2016 IEEE/ACM 13th Working Conference on Mining Software Repositories

Topic Modeling of NASA Space System Problem Reports

Research in Practice

Lucas Layman
Fraunhofer CESE
College Park, MD USA
llayman@cese.fraunhofer.org

Allen P. Nikora
Jet Propulsion Laboratory,
California Institute of Technology
Pasadena, CA USA
allen.p.nikora@jpl.nasa.gov

Joshua Meek
Fraunhofer CESE
College Park, MD USA
jmeek@fc-md.umd.edu

Tim Menzies
North Carolina State University
Raleigh, NC USA
tim@menzies.us

ABSTRACT

Problem reports at NASA are similar to bug reports: they capture defects found during test, post-launch operational anomalies, and document the investigation and corrective action of the issue. These artifacts are a rich source of lessons learned for NASA, but are expensive to analyze since problem reports are comprised primarily of natural language text. We apply *topic modeling* to a corpus of NASA problem reports to extract trends in testing and operational failures. We collected 16,669 problem reports from six NASA space flight missions and applied Latent Dirichlet Allocation topic modeling to the document corpus. We analyze the most popular topics within and across missions, and how popular topics changed over the lifetime of a mission. We find that hardware material and flight software issues are common during the integration and testing phase, while ground station software and equipment issues are more common during the operations phase. We identify a number of challenges in topic modeling for trend analysis: 1) that the process of selecting the topic modeling parameters lacks definitive guidance, 2) defining semantically-meaningful topic labels requires non-trivial effort and domain expertise, 3) topic models derived from the combined corpus of the six missions were biased toward the larger missions, and 4) topics must be semantically distinct as well as cohesive to be useful. Nonetheless, topic modeling can identify problem themes within missions and across mission lifetimes, providing useful feedback to engineers and project managers.

MSR'16, May 14-15, 2016, Austin, TX, USA

DOI: http://dx.doi.org/10.1145/2901739.2901760

CCS Concepts

•Information systems → Data mining; •Applied computing → Aerospace; •Software and its engineering → Maintaining software;

Keywords

topic modeling, data mining, defects, natural language processing, LDA

1. INTRODUCTION

The U.S. National Aeronautics and Space Administration (NASA) builds autonomous, dependable, and safety-critical systems whose engineering and operational lifecycles span years if not not decades. Mission operators and integration specialists record *problem reports* of off-nominal performance, deviations from design, and human errors that occur while building and operating these systems. Problem report repositories across NASA cumulatively contain millions of detailed records spanning hundreds of missions over the past 40 years. Problem reports are similar to software defect reports in issue trackers and bug repositories, though NASA problem reports contain hardware and human operations errors in addition to software defects.

Problem reports are a rich source of information on defects found in test and operations, and this information has the potential to improve the design of future systems. However, NASA centers and projects use different forms, engineers ignore some data fields and appropriate others from their original purpose, and the semantic quality of the information varies drastically.

Problem reports contain vital information in the form of natural language about the challenges faced by NASA projects, their root causes, and the history of corrective actions that have been found useful (or useless) in addressing those root causes. However, data heterogeneity and quality issues create significant obstacles to using problem reports to trend problem data. Currently, the analysis of problem reports is an intensive manual process.

In this study, we apply of *topic modeling* [5] to NASA problem reports to discover problem trends within and across

NASA missions. Our goal for topic modeling is to help answer the following questions:

1. What are the most common problem topics within projects?
2. Do problem topics change over the lifetime of a project?
3. What are the most common problem topics across projects?
4. What are the practical challenges and limitations in applying topic modeling to a large problem report corpus?

In this paper, we report our experience constructing topic models of 16,669 problem reports from six NASA missions. We encountered a number of challenges, such as the lack of guidance on selecting the appropriate number of topics. Analysis of the topic models yielded trends in the problems reported within and across projects. In the remainder of this paper, we describe our analysis methodology, challenges encountered in applying topic modeling to our corpus of NASA problem reports, and present some initial findings of topic trends in the systems under study.

2. BACKGROUND AND RELATED WORK

This section provides a brief summary of problem reporting at NASA, topic modeling with Latent Dirichlet Allocation, and the application of similar techniques to software repositories.

2.1 Problem reporting at NASA

The systems built by NASA include drones, sounding rockets, space transportation systems, deep-space telescopes, robotic explorers, planetary satellites, and more. Reliability is essential in most NASA systems, as space platforms cannot be physically repaired and software defects have been demonstrated to cripple entire missions (e.g., [27, 33]).

Problem reports capture software errors, physical hardware defects, operator error, or environment-induced failures. Problem reports also cover a range of severities, from unexpected screen font colors to total mission loss. Problem reporting is a required activity during system Integration and Test (I&T) and mission operations. All NASA systems undergo a lengthy I&T phase to assemble the system and verify that it will perform as expected in operation. Test failures, design nonconformances, inspection defects, and integration errors are recorded as *problem reports* during the I&T phase. Defects found in the I&T phase can be expensive to fix due to the cost of manufacturing specialized hardware and re-verification of software on the integrated platform.

Once a spaceflight platform has been launched or a ground system goes live, the system enters the Mission Operations phase. Problems encountered during mission operation are called *anomalies* and are formally defined as "an unexpected event, hardware or software damage, a departure from established procedures or performance, or a deviation of system, subsystem, or hardware or software performance outside certified or approved design and performance specification limits" [22].

Problem report contents vary considerably. There is no NASA standard governing the format of problem reports. NASA centers such as Goddard Space Flight Center and the Jet Propulsion Laboratory provide center-wide problem reporting systems for their projects to leverage, but usage of these systems is optional. Some individual projects use their own home-brewed solutions. It is our experience that metadata in problem reports, such as error type or phase discovered, is often missing or used inconsistently.

