatively confirmed by review of Table 3.5. Therefore, the MCSs of posterior reliability distributions for the component types in the *Function A* hierarchy were substituted for the *Function A* MCS in the RBD arrangement. Figure 3.16 shows the fitted system reliability distributions produced by the element-wise combination of *Function B* and *Function C* with component types 1, 2, 3, and 4, respectively.

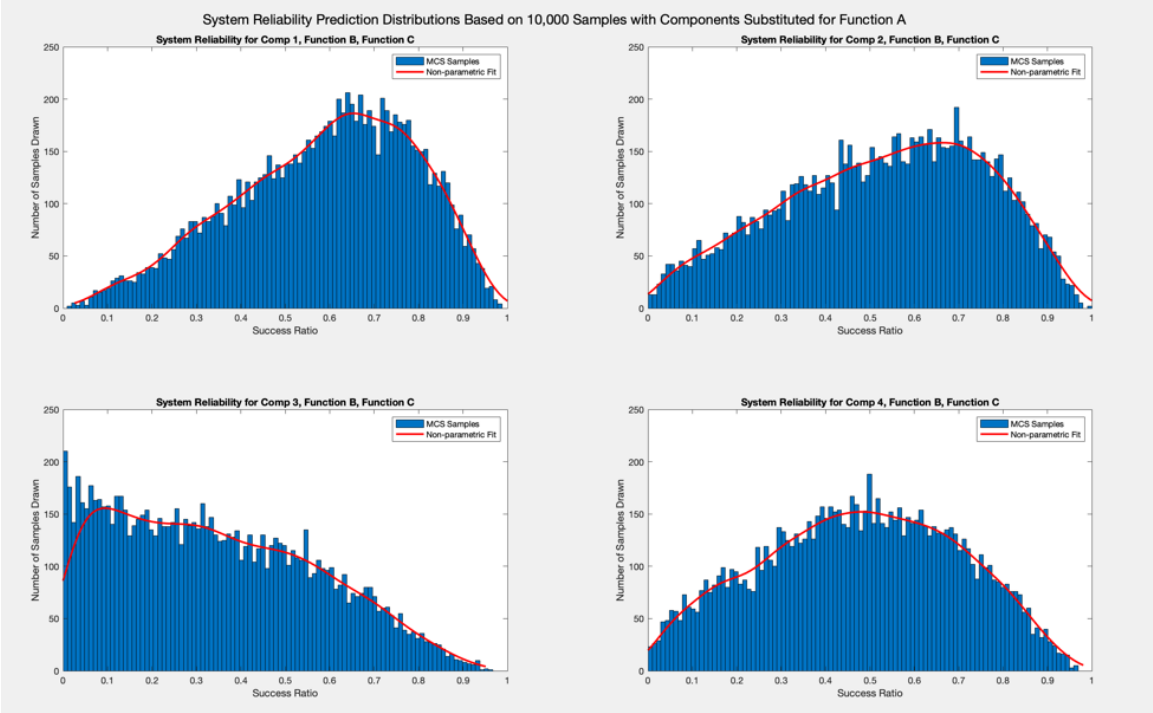


Figure 3.16. System Reliability Prediction Distribution Based on Components Substituted for Function A. These results are for a 10,000-sample MCS of each function and respective component type, binned into 100 intervals of 1% each, subsequently fitted with an appropriate distribution shown in red.

A qualitative review of the central tendencies of each plot in Figure 3.16 suggests that design solutions utilizing component type 3 are unlikely to meet the system reliability requirement since the bulk of its mass is to the left, around lower success ratios. The likelihood for each configuration to meet the system reliability requirement was calculated and the results are shown in Table 3.6. The calculations quantitatively confirm that component type 3 is

unlikely, specifically only 8% likely, to result in a reliable system, that is, one that meets its reliability requirement. Since an approximately 16% likelihood of meeting the requirement was unacceptable for the function-level assessment, the impetus for using component types for a finer assessment, it may be that an 18% likelihood deems component type 4 an undesirable solution, as well.

Table 3.6. Component Type Likelihood of Meeting Reliability Requirement

Likelihood of Meeting 70% Reliability	
Component Assessed	% Likelihood
Comp Type 1	33.2
Comp Type 2	26.8
Comp Type 3	8.0
Comp Type 4	18.0

Likelihood percentages were determined in the manner shown in Figure 3.15 with indicated component types substituted for *Function A*.

Thus, this analysis, with a presumed risk posture, concludes that design solutions utilizing component type 3 and component type 4 are unlikely to produce a system that meets its reliability requirement. Therefore, component types 3 and 4 should be removed from consideration for any configurations that progress through the AoA and any design database should be updated accordingly. In the event the likelihood of meeting a system reliability requirement is unacceptable for the remaining components, the engineer should return to the function-level likelihood assessment and identify the function that is next least likely to produce a reliable system. That function's reliability distribution should be replaced with that of its constituent component types as was demonstrated for *Function A*, and the process of elimination would continue. All subsequent analyses should only be inclusive of the components that were not previously eliminated. That is, for the example system, *Function B* had a lower likelihood of meeting the system requirement than *Function C*. Therefore, the MCS of *Function B*'s reliability would be replaced with MCSs of its component types, and aggregated with *Function C* and only component types 1 and 2 of *Function A* since component types 3 and 4 of *Function A* were eliminated.

3.6 Chapter Summary

This chapter presented the five steps of the CDRPM in detail, with execution illustrated through application to a small generic example system consisting of three abstract functions. The evidence to support the analysis was based on individual product success ratios which generalized the failure data in an appropriate manner since little is known about the specific use of a component during conceptual design and, usually, little is known about the precise cause of failure in published data. Therefore, it is best to have data that fully encompasses all possible environments, otherwise one should have data collected in a manner that does not impose false discrimination due to operating condition influence. This approach preserves the uncertainty in the data for a more inclusive, realistic reliability characterization at all levels of an HB model. The methodology illustration was successful in that it demonstrated it is mathematically possible to meaningfully related component reliability with function performance, to aggregate component and function reliability to produce a system reliability prediction, and to evaluate the prediction relative to a requirement. System reliability predictions were made with increasing configuration specificity to illustrate the process of identification and elimination of unreliable candidate solutions. This chapter did not, however, illustrate that the CDRPM can be applied to hierarchies of different sizes or constructed of different distributions. Furthermore, the CDRPM was ultimately intended to mirror the EDRPM, but there was no explicit comparison to confirm this intention was met. Therefore, to verify execution consistency between the CDRPM and EDRPM, as well as demonstrate the application versatility of the methodology developed herein, the abstract system presented in [13] is assessed by the CDRPM in Chapter 4.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Methodology Verification: Comparison to the EDRPM

This chapter serves as verification of the CDRPM. "Verification" and "validation" are words commonly heard within the SE discipline, often jointly abbreviated as "V&V," so the separate and distinct role of each activity is often muddled. Plainly speaking, verification asks, "Did I develop the system correctly?" and validation asks, "Did I develop the correct system?" [34]. Since the CDRPM was intended to serve as a generalized form of the EDRPM, it should at the very least be able to duplicate the results of a scenario where the EDRPM was applied. Anticipating some degree of discrepancy due to the CDRPM being a stochastic method while the EDRPM is deterministic, successful replication would indicate the CDRPM was developed in the manner intended; thus, successful replication verifies the CDRPM.

The abstract system used to illustrate the EDRPM's derivation in [13] was supported by failure rate data in terms of failures per millions of hours (failures/MHours), which were assumed to follow normal distributions. Three different frequency weighting cases were evaluated: an unweighted case where $\bar{C} = (1, 1, 1, 1)$, a moderately weighted case where $\bar{C} = (1, 1, 2, 5)$, and a heavily weighted case where $\bar{C} = (1, 1, 2, 50)$. Therefore, successful replication of the findings published in [13] would not only verify the methodology, but confirm the appropriate frequency weighting implementation method was selected for the CDRPM, as well as illustrate the flexibility of the CDRPM to accommodate a variety of data distributions and hierarchical arrangements.

The contents of this chapter are as follows. The nature of the system, specific evidence used for evaluation, and EDRPM results are presented in the first section. Next, the unweighted case is evaluated. It was speculated that the scale of the failure data led to convergence errors during MCMC sampling, but once the scale was modified, utilizing known properties of the normal distribution, the CDRPM results matched the EDRPM results with less than 1% error. Percent errors less than 2% were assessed as acceptable, indicative of successful replication without any significant accuracy loss from the simulation. The weighted cases are evaluated after the unweighted case, which were replicated with less success. A detailed review of the EDRPM results revealed an error in the execution of the

EDRPM. Once rectified with the approval of one of the EDRPM's authors, the CDRPM successfully replicated the corrected results from [13]. The chapter concludes with a summary presentation of findings demonstrating methodology verification and introduces how Chapter 5 will accomplish validation.

4.1 Abstract Example Used in EDRPM Development

The nature of the abstract system used in [13] is similar to the example system in Chapter 3. The EDRPM system consists of three functions, designated as *Function A*, *Function B* and *Function C*, related in an unspecified manner. Only the specific evidence for *Function A* is presented in [13] and duplicated in Table 4.1 so this chapter focuses on replication of the posterior results for this function, rather than replication of the conclusions for the system as a whole. The data in Table 4.1 is product failure rate evidence categorized into four generic component types. Each datum represents a mean failure rate that follows a normal distribution, all of which share an assumed standard deviation of 1.5 failures/MHours.

Table 4.1. EDRPM Function A Evidence. Source: [13].

Evidence (Failures/Mhours)			
Comp 1	Comp 2	Comp 3	Comp 4
1.0E-05	3.0E-06	3.0E-06	1.0E-06
9.0E-06	4.0E-06	5.0E-06	2.0E-06
9.0E-06	9.0E-06	3.0E-06	4.0E-06
8.0E-06	5.0E-06	2.0E-06	5.0E-06
7.0E-06	1.0E-06	2.0E-06	4.0E-06
9.0E-06	7.0E-06	na	6.0E-06
1.0E-05	2.0E-06	na	7.0E-06
1.1E-05	4.0E-06	na	5.0E-06
1.2E-05	1.0E-05	na	8.0E-06
8.0E-06	7.0E-06	na	4.0E-06

The units in the title and exponents in the data are likely double-accounting for evaluation per millions of hours since failure rates per one trillion hours (e.g., 7E-12 failures/hour) would be an extraordinarily low rate. The data was treated as if the title indicated units of failures/hour.

The HB model used for this function was a normally-distributed, two-tier hierarchy, presented in Figure 4.1. The hierarchy is considered two-tier even though there are three levels of abstraction (product, component type, and function) because the product level in the EDRPM is not modeled to have a posterior update; only the component type distribution parameters and function distribution hyperparameters have posterior updates. As a reminder, "hyper" is essentially synonymous with "parent"; the use of "hyper" was omitted during the CDRPM discussion in Chapter 3 because use of "parameter," "hyperparameter," and "hyperhyperparameter" seemed unnecessarily cumbersome for a three-tier hierarchy. The hyperparameters μ and τ represent the mean and standard deviation of the function-level normal distribution, while θ represents the component type's mean.

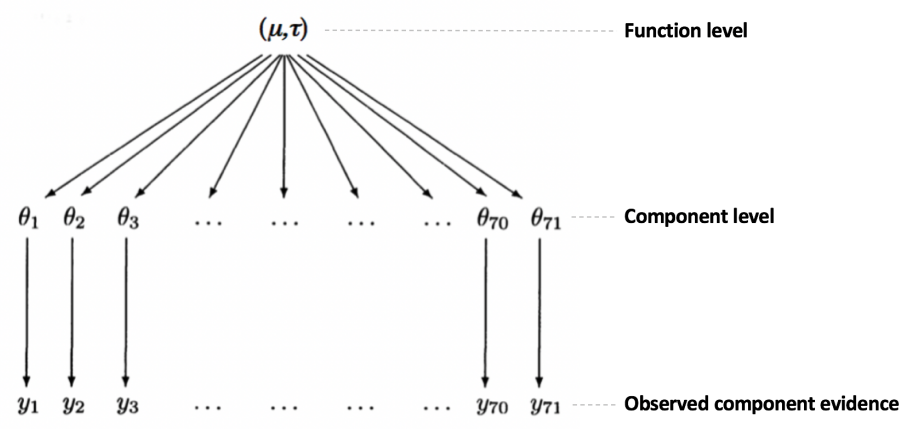


Figure 4.1. EDRPM HB Model. Source: [13].

The posterior parameters determined by the closed-form equations of [13] are shown in Table 4.2. These are the results the CDRPM results were initially compared against.

Table 4.2. EDRPM Function A Results. Source: [13].

Frequency Vector	Case 1 (1,1,1,1)	Case 2 (1,1,2,5)	Case 3 (1,1,2,50)
	Mean		
θ1	9.2766E-06	9.2723E-06	9.2711E-06
θ2	5.2020E-06	5.1997E-06	5.1965E-06
θ3	3.0312E-06	3.0227E-06	3.0202E-06
θ4	4.6058E-06	4.6015E-06	4.6002E-06
Function	5.5289E-06	4.8359E-06	4.6393E-06

4.2 Comparison to Unweighted Case

The CDRPM was first applied to the case with no frequency weighting, where $\overline{C} = (1, 1, 1, 1)$. The first obvious issue was that the MCMC sampling would not converge on the posterior distributions. Adjusting common sampling specifications such as the warm-up period and number of sampling iterations was ineffective. Modifying more nuanced execution parameters such as maximum tree depth and adaptation delta were equally ineffective. The most overt indication of non-convergence was the list of warnings that printed to the screen upon finishing the sampling cycle. A screenshot of such warnings is shown in Figure 4.2. This particular list of warnings is meant to be illustrative only; variations of these warnings, while heeding the recommendations provided, continued to occur.

```
Warning messages:
1: There were 6527 transitions after warmup that exceeded the maximum treedepth. Increase max_treedepth above 12. See
http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
2: There were 1 chains where the estimated Bayesian Fraction of Missing Information was low. See
http://mc-stan.org/misc/warnings.html#bfmi-low
3: Examine the pairs() plot to diagnose sampling problems

4: The largest R-hat is 4.12, indicating chains have not mixed.
Running the chains for more iterations may help. See
http://mc-stan.org/misc/warnings.html#r-hat
5: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be unreliable.
Running the chains for more iterations may help. See
http://mc-stan.org/misc/warnings.html#bulk-ess
6: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quantiles may be unreliable.
Running the chains for more iterations may help. See
http://mc-stan.org/misc/warnings.html#tail-ess
```

Figure 4.2. Example of Non-convergence Warnings. The particular MCMC execution for this set of warning was specified by the following parameters: four chains, 2500 iterations/chain, 500 iteration warm-up, no thinning, max tree depth = 12, adaptation delta = 0.95.

The perspectives suggested for convergence monitoring in Section 3.3.1, namely a pairwise scatter plot, trace plot, and correlogram, all confirmed the stationary distribution was not achieved. A review of each is useful and included in this section to illustrate what an unsuccessful simulation may look like. The figures that follow are from a simulation of two chains of 6000 iterations each, with a 2000 iteration warm-up, maximum tree depth of 15, and adaptation delta of 0.99.

The pairwise scatter plot of the hyperparameters, shown in Figure 4.3, shows not only a lack of a central tendency in the histograms, but non-adjacent binning. This is indicative of each chain only taking on a single value for the given parameter, though it is possible (while extremely unlikely) that each chain evenly alternated between the two values. Yellow dots in the scatter plots of Figure 4.3 indicate where maximum tree depth was exceeded between samples. Since the chains did not approach a central tendency, they did not converge to an appropriate stationary distribution.

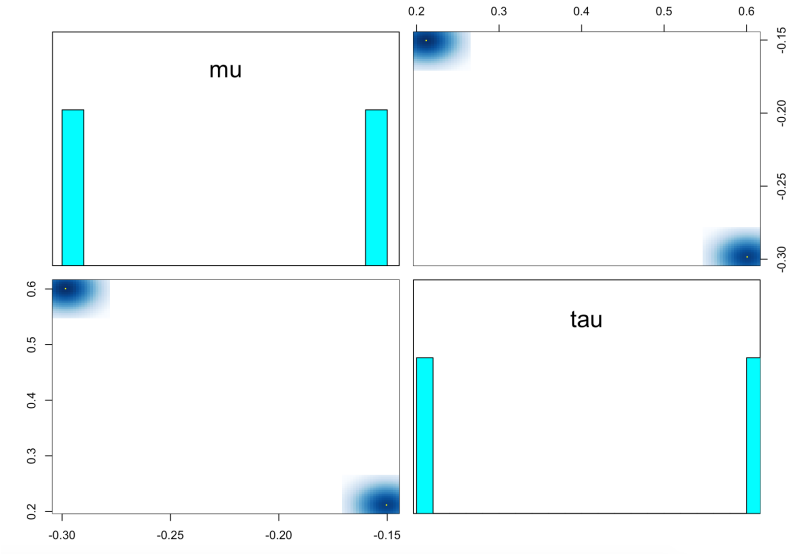


Figure 4.3. Non-convergent Pairwise Scatter Plot. Non-adjacent groupings in the histograms and scatter plots indicate the chains did not converge to an appropriate stationary distribution.

The interpreted histogram behavior was confirmed in the trace plots of Figure 4.4 for parameters μ and τ. Each chain takes on a single value, not only indicating lack of convergence, but lack of any movement within the parameter space, refuting the possibility that chains evenly alternated values. Therefore, the sampler's random walk for the function-level distribution was stuck at the same starting point. Like the trace plots reviewed in Chapter 3, the trace plots of the component-level parameters show relatively constant mean and variance, indicating that a transient was not in progress. However, since the chains were stuck at the hyperparameter level, little transient would be expected even with a shorter warm-up period prescribed.

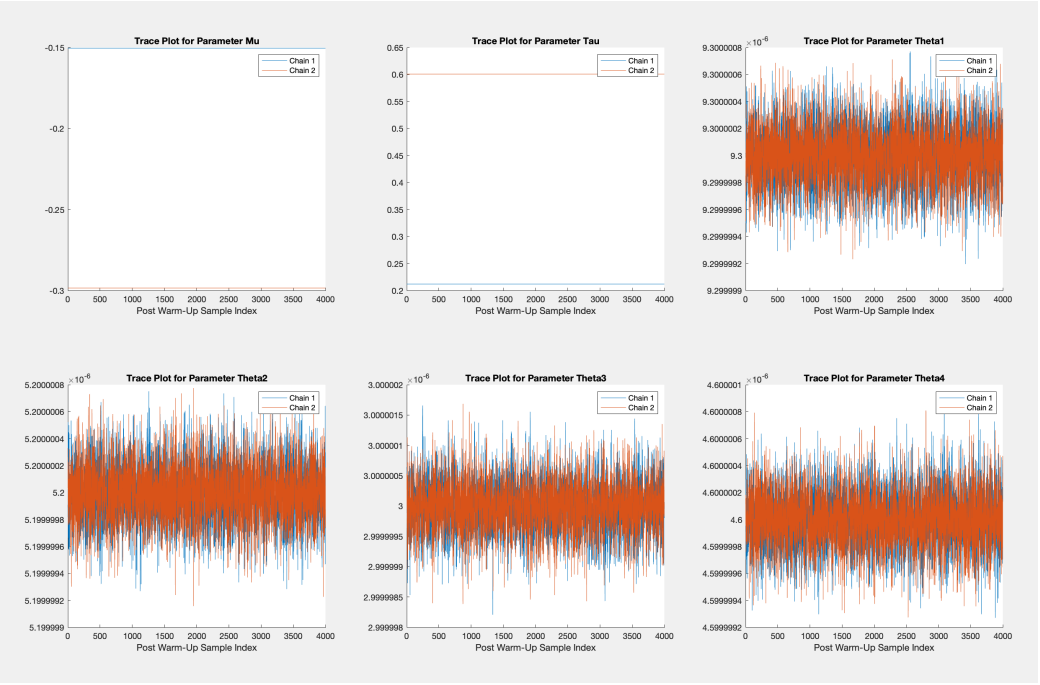


Figure 4.4. Non-convergent Trace Plots. Non-overlapping traces for μ and τ verify that each chain was stuck at a single value.

The last perspective reviewed to assess the movement of the MCMC, or lack thereof, was of the parameters' correlograms shown in Figure 4.5. For μ and τ, there is high correlation (effectively one) for the samples in the chains. The trace plots for the component θs in Figure 4.4 were dense so it was possible there were periods of flat-lined behavior, similar to behavior observed in Chapter 3. The correlograms for these parameters, however, showed a very low correlation between subsequent samples, indicating the sampler behaved as desired (for these parameters).

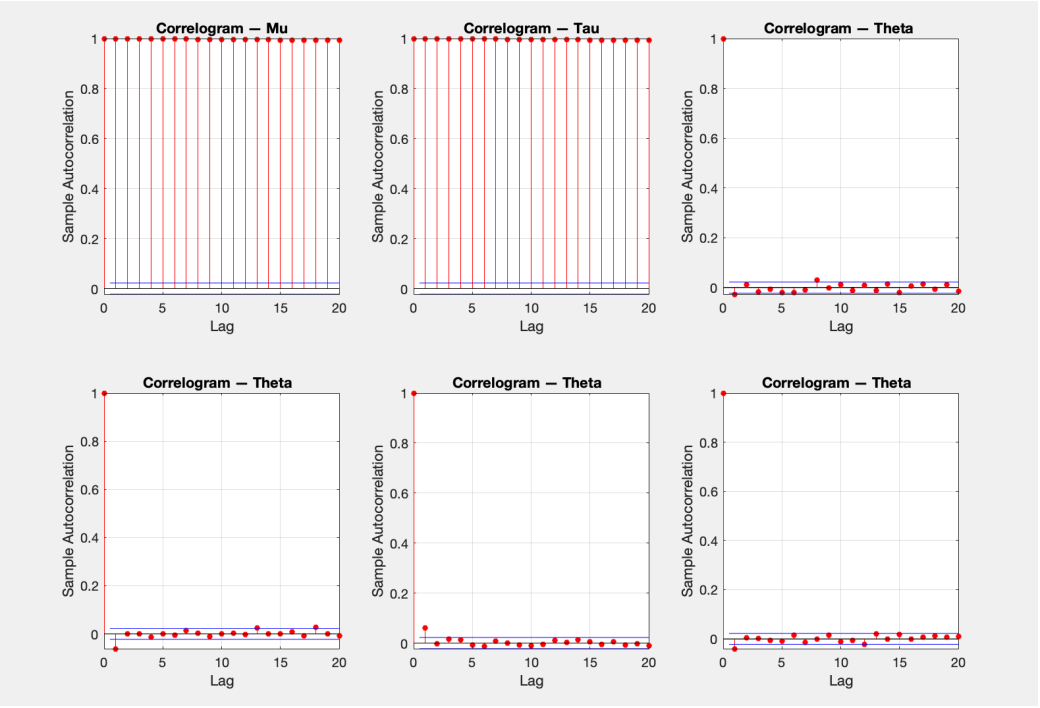


Figure 4.5. Non-convergent Correlograms. Low, near-zero correlation for the component parameters suggest the MCMC sampler behaved as desired for this level of the hierarchy, but high correlation for the hyperparameters confirms that each chain was stuck and not sampling appropriately.

Notwithstanding the obvious convergence issues, the results for the posterior component means, θ, were not dissimilar from the EDRPM's results for most of the simulation runs. However, the function means, μ, were orders of magnitude different. Occasionally, a posterior μ was conceptually irrational, such as a negative failure rate implying failure before operation. The posterior estimates from the simulation iteration reviewed in Figures 4.3–4.5 are summarized in Table 4.3.

It was suspected that the order of magnitude of the data was contributing to the errors encountered, specifically due to the scale of the standard errors of the component distributions. While the means were on the order of 10^{-6} (due to units of failures/MHours), the variances were the square of that order of magnitude, or 10^{-12}, which in practice became 10^{-13}. According to the *Stan User's Guide* [33], floating-point rounding permits 16-digits of accuracy, which encompassed the range of posterior estimates even with the

Table 4.3. Posterior Estimates from Non-convergent Simulation

Unweighted, $\overline{C}$ = (1,1,1,1)		
Mean	CDRPM Result	EDRPM Result
θ_1	9.30×10^{-6}	9.2766×10^{-6}
θ_2	5.20×10^{-6}	5.2020×10^{-6}
θ_3	3.00×10^{-6}	3.0312×10^{-6}
θ_4	4.60×10^{-6}	4.6058×10^{-6}
μ	-2.24×10^{-1}	5.5289×10^{-6}

All parameters have units of failures/MHours. These estimates are illustrative of numerous simulation attempts that did not converge. While the θ posteriors of the CDRPM were often similar, the hyperparameter posteriors were orders of magnitude different and the CDRPM estimation of μ was occasionally irrational.

nonsensical (unmixed) results. That is, even when the difference between a function-level mean, μ, and component standard error, V_c was 12 digits (from 10^{-1} to 10^{-13}), there was still sufficient accuracy for the MCMC random walk to make an appropriate estimate of the component-level distributions based on the evidence. This offered an explanation as to why the CDRPM component-level parameters trended with the EDRPM results, while the function-level hyperparameters did not. Therefore, it was speculated that if the sampling could begin closer to the true posterior hyperparameters, that is, at least the same relative order of magnitude, the MCMC would be more likely to achieve convergence and the CDRPM would produce the same results as the EDRPM.

The *Stan Reference Manual* [35] stated that the default seed selection for the MCMC algorithm was randomly generated from a $uniform(0.14, 7.4)$ distribution for variables that were declared to be constrained as positive. This confirmed that the posterior estimates were orders of magnitude smaller than their initially presumed values. When the seed was declared in the sampling initiation through a user option, specifically as 1×10^{-6}, the simulations were no more successful than the random seed runs. It was speculated that the errors were likely due to the very small initial step size required to characterize the HB, which was not an option for user specification. Therefore, in a final attempt to apply the CDRPM to the EDRPM example system, the data was modified for the tool, rather than the reverse.

Based on the known property of how the normal distribution scales, that given $X \sim N(\mu, \sigma^2)$ and c where $c \in \mathbb{R}$, $cX \sim N(c\mu, c^2\sigma^2)$, the scale of the data in Table 4.1 was modified for the means to have a base order of 10^0. This modification affected the results shown in Table 4.2 in a known, predictable manner, specifically only changing the order of magnitude, similar to how a unit change from failures/hours to failures/MHours would. After this data manipulation, convergence was achieved on the first attempt and yielded the posterior means shown in Table 4.4. Of note, the scaling relationship utilized for data in this example is only valid for the normal distribution. If other data of a similar order of magnitude has a non-normal distribution, a different scaling relationship may be required to manipulate it to a manageable level for the selected computing environment. Thus, the scaling is not a global solution but sufficient and acceptable for the specific purpose of comparing the CDRPM to the EDRPM results. Also, the scaling did not irrefutably confirm the speculated cause of errors encountered, but it increased the confidence with which the rationale was asserted.

The CDRPM results were re-scaled to their original order of magnitude by applying the reciprocal of the initial modifier and are shown in Table 4.4. Results from the CDRPM and EDRPM were quantitatively compared through percent errors, treating the EDRPM results as the "actual" values due to their deterministic nature. The greatest error was 0.58%, meeting the successful replication criteria of discrepancies not to exceed 2% of EDRPM results.

Table 4.4. Unweighted Results Comparison

Unweighted, $\bar{C}$ = (1,1,1,1)			
Mean	CDRPM Result	EDRPM Result	% Error
θ_1	9.2751×10^{-6}	9.2766×10^{-6}	-0.02
θ_2	5.1952×10^{-6}	5.2020×10^{-6}	-0.13
θ_3	3.0488×10^{-6}	3.0312×10^{-6}	0.58
θ_4	4.6058×10^{-6}	4.6058×10^{-6}	0.0
μ	5.5060×10^{-6}	5.5289×10^{-6}	-0.41

After data scale modification, the CDRPM approach to determining HB model posteriors yielded results that matched those produced by the EDRPM.

The unweighted comparison demonstrated successful posterior estimate replication, indicating the MCMC sampling approach to evaluating an HB model is reasonable and the stochastic simulation incurs a very low accuracy penalty. Evaluation of the unweighted case satisfied the objectives of assessing a different hierarchical arrangement (two-tier instead of three) founded on a different type of data (failure rate instead of success ratio) with different data distributions (all normal distributions). However, the concept of frequency weighting was one of the significant ways the EDRPM modified HB equations for a normal hierarchy found in [24]. Therefore, the CDRPM had to demonstrate replication of weighted EDRPM results before claiming it met the intent of being a generalized implementation.

4.3 Comparison to Weighted Cases

Evaluating weighted cases provided the opportunity to see if the frequency weighting implementation method selected in Chapter 3, that is, by type and not within type, reflected the closed-form implementation in [13]. The two weighted cases for the abstract system in [13] are referred by this work as "moderately weighted" ("case 2" in Table 4.2) and "heavily weighted" ("case 3" in Table 4.2).

4.3.1 Moderately Weighted, $\bar{C} = (1, 1, 2, 5)$

The moderately weighted case, where $\bar{C}$ = (1,1,2,5), was evaluated first. The same scaling technique was applied to achieve convergence to the posterior means shown in Table 4.5. As with the results of Table 4.4 for $\bar{C} = (1, 1, 1, 1)$, posterior means from both methods were quantitatively compared through a percent error calculation. All the posterior estimates met the error tolerance except one; the error for component type 3's mean, θ_3, exceeded the acceptance threshold by 0.19%. Otherwise, the errors were small and could reasonably be attributed to accuracy lost from using a stochastic method opposed to a deterministic one. Notwithstanding, the overall results were considered inclusive because the threshold of how many distributions had to meet the 2%-acceptability was never established, though a preponderance seemed a reasonable caveat.

Table 4.5. Moderately Weighted Results Comparison

Moderately Weighted, $\bar{C}$ = (1,1,2,5)			
Mean	CDRPM Result	EDRPM Result	% Error
θ_1	9.2440×10^{-6}	9.2723×10^{-6}	-0.31
θ_2	5.1953×10^{-6}	5.1997×10^{-6}	-0.08
θ_3	3.0890×10^{-6}	3.0227×10^{-6}	2.19
θ_4	4.6028×10^{-6}	4.6015×10^{-6}	0.03
μ	4.8495×10^{-6}	4.8359×10^{-6}	0.28

One CDRPM posterior distribution's result exceeded the 2% error acceptance threshold but overall unexpected behavior led to the comparison being considered inclusive.

The results were additionally considered inconclusive because it was peculiar that the greatest error was at the component type level and not the function level. Since the frequency weighting implementation methods evaluated in Section 3.3.5 demonstrated the greatest influence on the function-level posterior parameters, that was where discrepancies due to conceptual errors of methodology implementation were expected to be greatest. However, it was potentially not a coincidence that the greatest error was for the second most weighted component and that somehow the dominate weighting of component type 4 over-influenced the net shrinkage that should have occurred within the hierarchy at the component level.

4.3.2 Heavily Weighted, $\bar{C} = (1, 1, 2, 50)$

The last case to assess from the EDRPM was the heavily weighted scenario, $\bar{C} = (1, 1, 2, 50)$, where component type 4 overwhelmingly dominated the historical likelihood of fulfillment. Even if such a frequency mismatch was unlikely to be encountered in a real world scenario, its extremity offered insight into methodology behavior which was expected to be commensurately exaggerated. The CDRPM results after data modification and MCMC convergence are shown in in Table 4.6. The heavily weighted results did not trend as well as the other cases; two component-level posteriors did meet the percent error tolerance. Interestingly, the component-level parameters maintained the same relative order of percent error. This suggested that errors noted in the moderately weighted case were systemically

related to the CDRPM implementation and not due to accuracy loss affiliated with a simulation.

