SCIENTIFIC TEST & ANALYSIS TECHNIQUES
CENTER OF EXCELLENCE

Case Study on Bayesian Analysis and Test Planning for Binary Reliability Evaluation

July, 2025

Corinne Stafford, Ctr

To develop and apply independent, tailored Scientific Test & Analysis Techniques solutions to Test and Evaluation that deliver insight to inform better decisions.

About this Publication:
This work was conducted by the Scientific Test & Analysis Techniques Center of Excellence under contract FA8075-18-D-0002, Task FA8075-21-F-0074.

For more information:
Visit, www.AFIT.edu/STAT
Email, AFIT.ENS.STATCOE@us.af.mil
Call, 937-255-3636 x4736

Technical Reviewers:
Jim Theimer
Wayne Adams

Scientific Test & Analysis Techniques Center of Excellence
2950 Hobson Way
Wright-Patterson Air Force Base, Ohio

Version: 1, FY25

Abstract

Follow-on operational test and evaluation (FOT&E) and agile approaches to system development need rigorous ways to determine the amount of testing required. These systems change in small increments over time, and traditional frequentist methods for sizing tests that do not account for prior data result in unrealistically large sample sizes for testing the latest update. Bayesian methods mathematically account for previous system data, enabling reduced sample sizes while maintaining scientific rigor. This paper presents a Bayesian power prior framework for evaluating binary reliability of systems undergoing agile development. Using this framework, the paper discusses how to plan the number of trials required for a future update. Additionally, the results of application to a specific program are discussed.

Keywords: Bayesian analysis, follow-on test & evaluation, test planning, reliability

Table of Contents

Introduction

Many systems undergo follow-on operational test and evaluation (FOT&E), a phase of testing that occurs during fielding/deployment to ensure systems continue to meet operational needs after changes to environment, threats, or the system itself (AcqNotes, 2024). System changes may include hardware or software updates to the system to increase effectiveness. Continually updating the system in this way constitutes an agile approach to system development.

A frequent point of contention in FOT&E is determining the number of trials needed to evaluate an updated system's reliability before deployment. Traditionally, frequentist statistical methods are applied to size a test with acceptable risks of passing an unreliable system and failing a reliable system. However, frequentist methods do not account for the large body of data from testing a system prior to the most recent update, and test teams instinctively realize that frequentist size estimates are too large for this reason. As a result, test planning becomes a bargaining process between stakeholders to determine the number of runs that "feels right." The danger of this subjective process is that tests start to skew toward too few runs, where unreliable systems may pass undetected.

Bayesian statistical methods provide a solution to this problem with the capability to rigorously leverage previous data and quantify statistical risk associated with the size of a test. To account for differences between agile system updates, Bayesian power priors may be applied to downweight the influence of prior data when evaluating reliability of the most recent system. These methods provide a defensible way to reduce the size of tests compared to the frequentist approach.

This paper first provides background on key concepts in Bayesian statistics including Bayes' rule, conjugate priors, and power priors. Next the paper explains the Bayesian framework applied to evaluate reliability of systems undergoing agile development. Provided this framework, the paper then discusses how to plan the number of trials required for a future update. Additionally, the results of application to a specific program are discussed.

Background

Bayesian statistics expresses probability as a degree of belief, where beliefs can be influenced by prior data, expert knowledge, or natural/physical bounds. It is a different philosophy than frequentist statistics, which defines probability to be the long-run probability of an event after repeated experimentation. In contrast to frequentist statistics, Bayesian statistics provide more flexibility through its ability to incorporate prior data into analyses while also providing easier-to-interpret results.

Bayes' Rule

Bayes' rule is the theoretical backbone of Bayesian statistics and provides the mathematical mechanism for combining prior and current data. The key components of Bayes' rule are the prior distribution, which captures prior knowledge about the distribution of some parameter, the likelihood function, which captures the current data, and the posterior, which combines prior and current data into a new distribution describing updated beliefs about the parameter. The parameter refers to some parameter of a distribution or statistical model. For example, missiles either fire without issue (success) or experience some type of failure. This binary outcome is modeled with a binomial likelihood, with parameter θ, the probability of missile success.

While Bayes' rule can be written many ways, Equation 1 gives one version, where θ is the parameter, D is the current data, $p(\theta|D)$ is the posterior distribution of θ given the observed data, $L(\theta|D)$ is the likelihood of parameter values given the data, and $p(\theta)$ is the prior distribution of θ.

$$p(\theta|D) \propto L(\theta|D)p(\theta) \tag{1}$$

Equation 1 expresses Bayes' rule as a proportionality; to find the posterior exactly, the product on the right must be normalized to integrate to one, as required for a valid probability distribution. This normalization constant, however, can be difficult to compute analytically. Instead, numerical methods such as Markov Chain Monte Carlo (MCMC) are often employed to sample from the posterior, enabling analysis without a closed form expression.