Despite the diversity in problem report contents, we find that nearly all NASA problem reports contain the following data fields by one name or another: a *description* of the problem at the time of occurrence, the *date* of problem occurrence, the *severity* of the problem, the *responsible party* for investigating and correcting the problem, an *analysis of the cause* of the problem, a *description of the corrective action* taken, and a *date of closure* when the ultimate disposition of the problem was determined. These fields are those most often used by the projects themselves to track and resolve problems. The rich natural language content in the description, cause analysis, and corrective action are the focus of our analysis.

2.2 Topic modeling with LDA

Topic models are "probabilistic models for uncovering the underlying semantic structure of a document collection based on a hierarchical Bayesian analysis of the original texts" [4]. Throughout this paper, we are referring to Latent Dirichlet Allocation (LDA) [5] models when we refer to topic modeling. In LDA, a *word* is the basic unit of discrete data, a *document* is a sequence of words, and a *corpus* is a collection of documents. A topic model of a corpus describes the distribution of *topics* in each document, where each topic contains a distribution of words. An LDA topic model is defined by two probability distributions:

- $\theta \sim Dirichlet(\alpha)$: representing the topic distribution over documents
- $\phi \sim Dirichlet(\beta)$: representing the word distribution over topics

The generated topic model reveals: 1) the latent topics (distributions of words) throughout the corpus, and 2) which topics are most prevalent in each document. Since topics are merely collections of words, it is incumbent upon the analyst to assign semantic meaning to the topics. Thus, the meaningfulness of topics is dependent upon both the semantic cohesion of the topics and the domain knowledge of the analyst. Hindle et al. [12] found that topics derived from LDA models of software requirements documents were by and large meaningful to developers and managers with expertise in the sampled projects.

2.3 Information Retrieval and software repositories

Researchers have applied a variety of *information retrieval* (IR) techniques to software repositories to support knowledge discovery and artifact analysis. Vector-space model (VSM) representations of documents are the foundation for many IR techniques. including topic modeling. Runeson et al. [31] used VSM analysis to detect duplicate bug reports based on similarity scores, which was extended by Wang et al. [36] to incorprate date from execution traces. Menzies and Marcus [20] trained a rule-based learner to automatically assign severities to NASA problem reports based on VSMs. Others have applied similar IR techniques to support software requirements traceability [11], including tracing defect reports to requirements [39], and tracing bug reports to source code using a VSM of the abstract syntax tree [32].

Researchers have applied topic modeling to address a variety of defect management issues, including automated clas-

sification of bugs [30, 18], recommending which developers should fix a bug [38, 37], detection of duplicate reports [26], finding source code related to bug reports [17, 25], and evaluating the bug report coherence [15]. Others have applied topic modeling for other maintenance activities, such as measuring source code quality [6], tracking system evolution [16, 35], and maintaining artifact traceability [3, 28].

The focus of our paper is analyzing the *topic trends* within a corpus of information. Based on our brief literature search, papers focusing on analysis of topic trends are somewhat rare. Neuhaus and Zimmerman [23] use topic modeling to find trends in vulnerability reports, Martie et al. [19] trend Android bug report topics, and Zou et al. [41] analyze of StackOverflow to identify non-functional requirements of concern to developers. Trending of problem topics within and across NASA projects has practical use to NASA program managers for identifying problem areas in spacecraft development, directing process improvement and engineering education, and fostering agency-wide lessons learned.

3. NASA PROBLEM REPORT CORPORA

The problem reports used in this research were collected from problem reporting systems at Goddard Space Flight Center (GSFC) and the Jet Propulsion Laboratory (JPL). Problem reports from three missions at GSFC and three missions at JPL are used. These projects were selected for relevance to current and future missions. The first and second authors have previous experience analyzing the problem reports from these missions and thus possess the domain expertise necessary to interpret the semantic meaning of topics generated. All six of the missions analyzed were active at the time this paper was written (January 2016). We cannot share the problem report corpus due to U.S. government regulations. The mission names have been anonymized so that statistics in this report will not be used out of context.

The missions and their problem report data are summarized in Table 1. The distribution of problem reports by date for each mission is shown in Figure 1. In general, the JPL missions have significantly more problem reports than the GSFC missions because the JPL problem reports are almost all from the pre-launch Integration and Test (I&T) phase. NASA I&T can take years depending on the scale of the projects, and most of a project's problems are eliminated during this phase. In most cases, the JPL missions have longer data collection periods than the GSFC missions.

Table 1: Problem report corpora

Mission	Type	Reports	Date span
GSFC-PS	planetary spacecraft	715	40 months
GSFC-ST1	space telescope	312	83 months
GSFC-ST2	space telescope	908	91 months
JPL-PS	planetary spacecraft	3885	227 months
JPL-RE1	robotic explorer	3683	124 months
JPL-RE2	robotic explorer	7166	93 months

The JPL missions in Figure 1 show a trend common to many projects: a sharp rise in reported problems during test leading up to product launch. The notable exception is JPL-RE2, which was ramped up for an initial planned launch date (the first peak), but then launch was delayed for some time (the second peak). Post-launch problem reports for JPL are collected in a separate reporting system excepting those problems relevant for recurrence control of defects in components that will be reused in future missions. GSFC problem reports for pre-launch I&T problems are recorded in a separate system from the post-launch operational anomalies.

The GSFC missions in Figure 1 show another trend common in NASA systems [1, 2]: space systems exhibit a spike in problem reports in the first six months after launch, which typically covers the launch, cruise, and initial power-on of scientific instruments.[1] Alonso et al. [1] have shown that flight software issues change during the extended lifetime of NASA systems: design and implementation flaws manifest early in operations while hardware materials wear-out and software aging (i.e., memory overflow, upper limit overflow, unique race conditions) manifest after years of operation.

4. TOPIC MODELING METHODOLOGY

Topic modeling was conducted in R using the `topicmodels` package using scripts and procedures from Grün and Hornik [10].

Step 1 – Data preparation: The following fields were extracted from the GSFC problem reports and used to form the input corpora for topic modeling:

- Description - a description of the observed problem, usually entered by the mission operator.
- Root cause - a description of the root cause of the problem, usually entered after an investigation.
- Investigation log - an optional supplement describing the analysis process. Often this information is captured in the "root cause" field.
- Corrective action - A description of the corrective action taken, if any.