Table 4.6. Heavily Weighted Results Comparison

Heavily Weighted, $\bar{C}$ = (1,1,2,50)			
Mean	CDRPM Result	EDRPM Result	% Error
θ_1	8.8213×10^{-6}	9.2711×10^{-6}	-4.85
θ_2	5.1434×10^{-6}	5.1965×10^{-6}	-1.02
θ_3	3.5211×10^{-6}	3.0202×10^{-6}	16.58
θ_4	4.6059×10^{-6}	4.6002×10^{-6}	0.12
μ	4.6547×10^{-6}	4.6393×10^{-6}	0.33

CDRPM results suggested the unsatisfactory replication of the EDRPM.

4.4 Error Investigation

The most likely source of error for the weighted cases was the CDRPM frequency weighting implementation since the magnitude of the error trended with the magnitude of the component occurrence disparity. The alternative frequency weighting implementation, "within type" replication, was reconsidered. However, the "by type" implementation was selected because the within type implementation had a greater influence on the component-level posteriors, which was occasionally obviously inappropriate. The results in Tables 4.5 and 4.6 suggested that the CDRPM implementation was already over-influencing component-level posteriors. Upon closer examination of the heavily weighted EDRPM results, however, it seemed that they did not demonstrate the shrinkage expected with one component type so dominate relative to the others. That the component type 1 mean, θ_1, only changed in the fourth significant digit, despite being the furthest from the mode determined by the component type 4 mean, θ_4, was particularly perplexing.

Shrinkage, discussed in Section 2.3.2, occurs at all levels of a HB model—from a hyperparameter towards its closest mode and from subordinate parameters towards the hyperparameter. It is the natural manifestation of relating the levels in the hierarchy. While the EDRPM function-level mean, μ, shrunk towards θ_4 in the heavily weighted case, the other

three component types' means shifted very little towards the now-smaller function mean. With this notion in mind, the trend of relatively little shrinkage from the EDRPM component posteriors was observed in the moderately weighted case as well; this is especially notable when the cases are reviewed side by side in Table 4.2. This observation was consistent with the recognition of weighted case component-level errors maintaining the same relative magnitude, but now that trend could be affiliated with a mathematical phenomenon of a hierarchical model and not necessarily a simulation-related error.

It was hypothesized that the authors of [13] made an inadvertent error during the application of the EDRPM to the generic function for illustration. The author of this work developed the script in Appendix E to implement the EDRPM as she understood it from [13]. The results were different from those in Table 4.2, which was partially expected because execution error was the hypothesized source of result discrepancy. However, the results did not conclusively indicate who made an error because it was just as likely for this author's execution to be faulted; they only confirmed the values in Table 4.2 should be further interrogated before quantitatively comparing to the CDRPM.

Bryan O'Halloran, one of the authors of the EDRPM, provided the original MATLAB script used to generate the results published in [13] to the author of this work. In short, there were two implementation errors noted in the original script. Details are provided in Appendix E. The findings were reviewed and discussed with O'Halloran; he agreed with the corrections suggested. Once the corrections were made, the EDRPM results from the original script exactly matched the script shown in Appendix E.

All cases were re-evaluated using the corrected EDRPM implementation and an updated comparison to the CDRPM yielded the results in Table 4.7. Of the 15 parameter comparisons, only one had an error in excess of 2%. In the heavily weighted case, component type 3's mean, θ_3, was 6.69% larger for the CDRPM than EDRPM, but this was an improvement to the previously calculated error or 16.58%; the error was less than half the initial error. The corrected unweighted case results were nearly identical to the initial results. The calculated errors for the weighted cases were all lower in the comparison to the corrected EDRPM than to the published EDRPM results. As a reminder, the purpose of the quantitative comparison was to determine whether or not the CDRPM, a stochastic method, could satisfactorily generate the same results as the EDRPM, a deterministic method, for a

hierarchy based on the normal distribution. Therefore, the percent errors are representative of a loss of accuracy from using a simulation-based method.

Table 4.7. Corrected Comparison of CDRPM and EDRPM

Unweighted, $\bar{C}$ = (1,1,1,1)			
Mean	CDRPM Result	Corrected EDRPM Result	% Error
θ_1	9.2751×10^{-6}	9.2725×10^{-6}	0.03
θ_2	5.1952×10^{-6}	5.2024×10^{-6}	-0.14
θ_3	3.0488×10^{-6}	3.0366×10^{-6}	0.40
θ_4	4.6058×10^{-6}	4.6068×10^{-6}	-0.02
μ	5.5060×10^{-6}	5.5296×10^{-6}	-0.43
Moderately Weighted, $\bar{C}$ = (1,1,2,5)			
Mean	CDRPM Result	Corrected EDRPM Result	% Error
θ_1	9.2440×10^{-6}	9.2418×10^{-6}	0.02
θ_2	5.1953×10^{-6}	5.1953×10^{-6}	0.00
θ_3	3.0890×10^{-6}	3.0474×10^{-6}	1.37
θ_4	4.6028×10^{-6}	4.6031×10^{-6}	-0.01
μ	4.84950×10^{-6}	4.8386×10^{-6}	0.22
Heavily Weighted, $\bar{C}$ = (1,1,2,50)			
Mean	CDRPM Result	Corrected EDRPM Result	% Error
θ_1	8.8213×10^{-6}	8.8320×10^{-6}	-0.12
θ_2	5.1434×10^{-6}	5.1442×10^{-6}	-0.02
θ_3	3.5211×10^{-6}	3.3004×10^{-6}	6.69
θ_4	4.6059×10^{-6}	4.6045×10^{-6}	0.03
μ	4.6547×10^{-6}	4.6445×10^{-6}	0.22

CDRPM results were generated by four chains with 6000 iterations each, a warm-up of 2000 iterations, maximum tree depth of 15, and adaptation delta of 0.99. EDRPM results were generated using the code in Appendix E.

Overall, the comparison exercise was considered a successful demonstration of the ability of the CDRPM to replicate the EDRPM posterior results for a normally-distributed hierarchy.

The error affiliated with θ_3 for $\bar{C} = (1, 1, 2, 50)$ was undesirable, but in the context of the extremity of the case could reasonably be considered acceptable. To reiterate the significance of the frequency vector, $\bar{C} = (1, 1, 2, 50)$ conveys that, based on precedence, component type 4 is used to fulfill the function 50 times more often than component types 1 or 2, and 25 times more often than component type 3. From equations 2.2 and 2.3, the frequency weights, which represent historical probabilities of selection, are approximately 1.85%, 1.85%, 3.7%, and 92.6%, respectively. Therefore, the rationale could be that when in very rare situations, a greater accuracy loss that favors the dominant component characterization is acceptable. Furthermore, the component-level accuracy loss is completely transparent to the ultimate outcome of the CDRPM unless the system reliability prediction fails to meet the requirement, causing the function of concern to be decomposed to the component-level for re-evaluation.

4.5 Chapter Summary

This chapter reviewed the successful comparison of posterior parameter estimates generated by the CDRPM to those generated by the EDRPM. Success was determined by percent errors between the two methods of less than 2%, indicating that use of the stochastic method incurred an acceptably low accuracy penalty relative to the deterministic method. One parameter did not meet the threshold in the heavily weighted case, exceeding the threshold by 4.69%. Such a discrepancy in an extremely rare scenario is recommended to be an acceptable accuracy loss, handled with consideration in the event the system reliability prediction is made explicitly with non-dominant distributions from a heavily weighted function. The comparison demonstrated that, while the CDRPM was created to handle non-normal distributions, it is still possible to apply the method to normal distributions, as well as with different types of reliability data. On the whole, this exercise presented reasonable support for the CDRPM as a generalized form of the EDRPM. Two follow-on questions arose. First, how does the CDRPM scale to a system using real success ratio data? Second, is one data type preferable to the other when multiple types are available? That is, if failure rate data in a normally-distributed hierarchy and success ratio data in an HB like Chapter 3 are each assessed by the CDRPM, are the conclusions consistent? Chapter 5 explores these questions.

CHAPTER 5:
Methodology Validation: Case Study for CDRPM

This chapter serves as validation of the CDRPM. Through the examples in Chapter 3 and Chapter 4, this work demonstrated that it is possible to apply the CDRPM to data from a range of components to establish functional-physical relationships in system reliability performance for abstract systems. This case study applies the CDRPM to a coherent, purposeful system, a generic launcher, using product data from published sources, and assesses the findings to determine the extent to which the CDRPM addresses the gap identified in Chapter 2; it evaluates if the "correct system," or in this case, correct methodology, was developed. Thus, the case study validates the CDRPM.

Additionally, Chapters 3 and 4 demonstrated the CDRPM has the flexibility to assess different types of data. This chapter explores the implication of data type selection on resultant conclusions by using two different data types, success ratios and failure rates, for the exact same products in the functional arrangement of the case study system. While time-based failure rate data with a normality assumption lends itself to assessment by the EDRPM, Chapter 4 demonstrated that CDRPM results are consistent with the EDRPM, provided there is not large disparity in the component type frequency of occurrence. Therefore, the CDRPM was applied to both data types for consistency in system reliability prediction-requirement comparison and presentation of findings, with a presumed negligible impact of methodology selection on data type evaluation. Success ratio data are fully assessed first, followed by failure rate data. This chapter concludes with a comparison of the assessments and postulates scenarios in which use of one type of failure data may be preferable over another.

5.1 Case Study System and Evaluation Process

The system assessed in this case study is a generalized representation of a projectile launcher for a non-specific application. The reliability requirements against which the CDRPM predictions are compared are not intended to reflect real world systems, thus the conclusions are illustrative only of the methodology application and not of actual system performance. The component data applied to the CDRPM, however, came directly

from published commercial sources, namely the *NPRD* [31] and *EPRD* [30]. Details of the launcher's functional arrangement and supporting data are provided in the following subsections, fulfilling the activities of CDRPM step 1.

5.1.1 Launcher Functions and Reliability Block Diagram

Whether a launcher is intended for rockets, aerial vehicles, missiles, or other projectiles, the broad purpose of the system is to impart energy to an object to make it airborne. In this highly simplified representation, the system must perform three functions to fulfill its purpose. Foremost, the system must impart sufficient energy of an appropriate type (electrical, mechanical, etc.) to the object of interest to project it from the launcher; this function is hereby referred to as *Convert Energy*. The system must receive energy from a source, assumed to be electrical in this case, and modify it as necessary for use by the remainder of the launcher components; this is referred to as *Condition Energy*. Additionally, the system must meaningfully relay the received energy input to the components generating energy output; this function is referred to as *Actuate Energy*.

A simplified RBD for the launcher is shown in Figure 5.1. The dark gray boxes with solid outlines represent the functions identified above. The *Actuate Energy* function is represented in the system three times to nominally illustrate a safety-related intent with an in-line function and a reliability-related intent with a pair of parallel functions. For synthesis, a physical component that could fulfill each function is shown in lighter gray with dotted outlines; this arrangement is an example of a design alternative or design configuration for the launcher.

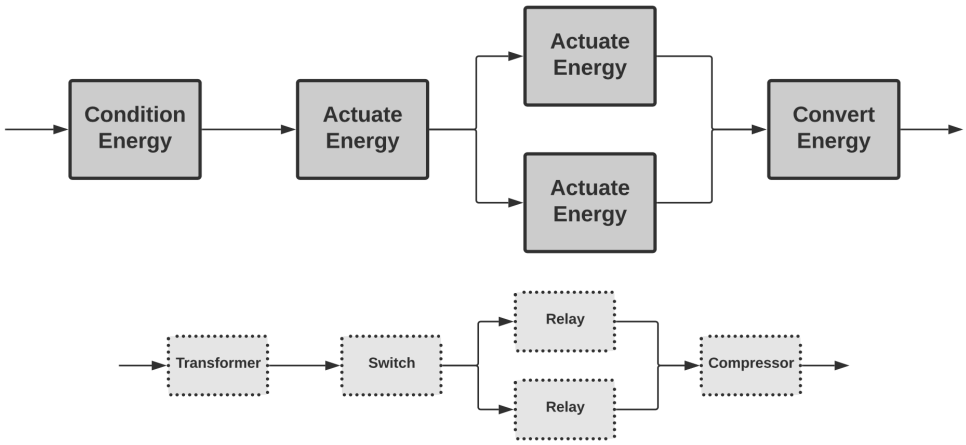


Figure 5.1. Launcher Reliability Block Diagram

For further simplification and brevity, the functions in Figure 5.1 were labeled as *Function A*, *Function B*, and *Function C* in the manner shown in Figure 5.2. The analysis in the following sections uses this nomenclature.

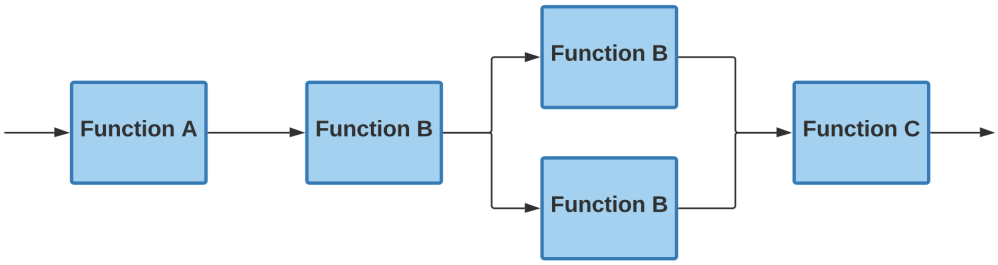


Figure 5.2. Generic Function Reliability Block Diagram

The system's reliability, according to its RBD, is represented by the expression below, equation 5.1. Representing the parallel *Function B* reliability terms individually, rather than with an exponent, is a deliberate measure related to execution of CDRPM step 4, addressed in Sections 5.2.2 and 5.3.2.

$$R_{sys} = R_A R_B (1 - (1 - R_B)(1 - R_B)) R_C \tag{5.1}$$

5.1.2 Launcher Reliability Requirement

For the purpose of this case study, the launcher system has a reliability requirement of 70%, alternatively stated as a 30% likelihood of failure during operation. The service life, or period over which its reliability will be assessed, is 1,200 hours. This time period is not the same as the system's MTBF, which, for failure rates on the scale of failures per millions of hours, leads to MTBFs generally on the scale of millions of hours. The rationale for a 1,200-hour evaluation was that continuous operation during daylight hours (nominally 12 hours a day) for a quarter calendar year (13 weeks) entails 1,092 hours of operation. Ideally, a quarterly preventative maintenance task would identify and address system issues prior to failure. Therefore, the benchmark for evaluation had to be at least 1,092 hours; an approximate week-long buffer seemed reasonable. The selection of a 70% reliability threshold was arbitrary. Neither of these requirement aspects are attempting to mimic real world launcher scenarios and any likeness is coincidental.

5.1.3 Supporting Data

A minimum of two component types were identified to potentially fulfill each function shown in Figure 5.1. The alternatives were not meant to be all inclusive since this exercise was focused on data implications and not the explicit predictions produced. Inverters and transformers were selected as potential physical solutions for the *Condition Energy* function with respective historical frequencies captured in the frequency vector $\bar{C} = (1, 2)$. Circuit breakers, switches, and relays were selected as potential physical solutions for the *Actuate Energy* function with frequencies assigned as $\bar{C} = (1, 2, 3)$. Lastly, fans, compressors, and pumps were selected for the *Convert Energy* function with frequencies assigned as $\bar{C} = (1, 2, 1)$. Physical solutions are henceforth only referred to as component types of their parent functions with a reference number (the j index presented in Chapter 3) to avoid specific conclusions from being drawn about the parts' characteristics during this case study.

The *EPRD* [30] and *NPRD* [31] provided all the product evidence. Both publications are arranged in a similar manner and their use of the word "part" is synonymous with this work's use of "product." Failure rate data for a part, such as a fan, is presented on three levels, referred to as "summarized," "roll-up," and "summary," from least to most aggregated, respectively. The summarized level represents "actual data from each data source" [30], [31]. These failure rates are given in units of failures/MHours and assumed

normally distributed with a standard deviation of 1.5. Roll-up entries are geometric means of summarized entries calculated on the basis of quality level (commercial, military, or industrial) or application environment (e.g., ground, airborne, spaceflight). Summaries are geometric means of the roll-ups, representing the entire body of data for the given part. Success ratios are only provided at the summarized level, but aggregated levels are easily calculated through arithmetic sums of the number of failures and population size tested for each summarized entry of interest.

At least three product evidence data points were selected for each component type identified. The complete data set used in this case study is shown in Appendix F. The following conditions were observed during data selection. These conditions were established to facilitate the case study comparison of data type implication while recognizing they induced a non-zero influence on the representativeness of the resultant reliability predictions. Since the case study launcher was for a non-specific application, prediction consistency was prioritized over prediction accuracy.

1. Only data with non-zero failures were selected. If none of the tested parts failed, the failure rate reported was qualified as "less than." However, there is no way to capture the qualification in the HB model. If only failure rate data were used for a system prediction, this aspect could subjectively be captured by recognizing the resultant prediction would likely be a conservative estimate since actual failure rates were likely lower than those used. This approach would have confounded the prediction comparison to that determined by success ratio data. Therefore, discrete failure rates were selected to facilitate comparison with discrete success ratios.
2. Only data that had a self-consistent population reported was used. For example, some parts details did not provide any population information, while others reported a population less than the number of reported failures. It is possible a component failed, was repaired, and was subsequently retested, repeatedly, but without that insight explicitly available, the data was excluded from consideration.
3. The highest level of data that met conditions (1) and (2) was used as product evidence. While this could have meant multiple products, quality levels, and test environments were represented in the data, they were still a subcategorization of the component type and therefore consistent with the hierarchical intent.

5.2 Assessment Using Success Ratios

Evaluation of each launcher function using success ratio data used same HB model developed in Chapter 3. For concision, as well as to focus on the process of elimination of unreliable candidate solutions, the posterior results of each function's HB model are presented summarily; interim results and convergence illustrations for the parameters of interest are not shown. That is, Figures 5.3 through 5.5 were produced by executing CDRPM steps 2 and 3 through code similar to that shown in Appendices B and C for each function in Figure 5.2.

5.2.1 Function-Level Characterization with Success Ratio Data

The posterior beta distribution parameters affiliated with *Function A*, supported by two components with a frequency vector of $\bar{C} = (1, 2)$, are shown in Table 5.1. The pdfs for these distributions are shown in Figure 5.3. The general shape of the two component distributions conveys that they are relatively similar and so the encompassing function distribution is also qualitatively quite similar with an unremarkable frequency weighting influence.

Table 5.1. Success Ratio Function A Posterior Results

Posterior Distribution Parameters		
Distribution	α	β
Function A	0.8126	0.4658
Comp Type A1	0.9284	0.2515
Comp Type A2	1.1633	0.2066

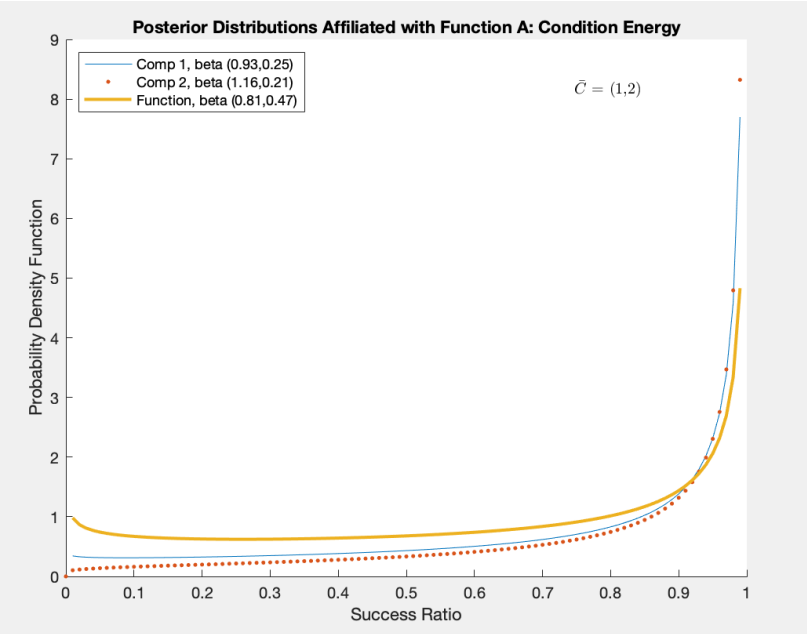


Figure 5.3. Success Ratio Function A Posterior Distribution. The MCMC execution for the graphed results was specified by the following parameters: two chains, 6000 iterations/chain, 2000 iteration warm-up, no thinning, max tree depth = 10, adaptation delta = 0.95.

The posterior beta distribution parameters affiliated with *Function B*, supported by three components with a frequency vector of $\bar{C} = (1, 2, 3)$, are shown in Table 5.2. The pdfs for these distributions are shown in Figure 5.4. While the three component distributions were similar, the slightly higher probability of a lower success ratio from component type B3 coupled with highest frequency ($c_3 = 3$) yielded a similar but noticeably different function posterior.

Table 5.2. Success Ratio Function B Posterior Results

Posterior Distribution Parameters		
Distribution	α	β
Function B	1.0417	0.5427
Comp Type B1	1.6712	0.3416
Comp Type B2	1.1749	0.2484
Comp Type B3	1.2387	0.3211

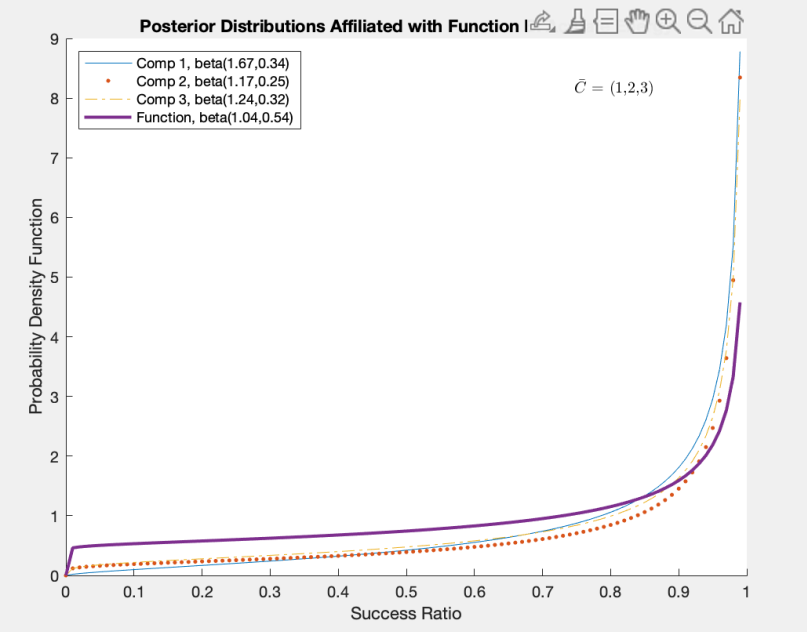


Figure 5.4. Success Ratio Function B Posterior Distribution. The MCMC execution for the graphed results was specified by the following parameters: two chains, 6000 iterations/chain, 2000 iteration warm-up, no thinning, max tree depth = 10, adaptation delta = 0.95.

The posterior beta distribution parameters affiliated with *Function C*, supported by three components with a frequency vector of $\bar{C} = (1, 2, 1)$, are shown in Table 5.3. The pdfs for these distributions are shown in Figure 5.5. The most notable aspect of this function's analysis was that distributions for component types C1 and C2 were nearly identical.

Table 5.3. Success Ratio Function C Posterior Results

Posterior Distribution Parameters		
Distribution	α	β
Function C	0.9025	0.5022
Comp Type C1	1.4132	0.3500
Comp Type C2	1.4311	0.3489
Comp Type C3	1.6024	0.3114

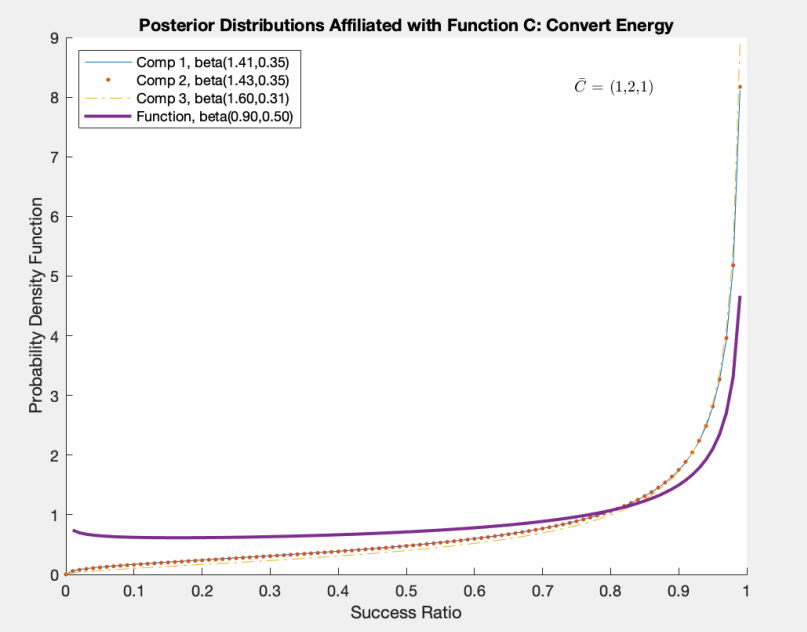


Figure 5.5. Success Ratio Function C Posterior Distribution. The MCMC execution for the graphed results was specified by the following parameters: three chains, 6000 iterations/chain, 2000 iteration warm-up, no thinning, max tree depth = 12, adaptation delta = 0.95.

5.2.2 System-Level Characterization with Success Ratio Data

In accordance with step 4 of the CDRPM, the function-level posterior reliability distributions were aggregated by representing each with an MCS and combining the respective sample iterations according to Equation 5.1. As a reminder, each sample from a function represents

a unique instantiation of that function. While *Function B* appears three times in the RBD, each appearance could be executed by a different component affiliated with *Function B*. Therefore, three separate MCSs of *Function B* were conducted, one for each appearance, to prevent one instantiation from over-influencing the system iteration's result.

Consistent with the presentation of Figure 3.14, Figure 5.6 shows the results of 10,000-sample MCSs of each function-level distribution and their combination to determine the system-level reliability prediction distribution. The higher density of low success ratios at the system-level, shown in the bottom right quadrant of Figure 5.6, reiterates the significance of considering each function's reliability in the context of the system RBD.

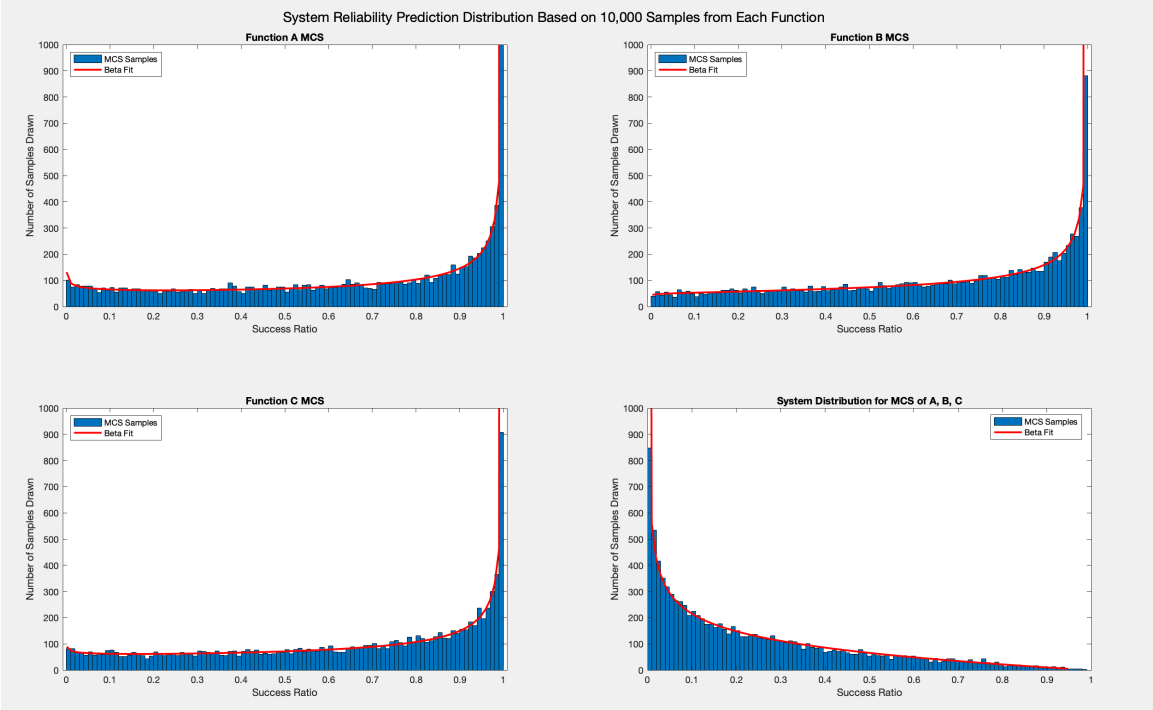


Figure 5.6. Success Ratio System Reliability Prediction Distribution. The prediction was based on a 10,000-sample MCS of *Functions A, B,* and *C* for each appearance in the RBD, binned into 100 intervals of 1% each, fitted with an appropriate distribution shown in red.

5.2.3 Identification of Unreliable Solutions with Success Ratio Data

The fifth step of the CDRPM commenced with calculating the likelihood of meeting the launcher's 70% reliability requirement using the cdfs of the function-level posterior reliability distributions and of the fitted system-level reliability distribution. These results are shown in Table 5.4. After qualitatively assessing the system-level distribution in Figure 5.6, the likelihood of the launcher meeting its reliability requirement based on a physical configuration randomly selected from the design alternatives is unsurprisingly low, specifically 4.9%.

Table 5.4. Likelihood of Meeting Reliability Requirement Based on Success Ratio Data

Likelihood of Meeting 70% Reliability	
Level Assessed	% Likelihood
System	4.9
Function A	51.5
Function B	53.2
Function C	51.8

Of the three functions, *Function A* had the lowest likelihood of meeting the requirement, so it was decomposed into its component types first. The MCSs for component type A1 and component type A2 were substituted for that of *Function A* in Equation 5.1, resulting in two new system-level reliability prediction distributions. The likelihood calculations for meeting the system requirement were repeated using the fitted system-level distributions for a configuration of *Function B* and *Function C* with either component type A1 or component type A2. These results are shown in Table 5.5. The likelihood with component type A2 was marginally better, that is, by approximately 2%, so component type A1 was eliminated from consideration. However, a 10.8% likelihood of meeting the requirement was still unacceptably low. Therefore, *Function C*, the second least-likely function to meet the requirement, was decomposed next.

Table 5.5. Success Ratio Likelihoods with Function A Components

Likelihood of Meeting 70% Reliability	
Component Assessed	% Likelihood
Comp Type A1	8.5
Comp Type A2	10.5

Requirement Likelihoods with *Function A* Components Using Success Ratio Data. Function-level posterior distributions represented *Function B* and *Function C*.

The MCSs for component types C1, C2, and C3 were substituted for that of *Function C* in equation 5.1. *Function B* was still represented by three independent MCSs at the function level. Since component type A1 was eliminated, the MCS for component type A2 now singularly represented *Function A*. The updated likelihoods of meeting the system requirement based on the fitted distributions of the new configurations are shown in Table 5.6. Since the posterior distributions for component types C1 and C2 were nearly identical, noted in the discussion of Figure 5.5, it followed that the component types lead to nearly equal likelihoods of meeting the system reliability requirement.