Conjugate Priors

A conjugate prior is a specific type of prior that yields a closed-form solution for the posterior. The distribution type of the conjugate prior (e.g., normal distribution, gamma distribution) is dictated by the likelihood function. The posterior distribution type is the same as the conjugate prior, and simple updating schemes can be derived to determine the new hyperparameters of the posterior distribution.

The binomial likelihood function has a beta distribution conjugate prior. The beta distribution $\mathrm{Beta}(\alpha, \beta)$ describes how θ, the probability of success, is distributed, based on two shape hyperparameters α and β, which represent the number of successes and failures, respectively. The posterior distribution is derived according to Equation 2, where s is the number of successes in the data, f is the number of failures in the data, α_1 and β_1 are the posterior hyperparameters, and α_0 and β_0 are the prior hyperparameters.

$$p(\theta|n,k) = \mathrm{Beta}(\alpha_1, \beta_1) = \mathrm{Beta}(\alpha_0 + s, \beta_0 + f) \tag{2}$$

To summarize, the beta posterior hyperparameters are the total number of successes and failures from the current data and prior combined.

A common issue faced in test and evaluation is that prior and current data are not interchangeable, in other words, a prior failure should not be "counted as much" as a current failure since the system underwent changes or fixes since prior failures were observed. This issue motivates the need for power priors.

Power Priors

A power prior (Ibrahim et al., 2015) is a type of prior that incorporates prior data, but downweights it in comparison to current data. Equation 3 gives the general form of a power prior that incorporates multiple previous datasets, where n is the number of previous datasets, D_i is the i^{th} dataset, w_i is the weight applied to the i^{th} dataset, and $p_0(\theta)$ is the initial prior.

$$p(\theta|D_1, \dots, D_n, w_1, \dots, w_n) \propto \left(\prod\nolimits_{i=1}^{n} L(\theta|D_i)^{w_i}\right) \cdot p_0(\theta) \tag{3}$$

Each weight is a real number between zero and one, where one indicates that the dataset receives full weight in the analysis and zero indicates no weight is given. The initial prior

represents beliefs before observing any data and may be non-informative or informative.

The posterior when using a power prior framework is given by Equation 4, where D_c is the current data. Unlike previous datasets, the current data is not downweighted.

$$p(\theta|D_c, D_1, \dots, D_n, w_1, \dots, w_n) \propto L(\theta|D_c)\left(\prod_{i=1}^{n} L(\theta|D_i)^{w_i}\right) \cdot p_0(\theta) \quad (4)$$

Bayesian Power Prior Framework

Systems undergoing agile development have a wealth of previous data to support reliability assessments. However, this previous data was collected prior to software or hardware updates to the system and is not interchangeable with new test data. Therefore, it is appropriate to use a power prior framework, where data from prior system versions is downweighted.

Within this power prior framework, a binomial likelihood and beta conjugate prior are appropriate for modelling a pass/fail reliability response. Equation 5 gives the equation for the prior using this framework, where s_i and f_i are the number of successes and failures, respectively, in the i^{th} prior dataset, $L_{\text{bin}}(\theta|s_i, f_i)$ is the binomial likelihood given s_i success and f_i failures, and $\text{Beta}(\alpha_0, \beta_0)$ is the initial Beta prior.

$$p(\theta|s_1, \dots, s_n, f_1, \dots, f_n, w_1, \dots, w_n) \propto \left(\prod_{i=1}^{n} L_{\text{bin}}(\theta|s_i, f_i)^{w_i}\right) \cdot \text{Beta}(\alpha_0, \beta_0) \quad (5)$$

The initial prior parameters α_0 and β_0 can be chosen so that the initial prior is weakly informative or non-informative. A good prior choice is often $\text{Beta}(1,1)$, which is flat, giving equal probability to all possible parameter values between zero and one.

Due to beta/binomial conjugacy, it can be shown that the posterior is a beta distribution with parameters as given in Equation 6, where s_c and f_c are the number of successes and failures, respectively, in the current dataset.

$$\text{Posterior} = \text{Beta}\left(\alpha_0 + \sum_{i=1}^{n} w_i s_i + s_c, \beta_0 + \sum_{i=1}^{n} w_i f_i + f_c\right) \quad (6)$$

To apply this framework, the weight w_i for each data set must be determined. These weights should be set according to the similarity between data set i and the current data set. A more intuitive way to set weights may be to first determine weights relative to intermediate versions in sequence. For example, consider a system with two prior hardware versions (v1.0, v2.0) each with a software update (v1.1, v2.1). Evidence may show that software changes do not have much effect on reliability so a higher weight (e.g., 0.8) may be suitable for v1.1 compared to v1. On the other hand, hardware changes (v1.1 to v2.0) may have a larger effect on reliability and are better suited to a small weight (e.g., 0.2). Weights defined relative to intermediate versions may be multiplied together to determine weights relative to the current version. When assessing the reliability of a new v3.0, for example, v2.1 receives a weight of 0.2 (hardware change), v2.0 receives a weight of 0.2*0.8 = 0.16 (hardware and software change).

Table 1 demonstrates for notional system data how the posterior is calculated, where the posterior hyperparameters are the sum of equivalent successes and failures contributed by the initial prior and each version's data.