The following fields were extracted from the JPL problem reports:

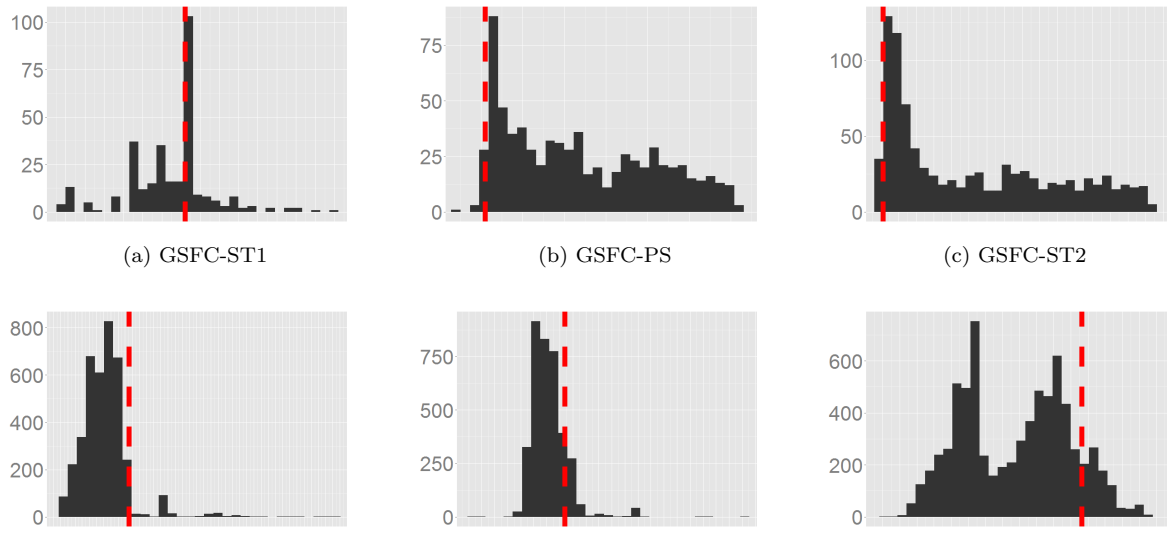
- Description - a description of the problem entered by the I&T engineer
- Investigation - an analysis of the cause of the problem
- Corrective action - a description of the corrective action taken, if any.

The verbosity of the fields varies substantially. For example, some problem descriptions contain entire error log file dumps, and cause analysis fields contain email chains between engineers. Other problem reports contain only a few pointed sentences.

The Grün and Hornik [10] scripts perform the following preprocessing on the input corpora prior to constructing the document-term matrix:

- conversion of all tokens to lowercase
- removal of punctuation and numbers
- stop word removal using the SMART list – http://goo.gl/qpShB2
- Porter stemming of terms using the Snowball algorithm – https://goo.gl/FVMui4
- omitting terms with length < 3. Menzies and Marcus [20] found that terms with length less than 3 often made no sense to business users.

[1]For the GSFC missions in our study, the projects became fully operational within six months. However, some planetary exploration missions have extended cruise phases to their destinations before powering up the scientific instruments and becoming fully operational.



(a) GSFC-ST1 (b) GSFC-PS (c) GSFC-ST2

(d) JPL-PS (e) JPL-RE1 (f) JPL-RE2

Figure 1: Problem Reports relative to launch date. x-axis = 6 month bins, y-axis = problem report count, red dashed line = mission launch date

Step 2 – Estimate topic modeling parameters: An LDA topic model is defined by two parameters:

- $\theta \sim Dirichlet(\alpha)$: representing the topic distribution over documents
- $\phi \sim Dirichlet(\beta)$: representing the word distribution over topics

The number of topics, k, in an LDA model must also be defined *a priori*. The θ and ϕ parameters are estimated in the Grün and Hornik [10] scripts using three methods:

1. variational expectation-maximization (VEM) where β is fixed at 0.1 and α is fixed at $50/k$ as recommended by [9]
2. VEM where β is fixed at 0.1 and α is estimated with a starting value of $50/k$
3. Gibbs sampling [9]

Grün and Hornik [10] implement 10-fold cross validation for each of the estimation methods and measure the model fits using *perplexity*, which captures how well a probability distribution predicts a sample using the log-likelihood of a held-out test set. We executed the 10-fold cross validation with $k = \{20, 30, 40, ..., 100, 125, 150, ..., 225\}$ and the three parameter estimation methods. An example of a resulting perplexity plot is shown in Figure 2. In general, the Gibbs sampling approach to parameter estimation resulted in significantly lower perplexities than the VEM estimations, and so we used the Gibbs approach to estimate the topic modeling parameters.

Step 3 – Select the number of topics: There is no de facto method for selecting k, the number of topics to model. Recommended methods range from using rules of thumb to a variety of statistical simulations. Our selection of the number of topics is based on the perplexity plots. The perplexity plots for all our corpora exhibited a positive linear positive trend as in Figure 2. We selected k for each corpora as the point on the line before the slope increased most drastically than the linear trend – this was done subjectively

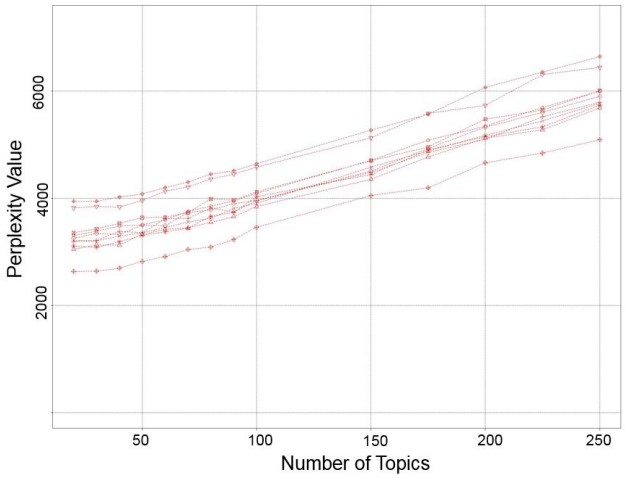


Figure 2: Example perplexity plot

and through visual inspection, but is similar to the rate of perplexity change metric proposed in [40]. If no such point was obvious, we set $k = min\{\frac{\#documents}{10}, 100\}$. Choosing a value for k has important implications for the utility of the topics models. Our subjective approach is a limitation, but the general lack of guidance in selecting the number of topics is a challenge discussed further in Section 6.