Table 5.6. Success Ratio Likelihoods with Function C Components

Likelihood of Meeting 70% Reliability	
Component Assessed	% Likelihood
Comp Type C1	16.5
Comp Type C2	16.7
Comp Type C3	18.4

Requirement Likelihoods with *Function C* Components Using Success Ratio Data. The posterior distribution for component type A2 represented *Function A* while the function-level posterior distribution represented *Function B*.

While a 2% improvement justified elimination of component type A1 from consideration, the same argument was not applied to *Function C* in the interest of preserving design alternatives. After all, the fundamental intent of the CDRPM was to eliminate unreliable component solutions and permit other performance metrics to determine selection of the optimal physical architecture, not to select the most reliable component solution. Rather than eliminate both C1 and C2 from further consideration for an improvement of less than 2%, all component types were retained as *Function C*. Further likelihood improvements were assessed through a decomposition of *Function B*.

The three *Function B* instantiations in Equation 5.1 were replaced by three MCSs of component types B1, B2, and B3, one a type time, to determine a new system prediction distribution. That is, all of the *Function B* distributions were replaced by iterations of component type B1's distribution for a system prediction based on component type A2, all component types of *Function C*, and component type B1. Similar to how *Function B* was consciously replicated independently, a number of permutations could have been evaluated for configurations utilizing different component types affiliated with *Function B* executing the different series and parallel *Function B* elements in the system. The granularity of such an analysis was speculated to be insignificant to the broader conclusions sought by this case study, but could be of use in a different scenario. The likelihoods to meet the system requirement based on component type A2, *Function C*, and one of *Function B*'s component types fulfilling all the *Function B* tasks, are shown in Table 5.7.

Table 5.7. Success Ratio Likelihoods with Function B Components

Likelihood of Meeting 70% Reliability	
Component Assessed	% Likelihood
Comp Type B1	23.0
Comp Type B2	23.7
Comp Type B3	20.8

Requirement Likelihoods with *Function B* Components Using Success Ratio Data. The posterior distribution for component type A2 represented *Function A* while the function-level posterior distribution represented *Function C*.

Recognizing the magnitude of the difference was relatively small—slightly greater than 2%, component types B1 and B2 were deemed more desirable than B3 due to their greater likelihood to yield a system that meets its reliability requirement. Once component type B3 was eliminated from further consideration, *Function B* was reassessed to reflect a random selection from component type B1 or B2. Rather than re-initiate CDRPM steps 2 and 3 for an HB model with only two component types, the MCSs for component types B1 and B2 were combined, from which 10,000 samples were randomly selected. This new population represented the *Function B* if it were to be executed by component types B1 or B2 only, and was coined *Function B'*. Note that by not re-analyzing the HB model, there was still some influence of component type B3 on the posterior distributions of the other two components used to determine *Function B'*. This influence was considered insignificant for the purposes of determining relative reliability contribution to the system-level prediction.

Assessment of the fitted system reliability prediction distribution generated by combining the reliability distributions for component type A2, *Function B'*, and *Function C*, yielded a 23.4% likelihood of meeting the system requirement. A histogram of the system reliability prediction with the fitted beta distribution used to facilitate the requirement likelihood calculation is shown in Figure 5.7.

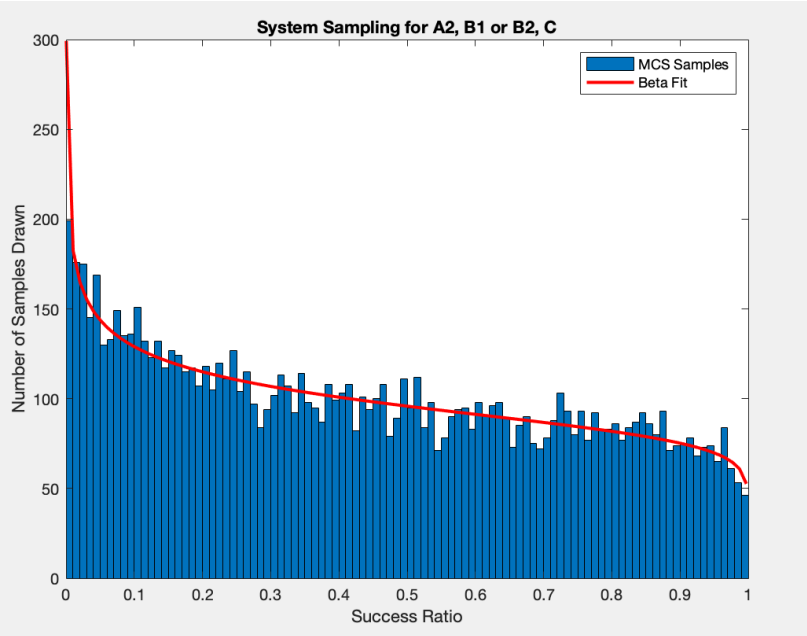


Figure 5.7. Success Ratio System Reliability Prediction Distribution, Semi-Final. The prediction was based on the posterior distributions for component type A2, *Function B'*, and *Function C*.

Out of curiosity, the component type substitution for *Function B* was re-assessed on the premise that *Function C* had been down-selected to component type C3, only. In this scenario, the likelihoods affiliated with each *Function B* component type were greater, but the component types retained their relative rank. The results are shown in Table 5.8, and the histograms for each combination are shown in Figure 5.8. They affirmed that retaining solutions to be inclusive in one functional area may influence the absolute system-level assessment, but should not affect the decisions within another functional area. Subsequently, the analysis for eliminating B3, thus a system comprised of component type A2, *Function B'*, and component type C3, resulted in a 38.7% likelihood of meeting the reliability requirement.

Table 5.8. Reassessment of Success Ratio Likelihoods with Function B Components

Likelihood of Meeting 70% Reliability	
Component Assessed	% Likelihood
Comp Type B1	38.4
Comp Type B2	39.4
Comp Type B3	34.9

Requirement Likelihoods with *Function B* Components Using Success Ratio Data, Reassessed. The posterior distribution for component type A2 represented *Function A* while the posterior distribution for component type C3 represented *Function C*.

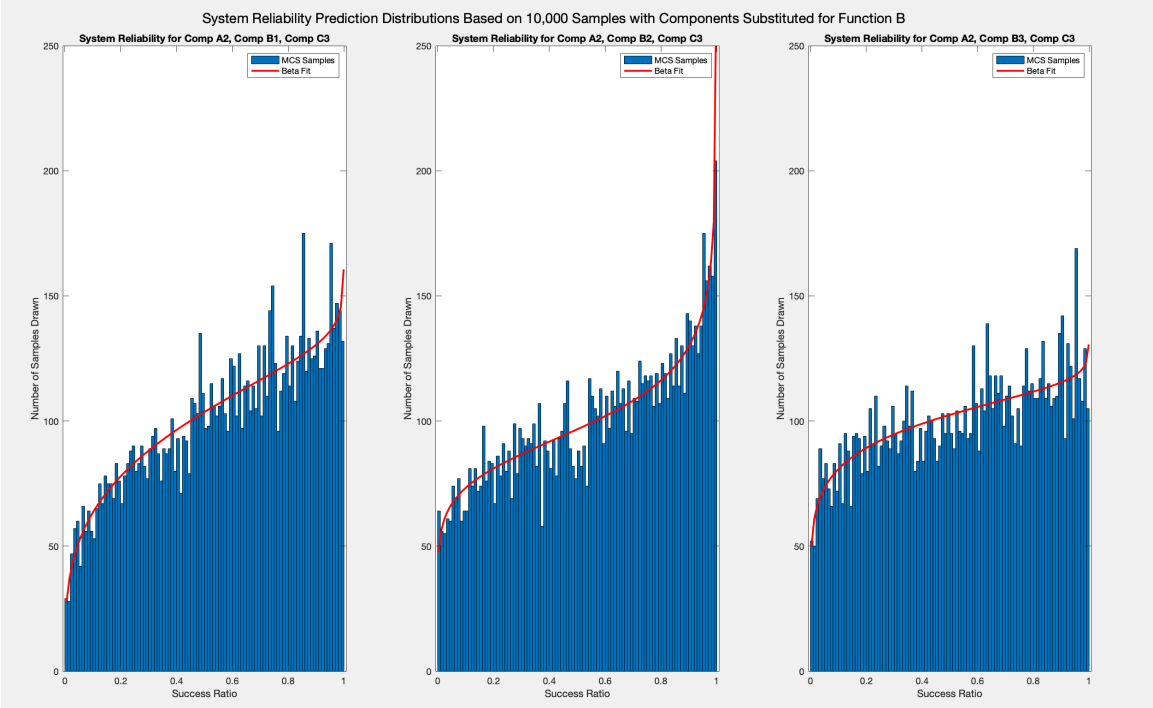


Figure 5.8. Success Ratio System Prediction, Circle Back to *Function B*. This analysis assumed *Function C* was down-selected to component type C3 prior to decomposing *Function B* into its component types. Component type B3 was still unfavorable.

The risk posture of the engineer determines when the identification of unreliable candidate solutions concludes. In this case, the most reliable design solution, comprised of component types A2, B2, and C3, only has a 39.4% likelihood of meeting the system-level reliability requirement of 70%, and a 38.7% if additional decision flexibility is retained for *Function B*. It is important the likelihood of meeting the requirement is not confused with the system's reliability, that is, the likelihood of the system operating as intended. Further, there is still a significant amount of decision space to influence the specific design configuration and, accordingly, the actual system reliability. For example, component type C3 is generally representative of a pump in this case study; it should be widely accepted that there are many different kinds of pumps with a wide range of performance characteristics. The CDRPM is not intended to prescribe a particular pump, only to illustrate that for the affiliated function, random selection of a pump is more likely to yield a system with an acceptable reliability than random selection of a compressor or fan. Therefore, for the sake of this case study, an approximately 40% chance of meeting a system level requirement based on the success ratio data for the indicated candidate solutions is deemed acceptable. The system reliability prediction distribution for the final configuration assessed in this case study, using component type A2, *Function B'*, and component type C3, resulting in a 38.7% likelihood of meeting a 70% reliability requirement, is shown in Figure 5.9

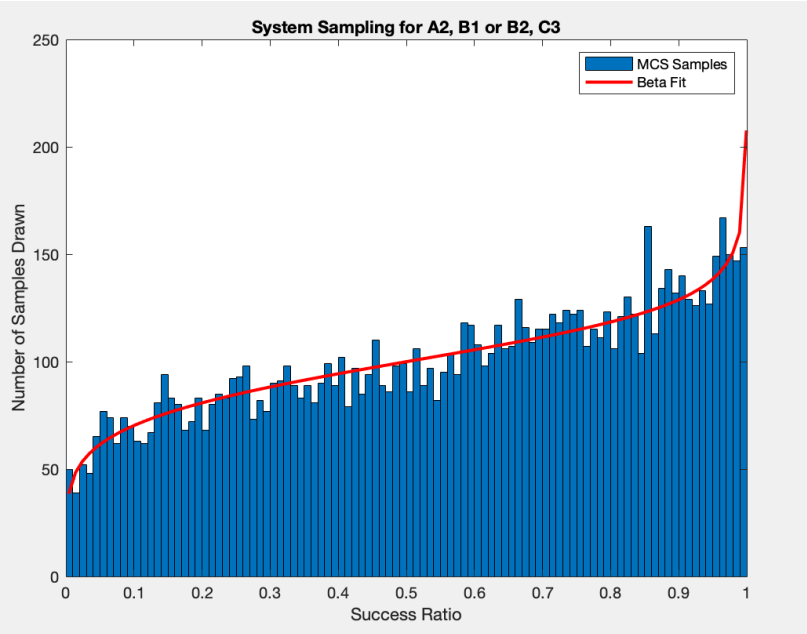


Figure 5.9. Success Ratio System Reliability Prediction Distribution, Final. After component type B3 was eliminated in the circle-back analysis, the final configuration recommended based on the success ratio data was a system utilizing component types A2, either B1 or B2, and C3.

5.3 Assessment Using Time-Based Failure Rates

Evaluation of each launcher function using failure rate data used the same HB model developed to facilitate the comparison with the EDRPM in Chapter 4. As with the presentation of findings in Section 5.2, the posterior results are presented summarily. While different characters were used to represent normal distribution parameters of the different levels in the hierarchy in Chapter 4, such as μ for the function-level normal distribution and θ for the component-level, all means are represented as μ and all standard deviations as σ in this case study.

5.3.1 Function-Level Characterization with Failure Rate Data

The posterior normal distribution parameters affiliated with *Function A*, supported by two components with a frequency vector of $\bar{C} = (1, 2)$, are shown in Table 5.9. The pdfs for these distributions are shown in Figure 5.10. While the tails of the component-level

distributions illustrate it would be extremely unlikely to encounter a negative failure rate, the encompassing function-level distribution shows the distinct possibility of a negative failure rate. Accounting for this behavior is addressed in the MCS representation of *Function A*, discussed in Section 5.3.2.

Table 5.9. Failure Rate Function A Posterior Results

Posterior Distribution Parameters		
Distribution	μ (Fail/MHours)	σ (Fail/MHours)
Function A	6.8478	3.1812
Comp Type A1	6.2911	0.7500
Comp Type A2	8.4818	0.4500

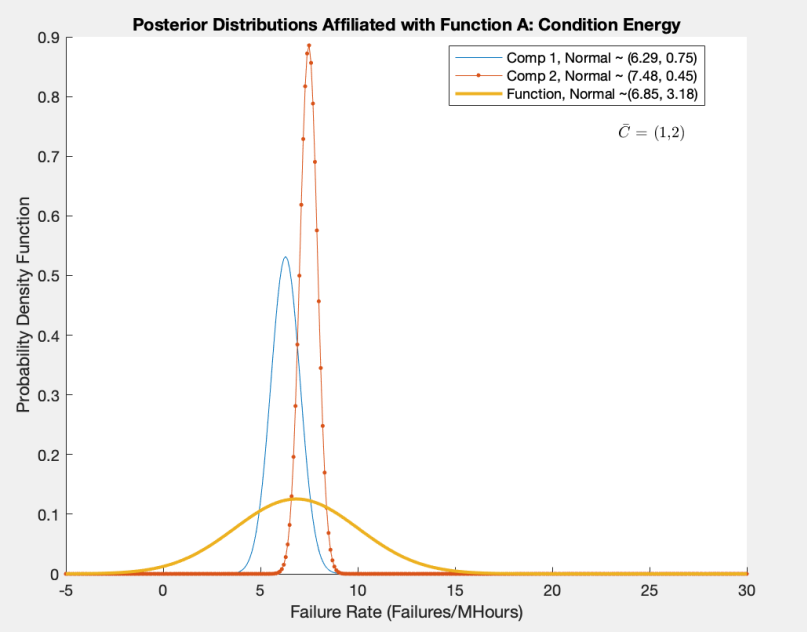


Figure 5.10. Failure Rate Function A Posterior Distribution. The MCMC execution for the graphed results was specified by the following parameters: three chains, 6000 iterations/chain, 3000 iteration warm-up, no thinning, max tree depth = 12, adaptation delta = 0.90.

The posterior normal distribution parameters affiliated with *Function B*, supported by three components with a frequency vector of $\bar{C} = (1, 2, 3)$, are shown in Table 5.10. The pdfs for these distributions are shown in Figure 5.11. The function-level posterior showed an even greater likelihood of encountering a negative failure rate than *Function A*, as well as for component type B2. Accounting for this behavior is addressed in Section 5.3.2.

Table 5.10. Failure Rate Function B Posterior Results

Posterior Distribution Parameters		
Distribution	μ (Fail/MHours)	σ (Fail/MHours)
Function B	15.0566	13.6223
Comp Type B1	15.2490	0.2500
Comp Type B2	3.5588	0.7500
Comp Type B3	23.0351	0.4500

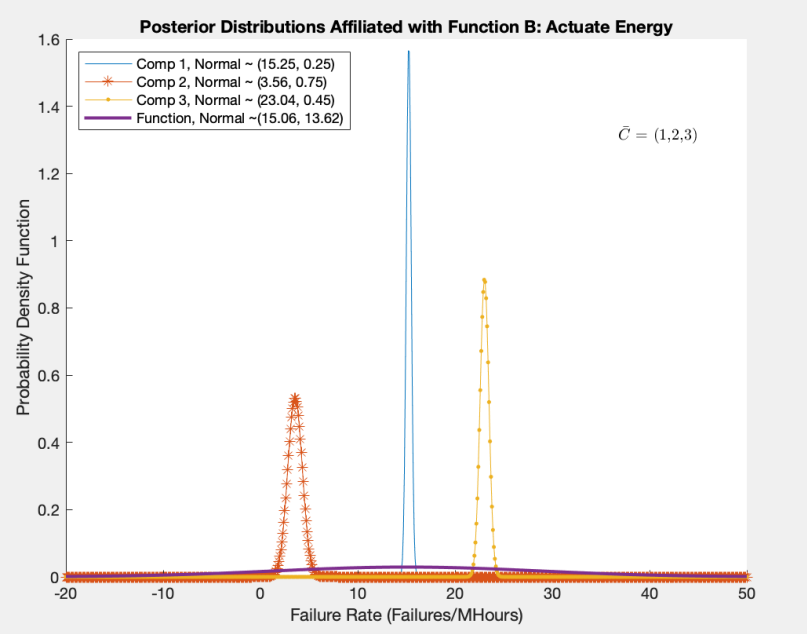


Figure 5.11. Failure Rate Function B Posterior Distribution. The MCMC execution for the graphed results was specified by the following parameters: three chains, 6000 iterations/chain, 3000 iteration warm-up, no thinning, max tree depth = 15, adaptation delta = 0.95.

The posterior normal distribution parameters affiliated with *Function C*, supported by three components with a frequency vector of $\bar{C} = (1, 2, 1)$, are shown in Table 5.11. The pdfs for these distributions are shown in Figure 5.12. As with the success ratio data, the posterior distributions for component types C1 and C2 were extremely similar. The potential for negative failure rates was noted at the function level, as well.

Table 5.11. Failure Rate Function C Posterior Results

Posterior Distribution Parameters		
Distribution	μ (Fail/MHours)	σ (Fail/MHours)
Function C	8.5610	10.5525
Comp Type C1	6.0473	0.5625
Comp Type C2	5.9051	0.4500
Comp Type C3	16.7972	0.2500

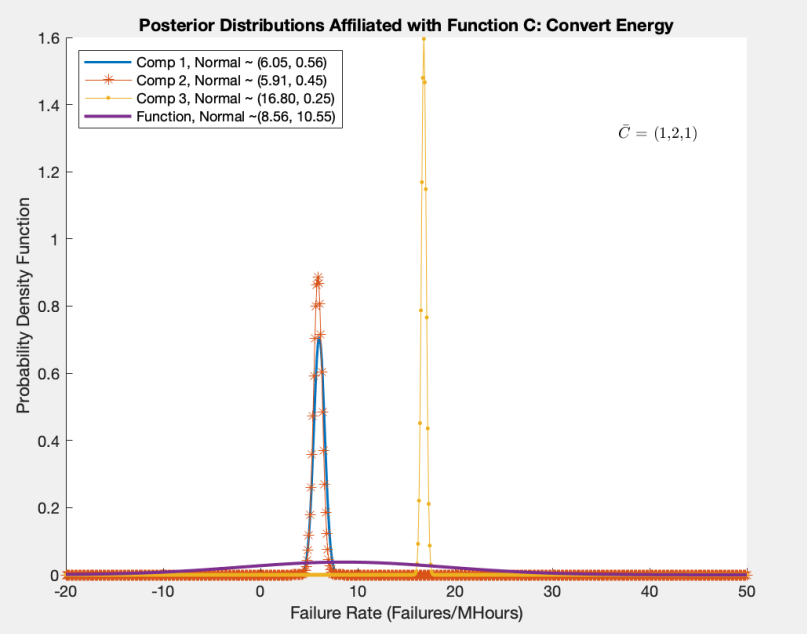


Figure 5.12. Failure Rate Function C Posterior Distribution. The MCMC execution for the graphed results was specified by the following parameters: three chains, 6000 iterations/chain, 3000 iteration warm-up, no thinning, max tree depth = 12, adaptation delta = 0.95.

5.3.2 System-Level Characterization with Failure Rate Data

Rather than performing MCSs of the function-level reliability distributions like in Section 5.2.1, MCSs were performed for the function-level failure rate distributions. Next, a reliability value was calculated for each sample in a function's simulation according to the Equation 2.8, producing 10,000 representations of the function's reliability, effectively creating the posterior reliability distribution for the function. These reliability distributions were then combined in the same manner discussed in Section 5.2.2 to create the system-level reliability prediction distribution.

The potential to randomly sample a negative failure rate was present for each of the function-level failure rate posteriors. The posterior normal distributions needed to be truncated to positive values while ensuring precisely 10,000 samples were available for subsequent element-wise operations. This was done by first performing a 10,000-sample MCS of the original posterior distribution. Then the negative rates were identified and replaced with the results from a new MCS of the original parameters. This cycle was repeated as many times as necessary until there were 10,000 positive failure rates sampled from the posterior distribution. Figure 5.13 shows a comparison of the positive samples in the original MCS in blue and positive samples used for reliability calculations for *Function A* in a semi-transparent yellow, thus the samples common to both sample populations appear green. The code and histogram comparisons for all of the case study functions are contained in Appendix F.

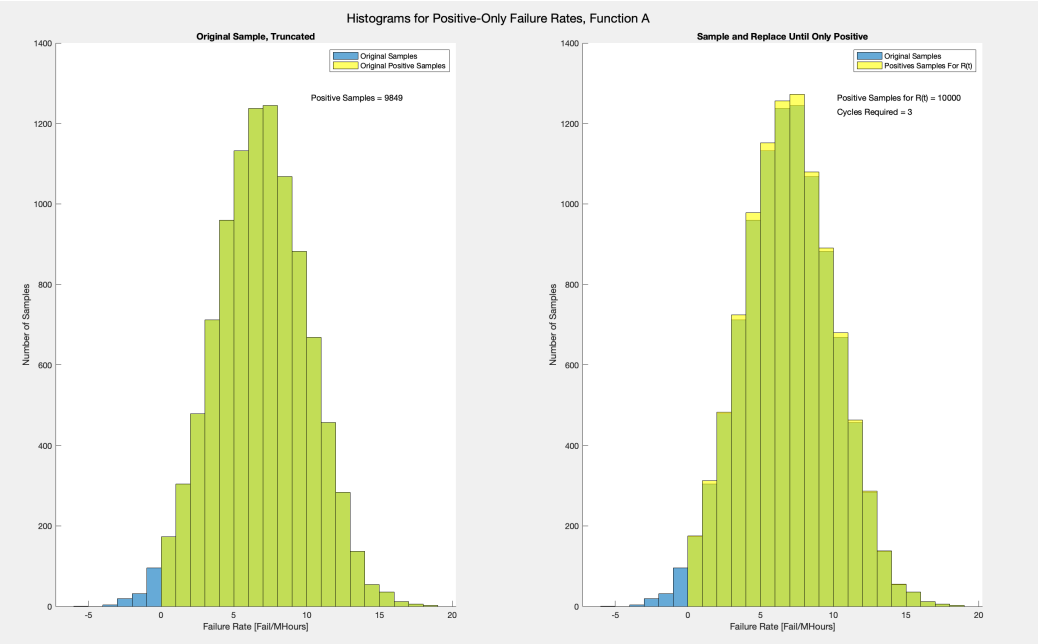


Figure 5.13. Positive Failure Rate Histograms for *Function A*. Original samples are shown in blue while positive samples available for subsequent analyses are shown in yellow. Green indicates the overlap of the two classifications, that is, original (blue) positive samples used in subsequent analyses (yellow). Strictly yellow regions were generated during the additional sampling cycles; they were not part of the original sample. It took three cycles of sampling and replacing negative failure rates to achieve a 10,000-sample population of positive failure rates for this function.

The positive-only samples induced a right-skew to the failure rate population descriptive statistics (e.g., mean and median), which was unavoidable due to truncation of negative values. Greater failure rates affiliated with the right skew would result in artificially lower reliability predictions; this was qualitatively kept in mind during follow-on analysis.

For each positive sampled failure rate, a corresponding reliability was calculated according to Equation 2.8 using a time of 1,200 hours, the benchmark for the reliability requirement. The resultant distributions of calculated reliabilities formed the function-level distributions combined according to the RBD in Figure 5.2 to form a system-level prediction. *Function B* was represented by three independent iterations, including complete reiteration of the cycle process for positive-only samples. Since the period of evaluation, 1,200 hours, was

so small relative to the failure rate order of magnitude, that is, evaluated per millions of hours, the reliability distributions were unsurprisingly high. The system-level reliability prediction distribution is shown in Figure 5.14. Note, this distribution was generated with the right-skewed failure rates that yielded lower calculated reliabilities; the "true" reliability predictions would be slightly greater than those shown in the histogram.

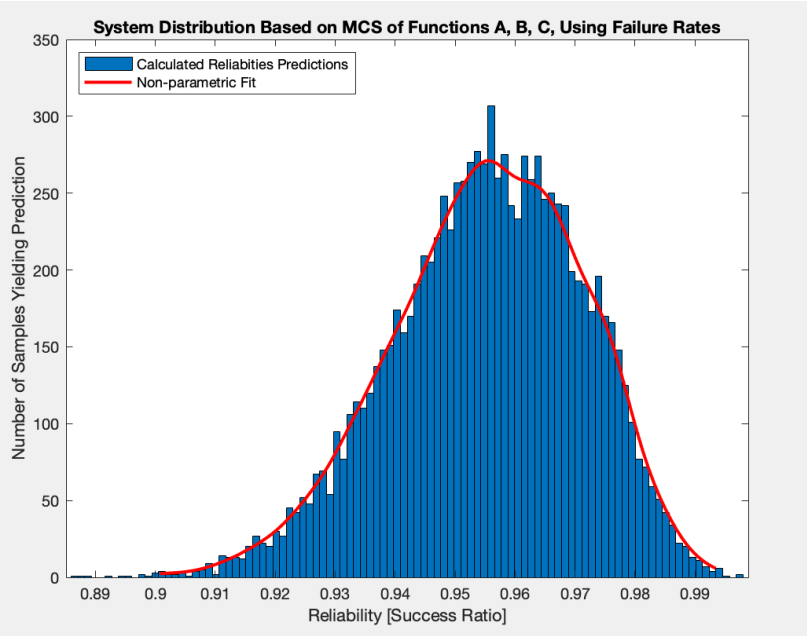


Figure 5.14. Predicted System Reliability Distribution Using Failure Rates. The prediction was based on 10,000 positive samples from MCSs of each function's posterior failure rate distribution, which were converted to a reliability based on a exponential reliability relationship evaluated at a time of 1,2000 hours. The results are binned into 100 intervals and fitted with a non-parametric distribution.

5.3.3 Identification of Unreliable Solutions with Failure Rate Data

A qualitative assessment of Figure 5.14 readily indicated that there was a very high (arguably certain) likelihood that the system would meet the 70% reliability requirement. Notwithstanding, the complement of the cdf evaluated at the requirement was calculated for CDRPM step 5. As expected, all functions, and thus the resultant system, had a 100% chance of meeting the requirement. These high (certain) likelihoods of meeting the requirement

indicate that all component types are reasonable design alternatives, as far as system reliability in concerned, and none are eliminated from consideration moving forward in the AoA.

5.4 Comparison of Results for Each Data Type

The conclusions in sections 5.2.3 and 5.3.3 clearly demonstrate that the type of failure data and associated assumptions influence the outcome of the CDRPM. Using success ratios provided a greater discrimination between functions and component types, leading some candidate solutions to be eliminated from further consideration. Assumptions affiliated with success ratios include an inherent success ratio at the product level that is represented by a binomial distribution, while all component- and function-level distributions are represented by beta distributions informed by vague beta and gamma priors. Since the success ratio was considered an inherent property, the evaluation time period for the requirement was inconsequential. Conversely, using failure rate data indicated that all candidate solutions were highly likely to produce a design configuration that met the system reliability requirement for the indicated time frame. This scenario assumed failure rates were normally distributed at all levels of the HB, informed by a vague uniform prior, and that product reliability could be expressed by an exponential reliability relationship, which assumes a constant failure rate.

It is worth repeating that only data with non-zero failures were selected for use in this case study. Products with zero failures have a success ratio of one under all conditions but their failure rates are not similarly expressed absolutely; failure rates are qualified as less than the inverse of the time period tested. This means that introduction of a time-based aspect will not affect the reliability predictions affiliated with those products when characterized by success ratios, but failure rates will eventually become significant over time. As mentioned in Section 5.3.3, the time period assessed in this case study was so small the reliability reduction implication would not have manifested anyway, but using zero-failure parts would have increased the reliability predictions assessed by success ratios, very likely leading to less disparity in the overall conclusions from each data type. That is, using zero-failure products would have made the success ratio configurations more likely to be reliable, but would have had very little effect on increasing the reliability predictions assessed by failure rates over the relatively short time of 1,200 hours.

There are a number of considerations to keep in mind when selecting a failure data type and overall approach for using the CDRPM. First of all, the requirement evaluation period may be much larger, over which the exponential reliability relationship in equation 2.8 will become more significant. Alternatively, the exponential reliability assumption may be wholly inappropriate and use of a log-normal or Weibull relationship may yield greater discrimination even for the relatively short time frame. Related, a reasonable representation of operational life may not exist for an emerging technology and it may be undesirable to assume one. If using a failure rate data type, all rates must have the same units, such as failures per hours which used in this case study, failures per cycles, etc., and this type of data may not be consistently available for all products of interest. However, all products can be combined using success ratios because the measure of operation is embedded in the context of its usage.

There are a number of logistical considerations, as well. Success ratio data availability significantly contributed to the product-level selections that were made to inform the HB models. For the publications used, many part summaries were missing a population count or had inconsistent failure-to-population counts, alluded to in Section 5.1.3. Additionally, [30] and [31] provide failure rates at various summary levels, but failure counts and populations are only provided at the lowest part level so aggregations had to be manually summed. While not a complex task, this is much more time-intensive than collecting failure rate data and may become an unreasonable investment when scaling the method to larger systems. All of these issues could probably be rectified by referencing the actual data sources, not just general reference collections like [30] and [31].

5.5 Chapter Summary

This chapter used the CDRPM to form system reliability predictions for a generic launcher using either success ratio data or failure rate data for a number of products that have historically been used to fulfill the functions affiliated with the launcher system. The exact same products were used in each application. As with any decision tool or prediction method, the nature of the data and affiliated assumptions affected the resultant conclusions. Success ratios yielded a more conservative prediction, underestimating reliability relative to the prediction made using failure rates for the designated evaluation period. Notwithstanding, the system reliability prediction in both scenarios was made without assuming a physical

architecture by establishing functional-physical reliability relationships, capturing data uncertainty through prediction distributions rather than point estimates, and facilitated comparison to a reliability requirement to inform the likelihood of meeting said requirement. While the notion of a "valid" method is subjectively assessed, the case study scenarios conformed to the tenets identified in the methodology gap assessment of Section 1.2. Therefore, the case study increased the confidence with which the CDRPM is asserted as a valid method to form system reliability predictions during conceptual design.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 6:
Summary

This chapter reiterates the main conclusions drawn from the preceding chapters, and highlights more nuanced considerations for CDRPM application. A few conclusions that may be drawn without appreciation for the assumptions and limitations inherent in the methodology development are explicitly identified and cautioned against. Limitations based on CDRPM application and execution experiences during this research are discussed thereafter. The final section, future work, suggests opportunities to correct or better understand said limitations to enhance the overall utility of the CDRPM.