Table 1

Notional Example of Posterior Calculation with Power Prior Framework

	Initial Prior	v1.0	v1.1	v2.0	v2.1	v3.0		Posterior
Successes	1	7	3	8	5	9		
Failures	1	3	2	2	0	1		
Weight (relative to next version)		0.8	0.2	0.8	0.2	1		
Weight (relative to v3.0)		0.0256	0.032	0.16	0.2	1		
Equivalent successes	1	0.1792	0.096	1.28	1	9	α	12.5552
Equivalent failures	1	0.0768	0.064	0.32	0	1	β	2.4608

The posterior, weighted likelihoods, and initial prior are shown in Figure 1. The posterior can be understood as combining all weighted likelihoods and the initial prior. The most down-weighted likelihoods appear nearly flat, meaning they contribute little to the shape of the resulting posterior. A plot of weighted likelihoods can also be helpful for determining if chosen weights are appropriate. For instance, if the prior data set's weighted likelihood shows a small or near-zero probability at a parameter value that would be plausible for the newest version, the weight may not be small enough.

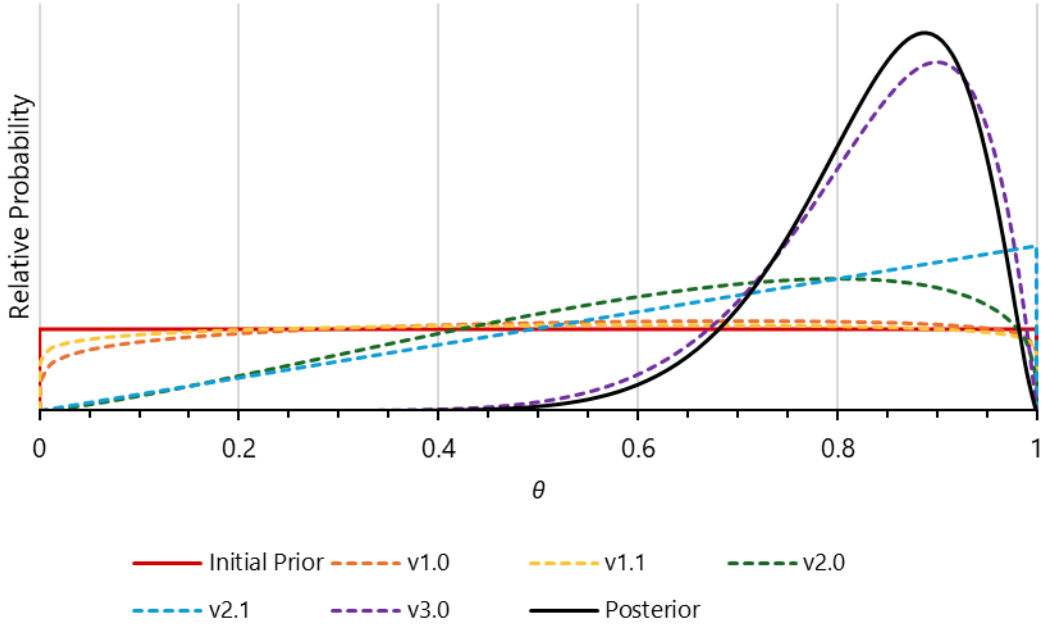

Figure 1

Initial Prior, Weighted Likelihoods (Dashed Lines), and Posterior for Notional Example

Another view is shown in Figure 2, which shows the posterior distribution obtained for each version. Figure 2 shows how the posterior shifts toward higher reliabilities as the system improves.