Step 4 – Fit model to corpora: Once the parameter estimation method and number of topics are chosen, the topic model is fit to the data. From the model, we can then extract: 1) θ_d: a vector representing distribution of topics for a problem report d, and 2) ϕ_k: a vector representing the distribution of words representing topic k. See Figure 3 for an illustrative of example of θ_d and topic k. The sum of the vector θ_d is 1. For both θ_d and ϕ_k there is a minimum probability value that indicates irrelevance, and we have removed these terms from our example. We use θ_d and ϕ_k to

Topic ID	Text
35	malindi antenna carrier late mal unit comput redump paolo aos recovered record acu replac manually contacts degrad sit
62	star tracker quaternion spike jump goto contractu fund sacsafepoint xrt corrupt arcminut degre theori voltag build disturb
28	malindi fucino jsc line network mal comm router asi transfer houston asinet equipment establish phone outag intermitt
12	circuit safeti trip bright star wider beta sirius cen grism mag april cathod great pcthold trends alpha blue cma mal across b
3	usn socket carrier cds router ptp establish nmc noop record los power combin equipment network recycl flow hung recor
52	xrt dpu chart xend xrtpos utc window build ack geni centroid offset mcp submodenon cathod ecr hete jan modeimageev

(a) Word tokens that compose a topic.

Problem Report	Topic_2	Topic_3	Topic_4	Topic_7	Topic_8	Topic_9	Topic_10	Topic_18	Topic_24	Topic_33	Topic_56	Topic_57
1						0.763037		0.029689	0.088357			0.088357
2				0.100154	0.597908						0.199705	
3						0.384182				0.595354		
4		0.910828										
5		0.230104										
6					0.563533		0.087141					

(b) Distribution of topics within documents. Topics with the minimum probability contribution are hidden for readability.

Figure 3: Examples of topics and topic distributions over problem reports

identify the most frequently occurring topics and to ascribe semantic meaning to the topics.

Step 5 – Create semantically meaningful labels for topics: LDA topics are collections of word tokens (see Figure 3.a). The word tokens provide some hints on what the topic represents, but reading the relevant problem reports is necessary to identify the semantic topic represented by the problem reports. Further, domain knowledge of space system engineering is necessary to understand the technical language and plethora of acronyms in the problem reports. Our experience reflects the observations of Hindle et al. [12], who suggest that domain experts are necessary to accurately label topics.

Initial label set created for JPL missions
actuators and motors
avionics electrical performance
cabling and connectors
electronic component assembly or workmanship
electronic part failure
flight software
ground software
ground support equipment
mechanical or structural integrity
mechanical performance
moving parts
power generation and distribution
RF or EMC compatibility
spacecraft I&T environment
Additional labels set needed for GSFC missions
avionics component
ground communications
mission planning
space network
spacecraft (S/C) fault detection
spacecraft (S/C) instrument operations

Table 2: Semantic labels for topics

The list of semantic labels for the most popular topics in our corpora is shown in Table 2. The second author examined the most frequently occurring topics for JPL missions and created an initial list of topic labels. The first author then applied and expanded the list to the GSFC missions – the expanded list contained additional labels for ground operations topics not found in the JPL corpora. Both authors discussed the expanded list and revised the labels they applied to the GSFC and JPL mission topics. The first author applied the topic labels to the topic models built from the combined corpus of all missions. A limitation of our analysis is that we did not systematically evaluate the accuracy of our topic labeling. Instead, we rely on the domain knowledge of the two authors, who have worked extensively with the problem reports on these missions.

5. ANALYSIS OF PROBLEM REPORT TOPICS

In this section, we examine the *most frequently occurring topics* in the topic models generated for our missions. We say that topic t occurs in a problem report d if $\theta_{d,t} > min(\theta_d)$. In other words, a topic occurs in a problem report if the cell representing topic t is non-blank for row d in Figure 3.a. We do not consider the strength of the probability value, which is a limitation discussed in Section 7. Table 3 contains a summary of the topic modeling results.

The JPL corpora are significantly larger than the GSFC corpora. The JPL problem reports were mostly recorded during I&T, whereas the GSFC problem reports were recorded during mission operations. In a successful mission, most of the problems will be discovered during integration test. The JPL problem reports contain more words because more information is available to a tester working with a component on a test bench than to a mission operator reviewing error logs generated on operational spacecraft and ground systems that are hundreds, thousands, or millions of miles away. Further, the JPL corpora had significantly higher perplexities (even for small values of k) than the GSFC missions, which is a reflection, in part, of the larger corpora size and the difficulty of fitting a precise topic model.

The word counts and word densities also reflect process differences in how problem reports are recorded. We have

Table 3: Topic modeling summary

Corpus	documents	word count	words / document	Topics	$\overline{perplexity}$
GSFC-ST1	312	178,406	572	30	1173.0
GSFC-PS	715	90,845	127	50	791.3
GSFC-ST2	908	205,436	226	90	768.5
JPL-PS	3885	1,081,897	278	100	2510.1
JPL-RE1	3683	1,586,430	431	100	2684.8
JPL-RE2	7166	3,585,891	500	100	4056.7
All missions	16,669	6,728,905	404	100*	5498.9

* We ended the simulation at k=150 topics for the "All missions" corpora due to extremely long computation times for higher values of k.

observed previously [14] that some teams record every deviation as a separate problem report, other teams combine multiple related deviations into a single problem report, while still other teams (particularly in the operations phase) avoid writing problem reports for perceived "trivial" problems. Thus, differences in the numbers of problem reports and document word densities are also a factor of the development phase in which the problem reports were recorded. Therefore, we must assume that the definition of a "reportable problem" is not consistent across missions. This limits the accuracy of the list of most frequently occurring topics learned from corpora of combined missions discussed in Section 5.4.