6.1 Conclusions

The consolidated body of conclusions from each chapter is as follows. Design changes are more costly the later they are made in a system's life cycle so it desirable to identify physical configurations that are unlikely to meet reliability requirements as soon as possible (Chapter 1). Many reliability prediction methods exist, but only one methodology reviewed, the EDRPM, was suitable for application during conceptual design, specifically before a physical architecture has been selected (Chapter 2). However, the EDRPM relies upon an assumption of normally-distributed data assessed by an HB model through a closed-form solution to identify unreliable functions (Chapter 2). The CDRPM was developed as a generalized form of the EDRPM to facilitate functional-physical relationships using data characterized by non-normal distributions in an HB model assessed by MCMC simulation (Chapter 3). Furthermore, MCSs of the functional posterior distributions can be combined to yield integrated system-level reliability prediction distributions (Chapter 3). Use of the CDRPM on normally-distributed data, which could otherwise be assessed deterministically with the EDRPM, does not significantly affect the precision of the posterior parameter determination (Chapter 4). The CDRPM is flexible enough to assess multiple failure data types, such as success ratios and failure rates, in different HB arrangements, like two-tier and three-tier models (Chapters 4 and 5). However, as with any decision tool, the assumptions inherent with each data type will consequently affect the conclusions from the assessment so the engineer should deliberately select a data type commensurate with her intentions for

using the CDRPM's prediction (Chapter 5).

A number of corollary conclusions are presented for consideration, as well. First, while the HB models assessed in this work were for conjugate distributions, there is nothing about the approach, that is, MCMC determination of posterior distributions, that precludes application to non-conjugate hierarchies. Therefore, the CDRPM is versatile enough to assess any type of failure data once it is arranged in a functional-physical hierarchy. Second, solving an HB model by MCMC simulation will always incur some finite accuracy loss relative to a deterministic solution when available. While this loss is typically inconsequential, it appears to increase with greater disparity between frequency weights for a given function's frequency vector. If the greater accuracy loss is considered unacceptable, a solution could be to re-evaluate the classification of the component type groupings under their parent functions. Large frequency disparities may indicate that the component types are historically used to solve fundamentally different functions and should not be assessed together. Third, the conclusions from a within-function assessment of component types are independent of the conclusions from other functions. That is, the elimination of component types in one functional area does not influence the relative reliability of component types in another function. While it is recommended to assess the functions in terms of their component types in order of those functions least likely to meet a system reliability requirement, it is possible to evaluate the functions in a different order (e.g., in descending order of number of component types). Assessing the function with the greatest number of component types affords more opportunities to find reliability improvements, but if it is a highly reliable function, the assessment would incur the largest time investment possible with potentially little return. The primary motivation for the systemic process prescribed is efficiency and repeatability, particularly for larger hierarchies, to reduce waste of an "Easter egg hunt." However, if it is desirable to retain greater decision space across all functions, eliminating one of two candidate solutions, as done in the launcher case study may be counterproductive.

On the other hand, the following conclusions are cautioned against. First, while success ratios resulted in a lower, more conservative reliability estimate than failure rate data in the launcher case study, this relationship is not universally true. Products that completed testing without failure will always have a success ratio of one, but their failure rate will eventually become significant, regardless of the life expectancy distribution. Therefore, the time period over which reliability is assessed and the product's life expectancy

characterization are integral in determining which type of data will provide a conservative estimate. Second, the implications of "eliminate from consideration" may be overly harsh. If the engineer is insistent on a particular component type, the reliability prediction may indicate that higher fidelity failure information, such as selection of a particular quality level or operating environment, is necessary. Alternatively, an unreliable disposition can help inform a make-buy decision. A buy decision would not be strongly supported because the canvas of products for a component types was summarily deemed unreliable. However, the engineer must then assess whether or not she has the resources to design and produce a product substantially more reliable than the majority of her contemporaries before endorsing a make-decision. Third, if, at the conclusion of the all function-level decompositions, the resultant system likelihood is still unsatisfactory, it does not imply that the system is destined to fail to meet its requirement. The scenario is akin to cautiously pursuing a low likelihood of success, not flagrantly pursuing a certainty of failure, which, depending on the technology maturity and risk posture of designing organization, may be entirely reasonable.

6.2 Limitations

The biggest limitation of the CDRPM is data availability. Since the premise of the functional-physical relationships is that the functional posterior distribution represents a range of component types as possible physical instantiations, exponentially more data is needed to assess the system reliability prediction than if a single design solution were assessed. The aggregation at the system-level, through combination of the function MCSs in accordance with the RBD, is no more complex as long as the function-level distributions are not decomposed to the component level. This limitation hinders application to complicated systems with more directly than complex systems. A complicated system with many parts for broader functions requires a great deal of data for each candidate solution. A complex system with more advanced parts for fewer specialized functions entails less data. Of course, complex systems may also have many parts, incurring the data volume challenge. Otherwise, provided the data is consistently available at the product-level, the CDRPM can be applied to any type of system, whether mechanical, electrical, control, etc.

A related limitation is the concept of "state space explosion" where the computing system conducting the MCMC sampling essentially runs out of memory while attempting to store the transitional steps for posterior distribution characterization [21]. The exact

threshold at which this would occur, making an MCMC approach no longer viable, is unknown for the CDRPM. The more complex the functional posterior (i.e., multi-modal or non-parametric distributions supported by multiple component types), the greater the computational memory required, thus the greater the likelihood of state space explosion being realized. Fortunately, the CDRPM applications thus far leverage simple, uni-modal distributions, namely beta, gamma, binomial, and normal distributions, so the volume of data necessary remains the primary issue.

A final hindrance to adoption is the number of computing environments utilized for the execution in this work. It should be stressed that the workflow inefficiencies for transitioning between environments was minimal relative to the time spent in any one environment for execution of a particular CDRPM step. Furthermore, many of the transitions could probably be programmed for automation, particularly for data transfers, once the specific environments for each task were selected. It was not the intention of this work to prescribe specific computational tools for execution.

6.3 Future Work

The launcher case study was a relatively small real world system. To better explore the implications and limitations of data availability, applying the CDRPM to a larger system, that is, one with more functions and more component types, is recommended for future work. An assessment with more representative data, specifically that incorporates zero-failure products and is not limited to products with data consistently available in multiple forms, is also recommended. The representative data could be further constrained to the appropriate operating environment, or the hierarchy could be expanded to a fourth tier for operating conditions or manufacturing quality. For depth rather than breadth, it may be of interest to further evaluate the case study system with a sensitivity analysis to see when the reliability predictions from each data type intersect. A final future activity for consideration is to test the notion that a functional posterior distribution could capture the performance of an emerging technology not yet developed. A body of products could be assessed in a functional HB and then classic statistical comparison methods could be used to determine whether or not the emerging technology, once prototyped, could reasonably belong to the function's posterior distribution.

APPENDIX A:
Initial Stan Model

The code that follows delineates an HB model written for Stan, subsequently referenced in follow-on code as "example.stan." This initial model was intended as a proof of concept before implementing the framework for the full *Function A* data set of the example system. The basic aspects of a Stan model, programmed in this order, are the data structure to declare variables for the known evidence, parameters that are unknown and will be determined by the estimation, transformed parameters that are partially observed by the estimation, and the delineation of the hierarchy in a way that resembles that of Figure 3.2. It took approximately 50 lines of code to construct the framework for just the first two observations of component type 1, and the first observation of component type 2. It is referred to as a "hard-coded" model because each datum was explicitly declared; vectors and *for* loops were not utilized.

```
data {
  // Number of trials, Nij
  int N11;
  int N21;
  int N12;
  // Number of successes, nij
  int n11;
  int n21;
  int n12;
}

parameters {
  // Thetas - theta_ij per component
  real<lower=0, upper=1> theta11;
  real<lower=0, upper=1> theta21;
  real<lower=0, upper=1> theta12;

  // Component parameters - mu_j and k_j per component beta distribution
  real<lower=0, upper=1> mu1;
  real<lower=0> k1;
  real<lower=0, upper=1> mu2;
  real<lower=0> k2;

  // Function parameters - muA and kA for function beta distribution
  real<lower=0, upper=1> muA;
  real<lower=0> kA;
}
```

```
transformed parameters{
  // Beta parameters - a_j and b_j per beta distribution
  real<lower=0> a1;
  real<lower=0> b1;
  real<lower=0> a2;
  real<lower=0> b2;
  real<lower=0> aA;
  real<lower=0> bA;
  a1 = mu1*k1;
  b1 = (1-mu1)*k1;
  a2 = mu2*k2;
  b2 = (1-mu2)*k2;
  aA = muA*kA;
  bA = (1-muA)*kA;
}

model {
  muA ~ beta(1,1);
  kA ~ gamma(5,5);
  mu1 ~ beta(aA,bA);
  mu2 ~ beta(aA,bA);
  k1 ~ gamma(5,5);
  k2 ~ gamma(5,5);
  theta11 ~ beta(a1,b1);
  theta21 ~ beta(a1,b1);
  theta12 ~ beta(a2,b2);
  n11 ~ binomial(N11,theta11);
  n21 ~ binomial(N21,theta21);
  n12 ~ binomial(N12,theta12);
}
```

The model progressed to the third step of the CDRPM, posterior determination. This step was initially attempted with MATLAB and MATLABStan to minimize the number of software programs utilized while executing the methodology. MATLAB was preferable to R based on the author's familiarity with each. However, recurrent compilation errors during the execution of the initial model (example.stan, shown above) led to abandonment of MATLAB for R which had a more robust user community for troubleshooting this particular application. RStudio was the specific R platform used. The exact R code written for data assignment and MCMC sampling initiation to determine the HB posteriors in Stan follows.

```
library("rstan")
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
setwd("/Users/bethrajchel/Thesis")
```

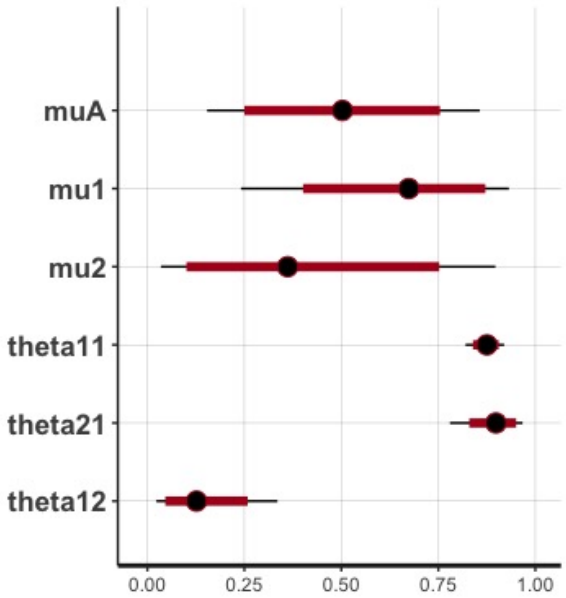
```
# Data assignment for declared data structure portion of Stan model
n11 <- 140
N11 <- 160
n21 <-36
N21 <- 40
n12 <- 2
N12 <- 16

data_list <- list(n11=n11, N11=N11, n21=n21, N21=N21, n12=n12, N12=N12)

# Compiling and producing posterior samples from the model.
stan_samples <- stan(file='example.stan', data = data_list)
```

The simulation yielded favorable results; results were considered favorable simply because the simulation ran without errors or warnings. Various results presentations were generated in R with the code below. The graphical outputs follow.

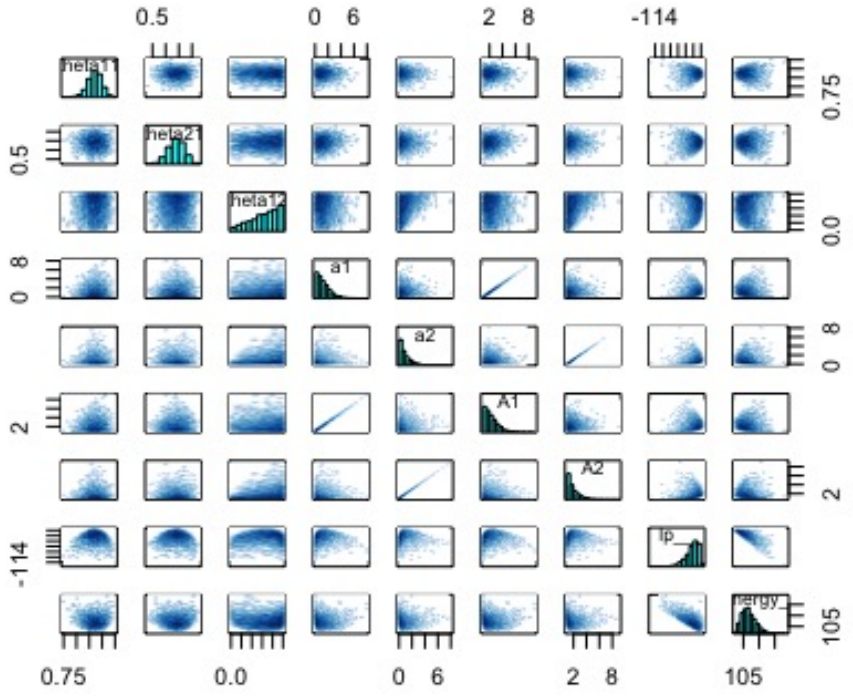
```
# print results to screen
stan_samples

# plot results with CI bars, CI level at 80%, outer level at 95%
plot(stan_samples, pars = c("muA", "mu1", "mu2", "theta11", "theta21", "theta12"))

# plot mutligraph
pairs(stan_samples)

# write to multidimension matrix
Samples_matrix <- as.matrix(stan_samples)
```

```
Inference for Stan model: 526b65a09a94f5b75e56ceba9efa2a3e.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

          mean se_mean   sd    2.5%     25%     50%     75%   97.5% n_eff Rhat
theta11   0.87    0.00 0.03    0.82    0.86    0.88    0.89    0.92  7438    1
theta21   0.89    0.00 0.05    0.78    0.86    0.90    0.93    0.97  7038    1
theta12   0.14    0.00 0.08    0.02    0.08    0.13    0.19    0.34  7339    1
mu1       0.65    0.00 0.18    0.24    0.54    0.67    0.79    0.93  6726    1
k1        1.18    0.01 0.46    0.47    0.84    1.11    1.46    2.20  6076    1
mu2       0.39    0.00 0.24    0.03    0.19    0.36    0.57    0.90  6035    1
k2        1.07    0.01 0.45    0.37    0.74    1.01    1.32    2.16  6311    1
muA       0.50    0.00 0.19    0.15    0.37    0.50    0.64    0.86  6195    1
kA        1.19    0.01 0.46    0.46    0.84    1.13    1.47    2.24  7033    1
a1        0.79    0.01 0.41    0.16    0.48    0.73    1.03    1.75  5827    1
b1        0.39    0.00 0.23    0.07    0.22    0.35    0.52    0.93  6390    1
a2        0.40    0.00 0.29    0.03    0.18    0.34    0.56    1.11  5910    1
b2        0.67    0.01 0.42    0.07    0.36    0.60    0.91    1.69  5580    1
aA        0.60    0.00 0.33    0.14    0.35    0.54    0.78    1.39  5381    1
bA        0.59    0.00 0.33    0.12    0.35    0.54    0.77    1.37  5875    1
lp__   -110.29    0.06 2.24 -115.55 -111.53 -109.97 -108.62 -106.96  1529    1
```

Output from Printing Results to Screen



Selected Results with Confidence Intervals from plot() Function



Selected Results in Pairwise Scatter Plot from pairs() Function

While this initial model demonstrated proof of concept for a reasonable hierarchical arrangement, it immediately demonstrated that the hard-coded approach was inflexible to changes in database size and highly likely to contain a transcription or subscript error. Since each “nij” and “Nij” were identified discretely and the product was explicitly assigned to a component type distribution, each new product would require a minimum of six additional lines; more component type classifications would incur additional lines as well.

For efficiency, vectors and indexed loops were attempted. Stan has a robust directory of documentation at https://mc-stan.org/users/documentation/ but the author was not able to identify a concrete example for indexed *for* loops at two levels of a three-tier hierarchy. Incorrectly established precedence resulted in compilation errors when simulation was initiated, such that even the initial hard-coded three-observation model could not be duplicated. This prompted the author to collaborate with Samuel Buttrey for the script shown in Appendix B. Some hard-coding was still required but the script alleviated the need to explicitly type each datum variable in the model and enabled data assignment by subscript indices.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B:
Model Writer Script

Samuel Buttrey, an Associate Professor in the Operations Research Department at NPS, assisted with the development of a script written in R to partially automate generation of an HB model as a data string, which significantly reduced the labor involved in defining larger models for Stan. That is, rather than having a Stan file (.stan) called by the stan() function in R, like shown in Appendix A, a model string containing the same information was called instead. Since the model writing script was written in R, the same environment used for CDRPM step 3, the script consolidates the activities of the second and third step into one program for model definition, data assignment, and simulation initiation. To reiterate, Stan is still called from R to execute the Bayesian estimation by MCMC sampling; R is not performing the sampling.

The script only partially automates model generation because some hard-coding was still required. The script is structured for four component types with an unlimited number of product observations of each type; the number of component types is one of the hard-coded aspects. The evidence is read in from a database in a comma separated value (.csv) format. Evidence is arranged by column with the explicit headers "Success_*j*" and "Trials_*j*." Prior distribution parameters are defined within the script. The model writing script in its entirety follows.

```
setwd("/Users/bethrajchel/Thesis")
#
# Start: top function level. Beta and gamma prior parameters for function level: A, B; S, R.
#
beta.params <- c(5,2)
gamma.params <- c(2, 2)
#
# Components priors informed by gamma distributions, defined by two parameters: g1 for S'_j, g2 for R'_j.
# These parameters are put into a data frame.
#
components <- data.frame (Num = 1:4, g1 = c(2, 2, 2, 2), g2 = c(2, 2, 2, 2))
#
# There can be multiple binomials for each component type; one binomial per product. Each binomial
# has an indicator of which component it comes from (j), plus a big N for number
# of trials, and a little n for number of successes, with n < N, each identified by index i.
#
evidence <- read.csv("Data_for_R.csv", header=TRUE)
n1 <- na.omit(evidence$Success_1)     # widgets as element i, success for j=1
N1 <- na.omit(evidence$Trials_1)      # widgets as element i, trials for j=1
I1 <- nrow(as.array(n1))              # index of i for number of widgets when j = 1
```

```
n2 <- na.omit(evidence$Success_2)
N2 <- na.omit(evidence$Trials_2)
I2 <- nrow(as.array(n2))
n3 <- na.omit(evidence$Success_3)
N3 <- na.omit(evidence$Trials_3)
I3 <- nrow(as.array(n3))
n4 <- na.omit(evidence$Success_4)
N4 <- na.omit(evidence$Trials_4)
I4 <- nrow(as.array(n4))
binoms <- data.frame (Num = rep (1:4, c(I1,I2,I3,I4)), n = c(n1,n2,n3,n4), N=c(N1,N2,N3,N4))
###########
theModelWriter <- function (beta, gamma, comp, binom) {
# The data lists the numbers of trials (Nij) and succeses (nij)
    binom <- binom[order (binom$Num),] # put in order by Num
# "run length encoding" (rle) tabulates runs in data. This call to sapply
# (inside unlist) generates a 1, 2, ..., number of trials sequence for
# each component, and we tack the component number onto the front.
    num.dex <- unique (binom$Num)
    num.dex.A <- c(num.dex, "A") # add A on the end
    if (any (binom$Num == "A")) stop ("'A' is not a valid component id")
    num.rle <- rle (binom$Num)
    trial.dex <- paste0 (binom$Num, "_",
        unlist (sapply (num.rle$lengths, function (t) 1:t)))
    n.dex <- paste0 ("n_", trial.dex)
    N.dex <- paste0 ("N_", trial.dex)
    data.string <- c("data {",
        paste0 ("int ", n.dex, ";"),
        paste0 ("int ", N.dex, ";"), "}")

# One theta for each trial, one mu and one k for each component,
# plus the global mu_A, k_A
    param.string <- c("parameters {",
        paste0 ("real<lower=0, upper=1> theta_", trial.dex, ";"),
        paste0 ("real<lower=0, upper=1> mu_", num.dex.A, ";"),
        paste0 ("real<lower=0> k_", num.dex.A, ";"), "}")

# One a and one b for each component, plus the global ones...
    trans.string  <- c("transformed parameters{",
        paste0 ("real<lower=0> a_", num.dex.A, ";"),
        paste0 ("real<lower=0> b_", num.dex.A, ";"),
# ...and then use the mu's and k's to construct those...
        paste0 ("a_", num.dex.A, "=mu_", num.dex.A, "*k_", num.dex.A, ";"),
        paste0 ("b_", num.dex.A, "=(1-mu_", num.dex.A, ")*k_",
                    num.dex.A, ";"), "}")

    model.string <- c("model {",
        "mu_A ~ beta(1,1);", "k_A ~ gamma(5,5);", # global
        paste0 ("mu_", num.dex, " ~ beta(a_A,b_A);"),
        paste0 ("k_", num.dex, " ~ gamma (", comp$g1, ",", comp$g2, ");"),
        paste0 ("theta_", trial.dex, " ~ beta (a_",
            binom$Num, ", b_", binom$Num, ");"),
        paste0 ("n_", trial.dex, " ~ binomial (N_", trial.dex,
                                    ", theta_", trial.dex, ");"), "}")

    final <- c (data.string, param.string, trans.string,
                model.string)
    return (list (model.string = paste (paste (final, collapse="\n"), "\n"),
        trial.dex = trial.dex))
}
# Use cat() here to make the new-lines print like new-lines
mod <- theModelWriter (beta, gamma, components, binoms)
cat (mod$model.string)
#
```

```
# Create data list with n_xx and N_xx names
#
nxx <- paste0 ("n_", mod$trial.dex, " = ", binoms$n)
Nxx <- paste0 ("N_", mod$trial.dex, " = ", binoms$N)
bigstring <- paste ("list (", paste (nxx, collapse = ","), ",",
                              paste (Nxx, collapse = ","), ")")
data.list <- eval (parse (text = bigstring))

library (rstan)
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
date ()
stan_samples <- stan(model_code = mod$model.string, data = data.list, iter=6000, chains=4, warmup =2000,
     control = list(adapt_delta=0.90))
date ()
```

The output of this script is the generated data string that represents the HB model passed to Stan, shown below. There are two minor notes on the subscript annotations in the model passed to Stan. First, an underscore was added to the ij convention to separate the indices for programming clarity, e.g., "i_j," because "113" could otherwise be interpreted as the eleventh observation of type 3 ($i = 11, j = 3$) or the first observation of type 13 ($i = 1, j = 13$) for larger models. Second, the variables generated by the string have subscripts ordered as "j_i,", which was inconsequential as long as the convention was maintained throughout the analysis.

```
data {
int n_1_1;
int n_1_2;
int n_1_3;
int n_1_4;
int n_1_5;
int n_1_6;
int n_1_7;
int n_1_8;
int n_1_9;
int n_1_10;
int n_2_1;
int n_2_2;
int n_2_3;
int n_2_4;
int n_2_5;
int n_2_6;
int n_2_7;
int n_2_8;
int n_2_9;
int n_2_10;
int n_3_1;
int n_3_2;
int n_3_3;
int n_3_4;
int n_3_5;
int n_4_1;
int n_4_2;
int n_4_3;
int n_4_4;
int n_4_5;
int n_4_6;
```

```
int n_4_7;
int N_1_1;
int N_1_2;
int N_1_3;
int N_1_4;
int N_1_5;
int N_1_6;
int N_1_7;
int N_1_8;
int N_1_9;
int N_1_10;
int N_2_1;
int N_2_2;
int N_2_3;
int N_2_4;
int N_2_5;
int N_2_6;
int N_2_7;
int N_2_8;
int N_2_9;
int N_2_10;
int N_3_1;
int N_3_2;
int N_3_3;
int N_3_4;
int N_3_5;
int N_4_1;
int N_4_2;
int N_4_3;
int N_4_4;
int N_4_5;
int N_4_6;
int N_4_7;
}
parameters {
real<lower=0, upper=1> theta_1_1;
real<lower=0, upper=1> theta_1_2;
real<lower=0, upper=1> theta_1_3;
real<lower=0, upper=1> theta_1_4;
real<lower=0, upper=1> theta_1_5;
real<lower=0, upper=1> theta_1_6;
real<lower=0, upper=1> theta_1_7;
real<lower=0, upper=1> theta_1_8;
real<lower=0, upper=1> theta_1_9;
real<lower=0, upper=1> theta_1_10;
real<lower=0, upper=1> theta_2_1;
real<lower=0, upper=1> theta_2_2;
real<lower=0, upper=1> theta_2_3;
real<lower=0, upper=1> theta_2_4;
real<lower=0, upper=1> theta_2_5;
real<lower=0, upper=1> theta_2_6;
real<lower=0, upper=1> theta_2_7;
real<lower=0, upper=1> theta_2_8;
real<lower=0, upper=1> theta_2_9;
real<lower=0, upper=1> theta_2_10;
real<lower=0, upper=1> theta_3_1;
real<lower=0, upper=1> theta_3_2;
real<lower=0, upper=1> theta_3_3;
real<lower=0, upper=1> theta_3_4;
real<lower=0, upper=1> theta_3_5;
real<lower=0, upper=1> theta_4_1;
real<lower=0, upper=1> theta_4_2;
real<lower=0, upper=1> theta_4_3;
real<lower=0, upper=1> theta_4_4;
```

```
real<lower=0, upper=1> theta_4_5;
real<lower=0, upper=1> theta_4_6;
real<lower=0, upper=1> theta_4_7;
real<lower=0, upper=1> mu_1;
real<lower=0, upper=1> mu_2;
real<lower=0, upper=1> mu_3;
real<lower=0, upper=1> mu_4;
real<lower=0, upper=1> mu_A;
real<lower=0> k_1;
real<lower=0> k_2;
real<lower=0> k_3;
real<lower=0> k_4;
real<lower=0> k_A;
}
transformed parameters{
real<lower=0> a_1;
real<lower=0> a_2;
real<lower=0> a_3;
real<lower=0> a_4;
real<lower=0> a_A;
real<lower=0> b_1;
real<lower=0> b_2;
real<lower=0> b_3;
real<lower=0> b_4;
real<lower=0> b_A;
a_1=mu_1*k_1;
a_2=mu_2*k_2;
a_3=mu_3*k_3;
a_4=mu_4*k_4;
a_A=mu_A*k_A;
b_1=(1-mu_1)*k_1;
b_2=(1-mu_2)*k_2;
b_3=(1-mu_3)*k_3;
b_4=(1-mu_4)*k_4;
b_A=(1-mu_A)*k_A;
}
model {
mu_A ~ beta(1,1);
k_A ~ gamma(5,5);
mu_1 ~ beta(a_A,b_A);
mu_2 ~ beta(a_A,b_A);
mu_3 ~ beta(a_A,b_A);
mu_4 ~ beta(a_A,b_A);
k_1 ~ gamma (2,2);
k_2 ~ gamma (2,2);
k_3 ~ gamma (2,2);
k_4 ~ gamma (2,2);
theta_1_1 ~ beta (a_1, b_1);
theta_1_2 ~ beta (a_1, b_1);
theta_1_3 ~ beta (a_1, b_1);
theta_1_4 ~ beta (a_1, b_1);
theta_1_5 ~ beta (a_1, b_1);
theta_1_6 ~ beta (a_1, b_1);
theta_1_7 ~ beta (a_1, b_1);
theta_1_8 ~ beta (a_1, b_1);
theta_1_9 ~ beta (a_1, b_1);
theta_1_10 ~ beta (a_1, b_1);
theta_2_1 ~ beta (a_2, b_2);
theta_2_2 ~ beta (a_2, b_2);
theta_2_3 ~ beta (a_2, b_2);
theta_2_4 ~ beta (a_2, b_2);
theta_2_5 ~ beta (a_2, b_2);
```

```
theta_2_6 ~ beta (a_2, b_2);
theta_2_7 ~ beta (a_2, b_2);
theta_2_8 ~ beta (a_2, b_2);
theta_2_9 ~ beta (a_2, b_2);
theta_2_10 ~ beta (a_2, b_2);
theta_3_1 ~ beta (a_3, b_3);
theta_3_2 ~ beta (a_3, b_3);
theta_3_3 ~ beta (a_3, b_3);
theta_3_4 ~ beta (a_3, b_3);
theta_3_5 ~ beta (a_3, b_3);
theta_4_1 ~ beta (a_4, b_4);
theta_4_2 ~ beta (a_4, b_4);
theta_4_3 ~ beta (a_4, b_4);
theta_4_4 ~ beta (a_4, b_4);
theta_4_5 ~ beta (a_4, b_4);
theta_4_6 ~ beta (a_4, b_4);
theta_4_7 ~ beta (a_4, b_4);
n_1_1 ~ binomial (N_1_1, theta_1_1);
n_1_2 ~ binomial (N_1_2, theta_1_2);
n_1_3 ~ binomial (N_1_3, theta_1_3);
n_1_4 ~ binomial (N_1_4, theta_1_4);
n_1_5 ~ binomial (N_1_5, theta_1_5);
n_1_6 ~ binomial (N_1_6, theta_1_6);
n_1_7 ~ binomial (N_1_7, theta_1_7);
n_1_8 ~ binomial (N_1_8, theta_1_8);
n_1_9 ~ binomial (N_1_9, theta_1_9);
n_1_10 ~ binomial (N_1_10, theta_1_10);
n_2_1 ~ binomial (N_2_1, theta_2_1);
n_2_2 ~ binomial (N_2_2, theta_2_2);
n_2_3 ~ binomial (N_2_3, theta_2_3);
n_2_4 ~ binomial (N_2_4, theta_2_4);
n_2_5 ~ binomial (N_2_5, theta_2_5);
n_2_6 ~ binomial (N_2_6, theta_2_6);
n_2_7 ~ binomial (N_2_7, theta_2_7);
n_2_8 ~ binomial (N_2_8, theta_2_8);
n_2_9 ~ binomial (N_2_9, theta_2_9);
n_2_10 ~ binomial (N_2_10, theta_2_10);
n_3_1 ~ binomial (N_3_1, theta_3_1);
n_3_2 ~ binomial (N_3_2, theta_3_2);
n_3_3 ~ binomial (N_3_3, theta_3_3);
n_3_4 ~ binomial (N_3_4, theta_3_4);
n_3_5 ~ binomial (N_3_5, theta_3_5);
n_4_1 ~ binomial (N_4_1, theta_4_1);
n_4_2 ~ binomial (N_4_2, theta_4_2);
n_4_3 ~ binomial (N_4_3, theta_4_3);
n_4_4 ~ binomial (N_4_4, theta_4_4);
n_4_5 ~ binomial (N_4_5, theta_4_5);
n_4_6 ~ binomial (N_4_6, theta_4_6);
n_4_7 ~ binomial (N_4_7, theta_4_7);
}
```

A complete discussion of viewing and evaluating the posteriors estimated by the MCMC sampling is provided in Appendix C.

APPENDIX C:
Complete Results for Function A Posterior Estimation

There are many ways to review and evaluate the posterior parameters for a HB model assessed by MCMC. Many software packages for Bayesian estimation have built-in tools and functions for such tasks. Some of the options available in R through the "bayesplot" and "rstan" libraries were utilized to review the full *Function A* estimation result. However, the majority of the analyses shown in this work were conducted in MATLAB as a matter of the author's programming and aesthetic preference. Her code and its affiliated output are provided in this Appendix.