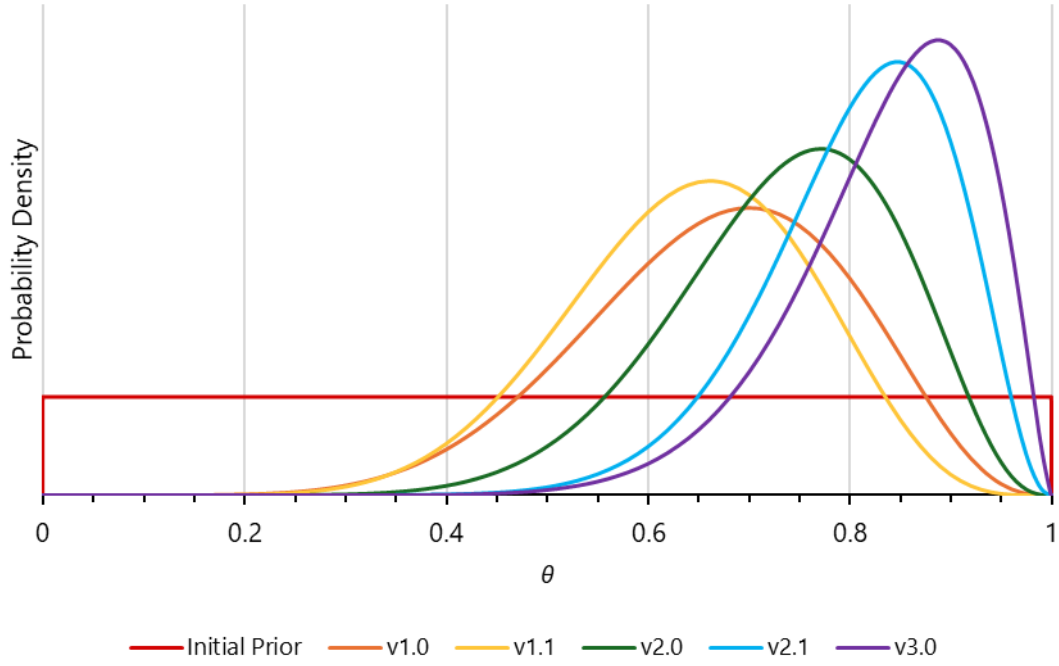

Figure 2
Posterior Distributions by Version for Notional Example

Bayesian Test Planning Process

Given the Bayesian power prior framework above, an important question becomes planning the number of trials required for a future phase of testing.

Planning Inputs

To determine the number of trials that sufficiently minimizes risk, several inputs are required:

- Initial prior: prior representing beliefs before observing any data
- Prior data: one or more phase of previous test data
- Weights: the amount of weight (0-1) carried by each prior data set relative to current data
- Rejectable Quality Limit (RQL): probability of success that the system must exceed to pass. May be equivalent to the threshold reliability.
- Acceptable Quality Limit (AQL): probability of success that is desirable. In general, a test plan is designed to have a high probability of passing AQL. May be equivalent to the objective reliability.
- Credible level: percentage probability (e.g., 80%) required to say the system is above the RQL
- Consumer's risk: the acceptable risk of passing the system if it performs at the RQL
- Producer's risk: the acceptable risk of failing a system if it performs at the AQL

Some of these inputs are named similarly to inputs required for the frequentist approach. This naming is to promote understanding for those familiar with frequentist approach, though the use

of these inputs differs slightly when used in the Bayesian framework.

The chosen number of runs should be such that there is a high chance of passing the updated system if it performs at the AQL (a "good" system), and a low chance of passing the system if it performs at the RQL (a "bad" system). Whether or not a system passes the RQL is assessed using a credible interval, as in Figure 3, which shows an 80% one-sided credible interval that passes the RQL.

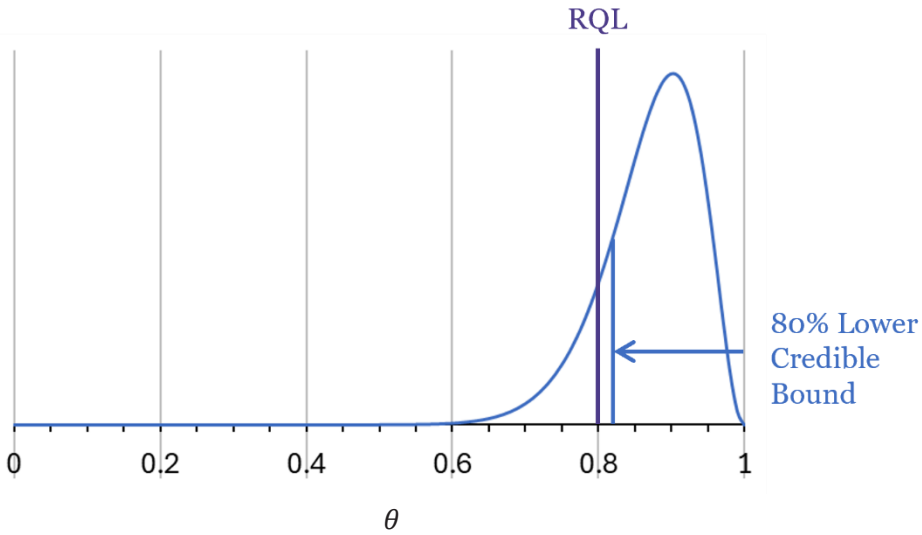

Figure 3
80% Credible Interval for Posterior Distribution that Passes the RQL

Number of Trials Determination

Given the Bayesian framework discussed and the planned evaluation of the RQL with a credible interval, the number of trials to meet acceptable risks can be determined. First, two intermediate steps will be discussed: determining the number of failures that can be accepted for a given run size and determining the probability of observing that many failures. Together, this enables evaluating the probability that the system will pass or fail, by number of runs, given the system performs at a certain reliability (e.g., at AQL or RQL).