5.1 Number of topics per document

A single problem report contains one or more topics. Table 4 shows the percentage of problem reports within each mission that are described by n={1,2,3,4,5,>5} topics. For 4/6 missions, most of the problem reports are described by one or two topics. This gives us hope that the topic models have some discriminative power and that we will be able to observe topic trends in our problem reports. We also observe that number of topics required to describe a document grows as the corpus size increases. This is a likely indicator k=100 topics is not large enough to discriminate between the many different topics that appear in the larger corpora in our data.

			% of reports with n topics					
Mission	Docs	Topics	1	2	3	4	5	>5
GSFC-PS	715	50	35	34	20	8	2	1
GSFC-ST1	312	30	37	40	18	4	0	0
GSFC-ST2	908	90	40	31	19	8	2	1
JPL-PS	3872	100	24	32	25	12	5	2
JPL-RE1	3679	100	15	32	28	15	7	4
JPL-RE2	7163	100	14	28	28	17	8	5

Table 4: Percentage of problem reports described by n topics

5.2 Problem report topics within projects

The top 10 topics for each mission, as measured by the number of problem reports containing those topics, is listed in Table 5. Although called problem reports, the "problems" reported for the GSFC missions are not necessarily failures, errors, or defects, but are "anomalies" that include any deviation from the desired behavior of the system [22]. For example, a tripped safety circuit trip because a sensitive camera pointed too close to the sun yields many problem reports in GSFC-ST2. This is designed behavior and may be expected in the mission planning for the day.

Our first observation is that one-third of the Top 10 topics are software-related. Previous research has suggested that approximately 25% of operational anomalies are software-related [14, 1, 7]. Flight software resides in the main flight computer and electronic control units (ECUs) onboard the spacecraft, whereas ground software resides in a variety of workstations and servers on Earth. Flight software is comprised of avionics and instrument operation software that controls the spacecraft after launch, while ground software is used for mission planning, communication, and scientific data collection. These topic models further point to the importance of software in space system engineering, which has long been dominated by hardware. By moving system control from hardware mechanisms to software, engineers are able to carry more scientific instrumentation within the strict weight and power consumption constraints of the spacecraft. However, this necessitates high reliability flight software with extensive verification and validation.

The majority of topics for the GSFC operational phase missions concern ground equipment, ground communications, and ground software. Stereotypical examples of these problems are loss of telemetry signal between the spacecraft and a ground station, network communication errors between the ground stations situated around the earth and the Mission Operations Center (MOC), and computer workstations freezing due to software errors. Problems on the ground are generally preferred to problems with the spacecraft because they are often less severe and easier to resolve. Ground systems can be rebooted and troubleshooted hands on. For spacecraft, physical contact is impossible, spacecraft telecommunications are timeshared among multiple missions on the space network, and round trip light time and bandwidth limitations extend the feedback cycle between the MOC and the spacecraft. While spacecraft problems have the potential to be much more severe (e.g., total loss of mission), the large numbers of ground system problems point to areas where significant maintenance costs might be saved if these systems were improved.

The dominant topic among the JPL problem reports, which are generated from I&T activities, are issues with cabling and connectors. These are mostly electronic wiring and pin connectors that carry signals to the spacecraft's various electronic controllers and actuators. Unlike the plug-and-play standards of consumer electronics, nearly every pin connector and cable is designed for a purpose unique to the space-

GSFC-ST1

Count	Topic
37	**flight software**
36	ground communications
35	ground support equipment
27	**flight software**
26	S/C fault detection
26	**ground software**
25	**flight software**
25	S/C fault detection
24	**mission planning**
23	ground communications

GSFC-PS

Count	Topic
111	ground support equipment
91	ground support equipment
81	ground support equipment
67	ground communications
63	**ground software**
63	**ground software**
46	ground communications
46	**ground software**
40	ground communications
37	**mission planning**

GSFC-ST2

Count	Topic
106	ground support equipment
84	ground communications
64	ground support equipment
62	avionics component
59	S/C fault detection
57	space network
56	avionics component
53	**ground software**
46	S/C fault detection
44	**mission planning**

JPL-PS

Count	Topic
351	cabling and connectors
260	electronic part failure
243	**flight software**
224	**flight software**
215	**flight software**
214	electronic component assembly or workmanship
214	mechanical/structural integrity
200	spacecraft I&T environment
193	spacecraft I&T environment
192	RF or EMC compatibility

JPL-RE1

Count	Topic
503	cabling and connectors
287	avionics electrical performance
259	electronic part failure
256	**ground software**
241	avionics electrical performance
240	actuators and motors
209	avionics electrical performance
208	ground support equipment
196	avionics electrical performance
192	**flight software**

JPL-RE2

Count	Topic
746	cabling and connectors
528	**flight software**
492	**flight software**
466	**flight software**
450	cabling and connectors
448	electronic component assembly or workmanship
447	**flight software**
421	**flight software**
413	electronic part failure
398	actuators and motors

Table 5: Top 10 most frequently occurring topics for each mission problem report corpus. **Boldface** topics are predominantly software-related

craft domain and customized to the mission profile. Each pin has a designated tolerance for current, frequency, and resistance that must be evaluated under a wide variety of operating conditions. A violation of any one of these tolerances due to a bent pin, poor material quality, or operator error is recorded.