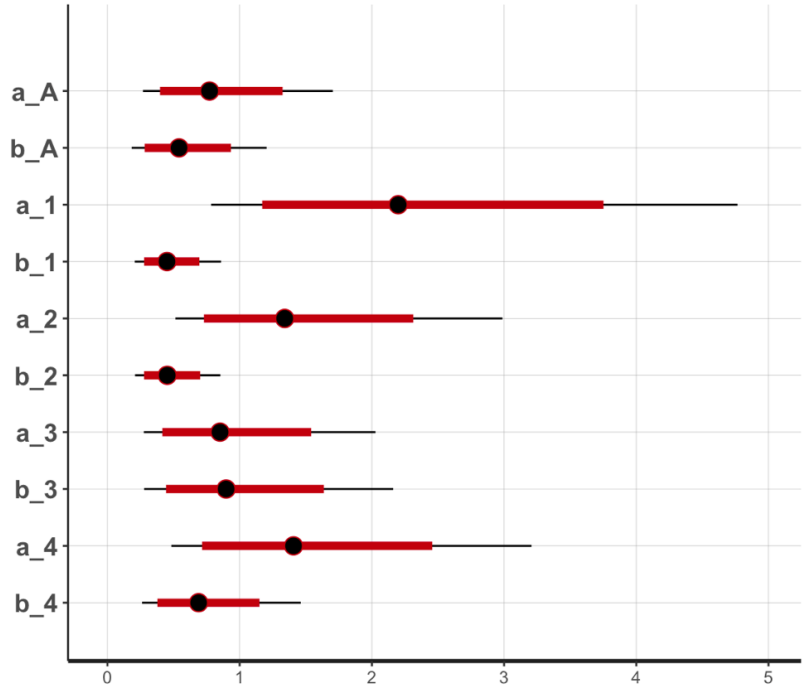
The R code used to print the MCMC results to the screen and conduct a preliminary assessment of convergence is shown below, followed by the generated displays. These three presentations, the (1) results list, (2) results plotted within confidence intervals (CIs), and (3) pairwise scatter plot, were the same ones used for the assessment of the initial hard-coded model in Appendix A. The pairwise scatter plot is shown in Figure 3.4 and is not repeated here.

```
# print results to screen
stan_samples
# plot results with CI bars
plot(stan_samples, pars = c("mu_A", "k_A", "a_1", "b_1", "a_2", "b_2", "a_3", "b_3","a_4", "b_4"),
     ci_level = 0.95, outer_level = 0.999)

# plot mutligraph
pairs(stan_samples, pars = c("mu_A", "k_A", "a_1", "b_1", "a_2", "b_2", "a_3", "b_3","a_4", "b_4"))

# write to multidimension to martix
Samples_matrix <- as.matrix(stan_samples)

# write to file
write.csv(Samples_matrix, file = "A_Data_Stan_.csv")
```

```
Inference for Stan model: 2270a50e38abcbf8f50c1c5e2c88f212.
4 chains, each with iter=6000; warmup=2000; thin=1;
post-warmup draws per chain=4000, total post-warmup draws=16000.

              mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff Rhat
theta_1_1     0.87    0.00 0.03     0.82     0.86     0.88     0.89     0.92 26551    1
theta_1_2     0.90    0.00 0.05     0.79     0.87     0.90     0.93     0.97 22141    1
theta_1_3     0.95    0.00 0.03     0.89     0.94     0.96     0.97     0.99 24230    1
```

theta_1_4	0.94	0.00	0.03	0.88	0.93	0.95	0.97	0.99	23333	1
theta_1_5	0.90	0.00	0.03	0.84	0.88	0.90	0.92	0.95	23340	1
theta_1_6	0.96	0.00	0.00	0.95	0.96	0.96	0.97	0.97	25527	1
theta_1_7	0.94	0.00	0.02	0.88	0.92	0.94	0.95	0.98	24700	1
theta_1_8	0.96	0.00	0.03	0.88	0.95	0.97	0.98	1.00	23203	1
theta_1_9	0.86	0.00	0.08	0.67	0.82	0.88	0.92	0.98	23345	1
theta_1_10	0.97	0.00	0.03	0.90	0.95	0.97	0.99	1.00	22318	1
theta_2_1	0.19	0.00	0.09	0.05	0.12	0.18	0.25	0.41	22879	1
theta_2_2	0.84	0.00	0.08	0.67	0.80	0.85	0.90	0.96	25357	1
theta_2_3	0.71	0.00	0.02	0.67	0.70	0.71	0.73	0.75	25826	1
theta_2_4	0.97	0.00	0.02	0.91	0.96	0.98	0.99	1.00	22354	1
theta_2_5	0.95	0.00	0.01	0.93	0.94	0.95	0.96	0.97	24132	1
theta_2_6	0.96	0.00	0.02	0.92	0.95	0.96	0.97	0.98	23316	1
theta_2_7	0.82	0.00	0.08	0.65	0.77	0.83	0.88	0.94	24266	1
theta_2_8	0.92	0.00	0.05	0.81	0.90	0.93	0.96	0.99	24129	1
theta_2_9	0.94	0.00	0.00	0.93	0.93	0.94	0.94	0.94	24422	1
theta_2_10	0.94	0.00	0.03	0.88	0.93	0.95	0.96	0.98	24290	1
theta_3_1	0.31	0.00	0.06	0.19	0.26	0.31	0.35	0.44	24676	1
theta_3_2	0.83	0.00	0.02	0.79	0.81	0.83	0.84	0.86	23863	1
theta_3_3	0.50	0.00	0.02	0.46	0.49	0.50	0.51	0.54	25538	1
theta_3_4	0.46	0.00	0.05	0.37	0.43	0.46	0.49	0.56	22554	1
theta_3_5	0.22	0.00	0.09	0.08	0.16	0.22	0.28	0.42	22357	1
theta_4_1	0.86	0.00	0.01	0.85	0.86	0.86	0.87	0.88	25843	1
theta_4_2	0.71	0.00	0.02	0.67	0.70	0.71	0.73	0.75	23020	1
theta_4_3	0.53	0.00	0.02	0.48	0.51	0.53	0.54	0.57	24385	1
theta_4_4	0.51	0.00	0.06	0.39	0.47	0.51	0.54	0.62	23704	1
theta_4_5	0.75	0.00	0.04	0.66	0.73	0.76	0.78	0.83	23165	1
theta_4_6	0.96	0.00	0.03	0.89	0.94	0.96	0.98	0.99	22562	1
theta_4_7	0.75	0.00	0.04	0.66	0.72	0.75	0.78	0.83	24727	1
mu_1	0.82	0.00	0.06	0.68	0.78	0.83	0.86	0.92	16207	1
mu_2	0.74	0.00	0.08	0.56	0.69	0.75	0.80	0.87	19116	1
mu_3	0.49	0.00	0.12	0.26	0.40	0.49	0.57	0.72	22556	1
mu_4	0.66	0.00	0.10	0.46	0.60	0.67	0.73	0.83	20642	1
mu_A	0.58	0.00	0.13	0.31	0.49	0.59	0.68	0.82	22616	1
k_1	2.83	0.01	1.13	1.08	2.00	2.67	3.49	5.42	19882	1
k_2	1.93	0.01	0.74	0.81	1.38	1.81	2.35	3.68	20341	1
k_3	1.91	0.01	0.82	0.69	1.32	1.78	2.36	3.85	23687	1
k_4	2.26	0.01	0.91	0.86	1.60	2.12	2.78	4.39	22473	1
k_A	1.41	0.00	0.50	0.61	1.05	1.35	1.70	2.57	23380	1
a_1	2.36	0.01	1.04	0.78	1.59	2.20	2.95	4.77	19752	1
a_2	1.45	0.00	0.64	0.51	0.99	1.34	1.81	2.99	19732	1
a_3	0.93	0.00	0.46	0.27	0.59	0.85	1.18	2.03	22776	1
a_4	1.52	0.00	0.70	0.48	1.01	1.41	1.92	3.21	22006	1
a_A	0.83	0.00	0.37	0.27	0.56	0.77	1.04	1.70	21777	1
b_1	0.47	0.00	0.17	0.21	0.35	0.45	0.57	0.86	19801	1
b_2	0.47	0.00	0.17	0.21	0.35	0.45	0.57	0.85	22667	1
b_3	0.98	0.00	0.49	0.28	0.63	0.90	1.25	2.16	23266	1
b_4	0.73	0.00	0.31	0.26	0.51	0.69	0.91	1.46	23008	1
b_A	0.58	0.00	0.26	0.18	0.39	0.54	0.73	1.20	22585	1
lp__	-4451.76	0.06	4.75	-4461.89	-4454.79	-4451.43	-4448.39	-4443.40	6495	1

Output from Printing Results to Screen



Selected Results with Confidence Intervals from plot() Function

The last two lines of R code above exported the full data set from R, which enabled further analysis in MATLAB. The full contents of the MATLAB script and the generated content in published format, follow. A description of the purpose for each section of code is embedded as a comment within the script section, below the section identifier. Some of the graphs shown in this appendix duplicate figures in Chapter 3 but are retained here in the published script.

MATLAB Script Contents

- Import Data from R for Analysis
- Setup the Import Options and Import the Data
- Declare
- Extract for Plotting and Print to Screen
- Make Chain Assignments
- Trace Plots by Parameter - All Chains
- Trace Plots by Parameter - Individual Chains
- Correlograms
- Component Posteriors Overlaid with Evidence Data Points:
- Component and Function Posteriors Together:

Import Data from R for Analysis

```
  filename: /Users/bethrajchel/Thesis/Post_AData_1111.csv


% This script will import the data from a Stan execution in R, saved to the
% filename indicated above.  This script is explicitly for a the Function A
% hierarchy (3 levels, 4 component types, components and functions as beta
% distributions, products as binomials). This script assumes 4 successful
% chains in Stan.

% Evidence: Full A Data Set
% Priors: A,B = 0.5. S,R = 5. S',R'=2.
% C = (1,1,1,1)
```

Setup the Import Options and Import the Data

This section governs the import of data into the designated format.

```
opts = delimitedTextImportOptions("NumVariables", 54);

% Specify range and delimiter
opts.DataLines = [2, Inf];
opts.Delimiter = ",";

% Specify column names and types
opts.VariableNames = ["Var1", "theta_1_1", "theta_1_2", "theta_1_3",...
    "theta_1_4", "theta_1_5", "theta_1_6", "theta_1_7", "theta_1_8",...
    "theta_1_9", "theta_1_10", "theta_2_1", "theta_2_2", "theta_2_3",...
    "theta_2_4", "theta_2_5", "theta_2_6", "theta_2_7", "theta_2_8",...
    "theta_2_9", "theta_2_10", "theta_3_1", "theta_3_2", "theta_3_3",...
    "theta_3_4", "theta_3_5", "theta_4_1", "theta_4_2", "theta_4_3",...
    "theta_4_4", "theta_4_5", "theta_4_6", "theta_4_7", "mu_1", "mu_2",....
    "mu_3", "mu_4", "mu_A", "k_1", "k_2", "k_3", "k_4", "k_A", "a_1",...
```

```
    “a_2”, “a_3”, “a_4”, “a_A”, “b_1”, “b_2”, “b_3”, “b_4”, “b_A”, “lp__”];
opts.SelectedVariableNames = [“theta_1_1”, “theta_1_2”, “theta_1_3”,...
    “theta_1_4”, “theta_1_5”, “theta_1_6”, “theta_1_7”, “theta_1_8”,...
    “theta_1_9”, “theta_1_10”, “theta_2_1”, “theta_2_2”, “theta_2_3”,...
    “theta_2_4”, “theta_2_5”, “theta_2_6”, “theta_2_7”, “theta_2_8”,...
    “theta_2_9”, “theta_2_10”, “theta_3_1”, “theta_3_2”, “theta_3_3”...
    , “theta_3_4”, “theta_3_5”, “theta_4_1”, “theta_4_2”, “theta_4_3”,...
    “theta_4_4”, “theta_4_5”, “theta_4_6”, “theta_4_7”, “mu_1”, “mu_2”,...
    “mu_3”, “mu_4”, “mu_A”, “k_1”, “k_2”, “k_3”, “k_4”, “k_A”, “a_1”,...
    “a_2”, “a_3”, “a_4”, “a_A”, “b_1”, “b_2”, “b_3”, “b_4”, “b_A”, “lp__”];
opts.VariableTypes = [“string”, “double”, “double”, “double”, “double”,...
    “double”, “double”, “double”, “double”, “double”, “double”, “double”,...
    “double”, “double”, “double”, “double”, “double”, “double”, “double”,...
    “double”, “double”, “double”, “double”, “double”, “double”, “double”,...
    “double”, “double”, “double”, “double”, “double”, “double”, “double”,...
    “double”, “double”, “double”, “double”, “double”, “double”, “double”,...
    “double”, “double”, “double”, “double”, “double”, “double”, “double”];

% Specify file level properties
opts.ExtraColumnsRule = “ignore”;
opts.EmptyLineRule = “read”;

% Specify variable properties
opts = setvaropts(opts, “Var1”, “WhitespaceRule”, “preserve”);
opts = setvaropts(opts, “Var1”, “EmptyFieldRule”, “auto”);

% Import the data (MASTER: A_Data_Stan_1111.csv)
ADataStan = readtable(“/Users/bethrajchel/Thesis/Post_AData_1111.csv”, ...
    opts);

%Clear temporary variables
clear opts
```

Declare

These variables would be established if the read-in occured after running a script for data manipulation for FW. Declared here again for certainty.

```
J = 4;
C = [1,1,1,1];
```

Extract for Plotting and Print to Screen

Extract parameters of interest from array and print means to screen.

```
aA = ADataStan{:,47};
bA = ADataStan{:,52};
a1 = ADataStan{:,43};
a2 = ADataStan{:,44};
a3 = ADataStan{:,45};
a4 = ADataStan{:,46};
b1 = ADataStan{:,48};
b2 = ADataStan{:,49};
b3 = ADataStan{:,50};
b4 = ADataStan{:,51};

a_all = {a1, a2, a3, a4, aA};
b_all = {b1, b2, b3, b4, bA};

a = cell(1,J+1);
b = cell(1,J+1);

for i = 1:length(a)
    a{i} = mean(a_all{i},'all');
    b{i} = mean(b_all{i},'all');
end
```

```
a = cell2mat(a);
b = cell2mat(b);

fprintf('Stan results with frequency (%d %d %d %d):\n', C)
fprintf(' aA = %d\n bA = %e\n\n', a(J+1), b(J+1))
for i = 1:J
    fprintf('  a %d = %e\n',i,a(i))
    fprintf('  b %d = %e\n\n',i,b(i))
end
fprintf('\n')
fprintf('\n')
```

```
Stan results with frequency (1 1 1 1):
 aA = 8.276042e-01
 bA = 5.813961e-01

  a 1 = 2.355622e+00
  b 1 = 4.720478e-01

  a 2 = 1.452210e+00
  b 2 = 4.738155e-01

  a 3 = 9.266678e-01
  b 3 = 9.821122e-01

  a 4 = 1.520651e+00
  b 4 = 7.344584e-01
```

Make Chain Assignments

If multiple chains, make chain assignments to plot traces by chain.

```
move_a = []; move_b = [];
a_chain1 = {}; a_chain2 = {}; a_chain3 = {}; a_chain4 = {};
b_chain1 = {}; b_chain2 = {}; b_chain3 = {}; b_chain4 = {};

for j = 1:J+1
    move_a = a_all{j};
    a_chain1{j} = move_a(1:4000);
    a_chain2{j} = move_a(4001:8000);
    a_chain3{j} = move_a(8001:12000);
    a_chain4{j} = move_a(12001:16000);
    move_b = b_all{j};
    b_chain1{j} = move_b(1:4000);
    b_chain2{j} = move_b(4001:8000);
    b_chain3{j} = move_b(8001:12000);
    b_chain4{j} = move_b(12001:16000);
end
```

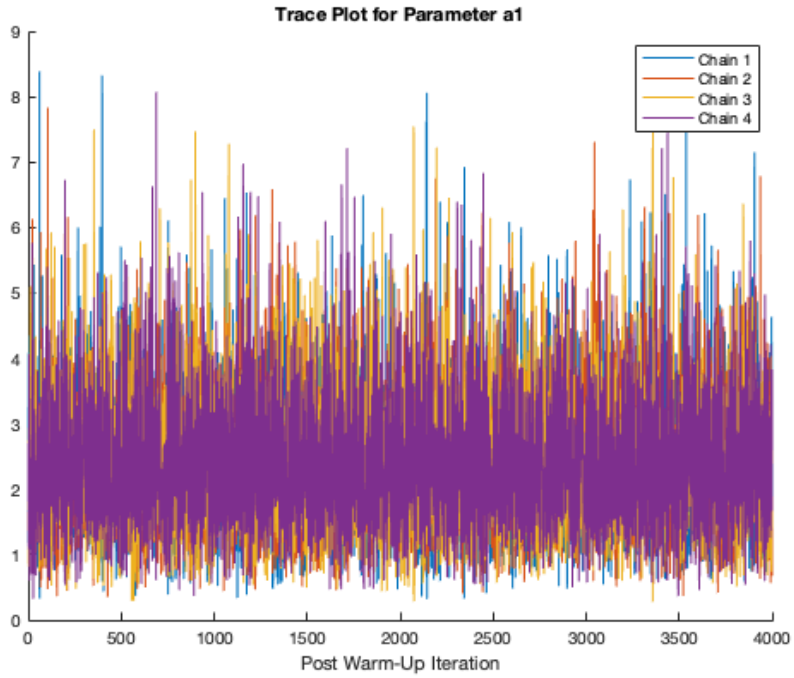
Trace Plots by Parameter - All Chains

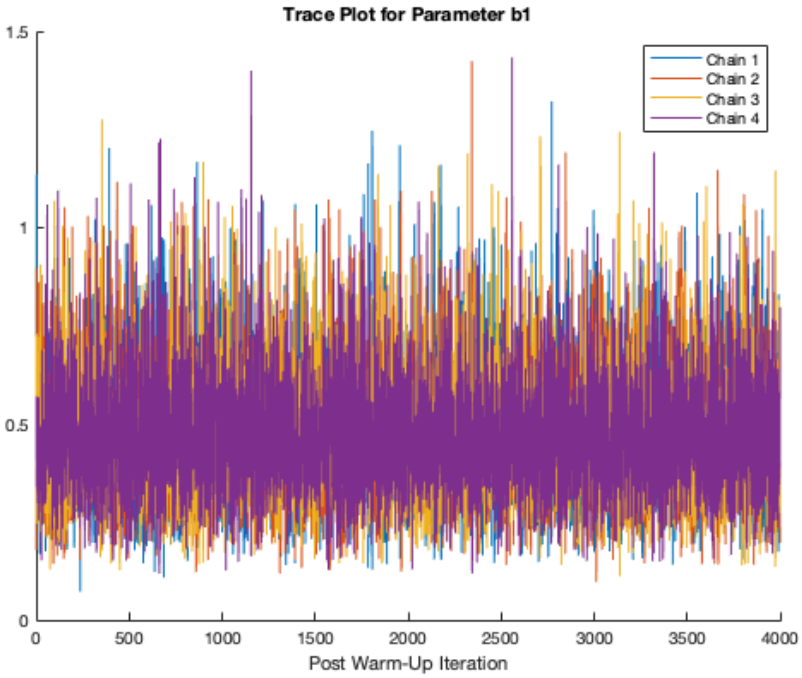
Trace plots of chains of a parameter should overlap as an indication of convergence (achieving stationary distribution).

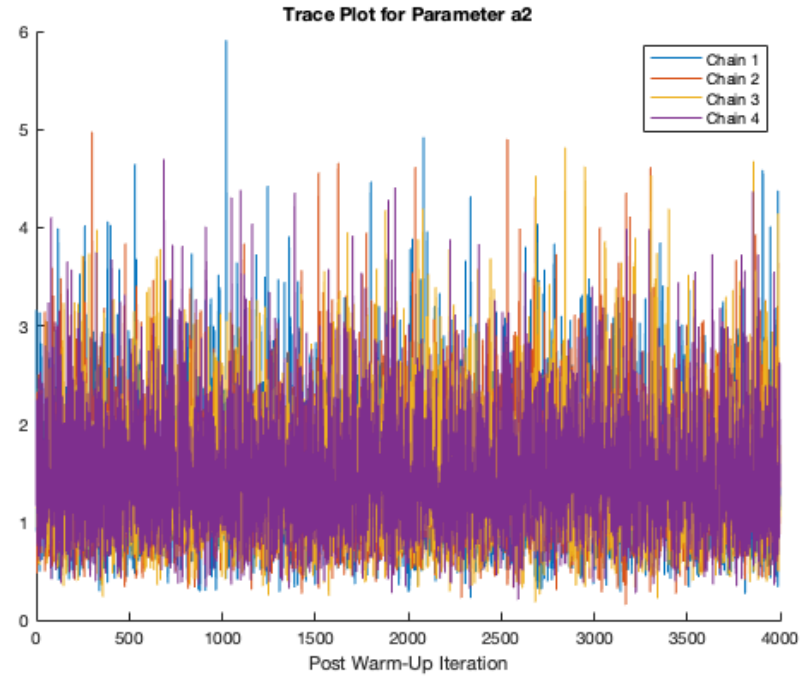
```
for j = 1:J
    figure
    hold on
    plot(a_chain1{j})
    plot(a_chain2{j})
    plot(a_chain3{j})
    plot(a_chain4{j})
```

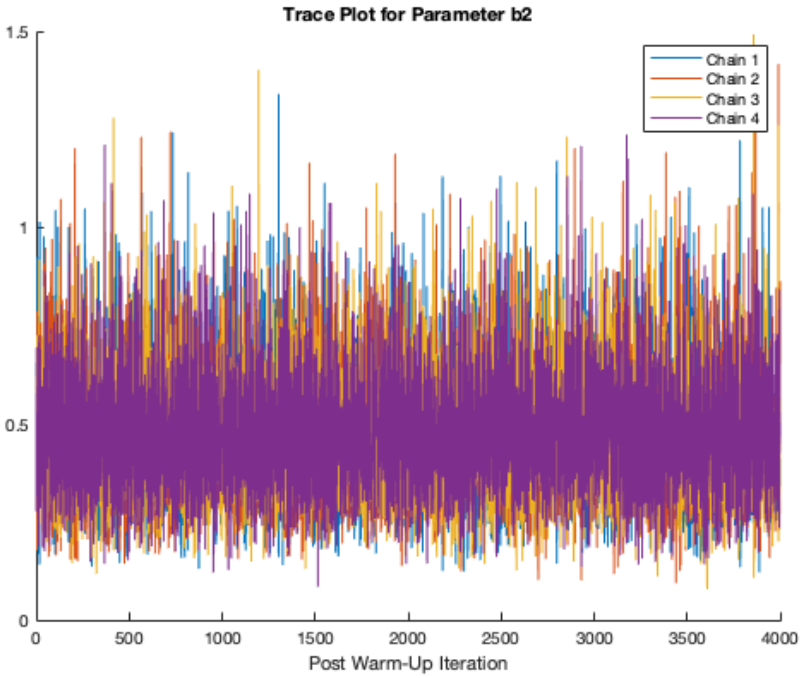
```
    legend('Chain 1', 'Chain 2', 'Chain 3', 'Chain 4')
    titlestr = sprintf('Trace Plot for Parameter a%d',j);
    title(titlestr)
    xlabel('Post Warm-Up Iteration')
    hold off
    figure
    hold on
    plot(b_chain1{j})
    plot(b_chain2{j})
    plot(b_chain3{j})
    plot(b_chain4{j})
    legend('Chain 1', 'Chain 2', 'Chain 3', 'Chain 4')
    titlestr = sprintf('Trace Plot for Parameter b%d',j);
    title(titlestr)
    xlabel('Post Warm-Up Iteration')
    hold off
end

figure
    hold on
    plot(a_chain1{5})
    plot(a_chain2{5})
    plot(a_chain3{5})
    plot(a_chain4{5})
    legend('Chain 1', 'Chain 2', 'Chain 3', 'Chain 4')
    titlestr = sprintf('Trace Plot for Parameter aA');
    title(titlestr)
    xlabel('Post Warm-Up Iteration')
    hold off
    figure
    hold on
    plot(b_chain1{5})
    plot(b_chain2{5})
```

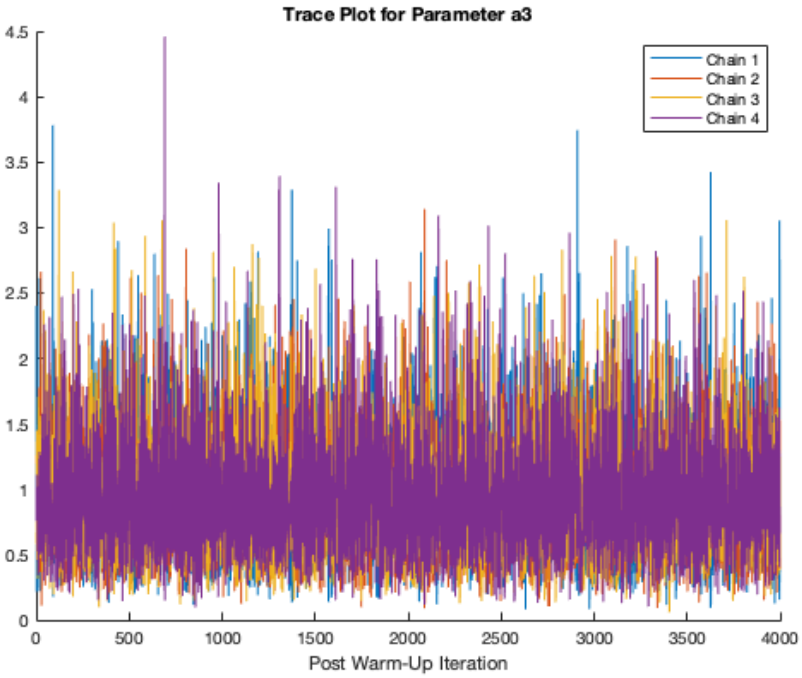
```
plot(b_chain3{5})
plot(b_chain4{5})
legend('Chain 1', 'Chain 2', 'Chain 3', 'Chain 4')
titlestr = sprintf('Trace Plot for Parameter bA');
title(titlestr)
xlabel('Post Warm-Up Iteration')
hold off
```

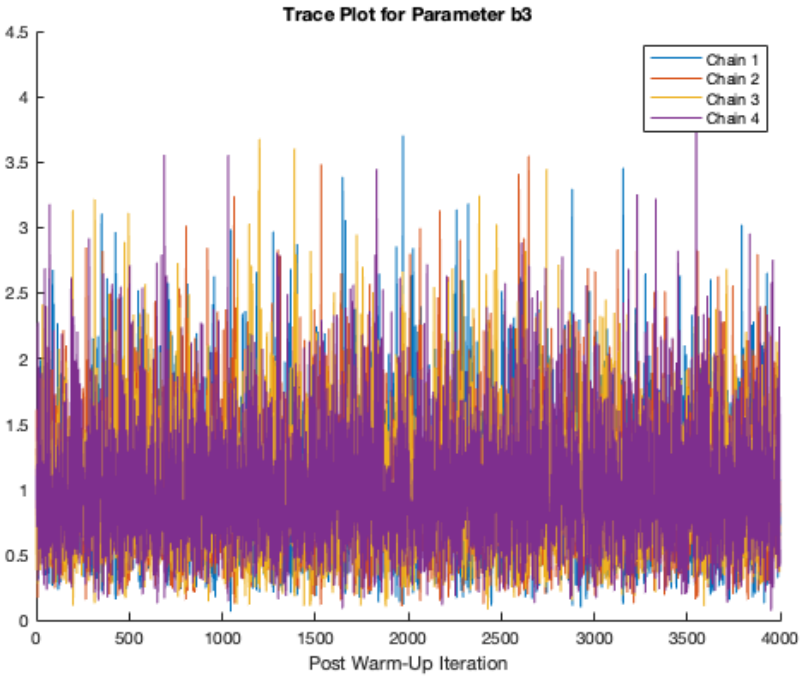


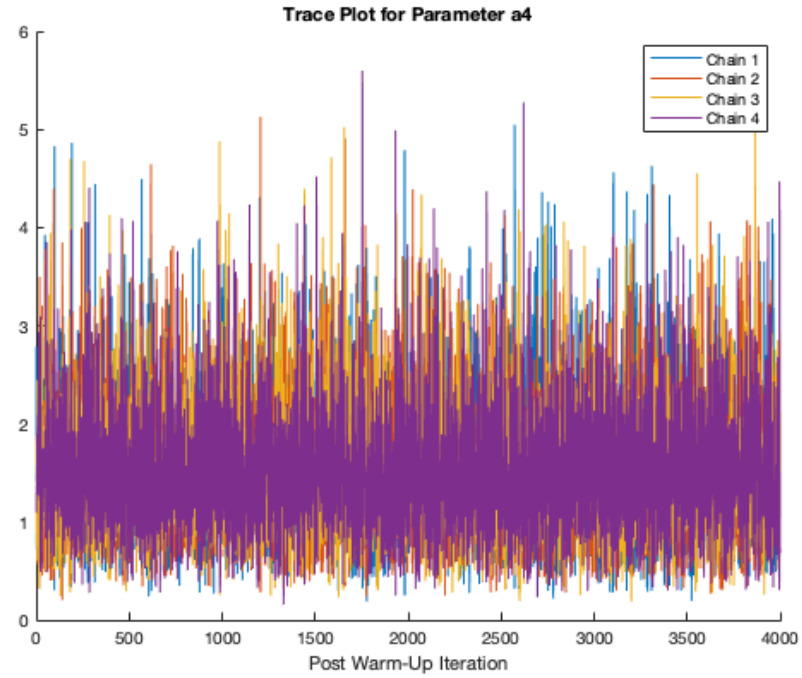

Trace Plot for Parameter b1
Chain 1
Chain 2
Chain 3
Chain 4
1.5
1
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration

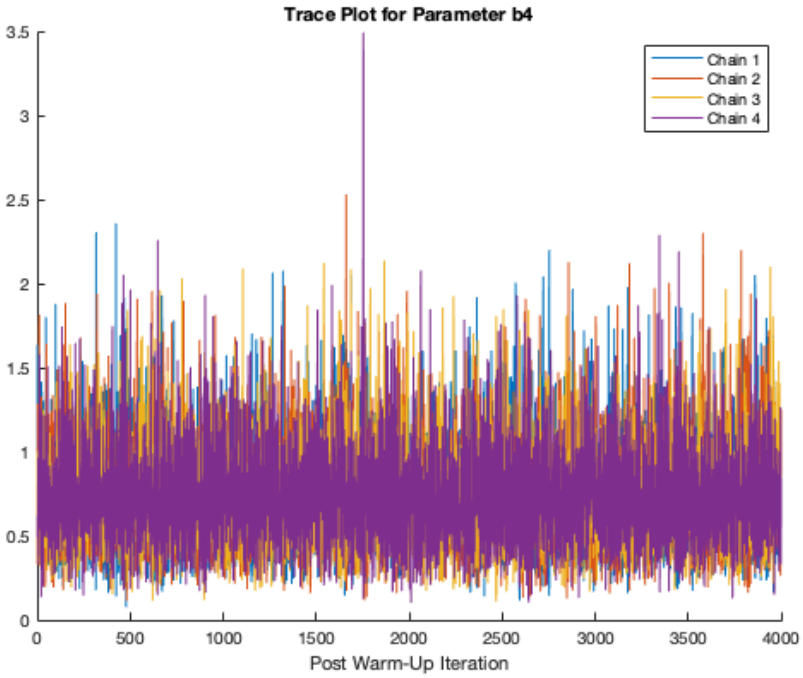

Trace Plot for Parameter a2
Chain 1
Chain 2
Chain 3
Chain 4
6
5
4
3
2
1
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration

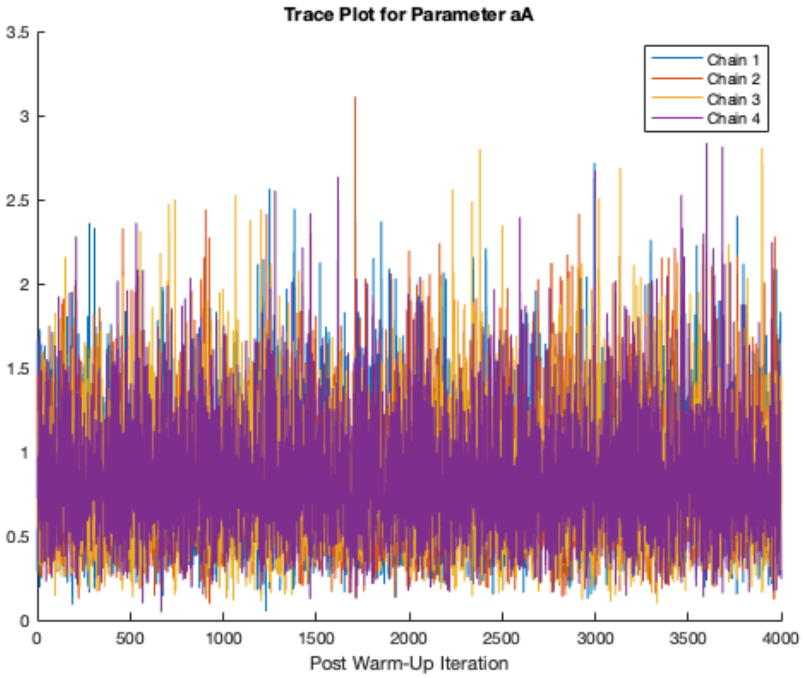

Trace Plot for Parameter b2
Chain 1
Chain 2
Chain 3
Chain 4
1.5
1
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration

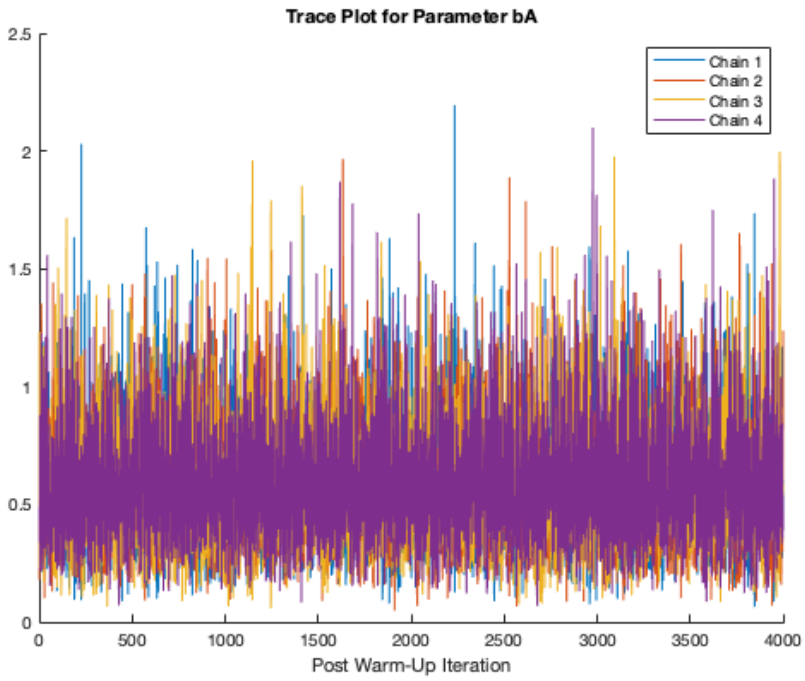

Trace Plot for Parameter a3
Chain 1
Chain 2
Chain 3
Chain 4
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration


Trace Plot for Parameter b3
Chain 1
Chain 2
Chain 3
Chain 4
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration


Trace Plot for Parameter a4
Chain 1
Chain 2
Chain 3
Chain 4
6
5
4
3
2
1
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration


Trace Plot for Parameter b4
Chain 1
Chain 2
Chain 3
Chain 4
3.5
3
2.5
2
1.5
1
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration


Trace Plot for Parameter aA
Chain 1
Chain 2
Chain 3
Chain 4
3.5
3
2.5
2
1.5
1
0.5
0
0
500
1000
1500
2000
2500
3000
3500
4000
Post Warm-Up Iteration



Trace Plots by Parameter - Individual Chains

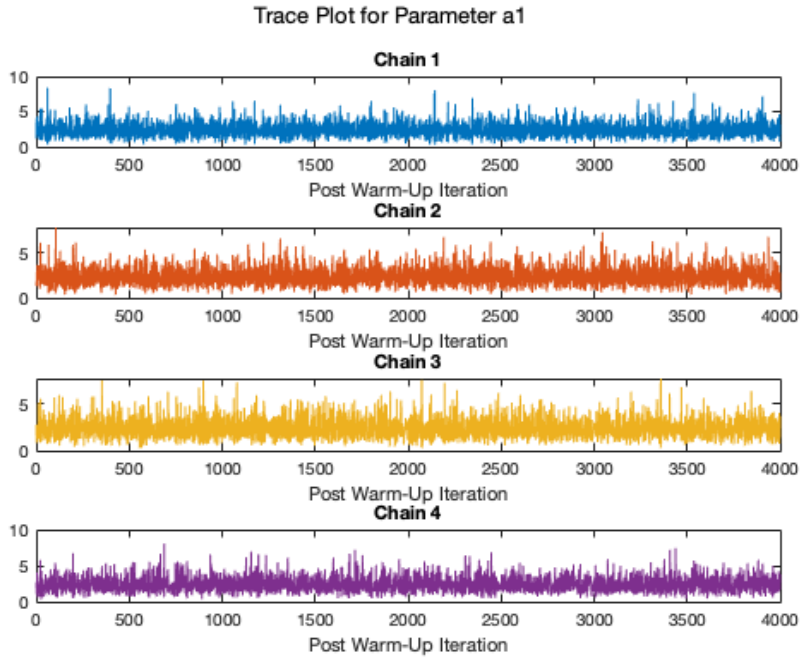
Any given chain trace plot should show relatively constant mean and variance (no transient behavior) as an indication of convergence.

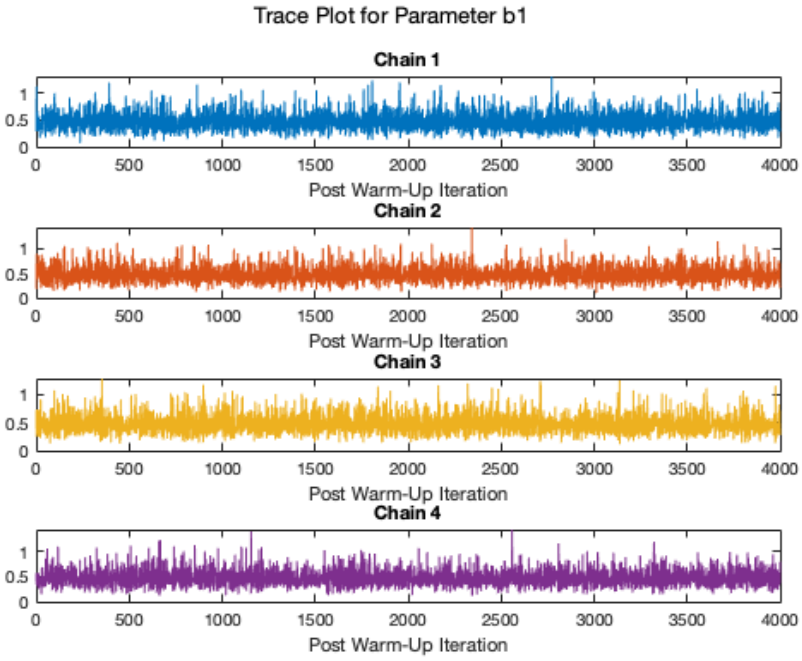
```
for j = 1:J
    figure
    hold on
    subplot(4,1,1)
    plot(a_chain1{j})
    title('Chain 1')
    xlabel('Post Warm-Up Iteration')

    subplot(4,1,2)
    plot(a_chain2{j},'color','#D95319')
    title('Chain 2')
    xlabel('Post Warm-Up Iteration')
```

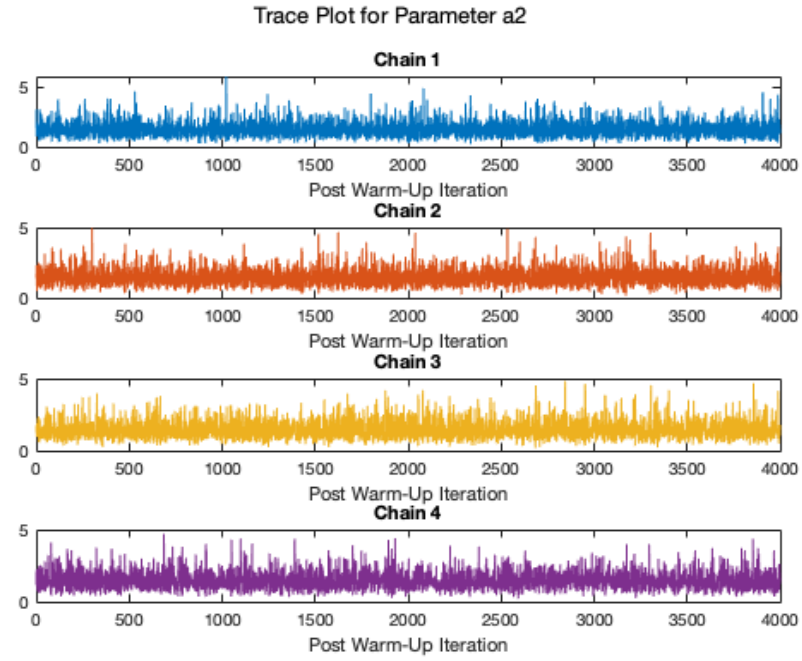
```
subplot(4,1,3)
plot(a_chain3{j},'color','#EDB120')
title('Chain 3')
xlabel('Post Warm-Up Iteration')

subplot(4,1,4)
plot(a_chain4{j},'color','#7E2F8E')
title('Chain 4')
xlabel('Post Warm-Up Iteration')

titlestr = sprintf('Trace Plot for Parameter a%d',j);
sgtitle(titlestr);
hold off

figure
hold on
subplot(4,1,1)
plot(b_chain1{j})
title('Chain 1')
xlabel('Post Warm-Up Iteration')

subplot(4,1,2)
plot(b_chain2{j},'color','#D95319')
title('Chain 2')
xlabel('Post Warm-Up Iteration')

subplot(4,1,3)
plot(b_chain3{j},'color','#EDB120')
title('Chain 3')
xlabel('Post Warm-Up Iteration')

subplot(4,1,4)
```

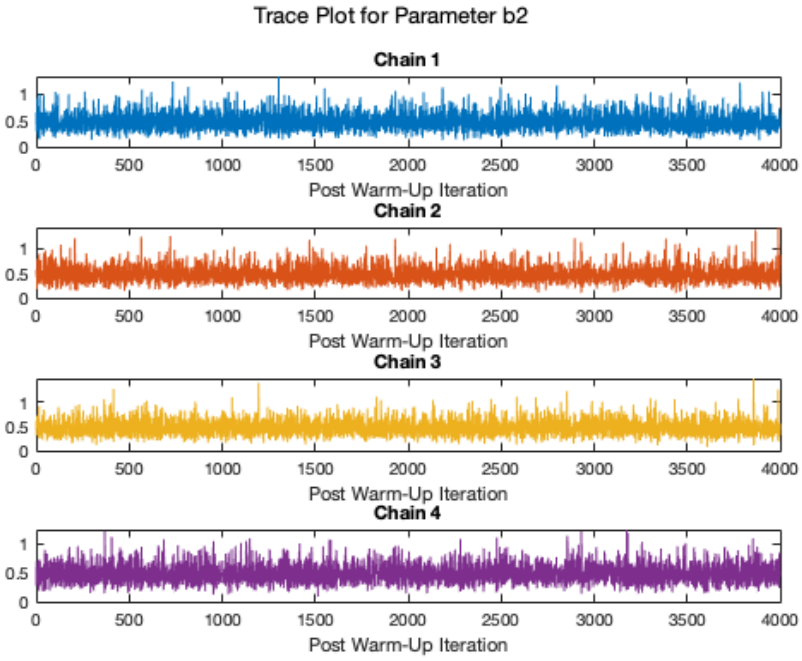
```
    plot(b_chain4{j},'color','#7E2F8E')
    title('Chain 4')
    xlabel('Post Warm-Up Iteration')

    titlestr = sprintf('Trace Plot for Parameter b%d',j);
    sgtitle(titlestr);
    hold off
end

figure
    hold on
    subplot(4,1,1)
    plot(a_chain1{5})
    title('Chain 1')
    xlabel('Post Warm-Up Iteration')

    subplot(4,1,2)
    plot(a_chain2{5},'color','#D95319')
    title('Chain 2')
    xlabel('Post Warm-Up Iteration')

    subplot(4,1,3)
    plot(a_chain3{5},'color','#EDB120')
    title('Chain 3')
    xlabel('Post Warm-Up Iteration')

    subplot(4,1,4)
    plot(a_chain4{5},'color','#7E2F8E')
    title('Chain 4')
    xlabel('Post Warm-Up Iteration')

    titlestr = sprintf('Trace Plot for Parameter bA');
    sgtitle(titlestr)
```

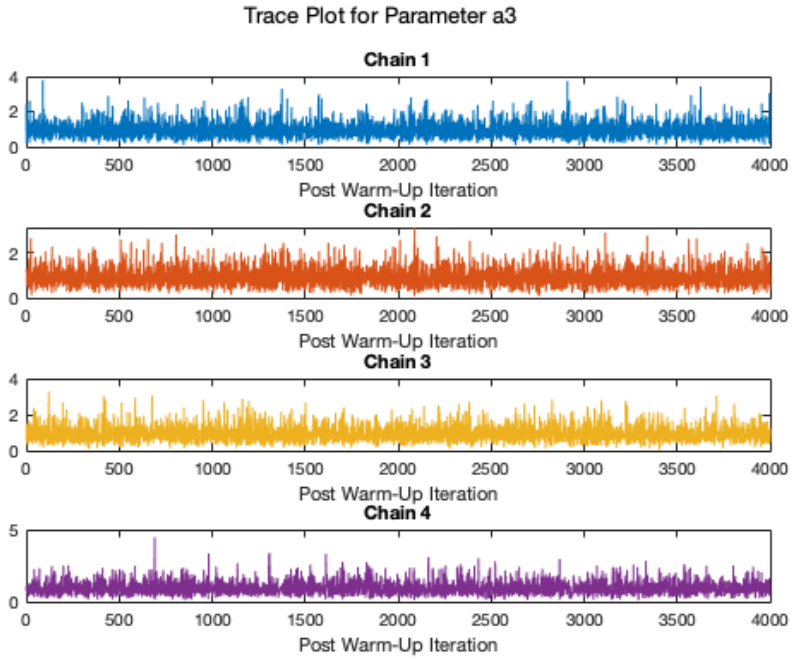
```
hold off

figure
hold on
subplot(4,1,1)
plot(b_chain1{5})
title('Chain 1')
xlabel('Post Warm-Up Iteration')

subplot(4,1,2)
plot(b_chain2{5},'color','#D95319')
title('Chain 2')
xlabel('Post Warm-Up Iteration')

subplot(4,1,3)
plot(b_chain3{5},'color','#EDB120')
title('Chain 3')
xlabel('Post Warm-Up Iteration')

subplot(4,1,4)
plot(b_chain4{5},'color','#7E2F8E')
title('Chain 4')
xlabel('Post Warm-Up Iteration')

titlestr = sprintf('Trace Plot for Parameter bA');
sgtitle(titlestr)
hold off
```

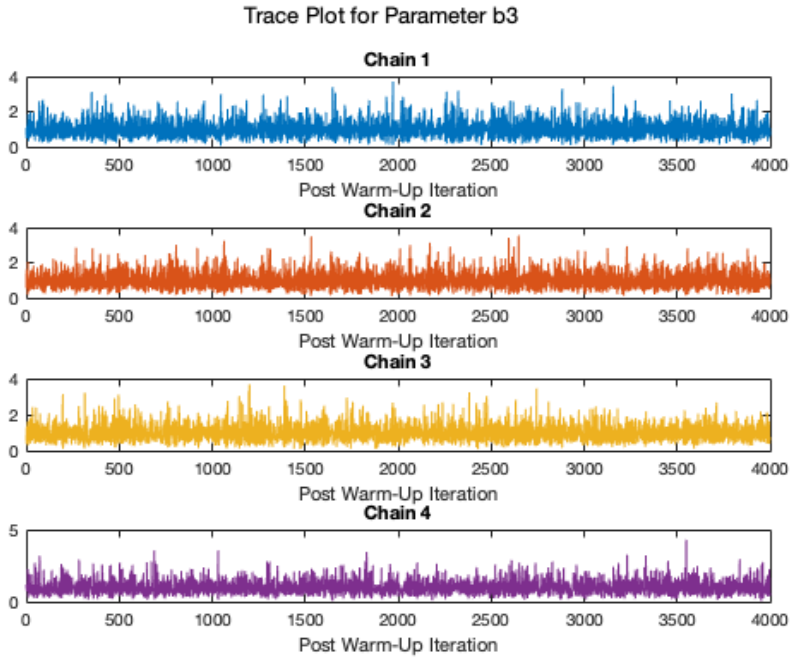

Trace Plot for Parameter a1
Chain 1
Post Warm-Up Iteration
Chain 2
Post Warm-Up Iteration
Chain 3
Post Warm-Up Iteration
Chain 4
Post Warm-Up Iteration

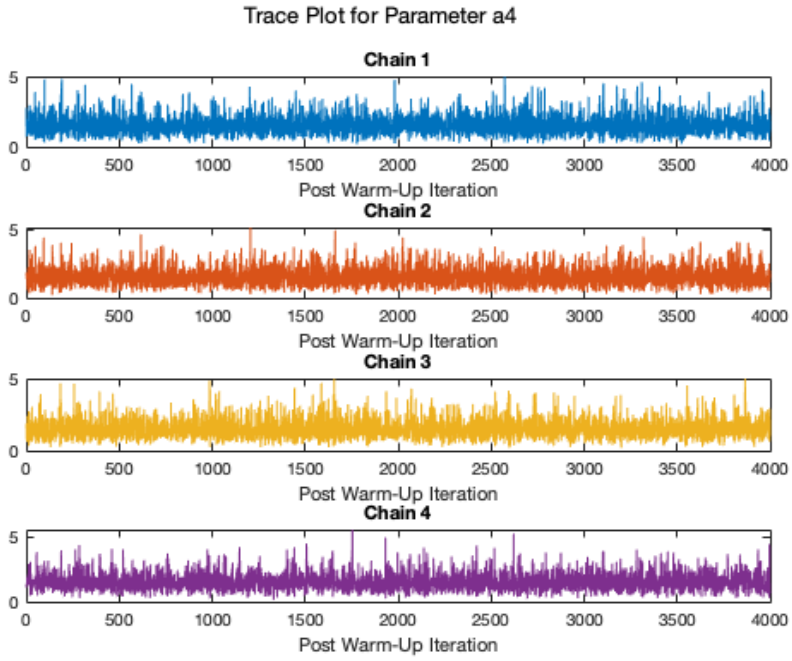

Trace Plot for Parameter b1
Chain 1
Post Warm-Up Iteration
Chain 2
Post Warm-Up Iteration
Chain 3
Post Warm-Up Iteration
Chain 4
Post Warm-Up Iteration

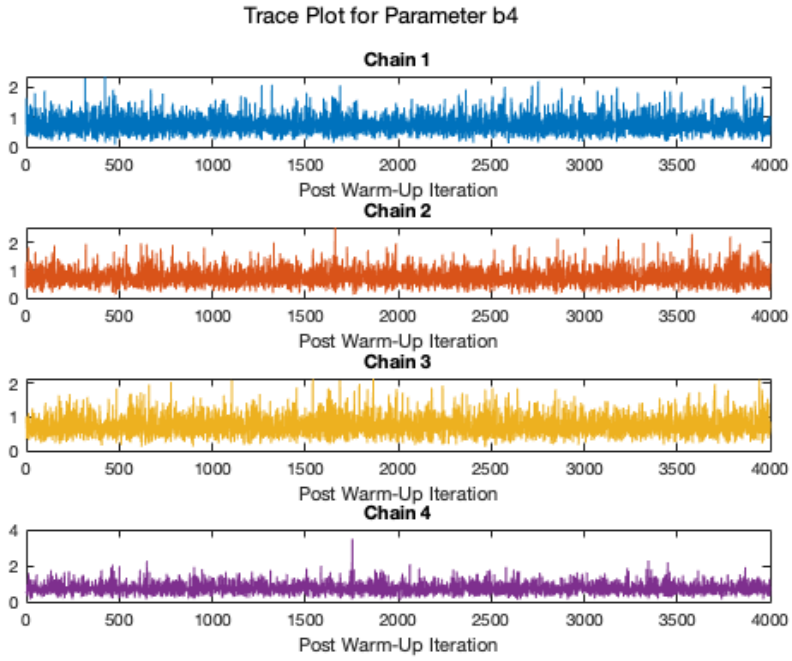

Trace Plot for Parameter a2
Chain 1
Chain 2
Chain 3
Chain 4
Post Warm-Up Iteration

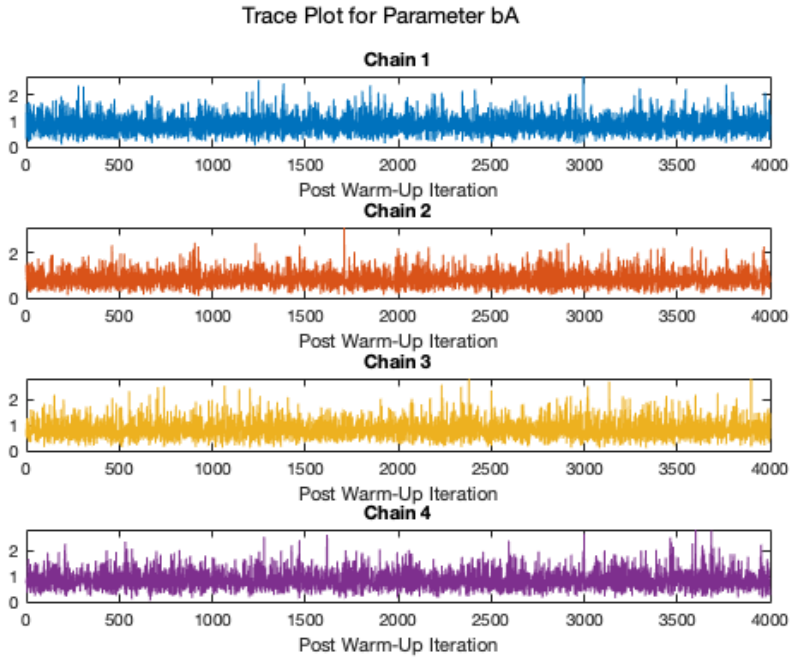

Trace Plot for Parameter b2
Chain 1
Chain 2
Chain 3
Chain 4
Post Warm-Up Iteration

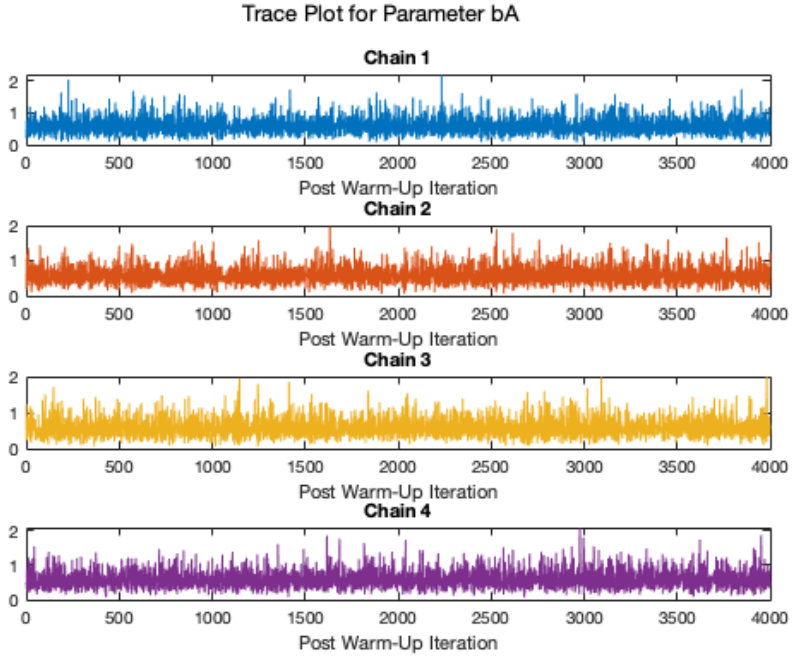

Trace Plot for Parameter a3
Chain 1
Chain 2
Chain 3
Chain 4
Post Warm-Up Iteration


Trace Plot for Parameter b3
Chain 1
Chain 2
Chain 3
Chain 4
Post Warm-Up Iteration


Trace Plot for Parameter a4
Chain 1
Chain 2
Chain 3
Chain 4
Post Warm-Up Iteration


Trace Plot for Parameter b4
Chain 1
Chain 2
Chain 3
Chain 4
Post Warm-Up Iteration

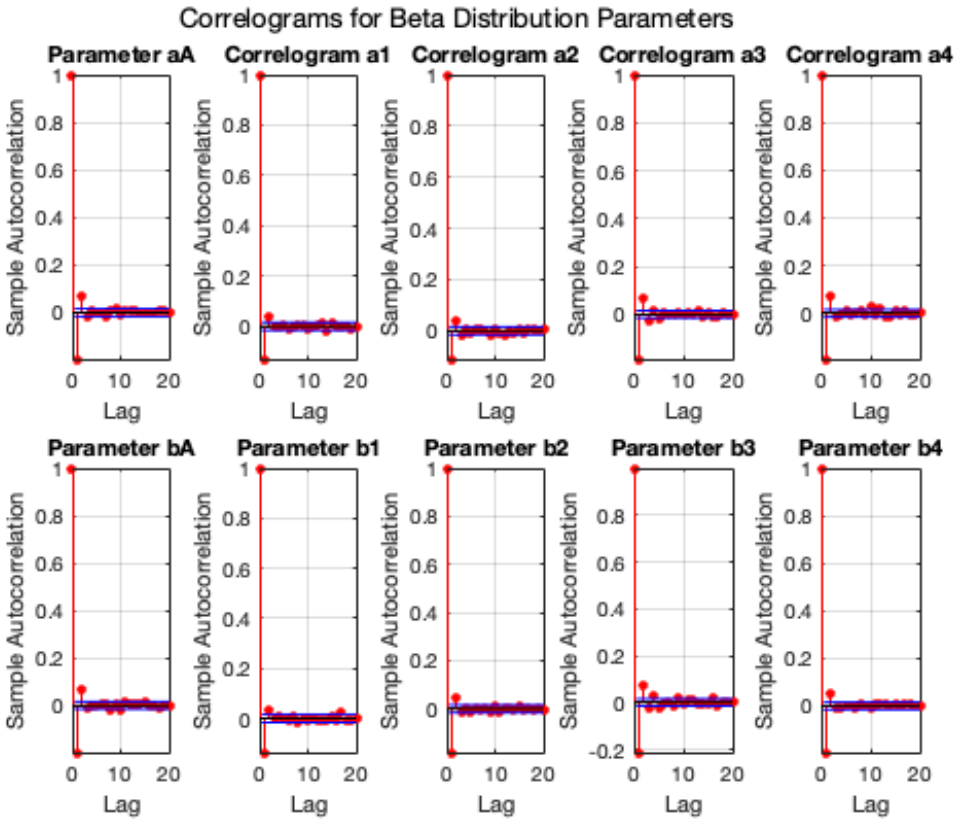

Trace Plot for Parameter bA
Chain 1
Post Warm-Up Iteration
Chain 2
Post Warm-Up Iteration
Chain 3
Post Warm-Up Iteration
Chain 4
Post Warm-Up Iteration


Trace Plot for Parameter bA
Chain 1
Post Warm-Up Iteration
Chain 2
Post Warm-Up Iteration
Chain 3
Post Warm-Up Iteration
Chain 4
Post Warm-Up Iteration

Correlograms

Generate correlograms for parameters governing function and component posteriors. High correlation undesirable; low/zero correlation desirable (indicates random walk during MCMC sampling).

```
figure;
subplot(2,5,1); autocorr(aA);
    axis tight
    title(sprintf(['Parameter aA']));
subplot(2,5,6); autocorr(bA);
    axis tight
    title(sprintf(['Parameter bA']));
for j = 1:J
    subplot(2,5,j+1)
    autocorr(a_all{j});
    axis tight
    title(sprintf(['Correlogram a%d'],j));
end
for j = 1:J
    subplot(2,5,j+6)
    autocorr(b_all{j});
    axis tight
    title(sprintf(['Parameter b%d'],j));
end
sgtitle('Correlograms for Beta Distribution Parameters');
```



Component Posteriors Overlaid with Evidence Data Points:

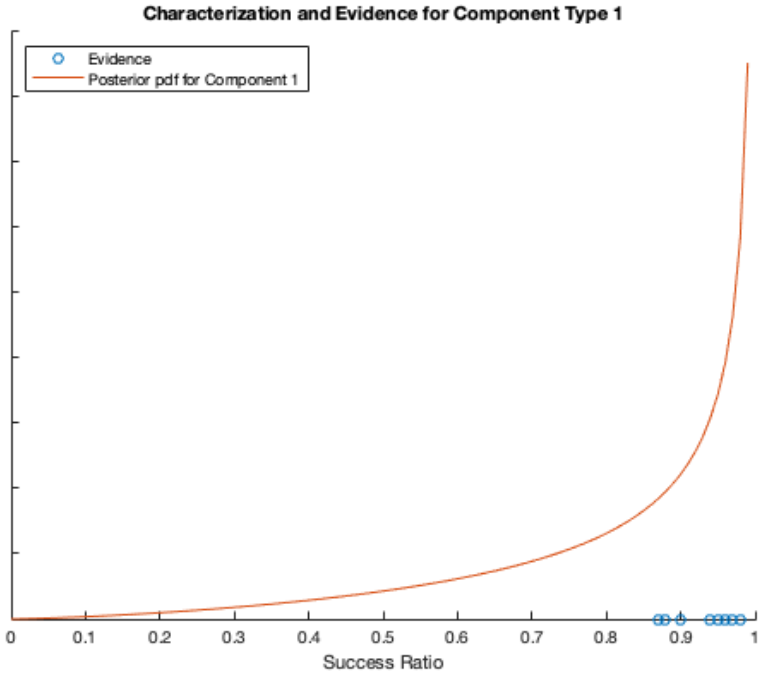
Evaluate the posterior distributions of each component in the context of the original evidence - provides a “gut check”; not part of convergence monitoring.