For a given run size, credible level, RQL, prior data, and weight(s), the number of allowable failures can be calculated directly. R code for this calculation is provided in the Appendix, or Equation 7 gives the number of acceptable failures f_{allowed}, where $\text{LCB}(f)$ is the lower credible bound given f failures out of n_{runs}, CDF^{-1} is the inverse cumulative distribution function for the beta distribution, CL is the credible level, and $\alpha(f)$ and $\beta(f)$ are the posterior hyperparameters after observing f failures.

$$f_{\text{allowed}} = -1 + \sum_{f=0}^{n_{\text{runs}}} \begin{cases} 1, & \text{LCB}(f) > \text{RQL} \\ 0, & \text{LCB}(f) \leq \text{RQL} \end{cases}$$

$$\text{where } \text{LCB}(f) = \text{CDF}^{-1}(1 - \text{CL};\ \alpha(f), \beta(f)), \quad \alpha(f) = \alpha_0 + \sum_{i=1}^{n} w_i s_i + n_{\text{runs}} - f, \quad \beta(f) = \beta_0 + \sum_{i=1}^{n} w_i f_i + f \tag{7}$$

Once the number of acceptable failures for a given run size is known, the probability of observing that many failures can be calculated. This calculation requires the additional input of the AQL. In other words, the calculation gives the probability of passing the system if it performs at the AQL. Equivalently, one can ask, "what is the probability of passing the system if it performs at the RQL?"

The probability of passing is given in Equation 8, where p_{exp} is the expected reliability of the upgraded system and will be set to equal either the AQL or RQL to evaluate passing probabilities.

$$P(\text{pass}|p_{\text{exp}}) = \sum_{f=0}^{f_{allowed}} \binom{n_{\text{runs}}}{n_{\text{runs}} - f} p_{\text{exp}}^{n_{\text{runs}}-f} (1 - p_{\text{exp}})^{f} \tag{8}$$

Together, equations 7 and 8 enable evaluation of the passing probability of a system performing at the AQL or RQL for a given number of runs. The passing probability can be plotted against many possible run sizes, as in Figure 4, provided the inputs shown.

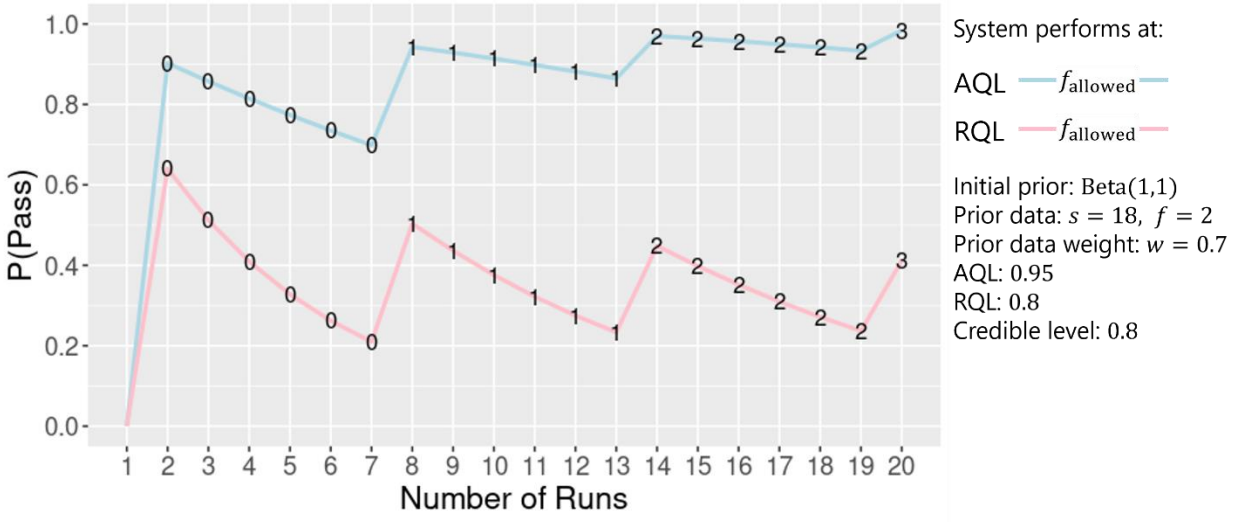

Figure 4

Example of Passing Probability Versus Run Size for Different Updated System Reliabilities

Figure 4 also shows the number of failures that can be allowed while still passing the system. More failures can be accepted the more runs that are performed. When the probability to pass is zero, even if all runs succeed, there is not enough evidence to say that the system passes the RQL. The zig zag pattern seen in Figure 4 appears, with a down slope each time another failure can be accepted, because a failure is more likely to be seen the more tests that are performed. In other words, it is more likely for a system to succeed twice in a row compared to seven times in a row if the probability of success stays constant. Thus, while increasing the number of runs from 2 to 7 makes it less likely to pass the good system that performs at the AQL, it also makes it less likely to pass the bad system that performs at the RQL. A good test plan should achieve sufficient separation between the good and bad systems, such that the risk of failing a good system (producer's risk) and the risk of passing a bad system (consumer's risk) are sufficiently small.

Figure 4 shows the minimum risk of passing a bad system occurring at 7 runs, though the probability to pass is still greater than 20%. To decrease this further, a higher credible level could be used, as shown in Figure 5 where the credible level is increased to 0.9.