Another common topic among the JPL reports is flight software. Autonomous spacecraft control has become increasingly common in NASA missions, and even more so in the planetary explorers produced at the JPL. The round trip light time between Earth, Mars, and the outer planets is sufficiently long to make synchronous commanding impossible. The responsibility of system control falls to the flight software, which follows mission plans uploaded by the mission operations center. So, not only is the flight software responsible for commanding, it must be able to automatically detect, isolate, and recover from faults while managing an increasing variety of scientific instruments. This is the primary driver behind the exponential code size increase in the Mars rover family of missions [13]. Thus, we are not surprised that flight software is the topic of many problem reports during JPL I&T, which must verify that these functions critical to mission success behave as intended prior to operation.

The differences between the JPL and GSFC topics are encouraging from a program management perspective. The I&T problem reports from JPL are replete with physical issues, part failures, and flight software changes, whereas the operations phase GSFC problems are mostly ground related. Mechanical problems are impossible to fix in operations, so eliminating them during test is essential. Similarly, flight software is risky and expensive to patch during operations since a failed patch can cripple a spacecraft and the patching process itself takes time away from scientific data collection, which is every mission's primary purpose. This does not mean that mechanical problems do not occur in the GSFC operational missions, only that they did not bubble to the top of the most frequently occurring topics. This hints at a limitation of topic modeling to detect rare yet possibly critical documents within a corpus. We discuss this implication for topic modeling of defect data further in Section 6.

5.3 Problem report trends over time

One of our goals in applying topic modeling to the problem reports was to detect topic trends over the lifetime of a mission. Previous research [1] has suggested that the types of failures encountered early in a mission's operational lifetime tend to be design and implementation related, whereas failures encountered during extended operations (sometimes years after launch) are more concerned with material breakdown and "software aging" [29].

Figure 4 provides an illustration of the density of 10 most frequently occurring topics in the GSFC-ST2 mission over time. The ground equipment, communication, and ground software issues (top 3 rows) are spread throughout the mission lifetime. In contrast, the mission planning issues (last row) seem to have been completely eliminated in the first 2.5 years of operation. Space network (row 6) and some spacecraft fault detection issues (row 9) are concentrated in the first year of operation, and then sporadically crop up

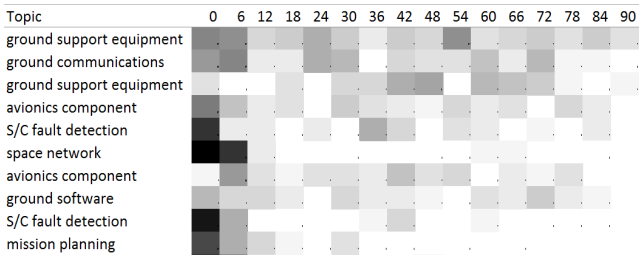


Figure 4: Density plot of the Top 10 most frequently occurring topics for GSFC-ST2. Columns represent 6 month ranges. Darker cells indicate that topic appeared in more problem reports.

some time later. We observed some less popular topics that occur late in the operational lifetime, for example, a recurring issue at a ground station where the receiver could not acquire spacecraft telemetry (23 problem reports) and an issue with an email server that failed to receive or send its daily message digests (20 problem reports).

Ultimately, we elected to perform a limited analysis of topics over time due to concerns on how strongly topic density is influenced by the selection of k, the number of topics. This issue is discussed more in Section 6.

5.4 Problem report topics across projects

We build a topic model from a corpus of all the problem reports from our six missions. Our goal was to determine if topics common across the missions would emerge, which might point to systematic problems across the NASA enterprise regardless of mission type. Table 6 shows the top 10 most frequently occurring topics learned from the combined mission corpus. The percentages of problem reports that contain the topic within each mission are expressed in columns 3-8 of Table 6. These percentages are summarized in the variance (σ^2) column, which provides a crude measure of the dispersion of the topic: a lower variance indicates more even dispersion of the topic across missions. Predictably, the numerical superiority of the JPL problem reports led to JPL problem topics dominating the cross-mission topic model. We discuss the challenges and limitations of working with a mixed corpus in Section 6.

Cabling and connector problems and electrical issues (rows 1 and 2) account for approximately 10% of each JPL missions' problem reports. Several rows pertain to flight software, but the flight software topics in rows 3, 6, 8, 9, and 10 pertain primarily to one of the three JPL missions. This is an artifact of different terminology between missions. These topics often refer to specific electronic components, subroutines, or functional responsibilities relevant to the flight software of a particular mission. Thus, the topics become as much an identifier for the mission as an indicator of where the problem occurred.

Finally, we note that there is some evidence that problem topics may be shared within product families or product lines. Many of NASA's missions inherit designs, hardware, and software from previous missions. JPL-RE1 and JPL-RE2 are two such missions in a product line. The proportions of problem reports pertaining to power generation (row 4), actuators and motors (row 5), and mechanical integrity (row 7) are more similar between JPL-RE1 and JPL-RE2 than when compared to JPL-PS. This is likely a function of the design and components of the three missions: JPL-PS is a planetary orbiter whereas JPL-RE1 and JPL-RE2 are robotic explorers. Further research is needed, but if problem report topics bear the same trends for product lines or similar mission profiles, problem reports categorized by topic may be used to help improve the design of inherited systems.

6. DISCUSSION - IMPLICATIONS FOR APPLIED TOPIC MODELING

Topic modeling of NASA problem reports is a promising approach for drawing lessons learned from a large, unstructured corpus of data in order to improve the design and execution of future NASA missions. Our topic modeling of the NASA problem report corpus is not so different from topic modeling of software defect repositories. We hope that the following observations will be useful for practitioners to understand the limits of topic modeling, and that these challenges serve as a basis for research that will make topic modeling more generally useful to practitioners.

6.1 Document examples for topics must be read to ensure accurate semantic labels

While the word tokens that comprise topics can provide clues, accurate semantic labeling of topics in our corpora required reading some of the problem reports associated with that topic. For example, a number of topics contain words one would associate with a ground station, but it was only by reading the details of the problem reports that we could distinguish between ground station communication, hardware, and software topics. NASA problem reports are filled with jargon and acronyms that identify specialized components, and thus topic meaning is difficult to infer without context and a dictionary of terms.