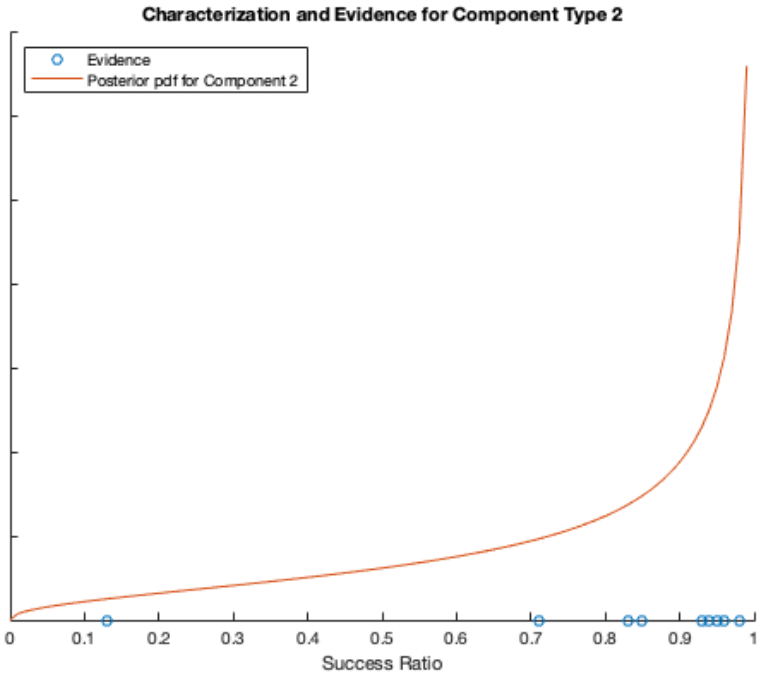
```
orig_A_data = readtable('A_Data.csv');
X = 0:0.01:1;

pd = cell(2,J+1);
for j = 1:J+1
    pd{1,j} = makedist('Beta','a',a(j),'b',b(j));
    pd{2,j} = pdf(pd{1,j},X);
end
```

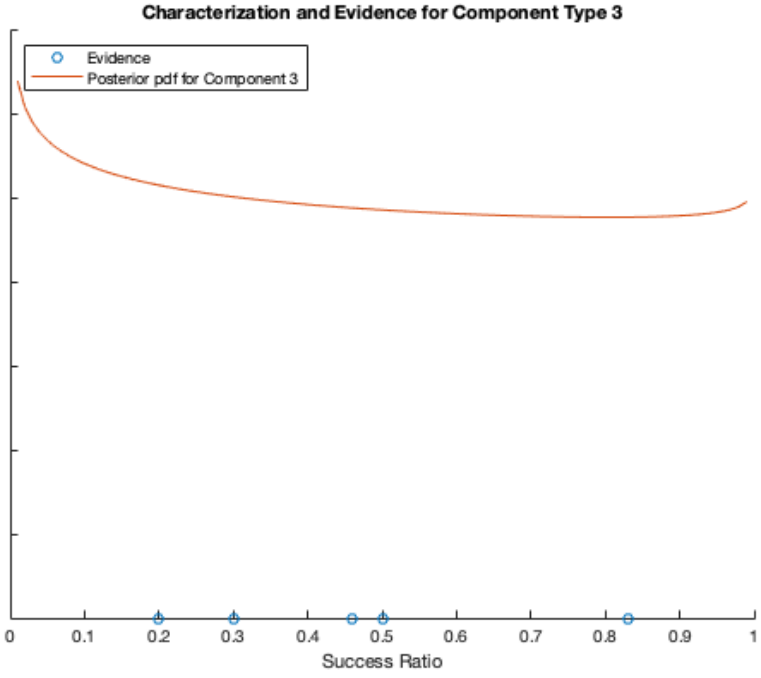
```
r = cell(1:J);
r{1} = sort(rmmissing(orig_A_data{1:end,'Ratio_1'}));
r{2} = sort(rmmissing(orig_A_data{1:end,'Ratio_2'}));
r{3} = sort(rmmissing(orig_A_data{1:end,'Ratio_3'}));
r{4} = sort(rmmissing(orig_A_data{1:end,'Ratio_4'}));

for j = 1:J
    figure
    hold on
%     scatter(r{j},1:length(r{j}))
    z = zeros(1,length(r{j}));
    scatter(r{j},z)
    plot(X,pd{2,j})
    legendstr = sprintf('Posterior pdf for Component %d',j);
    legend('Evidence',legendstr,'location','NorthWest')
    xlabel('Success Ratio')
    ay = gca;
    ay.YTickLabel = [];
    ts = sprintf('Characterization and Evidence for Component Type %d',j);
    title(ts)
    hold off
end

figure
subplot(2,2,1)
    hold on
%     scatter(r{1},1:length(r{1}))
    z = zeros(1,length(r{1}));
    scatter(r{1},z)
    plot(X,pd{2,1})
    legendstr = sprintf('Posterior pdf for Component 1');
    legend('Evidence',legendstr,'location','NorthWest')
    xlabel('Success Ratio')
```

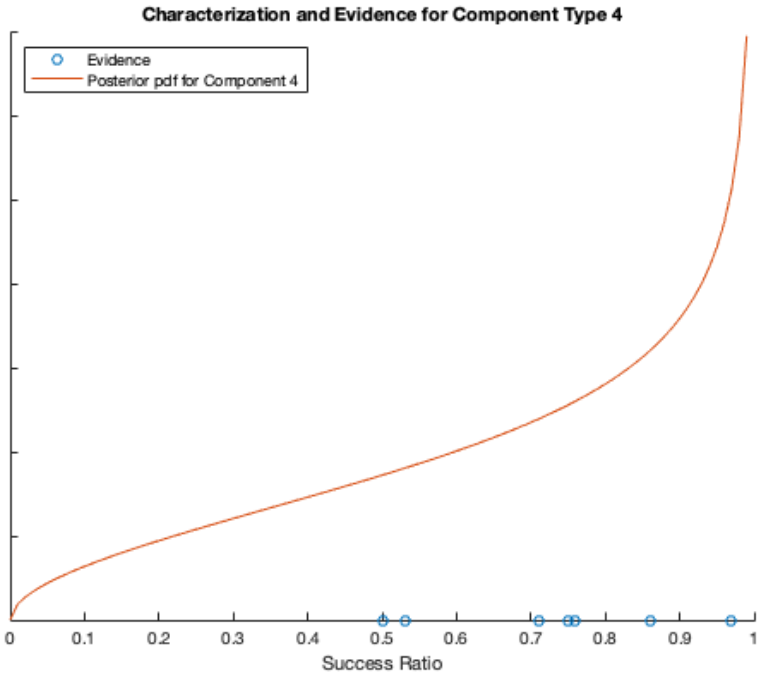
```
    ay = gca;
    ay.YTickLabel = [];
    titlestr = sprintf('Component Type 1');
    title(titlestr)
    hold off
subplot(2,2,2)
    hold on
%     scatter(r{2},1:length(r{2}))
    z = zeros(1,length(r{2}));
    scatter(r{2},z)
    plot(X,pd{2,2})
    legendstr = sprintf('Posterior pdf for Component 2');
    legend('Evidence',legendstr,'location','NorthWest')
    xlabel('Success Ratio')
    ay = gca;
    ay.YTickLabel = [];
    titlestr = sprintf('Component Type 2');
    title(titlestr)
    hold off
subplot(2,2,3)
    hold on
%     scatter(r{3},1:length(r{3}))
    z = zeros(1,length(r{3}));
    scatter(r{3},z)
    plot(X,pd{2,3})
    legendstr = sprintf('Posterior pdf for Component 3');
    legend('Evidence',legendstr,'location','NorthWest')
    xlabel('Success Ratio')
    ay = gca;
    ay.YTickLabel = [];
    titlestr = sprintf('Component Type 3');
    title(titlestr)
    hold off
```

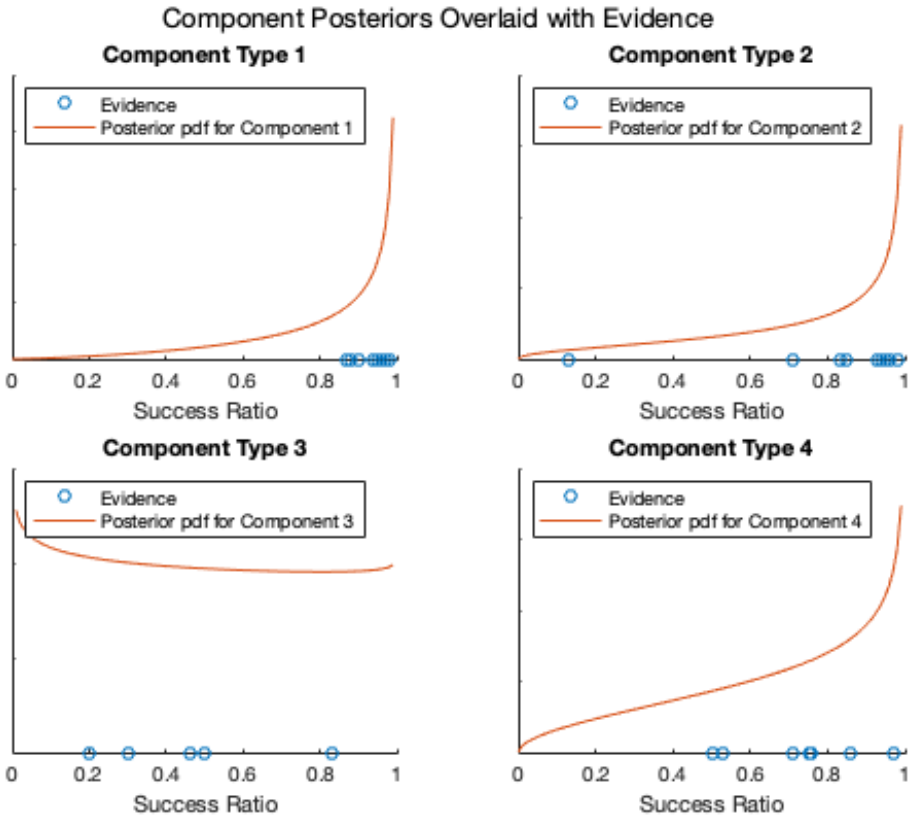
```
subplot(2,2,4)
    hold on
%     scatter(r{4},1:length(r{4}))
    z = zeros(1,length(r{4}));
    scatter(r{4},z)
    plot(X,pd{2,4})
    legendstr = sprintf('Posterior pdf for Component 4');
    legend('Evidence',legendstr,'location','NorthWest')
    xlabel('Success Ratio')
    ay = gca;
    ay.YTickLabel = [];
    titlestr = sprintf('Component Type 4');
    title(titlestr)
    hold off
sgtitle('Component Posteriors Overlaid with Evidence');
```




Characterization and Evidence for Component Type 2
Evidence
Posterior pdf for Component 2
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Success Ratio


Characterization and Evidence for Component Type 3
Evidence
Posterior pdf for Component 3
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Success Ratio


Characterization and Evidence for Component Type 4
Evidence
Posterior pdf for Component 4
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Success Ratio

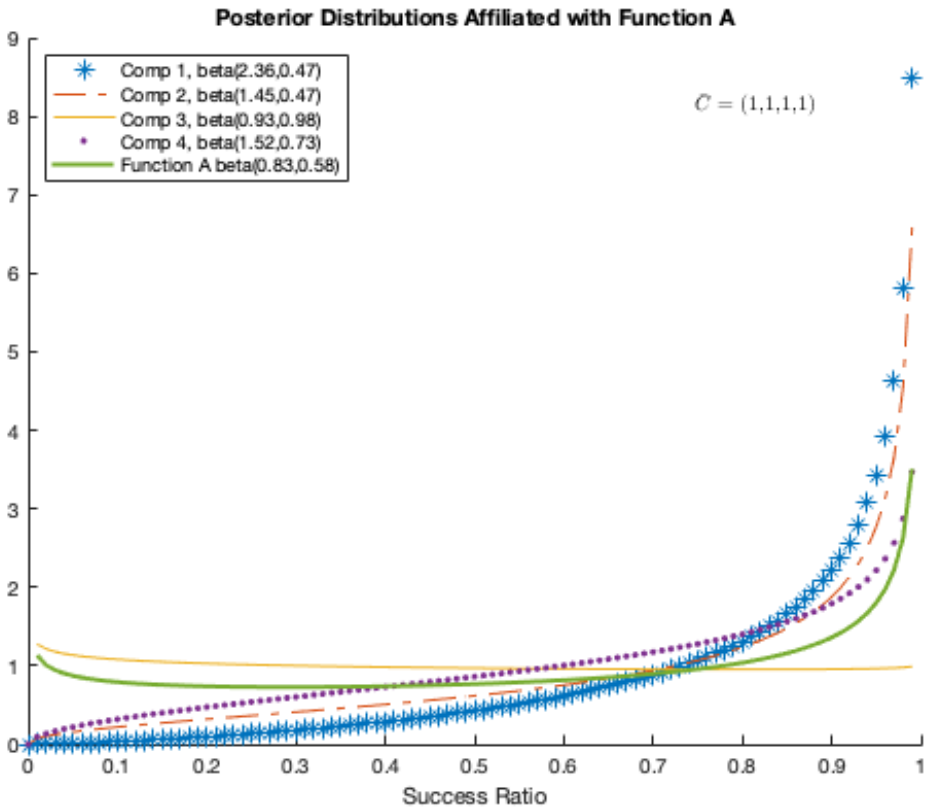


Component and Function Posteriors Together:

Display pdfs of component and function posterior distributions together

```
str = struct([]);
figure
hold on
plot(X,pd{2,1},'*')
plot(X,pd{2,2},'-.')
plot(X,pd{2,3})
plot(X,pd{2,4},'.')
plot(X,pd{2,5},'LineWidth',2)

str = struct([]);
```

```
for i = 1:J
    str{i} = ['Comp ' sprintf('%d,',i) ' beta' sprintf('(%1.2f,%1.2f)',...
        a(i),b(i))];
end
str{5} = ['Function A beta' sprintf('(%1.2f,%1.2f)', a(5),b(5))];
leg = legend(str{1},str{2},str{3},str{4},str{5},'location','NorthWest');
title('Posterior Distributions Affiliated with Function A');
annotation('textbox',[0.7 0.8 0.2 0.07],'String', ...
    {['$$\bar{C}$$ = ' sprintf('(%d,%d,%d,%d)',C)]},'interpreter','latex',...
    'LineStyle','none');
xlabel('Success Ratio')
hold off
```



APPENDIX D:
Frequency Weighting Implementation

Frequency weighting was a concept introduced in [13] to allow the engineer to subjectively influence a function-level posterior by accounting for the frequency with which a component type historically solved a function, thus accounting for the likelihood of a component type being selected over others. This was done through a discrete modification to the closed-form calculations in the EDRPM. Since the CDRPM uses MCMC sampling for Bayesian estimation of the hierarchical posterior parameters, the discrete modification could not be inserted into the process. Instead, the intent of frequency weighting was accomplished through data set manipulation by replicating data consistent with the historical frequencies.

Two different approaches to data replication were considered. The "within type" approach replicated the evidence within the component type, effectively multiplying the number of data points for a component type while maintaining the same total number of component types in the hierarchy. The "by type" approach replicated the whole component type data group, effectively multiplying the number of component types within the hierarchy. Both approaches were accomplished through MATLAB scripts rather than by manual manipulation of the original database to reduce the likelihood of errors for large frequencies and provide flexibility to evaluate different frequency vectors, $\bar{C}$s.

The MATLAB script for the two frequency weighting implementation methods is shown below in published format.

MATLAB Script Contents

- Data Manipulation for FW to be used by R (Function A Analysis)
- Set these Parameters and Read in Original Data
- FW Implementation: By Type
- FW Implementation: Within Type

Data Manipulation for FW to be used by R (Function A Analysis)

`MATLAB 2020a`

```
% This code produces two data sets accordint to two different
% implementation methods for frequency weighting - a manipulation within
% type and a manipulation by type.

% The within type replication requires the user-defined padcat() function
% in the MATLAB path.  The function concatenate vectors with different
% lengths by padding with NaN.

clear all, close all
```

Set these Parameters and Read in Original Data

```
C = [1,1,4,2];                      % Frequency vector
J = 4;                              % Original number of component types
evid = readtable('A_Data.csv');     % Original Data as A_Data.csv
```

FW Implementation: By Type

```
type1 = table2array(evid(:,1:3));
type2 = table2array(evid(:,4:6));
type3 = table2array(evid(:,7:9));
type4 = table2array(evid(:,10:12));

rep1 = [];
rep2 = [];
rep3 = [];
rep4 = [];

for i = 1:C(1)
    fill1 = type1;
    rep1 = horzcat(rep1,fill1);
    i = i+1;
end
```

```
for i = 1:C(2)
    fill2 = type2;
    rep2 = horzcat(rep2,fill2);
    i = i+1;
end

for i = 1:C(3)
    fill3 = type3;
    rep3 = horzcat(rep3,fill3);
    i = i+1;
end

for i = 1:C(4)
    fill4 = type4;
    rep4 = horzcat(rep4,fill4);
    i = i+1;
end

rep_data = horzcat(rep1,rep2,rep3,rep4);

% Arrange Replicated Data as Table for R
W = sum(C);

i = 1;
k = 1;
str = struct([]);
for c = 1:W
      str{k} = sprintf('Success_%d',i);
      str{k+1} = sprintf('Trials_%d',i);
      str{k+2} = sprintf('Ratio_%d',i);
      k = k+3;
      i = i+1;
```

```
end

R_Data= array2table(rep_data,'VariableNames',str);

% Write To File - Check name, will overwrite
str1 = sprintf('Data_for_R_%d%d%d%d_by_type.csv',C);
writetable(R_Data,str1);
```

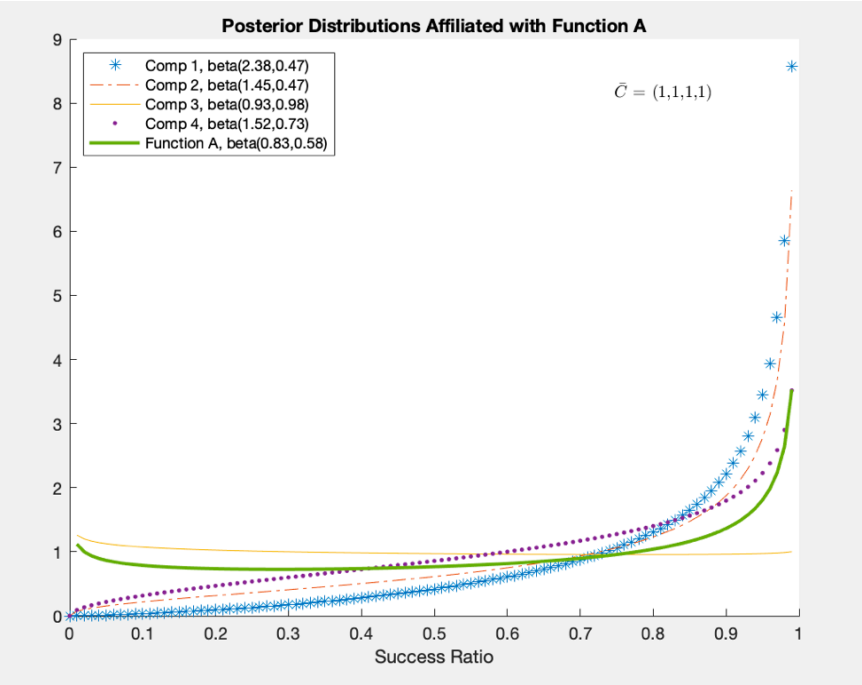
FW Implementation: Within Type

```
raw = struct([]);          % initialized for data without NaN
rep = struct([]);          % initialized weighted theta matrix
k = 0;

for j = 1:J
    for i = 1:3                        % for each column of evidence
        raw{k+i} = rmmissing(evid{:,k+i});      % data without NaN
        fill = ones(size(raw{k+i},1),C(j));
        filled = raw{k+i}.*fill;
        rep{k+i} = reshape(filled,[],1);
    end
    k = k+3;
end

weighted = padcat(rep{:});

R_Data2= array2table(weighted,'VariableNames',...
    evid.Properties.VariableNames);

% Write to File - Check name, will overwrite
str2 = sprintf('Data_for_R_%d%d%d%d_within_type.csv',C);
writetable(R_Data2,str2);
```

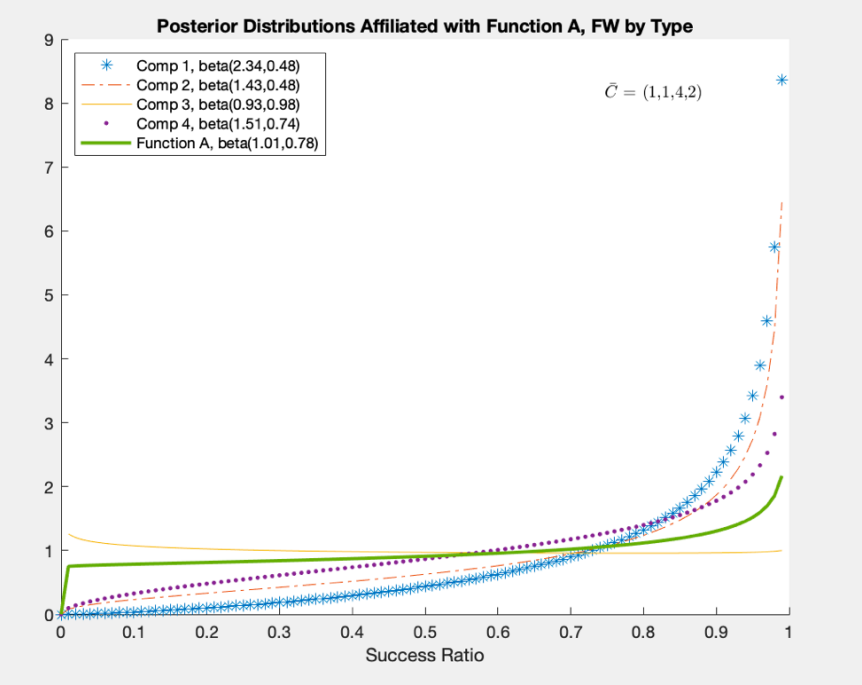
The modified data sets (in .csv format) were then imported into R in the same manner as the original data set was through a model writing code similar to that shown in Appendix B. The posteriors of the HB model were assessed for convergence by the same means shown in Appendix C. Significant aspects relative to frequency weighting implementation are most evident in the component posteriors with evidence graphs, as well as the component and function posterior pdf graphs. These graphs are consolidated in this appendix to illustrate the different implementation methods' influence on posterior distribution estimation.

The analysis that follows uses the frequency vector $\bar{C} = (1, 1, 4, 2)$ for *Function A* defined in Chapter 3.

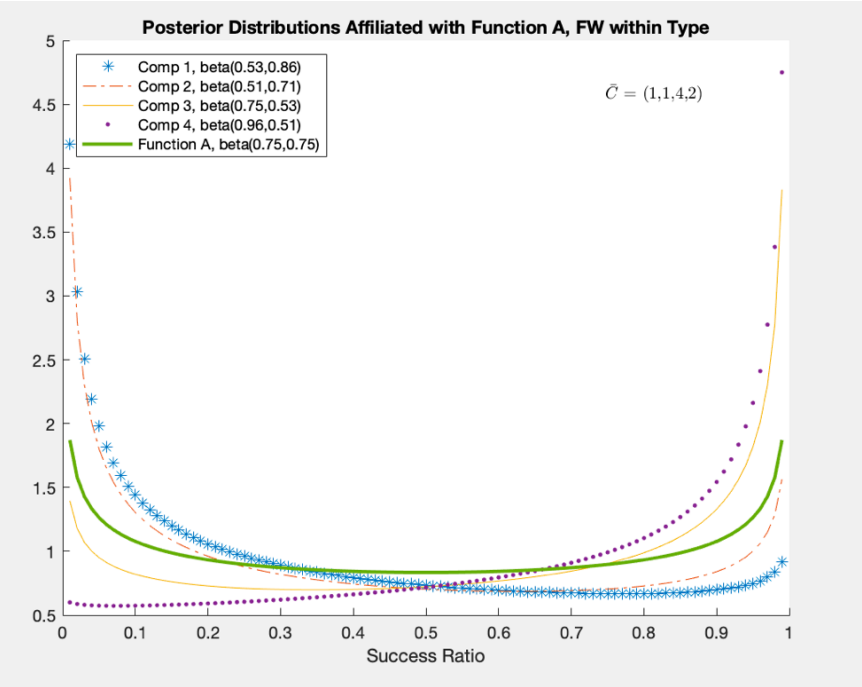
The component and function pdf graphs for each method are shown first, with the graph of unweighted results included as a point of reference. Since component type 3 had the greatest frequency, it was expected to see the function-level distribution more closely reflect the type 3 distribution. The within type implementation induced a greater influence on the function-level distribution than the by type implementation. This is likely because the within type implementation artificially reduced the variation in the component characterization since the data points are replicated. (This reduction of uncertainty was an undesirable consequence since it is unlikely an engineer's observation data research will always be exhaustive but was not significant enough to rule out the implementation method on its own.) The function-level distribution must then closely mirror the weighted component's distribution in order for the component type's more precise parameters to be drawn from it. Since the function posterior shifted significantly, it induced a greater amount of shrinkage on the other components, such that all posterior distributions had a greater resemblance of the most heavily weighted component type. The by type implementation showed the same trends but to a lesser extent, which suggested that the by type method was preferable so as to not over influence the hierarchy.



Unweighted Posterior Distributions for Reference

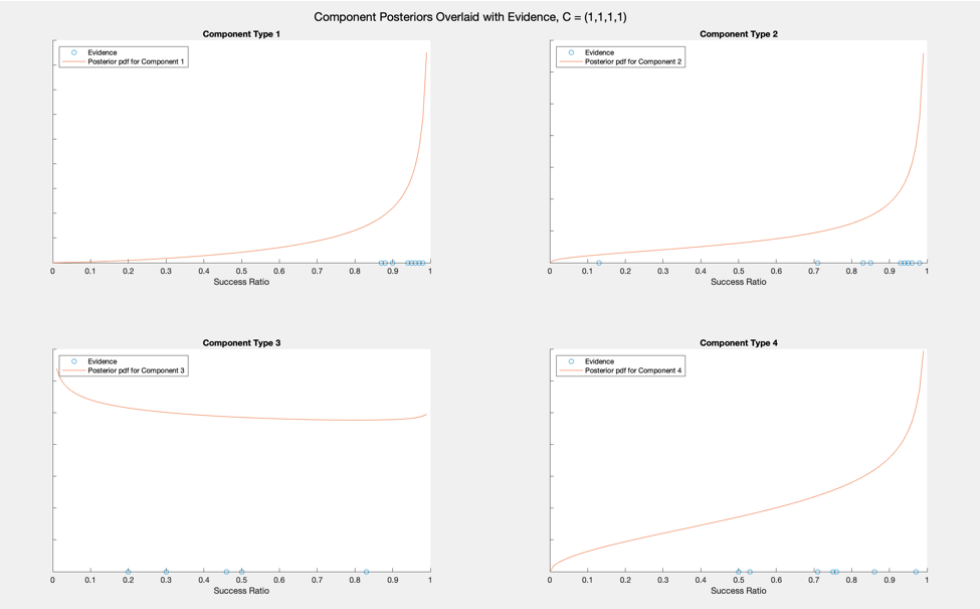


Posteriors Distributions for $\bar{C} = (1, 1, 4, 2)$ Implemented By Type

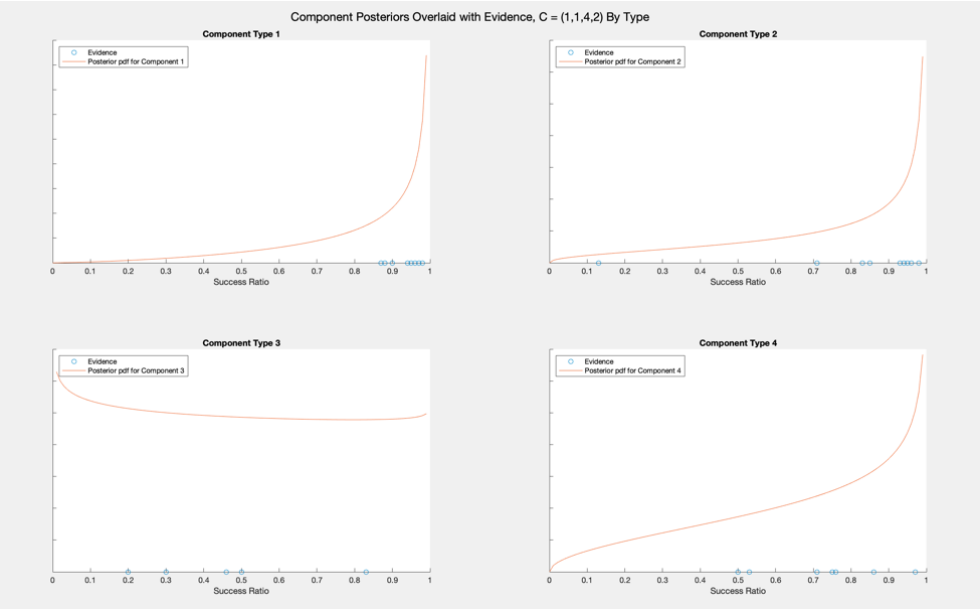


Posterior Distributions for $\bar{C} = (1, 1, 4, 2)$ Implemented Within Type

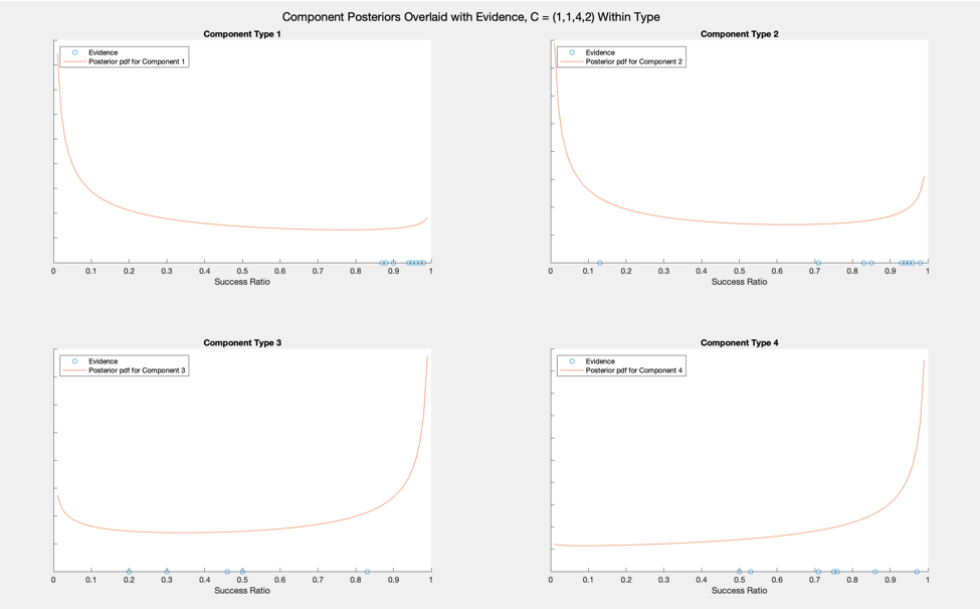
A closer look at the component type posteriors overlaid with evidence, the "gut-check" presentation, illustrated that the within type implementation was inappropriate. The by type implementation posteriors still reasonably trended with the original evidence. The within type implementation posteriors, specifically for component types 1 and 2, were so different from the unweighted case that they did not reflect original data clusters. Frequency weighting by type was therefore the implementation method selected for the CDRPM.



Unweighted Component Posteriors for Reference



Component Posteriors for $\bar{C} = (1, 1, 4, 2)$ Implemented BY Type



Component Posteriors for $\bar{C} = (1, 1, 4, 2)$ Implemented Within Type

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E:
Corrected EDRPM Implementation Script

The following script was generated for MATLAB to implement the posterior determination aspect of the EDRPM for the generic system presented in Chapter 4, originally presented in [13]. Not only did this script enable the subsequent application of the EDRPM for use in Chapter 5's case study, but it verified the scale modification used to manipulate data for the MCMC tool.