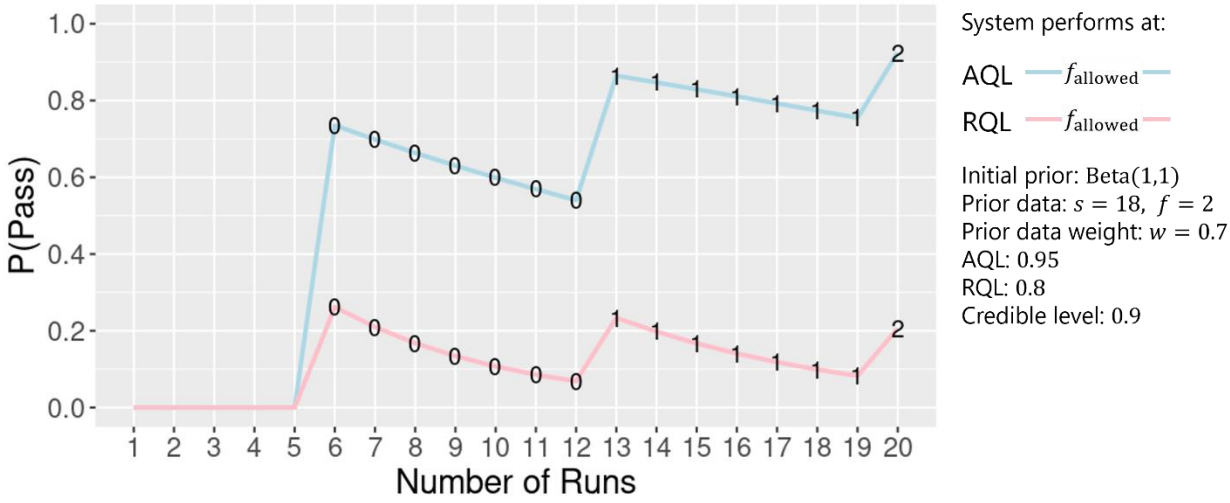

Figure 5

Example of Passing Probability Versus Run Size With Higher Credible Level

As seen in Figure 5, increasing the credible level also increases the burden of proof required to pass the good system, and therefore increases risk to fail the good system for lower numbers of runs. It can be noted that planning 20 runs using the inputs in Figure 5 results in similar risks to planning 19 runs with Figure 4.

Similarly, putting less weight on the prior also increases the burden of proof and decreases the risk of passing a bad system, as seen in Figure 6.

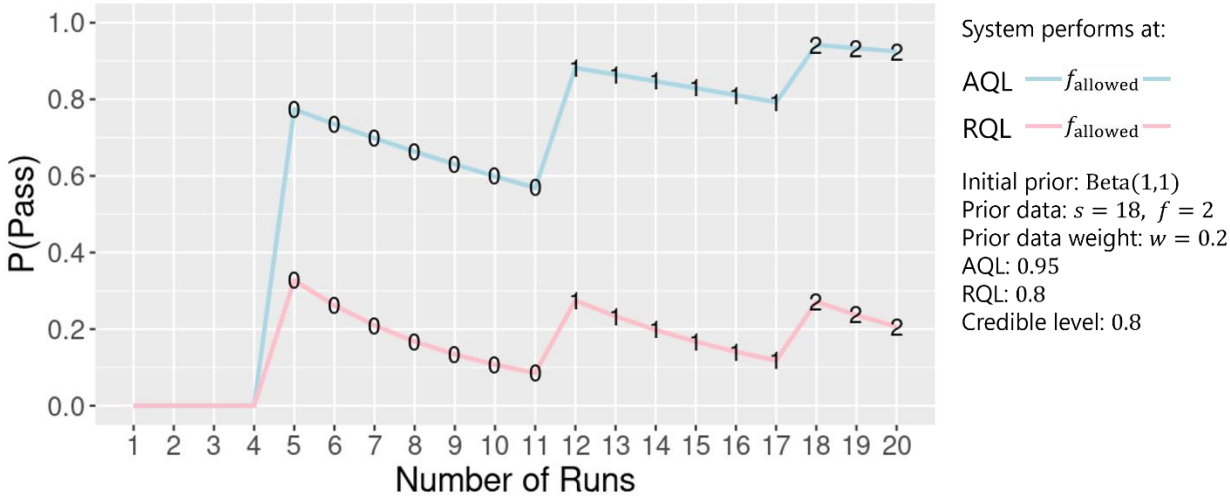

Figure 6

Example of Passing Probability Versus Run Size With Less Weight on Prior

On the other hand, if a very high weight is placed on successful previous data ($s = 19,\ f = 1$), as in Figure 7, the burden of proof shrinks. Figure 7 shows that for 1 run, whether it passes or

fails, the system will still pass. Thus, a one run test would be ineffective since the outcome of the test has no impact on the resulting decision.