Topics will not necessarily conform to a predefined labeling ontology. Our corpus is problem reports, so we might hope for a set of topics that pertain to "where" the problem occurred (e.g., ground equipment, flight software), but in reality a topic label may more accurately describe the "type" of failure (e.g., spacecraft fault tolerance, moving parts). One might just as easily find a topic where the color "blue" or "problems reported by John Smith" is the unifying theme. Reading the documents associated with an open mind helps to ensure that the semantic label accurately reflects the topic, instead of some preconceived notions of what topics should be on the part of the analyst.

Therefore, we suggest that reading examples of topic documents is necessary for accurate semantic labeling. Reading the topics alone is not enough. Unfortunately, reading the documents can be an expensive process, especially when the corpus is large. The challenge is to identify a representative, yet minimal set of documents to help the analyst converge on an accurate topic label.

6.2 Semantic coherence vs. distinction

Assigning a semantically meaningful label to a topic is a necessary first step to trending topics. Determining if the collection of words in a topic is *semantically coherent* is usually a human task, though researchers have designed and validated automated measures of topic coherence [24, 21]. We did not measure topic coherence in our work, but, anec-

		GSFC			JPL			
Topic	Reports	ST1	ST2	PS	PS	RE1	RE2	σ^2*
cabling and connectors	1690	0.3%	2.1%	0.7%	9.4%	14.3%	10.9%	3.00
avionics electrical performance	1507	1.0%	0.6%	1.2%	11.1%	8.3%	10.6%	2.14
flight software	1149	1.3%	2.4%	1.3%	25.2%	1.1%	1.3%	7.84
power generation and distribution	958	0.0%	0.6%	0.3%	13.2%	3.6%	4.3%	2.09
actuators and motors	901	2.2%	1.5%	0.8%	1.9%	9.3%	6.5%	0.96
flight software	866	1.0%	0.3%	0.6%	0.8%	13.1%	4.8%	2.12
mechanical or structural integrity	865	1.0%	0.7%	0.6%	8.8%	4.7%	4.8%	0.89
flight software	864	0.0%	0.1%	0.1%	0.1%	0.1%	12.0%	1.99
flight software	861	5.4%	1.3%	2.2%	0.8%	14.4%	3.5%	2.16
flight software	829	0.6%	0.6%	0.1%	1.2%	0.6%	10.6%	1.40

*variance of percentages is multiplied by 1000 for readability

Table 6: Top 10 most frequently occurring problem reports from a corpus of all six missions. Columns 3-8 show percentages of problem reports with topic per mission.

dotally, the most popular topics were semantically coherent. We also observed that less popular topics were less cohesive. This is due in part to having fewer document examples on which to base the semantic label. If topic coherence cannot be improved (e.g., through changing the number of topics in the topic model), then topic analysis may be restricted to focusing on only the most frequently occurring topics. This is a drawback since finding rare topics may be desirable, for example when surveying problem reports for topics pertaining to rare yet critical failures across the history of a mission.

A related issue is whether or not the topics are *semantically distinctive*. For example, three topics in GSFC-ST2 related to problems at a single overseas ground station, and these topics clearly distinguished between network troubles at the station, computer failures at the station, and software problems at the station. Other topics seemed repetitive and indistinct, such as the spacecraft fault detection topics for GSFC-ST2 (both topics pertained to a single camera) and the flight software issues in the JPL missions. Even after reading a number of problem reports from each topic, a semantic distinction was difficult to make. Hindle et al. [12] in a study of topic modeling of software artifacts, recommend that such confusing, irrelevant, and duplicated topics should be removed.

We observe that the semantic distinction of topics is undoubtedly dependent on the selection of k, the number of topics, relative to the size of the corpus. For topics analysis results to be more useful for trending and lessons learned, we need a validated measure of semantic distinction alongside semantic cohesion to serve as evaluation criteria for the quality of a topic model.

6.3 Selecting k, the number of topics

The number of topics modeled, k, in a corpus impacts the distribution of topics. Consider Table 4, which shows that problem reports in the larger JPL missions are less likely to be represented by a single topic. This contributes to a "dilution" of topics and less distinct density patterns compared to those in Figure 4. Conversely, selecting a large k relative to the number of documents results in a sparse matrix because the topics do not congeal in sets of problem reports. For our NASA reports, the practical implications of choosing the "correct" k are significant since it influences both the utility and the accuracy of the learned topics.

Unfortunately, there is no single recommended method for selecting the number of topics to model (c.f. [10, 8, 34, 28]. A definitive answer is likely not forthcoming. However, better toolsets and guidelines for evaluating topic modeling results (e.g., via measures of semantic coherence and semantic distinction) while changing parameters would help speed topic modeling into more practical applications.

6.4 Mixed corpora are dangerous

In Section 5.4, we analyzed topics learned from the combined corpora of all six missions. Not surprisingly, this lead to statistical biases in the topics produced, which can be seen in the percentage columns of Table 6. This statistical bias could potentially be addressed by undersampling, but we have not investigated that in this paper. For NASA, a meta analysis of the topics derived from individual missions (e.g. a structured comparison of the findings in Section 5.2) is still useful to project managers and engineers.

The true concern is when the analyst may not be aware that they possess a mixed corpus. How can one detect if the topics are interesting semantically different topics, or merely an artifact of disparate processes for generating the corpora? In our combined corpus of all mission problem reports, we knew that the JPL problem reports were derived from I&T and the GSFC problem reports were operational anomalies. We discussed in Section 5.2 why these two phases contributed different problem report topics. We were also aware of the product families and reporting process disparities in the corpora. There may be other systematic, process-related differences in problem report generation of which we are not aware.

More generally, consider a researcher who is mining GitHub issues that have an unbalanced representation of issues from embedded device controllers, mobile phone games, and programming tools. The issue-generation processes and writing conventions may be drastically different among these domains, and the learned topic model from the combined corpus may be biased in unknown ways to the analyst. One must exercise caution in drawing lessons learned from such mixed corpora, or making general statements about the popularity of one topic or another when comparing projects.