The two specific errors noted in the original script are detailed here and commented in the actual code. The first was in passing the expected value of τ from equation 2.4 to equations 2.5 and 2.6. Since equations 2.5 and 2.6 are functions of τ, equation 2.4 could be written as a very complicated function of τ, as alluded to [24] for the unmodified versions of the equations (no frequency weighting). Implementing this approach is highly error prone using symbolic math in a programming language, so the EDRPM prescribes solving equation 2.4 for a range of values and then calculating the expected value of τ. The expected value of τ is then substituted into equations 2.5 and 2.6. In the original script, the last value of τ from a *for loop* for $p(\tau|y)$ was passed to calculate μ and V_μ, rather than the expected value of τ. The second error was that equation 2.6 is for the inverse of V_μ, and the "J" modifier for frequency weighting was inconsistently applied. That is, the equation in the script was the argument for V_μ, not the inverse, but it was multiplied by J^{-1}.

MATLAB Contents

Corrected version of EDRPM

This script was developed using MATLAB 2020a and relies on the user-defined function padcat(), which concatenates vectors with different lengths by padding with NaN. This script implements the equations in the EDRPM by O'Halloran, et al, for Bayesian estimation of

reliability posteriors. It also served to verify the effect of the scalar modification of data for Stan.

```
clear, close
```

Parameters to Set

```
C = [1,1,1,1];        % FW vector for FW section
scale = 1E-6;            % Scale for units
```

Data and Early Arithmetic for FW

```
% Data from EDRPM Table 1, component means in terms of failures/Mhours:
ys1 = [10,9,9,8,7,9,10,11,12,8];    % comp1
ys2 = [3,4,9,5,1,7,2,4,10,7];       % comp2
ys3 = [3,5,3,2,2];                  % comp3
ys4 = [1,2,4,5,4,6,7,5,8,4];        % comp4

ys = padcat(ys1,ys2,ys3,ys4);       % each row is a component, fills in na

% Frequency Weighting
W=sum(C);                           % sum of all frequencies

% Count Comp
J=length(C);                        % total number of comp for the function

% Data scale manipulation
y = ys.*scale;
ss = 1.5*scale;                     % assumed by EDRPM

% As cell arrays...
yb = cell(1,J);
s = cell(1,J);
```

```
w = cell(1,J);

%Sample variance (ss), mean (yb), and freuquency weight (w) by comp type
for i = 1:J
    yb{i} = mean(y(i,:),'omitnan');
    s{i} = ss^2/size(rmmissing(ys(i,:)),2);
    w{i} = C(i)/W;
end

% As numerical arrays...
yb = cell2mat(yb);
s = cell2mat(s);
w = cell2mat(w);
```

Bayesian Estimation

```
% Set up for loop to determine expected value of tau
step = scale/20;
N = scale*100;
pt = 1;             % uniform prior of 1
tt = 0;
t2 = [];

norm = J;           % Normalization factor for FW; experimented with during
                    % error interrogation
                    % Set to J for EDRPM equations

% For loop for tau for various values
 for t = 0:step:N
    tt=tt+1;

    NumeratorMuHatA = cell(1,J);
    DenominatorMuHatA = cell(1,J);
```

```
    VmuA_arg = cell(1,J);

    %Estimated function level mean (MuHat) and error (Vmu)
    for i = 1:J
        NumeratorMuHatA{i} = sum((1/(s(i)+t^2))*w(i)*yb(i));
        DenominatorMuHatA{i} = sum((1/(s(i)+t^2))*w(i));
        VmuA_arg{i} = (w(i)/(s(i)+t^2));
    end

    NumeratorMuHatA = cell2mat(NumeratorMuHatA);
    DenominatorMuHatA = cell2mat(DenominatorMuHatA);
    VmuA_arg = cell2mat(VmuA_arg);

    MuHatA=sum(NumeratorMuHatA)/sum(DenominatorMuHatA);

    VmuA=1/(norm*sum(VmuA_arg));

    Tau_arg = cell(1,J);

    for i = 1:J
        Tau_arg{i} = (1/sqrt(s(i)+t^2)*exp(-1*(yb(i)-MuHatA)^2/...
        (2*(s(i)+t^2))))^w(i);
    end

    Tau_arg = cell2mat(Tau_arg);

    ptyA(tt) = pt*sqrt(VmuA)*(prod(Tau_arg)^norm);

    t2=[t2;t];
 end

% Calculate expected value of tau
ExtA=sum(ptyA*t2)/sum(ptyA);
```

```
    % This is the value that should be passed the following equations;
    % t2 was instead (the last value from the for loop)

% Used expectued value of tau to calculate mu_hat and V_mu
for i = 1:J
    NumeratorMuHatA(i) = sum((1/(s(i)+ExtA^2))*w(i)*yb(i));
    DenominatorMuHatA(i) = sum((1/(s(i)+ExtA^2))*w(i));
    VmuA_arg(i) = (w(i)/(s(i)+ExtA^2));
end

MuHatA=sum(NumeratorMuHatA)/sum(DenominatorMuHatA);

VmuA=1/(norm*sum(VmuA_arg));
    % This equation was not written as the inverse but the inverse of J
    % (norm in this form) was applied.

% Continue to Component Level for theta (Th) and V_c
Th = cell(1,J);
Vc = cell(1,J);

for i = 1:J
  Th{i} = ((1/s(i))*yb(i)+(1/ExtA^2)*MuHatA)/((1/s(i))+(1/ExtA^2));
  Vc{i} = 1/((1/s(i))+(1/ExtA^2));
end

Th = cell2mat(Th);
Vc = cell2mat(Vc);

fprintf('Closed form results with frequncies  [')
fprintf('%d ', C)
fprintf(']\t')
fprintf('Scale of %1.0e:\n',scale)
fprintf(' MuHatA = %4.4e\n ExtA = %4.4e\n VmuA = %4.4e\n', MuHatA, ExtA, VmuA)
```

```
for i = 1:J
    fprintf('  Theta %d = %4.4e\n',i,Th(i))
end
for i = 1:J
    fprintf('   Vc %d = %4.4e\n',i,Vc(i))
end
fprintf('\n')
```

```
Closed form results with frequncies  [1 1 1 1 ] Scale of 1e-06:
 MuHatA = 5.5296e-06
 ExtA = 5.5326e-06
 VmuA = 7.7227e-12
  Theta 1 = 9.2725e-06
  Theta 2 = 5.2024e-06
  Theta 3 = 3.0366e-06
  Theta 4 = 4.6068e-06
   Vc 1 = 2.2336e-13
   Vc 2 = 2.2336e-13
   Vc 3 = 4.4348e-13
   Vc 4 = 2.2336e-13
```

APPENDIX F:
Case Study: Launcher System Reliability Prediction

The complete data set used to facilitate the launcher case study is shown in the table below.

C	ID	Publication Label	Failure Rate (Fail/MHours)	Total Failed	Total Tested	Total Passed	Success Ratio	Source (If Less Than Highest Roll Up)	Reference
	A	**Condition Energy**							
1	**A1**	**Comp Type 1: Inverter**							
	A11	Inverter, Power	4.141067	24	48	24	0.50000	(Commercial, GF)	NPRD-2016
	A21	Inverter, Power, Static, 2.50KVA	6.054747	22	5411	5389	0.99593		NPRD-2016
	A31	Inverter	7.407239	25	5411	5386	0.99538	(Military, AC)	NPRD-2016
2	**A2**	**Comp Type 2: Transformer**							
	A12	Transformer Assembly	19.980291	299	840	541	0.64405	(Military, N, 800107-000)	NPRD-2016
	A22	Transformer Set	1.631133	9	129	120	0.93023		NPRD-2016
	A32	Transformer Subassembly	0.348281	1	5411	5410	0.99982		NPRD-2016
	A42	Transformer	15.459774	3	3070	3067	0.99902	(Military, AIF,18212-000)	EPRD-2014
	A52	Transformer, Power	0.395223	6	1401	1395	0.99572	(Military, AUF)	EPRD-2014
	B	**Actuate Energy**							
1	**B1**	**Comp Type 1: Circuit Breaker**							
	B11	Circuit Breaker, Current Trip	5.264201	42	3200	3158	0.98688		NPRD-2016
	B21	Circuit Breaker, Current / Voltage Trip	25.028537	27	1250	1223	0.97840		NPRD-2016
	B31	Circuit Breaker, Fixed, NO	0.617280	10	1703	1693	0.99413		NPRD-2016
	B41	Circuit Breaker, Metal Clad, NO	1.415562	17	124	107	0.86290		NPRD-2016
	B51	Circuit Breaker, Molded Case	1.107460	29	3864	3835	0.99249		NPRD-2016
	B61	Circuit Breaker, Power Switch	2.159443	76	4104	4028	0.98148		NPRD-2016
	B71	Circuit Breaker, Trip Free	97.938106	144	555	411	0.74054	(Military, A)	NPRD-2016
	B81	Circuit Breaker, Under Voltage	1.870033	8	350	342	0.97714	(Military)	NPRD-2016
	B91	Circuit Breaker, Vacuum, NO	1.857448	51	118	67	0.56780		NPRD-2016
2	**B2**	**Comp Type 2: Switch**							
	B12	Switch	9.675859	2	444	442	0.99550	(Military, AIA)	EPRD-2014
	B22	Switch, Arm and Disarm	0.537599	35	2015	1980	0.98263		EPRD-2014
	B32	Switch, Electric, Breaker Type, Non-Knife	0.300270	54	1163	1109	0.95357		EPRD-2014
3	**B3**	**Comp Type 3: Relay**							
	B13	Relay, Electromagnetic, General Purpose	18.882759	13	538	525	0.97584	(Military)	EPRD-2014
	B23	Relay, Electromechanical, Armature	23.283610	6	908	902	0.99339	(Military)	EPRD-2014
	B33	Relay, Electronic	67.256603	7	13	6	0.46154	(Military, A)	EPRD-2014
	B43	Relay, Solenoid, Solenoid-Driven	4.140519	64	1234	1170	0.94814		EPRD-2014
	B53	Relay, Time Delay	1.682544	9	384	375	0.97656	(Commercial, GF, 18354-000)	EPRD-2014
	C	**Convert Energy**							
1	**C1**	**Comp Type 1: Fan**							
	C11	Fan, Axial	2.211246	39	2192	2153	0.98221	(Unknown, NS)	NPRD-2016
	C21	Fan, Blade	6.880370	10	114	104	0.91228		NPRD-2016
	C31	Fan, Centrifugal	13.880011	20	239	219	0.91632	(Unknown, GF, 18459-000)	NPRD-2016
	C41	Fan Assembly, Centrifugal	1.158800	5	129	124	0.96124		NPRD-2016
2	**C2**	**Comp Type 2: Compressor**							
	C12	Compressor	10.716847	3	327	324	0.99083	(Unknown)	NPRD-2016
	C22	Compressor, Air	11.004983	27	129	102	0.79070	(Military)	NPRD-2016
	C32	Compressor, Centrifugal	1.268456	3	129	126	0.97674	(Military)	NPRD-2016
	C42	Compressor, Rotary	5.243973	135	700	565	0.80714		NPRD-2016
	C52	Compressor, Assembly	1.228916	3	129	126	0.97674		NPRD-2016
1	**C3**	**Comp Type 3: Pump**							
	C13	Pump, Axial	0.422819	1	129	128	0.99225		NPRD-2016
	C23	Pump, Centrifugal	5.069158	127	1320	1193	0.90379	(Military, N, 800108-000)	NPRD-2016
	C33	Pump, Compressor	30.832536	71	140	69	0.49286		NPRD-2016
	C43	Pump, Electric	2.486788	6	129	123	0.95349		NPRD-2016
	C53	Pump, Fuel	3.756554	22	5411	5389	0.99593	(Military, AC)	NPRD-2016
	C63	Pump, Hydraulic, Piston, Axial	7.722008	2	28	26	0.92857		NPRD-2016
	C73	Pump, Pneumatic, Air	69.169252	237	2314	2077	0.89758		NPRD-2016
	C83	Pump, Pressure	0.986149	6	5411	5405	0.99889		NPRD-2016
	C93	Pump, Rotary	30.858781	11	839	828	0.98689	(Military, AA)	NPRD-2016

The following code was used to ensure there were 10,000 positive failure rates, only, drawn from the failure rate posterior distributions to support further analyses.

Illustration of Positive-Only Failure Rate Samples

Contents

Function A, Condition, C = (1,2)

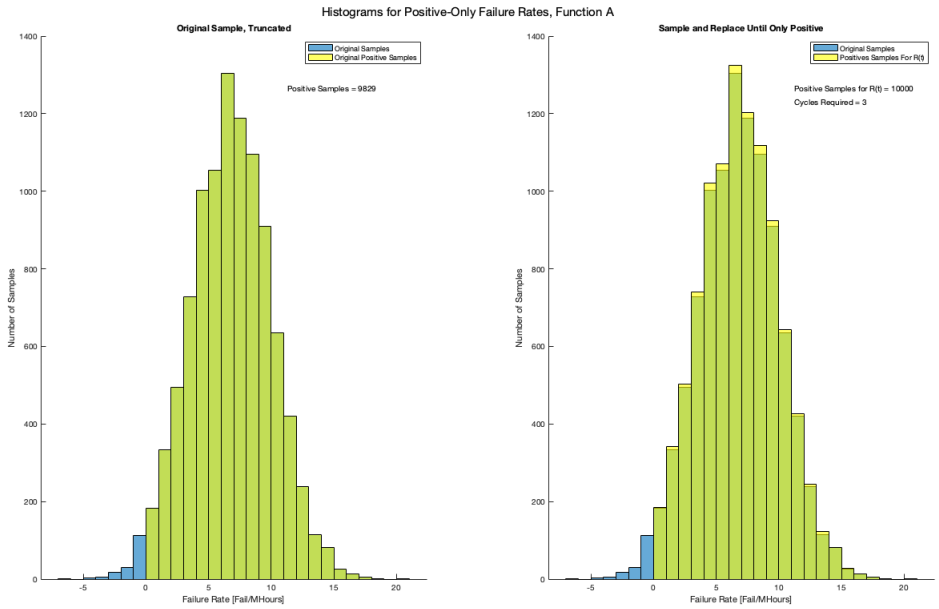
Mu, Tau, Theta 1, Theta 2

```
Cond = [6.84775622884608,3.18116370381325,6.29108939733541,7.48178001049368];

MuA = Cond(1);
TauA = Cond(2);
sA = normrnd(MuA,TauA,[10000,1]);
check = find(sA > 0);
posA = sA(check);

fprintf('Number of positive elements in original: %d\n', length(check))

use = sA;
neg = find(use < 0);
off = length(neg);

count = 1;
while off ~= 0
    replace = normrnd(MuA,TauA,[length(neg),1]);
    use(neg) = replace;
    neg = find(use < 0);
```

```
    off = length(neg);
    count = count+1;
end

check2 = find(use > 0);

fprintf('Number of positive elements in final: %d\n',length(check2))
fprintf('Cycles for to replace: %d\n',count)

figure
set(gcf,'WindowState','fullscreen')
subplot(1,2,1)
hold on
histogram(sA,'BinWidth',1)
histogram(posA,'BinWidth',1,'FaceColor','y')
legend('Original Samples','Original Positive Samples')
annotation('textbox',[0.34 0.82 0.13 0.02],'String', ...
    {['Positive Samples = ' sprintf('%d',length(check))]},...
    'LineStyle','none');
ylabel('Number of Samples')
xlabel('Failure Rate [Fail/MHours]')
title('Original Sample, Truncated')
hold off
subplot(1,2,2)
hold on
histogram(sA,'BinWidth',1)
histogram(use,'BinWidth',1,'FaceColor','y')
legend('Original Samples','Positives Samples For R(t)')
annotation('textbox',[0.78 0.82 0.13 0.02],'String', ...
    {['Positive Samples for R(t) = ' sprintf('%d',length(check2))]}, ...
    'LineStyle','none');
annotation('textbox',[0.78 0.8 0.13 0.02],'String', ...
    {['Cycles Required = ' sprintf('%d',count)]}, ...
```

```
    'LineStyle','none');
ylabel('Number of Samples')
xlabel('Failure Rate [Fail/MHours]')
title('Sample and Replace Until Only Positive')
hold off
sgtitle('Histograms for Positive-Only Failure Rates, Function A')
```

```
Number of positive elements in original: 9829
Number of positive elements in final: 10000
Cycles for to replace: 3
```



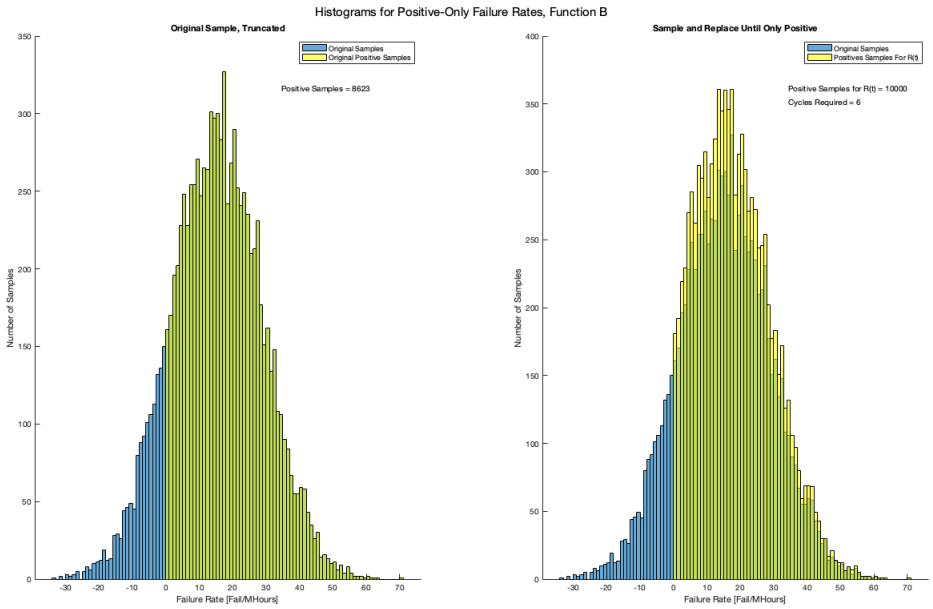
Function B, Actuate, C = (1,2,3)

Mu, Tau, Theta1, Theta2, Theta 3

```
Act = [15.0566069450050,13.6223233917751,15.2489807461578,...
3.55877735964277,23.0351290938322];
```

```
MuB = Act(1);
TauB = Act(2);
sB = normrnd(MuB,TauB,[10000,1]);
check = find(sB > 0);
posB = sB(check);

fprintf('Number of positive elements in original: %d\n', length(check))

use = sB;
neg = find(use < 0);
off = length(neg);

count = 1;
while off ~= 0
    replace = normrnd(MuB,TauB,[length(neg),1]);
    use(neg) = replace;
    neg = find(use < 0);
    off = length(neg);
    count = count+1;
end

check2 = find(use > 0);

fprintf('Number of positive elements in final: %d\n',length(check2))
fprintf('Cycles for to replace: %d\n',count)

figure
set(gcf,'WindowState','fullscreen')
subplot(1,2,1)
hold on
histogram(sB,'BinWidth',1)
histogram(posB,'BinWidth',1,'FaceColor','y')
```

```
legend('Original Samples','Original Positive Samples')
annotation('textbox',[0.34 0.82 0.13 0.02],'String', ...
    {['Positive Samples = ' sprintf('%d',length(check))]},...
    'LineStyle','none');
ylabel('Number of Samples')
xlabel('Failure Rate [Fail/MHours]')
title('Original Sample, Truncated')
hold off
subplot(1,2,2)
hold on
histogram(sB,'BinWidth',1)
histogram(use,'BinWidth',1,'FaceColor','y')
legend('Original Samples','Positives Samples For R(t)')
annotation('textbox',[0.78 0.82 0.13 0.02],'String', ...
    {['Positive Samples for R(t) = ' sprintf('%d',length(check2))]},'LineStyle','no
annotation('textbox',[0.78 0.8 0.13 0.02],'String', ...
    {['Cycles Required = ' sprintf('%d',count)]},'LineStyle','none');
ylabel('Number of Samples')
xlabel('Failure Rate [Fail/MHours]')
title('Sample and Replace Until Only Positive')
hold off
sgtitle('Histograms for Positive-Only Failure Rates, Function B')


Number of positive elements in original: 8623
Number of positive elements in final: 10000
Cycles for to replace: 6
```



Function C, Convert, C = (1,2,1)

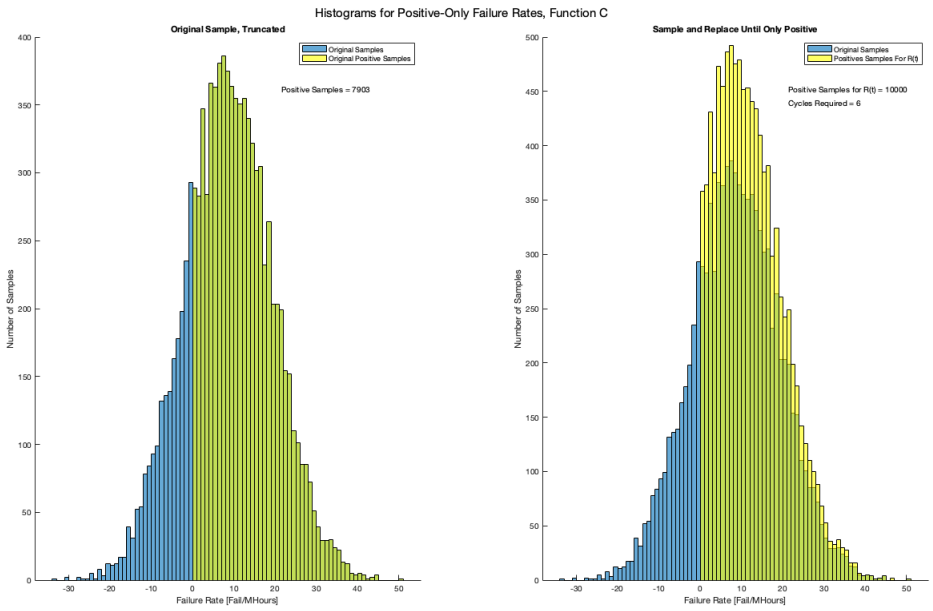
Mu, Tau, Theta1, Theta2, Theta 3

```
Conv = [8.56099537402864,10.5525194141036,6.04729258406312,...
5.90510504277764,16.7971826736931];

MuC = Conv(1);
TauC = Conv(2);
sC = normrnd(MuC,TauC,[10000,1]);
check = find(sC > 0);
posC = sC(check);

fprintf('Number of positive elements in original: %d\n', length(check))

use = sC;
neg = find(use < 0);
off = length(neg);
```

```
count = 1;
while off ~= 0
    replace = normrnd(MuC,TauC,[length(neg),1]);
    use(neg) = replace;
    neg = find(use < 0);
    off = length(neg);
    count = count+1;
end

check2 = find(use > 0);

fprintf('Number of positive elements in final: %d\n',length(check2))
fprintf('Cycles for to replace: %d\n',count)

figure
set(gcf,'WindowState','fullscreen')
subplot(1,2,1)
hold on
histogram(sC,'BinWidth',1)
histogram(posC,'BinWidth',1,'FaceColor','y')
legend('Original Samples','Original Positive Samples')
annotation('textbox',[0.34 0.82 0.13 0.02],'String', ...
    {['Positive Samples = ' sprintf('%d',length(check))]},...
    'LineStyle','none');
ylabel('Number of Samples')
xlabel('Failure Rate [Fail/MHours]')
title('Original Sample, Truncated')
hold off
subplot(1,2,2)
hold on
histogram(sC,'BinWidth',1)
histogram(use,'BinWidth',1,'FaceColor','y')
```

```
legend('Original Samples','Positives Samples For R(t)')
annotation('textbox',[0.78 0.82 0.13 0.02],'String', ...
    {['Positive Samples for R(t) = ' sprintf('%d',length(check2))]}, ...
    'LineStyle','none');
annotation('textbox',[0.78 0.8 0.13 0.02],'String', ...
    {['Cycles Required = ' sprintf('%d',count)]},'LineStyle','none');
ylabel('Number of Samples')
xlabel('Failure Rate [Fail/MHours]')
title('Sample and Replace Until Only Positive')
hold off
sgtitle('Histograms for Positive-Only Failure Rates, Function C')
```

```
Number of positive elements in original: 7903
Number of positive elements in final: 10000
Cycles for to replace: 6
```



THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] National Materials Advisory Board, "Enabling technologies for unified life-cycle engineering of structural components," in *Publication NMAB-445*, National Research Council. Washington, DC, USA: National Academy Press, 1991.

[2] B. S. Blanchard and W. J. Fabrycky, *Systems Engineering and Analysis*, 5th ed. Edinburgh Gate, Essex, England: Pearson, 2014.

[3] G. Molcho, A. Cristal, and M. Shpitalni, "Part cost estimation at early design phase," *CIRP Annals - Manufacturing Technology*, vol. 63, pp. 153–156, Jan 2014. Available: https://doi.org/10.1016/j.cirp.2014.03.107

[4] M. E. Saravi, L. B. Newnes, A. R. Mileham, and Y. M. Goh, "Estimating cost at the conceptual design stage to optimize design in terms of performance and cost," in *Collaborative Product and Service Life Cycle Management for a Sustainable World*. London, England: Springer, 2008, pp. 123–130. Available: https://doi.org/10.1007/978-1-84800-972-1_11

[5] D. G. Ullman, *The Mechanical Design Process*. Boston, MA, USA: McGraw-Hill, 2003.

[6] AcqNotes, "Program management: Total life cycle systems management," 2017, accessed April 8, 2020. Available: http://acqnotes.com/acqnote/careerfields/total-life-cycle-systems-management-tlcsm

[7] P. O'Connor and A. Kleyner, *Practical Reliability Engineering*. John Wiley & Sons, 2012.

[8] M. Modarres, *Risk Analysis in Engineering: Techniques, Tools, and Trends*. CRC press, 2006.

[9] L. M. Leemis, *Reliability: Probabilistic Models and Statistical Methods*. New Jersey, USA: Prentice Hall, 1995.

[10] *Report to the President by the Presidential Commission On the Space Shuttle Challenger Accident*, Washington, DC, USA, June 6, 1986, colloquially referred to as the "Rogers Commission Report" in honor of Committee Chairman William P. Rogers. Available: https://spaceflight.nasa.gov/outreach/SignificantIncidents/assets/rogers_commission_report.pdf.

[11] B. M. O'Halloran, R. B. Stone, and I. Y. Tumer, "Early design stage reliability analysis using function-flow failure rates," in *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2011, pp. 455–464.

[12] B. M. O'Halloran, C. Hoyle, R. B. Stone, and I. Y. Tumer, "A method to calculate function and component failure distributions using a hierarchical bayesian model and frequency weighting," in *ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2012, pp. 727–736.

[13] B. M. O'Halloran, C. Hoyle, I. Y. Tumer, and R. B. Stone, "The early design reliability prediction method," *Research in Engineering Design*, vol. 30, pp. 489–508, 2019. Available: https://doi.org/10.1007/s00163-019-00314-8

[14] Y. Cheng and X. Du, "System reliability analysis with dependent component failures during early design stage - a feasibility study," *Journal of Mechanical Design*, vol. 138, no. 5, pp. 051 405–1–051 405–12, 2016. Available: https://doi.org/10.1115/1.4031906

[15] B. Bergman and M. Ringi, "System reliability prediction using data from non-identical environments," *Reliability Engineering and System Safety*, vol. 58, no. 3, pp. 185–190, 1997. Available: https://doi.org/10.1016/S0951-8320(97)00051-3

[16] V. E. Johnson, A. Moosman, and P. Cotter, "A hierarchical model for estimating the early reliability of complex systems," *IEEE Transactions on Reliability*, vol. 54, no. 2, pp. 224–231, June 2005. Available: https://doi.org/10.1109/TR.2005.847262

[17] Boeing, "Boeing statement on AOA disagree alert," May 15, 2019, accessed on April 14, 2020. Available: https://boeing.mediaroom.com/news-releases-statements?item=130431

[18] *Navy Columbia (SSBN-826) Class Ballistic Missile Submarine Program: Background and Issues for Congress*. Washington, DC, USA: Congressional Research Service, Update March 22, 2020. Available: https://fas.org/sgp/crs/weapons/R41129.pdf

[19] W. O. Baun III, "Method for the early development of a bayesian prior time-to-failure distribution for an evolutionary product design," in *International Mechanical Engineering Congress and Exposition*. Seattle, WA, USA: ASME, November 11-15, 2017, vol. 14, Safety Engineering, Risk Analysis and Reliability Methods, pp. 23–31. Available: https://doi.org/10.1115/IMECE2007-41022

[20] M. Mayda and S.-K. Choi, "A reliability-based design framework for early stages of design process," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 39, no. 6, pp. 2105–2120, 2017. Available: https://doi.org/10.1007/s40430-017-0731-y

[21] D. Marquez, M. Neil, and N. E. Fenton, "A new Bayesian network approach to reliability modelling," 2007. Available: https://www.eecs.qmul.ac.uk/~norman/papers

[22] C. J. Bashe, L. R. Johnson, J. H. Palmer, and E. W. Pugh, *IBM's Early Computers*. Cambridge, MA, USA: MIT Press, 1986.

[23] Computer History Museum, "1991: Solid state drive module demonstrated," 2020, accessed April 16, 2020. Available: https://www.computerhistory.org/storageengine/solid-state-drive-module-demonstrated/

[24] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*, 3rd ed. (Texts in Statistical Science). Boca Raton, FL, USA: CRC Press, 2014.

[25] A. G. Papadopoulos, "A hierarchical approach to the study of the exponential failure model," *Communications in Statistics - Theory and Methods*, vol. 18, no. 12, pp. 4375–4392, 1989. Available: http://www.tandfonline.com/doi/abs/10.1080/03610928908830161

[26] J. K. Kruschke and W. Vanpaemel, *The Oxford Handbook of Computational and Mathematical Psychology: Bayesian Estimation in Hierarchical Models*, 1st ed. Oxford, England: Oxford University Press, 2015. Available: https://doi.org/10.1093/oxfordhb/9780199957996.013.13

[27] J. Orloff and J. Bloom, "Conjugate priors: Beta and normal," class notes for Class 15, 18.05 Introduction to Probability and Statistics, MIT OpenCourseWare, Spring 2014. Available: https://ocw.mit.edu

[28] J. Rocca, "Bayesian inference problem, mcmc and variational inference," July 1, 2019, accessed on April 8, 2020. Available: https://towardsdatascience.com/bayesian-inference-problem-mcmc-and-variational-inference-25a8aa9bce29

[29] R. Argwal, "MCMC intuition for everyone," June 3, 2019, accessed on April 8, 2020. Available: https://towardsdatascience.com/mcmc-intuition-for-everyone-5ae79fff22b1

[30] D. Mahar, W. Fields, and J. Reade, *Electronic Parts Reliability Data 2014* (Reliability Databook Series). Utica, NY, USA: Quanterion Solutions Incorporated, 2014.

[31] D. Mahar, W. Fields, and J. Reade, *Non-electronic Parts Reliability Data Publication 2016* (Reliability Databook Series). Utica, NY, USA: Quanterion Solutions Incorporated, 2016.

[32] J. K. Kruschke, *Doing Bayesian Data Analysis : A tutorial with R, JAGS, and Stan*, 2nd ed. Amsterdam, North Holland, Netherlands: Elsevier Science & Technology, December 30, 2014.

[33] *Stan User's Guide*. Version 2.23 [Online]. Available: https://mc-stan.org/docs/2_23/stan-users-guide/index.html#overview. Accessed April 26, 2020.

[34] A. Hernandez, "Test and evaluation basics," class notes for SE3100 Fundamentals of Systems Engineering, Dept. of Sys. Eng., Naval Postgraduate School, Monterey, CA, USA, fall 2018.

[35] *Stan Reference Manual*. Version 2.23 [Online]. Available: https://mc-stan.org/docs/2_23/reference-manual/index.html. Accessed April 26, 2020.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California