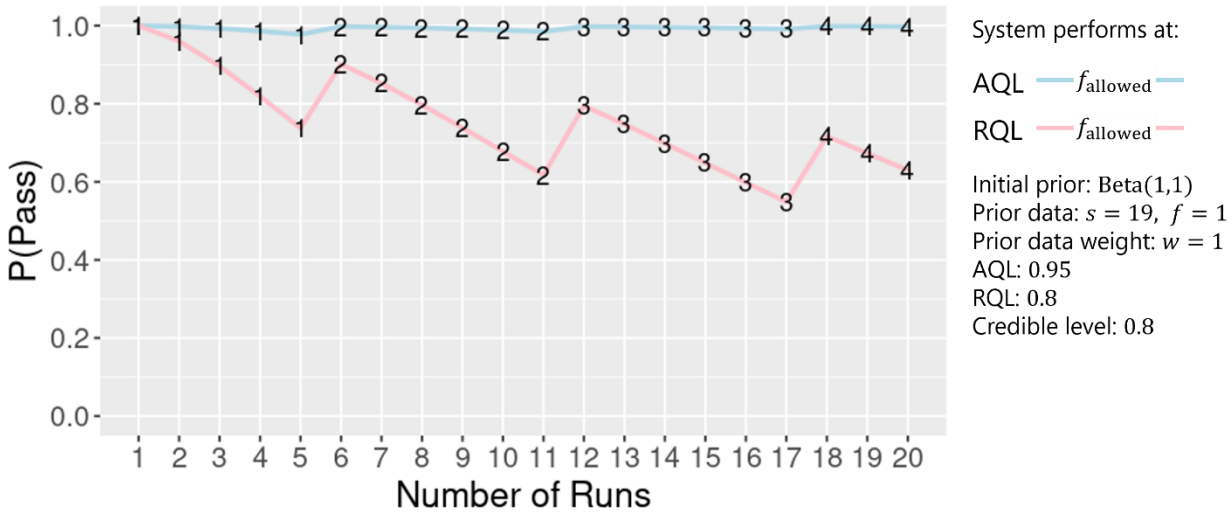

Figure 7
Example of Passing Probability Versus Run Size With Stong Successful Prior Data

Figures such as the above allow for evaluation of the tradeoffs between different types of risk. For instance, due to a limited number of runs, the risk of passing a bad system may be higher than desired but can still be reduced within the available trade space.

The code for producing the above graphs is provided in the appendix. Additionally, the effect of different inputs is summarized below:

- Prior data: A prior that supports the requirement will require less proof for a system to pass, while a prior that is below the requirement will require more proof for a system to pass
- Weights: downweighing a prior that supports the requirement increases the burden of proof; downweighing a prior that does not meet the requirement decreases the burden of proof
- Rejectable Quality Limit (RQL): decreasing the RQL decreases the burden of proof
- Acceptable Quality Limit (AQL): increasing the AQL increases the probability of a good system passing, but does not change the locations of peaks/valleys on above graphs
- Credible level: increasing the credible level increases the burden of proof and decreases the risk of passing a bad system

Results of Test Planning for a System in FOT&E

The methodology described above was applied to plan the number for trials required for FOT&E of a system. Specifically, the Scientific Test & Analysis Center of Excellence (STAT COE) worked collaboratively with the Institute for Defense Analyses (IDA) to recommend the minimum number of trials to reduce risk to acceptable levels.

Due to limited testing resources, the minimum recommended test sizes assumed no failures would be observed during testing. However, if a failure is observed, the number of runs could be increased, and if there are no further failures, the system could still pass with the same or lower

levels of risk. An example of this is shown in Figure 8 (all data is notional).

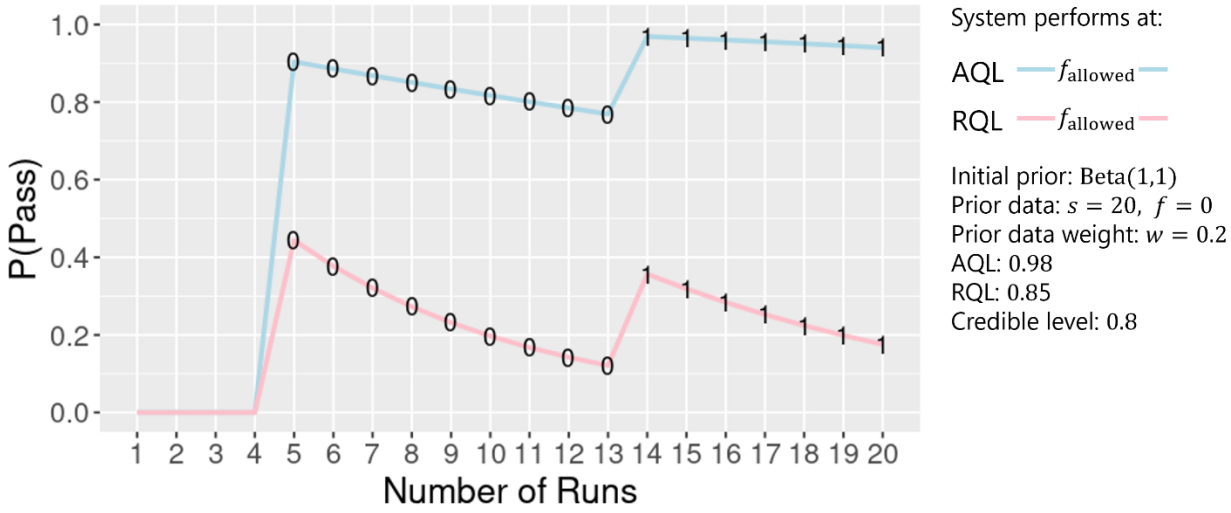

Figure 8

Notional Example of Minimum Number of Runs With Zero or One Allowed Failures

As seen in Figure 8, at least five runs must be performed to pass the system, and it will only pass if all five of those runs are successful. However, if one of those five runs fails or if lower risk is desired in the initial test plan, Figure 8 shows that the number of runs could be increased to 14, with one allowed failure. Assuming only one failure is observed out of 14 runs, the system would pass.

The STAT COE and IDA made recommendations using this methodology to the program. The number of runs recommended was higher than previous plans, indicating that while using this methodology supports a low number of runs, it ensures that the number of runs is not so low that risks are unacceptable. In other words, it provides objectivity to the intuition that fewer runs are needed for an agile update.

Conclusion

FOT&E and agile approaches to system development need rigorous ways to determine the amount of testing required that accounts for previous test data. Bayesian power priors provide a way to combine multiple prior data sets, each with different weights relative to the current system. Using these priors, future tests can be sized based on inputs including the RQL, AQL, credible level, and acceptable consumer's and producer's risk. Different risks can be balanced to create a test that is acceptable to a program. This methodology was applied to make recommendations for the minimum number of runs required for FOT&E of system, showing a reduced number of runs compared to frequentist methods with objective risk quantification. While this paper is focused on binary reliability, this framework could also be extended to continuous reliability problems.

References

AcqNotes. (2024). *Follow-on Operational Test and Evaluation (FOT&E).* AcqNotes. https://acqnotes.com/acqnote/careerfields/follow-on-operational-test-and-evaluation

Ibrahim, J. G., Chen, M., Gwon, Y., & Chen, F. (2015). The power prior: theory and applications. *Statistics in Medicine*. 34(28), 3724–3749. https://doi.org/10.1002/sim.6728.

Appendix

Code for Bayesian Test Planning Plots

```
#libraries
library(tidyverse)

#functions-----------------------------------------------------------
find_allowed_fails <- function(n_runs,a0,b0,a1,b1,w1,AQL,RQL,cred){
  failures <- 0:n_runs
  #calculate (vectors of) new hyperparameters
  a2 <- a0 + a1*w1 + (n_runs-failures)
  b2 <- b0 + b1*w1 + failures
  #calculate (vector of) LCBs
  LCB <- qbeta(1-cred,a2,b2)
  pass <- LCB>RQL
  #find allowed fails
  if(pass[1]==FALSE){n_allowed_fails <-NA}
  else{n_allowed_fails <-sum(pass)-1}
}

find_pass_prob <-
function(n_runs,n_a_fails,a0,b0,a1,b1,w1,AQL,RQL,cred){
  if(is.na(n_a_fails)){p_pass <- 0}
  else{
    possible_fails <- 0:n_a_fails
    n_successes <- n_runs - possible_fails
    p_fails <- choose(n_runs,n_successes)*(AQL)^n_successes*(1-
AQL)^(n_runs-n_successes)
    p_pass <- sum(p_fails)
  }
  return(p_pass)
}


#test planning plot ------------------------------------------------
#number of runs to plot
n_max <- 20
run_sizes <- 1:n_max

#initial prior
a0 <- 1
b0 <- 1
#prior data
a1 <- 20
b1 <- 0
#weight
```

```
w1 <- 0.2
#AQL
AQL <- 0.98
#RQL
RQL <- 0.85
#Credible %
cred <- 0.8

#find acceptable number of failures for each run size
acceptable_fails <-
map_dbl(run_sizes,find_allowed_fails,a0,b0,a1,b1,w1,AQL,RQL,cred)

#find probability of observing #of allowed failures or fewer
(probability to pass threshold) at both AQL and RQL
AQL_pass_prob <- map2_dbl(run_sizes, acceptable_fails, find_pass_prob,
a0,b0,a1,b1,w1,AQL,RQL,cred)
RQL_pass_prob <- map2_dbl(run_sizes, acceptable_fails, find_pass_prob,
a0,b0,a1,b1,w1,RQL,RQL,cred)

#plot results
plot_data <- data.frame(run_sizes,acceptable_fails, AQL_pass_prob,
RQL_pass_prob)
ggplot(plot_data, aes(x=run_sizes)) + geom_line(
aes(y=AQL_pass_prob),color="light blue",linewidth=1) +
geom_text(aes(y=AQL_pass_prob,label=acceptable_fails),size=4) +
  geom_line( aes(y=RQL_pass_prob),color="pink",linewidth=1) +
geom_text(aes(y=RQL_pass_prob,label=acceptable_fails),size=4) +
  xlab("Number of Runs") + ylab("P(Pass)") + scale_x_continuous(limits
= c(1, n_max), breaks = seq(1,n_max,1),minor_breaks=NULL) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0,1,0.2)) +
theme(text=element_text(size=16))
```