7. STUDY LIMITATIONS

We have highlighted a number of limitations in our topic modeling and analysis approach throughout the text, which

we summarize here.

We have reused the topic modeling methodology approach provided by Grün and Hornik [10] to minimize experimentation bias. The scripts used include standard text preprocessing: stemming, stopword removal, punctuation removal, and the like. The scripts we used for topic modeling and analysis can be found at https://goo.gl/KK5KrN.

We did not objectively derive a labeling scheme for the most popular topics, instead agreeing upon a label set after a few iterations and informal discussion. We did not measure interrater agreement when applying our labels to the topics, instead relying on the domain expertise of the first two authors. A rigorous qualitative coding approach may yield a more distinctive labeling scheme. We did not validate our labels or their assignment to problem reports with mission personnel due to their availability and the time required.

Our topic analysis is built upon binary yes/no answers of whether a topic appears in a problem report or not, but we do not use the probability values to gauge the strength of the topic's contribution to a document. Setting a minimum probability threshold would lower the number of times the topics appeared in problem reports. This may yield results that are free of "noise", however, we do not have a methodology for determining an appropriate minimum threshold. We have not performed a sensitivity analysis to determine how various minimum probability thresholds would affect the rankings in Tables 5 and 6.

As discussed throughout the paper the different missions have different problem report generating processes. We discussed the perils of statistical bias in our combined corpora in Section 6.4 both as a limitation of our own study and as a caution to researchers working in large, freely-available repositories. Therefore, the topic ranking found in Table 6 should be viewed skeptically.

The most frequently occurring topics are representative of these NASA missions and may not generalize to systems beyond the space system domain. Further, the NASA problem reports we modeled are from highly specialized systems and the problem reports often use unique acronyms and spacecraft jargon unlikely to be found on other defect repositories. These problem reports are long and detailed even after stop word removal (see Table 3). Therefore, we cannot claim that this topic trending analysis will yield as much useful information for corpora with other lexical features.

8. CONCLUSION

In this paper, we describe our experience applying Latent Dirichlet Allocation topic modeling to 16,669 problem reports from six NASA missions. Our goal was to look for topic trends in problem reports within each NASA mission, over the life time of the missions, and trends across missions. We found that post-launch, operations phase anomalies in our sample consisted mainly of ground equipment and ground software problems. The pre-launch I&T problem reports exhibited a wider variety of problem topics, including flight software, cabling and connection issues, and electronic and mechanical part failure. Software is a contributing factor to many of the problem reports, which highlights the growing trend to move more spaceflight control and failsafe functionality into software instead of traditional hardware control.

We encountered a number of challenges that require further research in order to help make topic modeling more readily applicable for trend analysis and lessons learned generation:

- Defining accurate semantic labels for topics requires reading the underlying documents and domain expertise. The cost of labeling grows with the number of topics.
- Semantic distinction between topics helps to avoid duplicate topics and isolate trends. A measure of semantic distinction is needed together with semantic cohesion [24] to evaluate topic models.
- Selection of k, the number of topics to model, significantly impacts the internal validity of trend analysis. Definitive methods and usable tools for evaluating k are needed.
- A corpus comprised of documents generated by disparate processes (e.g., testing defects vs. post-release defects, mobile applications vs. embedded systems) may yield a topic model unknowingly biased to the processes which generated the most documents.

NASA has an enormous corpus of problem reports in databases spread throughout the agency serving 40+ years of missions. The primary use of these databases has been to track problems that occur during testing and operations. Topic modeling is a promising technology for analyzing this large corpus of data to extract lessons learned and failure trends in order to design more reliable systems for future missions. This study is a first step in understanding the practical limitations of our approach and what advances in the state-of-the-art are necessary to mature applied topic modeling for use across NASA enterprise.

9. ACKNOWLEDGMENTS

This work was supported by the NASA OSMA Software Assurance Research Program through grants NNX11AP93G and NNX15AC81G. Cross-project data analysis and undergraduate student support was partially supported by NSF grant 1302169. Part of this work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and sponsored both by the National Aeronautics and Space Administration's Office of Safety and Mission Assurance Software Assurance Research Program (SARP) and JPL's Office of the Management System. The JPL effort funded by SARP is managed locally by JPL's Assurance Technology Program Office.

10. REFERENCES

[1] J. Alonso, M. Grottke, A. P. Nikora, and K. S. Trivedi. The nature of the times to flight software failure during space missions. In *Proc. 23rd IEEE International Symposium on Software Reliability Engineering Workshops*, pages 331–340, Dallas, TX, 2012.

[2] J. Alonso, M. Grottke, A. P. Nikora, and K. S. Trivedi. An empirical investigation of fault repairs and mitigations in space mission system software. In *Proc. 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 1–8, Budapest, Hungary, 2013.

[3] H. U. Asuncion, A. U. Asuncion, and R. N. Taylor. Software traceability with topic modeling. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - ICSE '10*, pages 95–104, Cape Town, South Africa, may 2010. ACM Press.

[4] D. M. Blei and J. D. Lafferty. Topic Models. In A. N. Srivastava and M. Sahami, editors, *Text Mining: Classification, Clustering, and Applications*, pages 71–94. CRC Press, New York, NY, 2009.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3:993–1022, mar 2003.

[6] T.-H. Chen, S. W. Thomas, M. Nagappan, and A. E. Hassan. Explaining software defects using topic models. In *2012 9th IEEE Working Conference on Mining Software Repositories (MSR)*, pages 189–198, Zurich, Switzerland, jun 2012. IEEE.

[7] D. Falessi and L. Layman. Automated classification of NASA anomalies using natural language processing techniques. In *2013 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 5–6, Pasadena, CA, nov 2013. IEEE.

[8] S. Grant and J. R. Cordy. Estimating the Optimal Number of Latent Concepts in Source Code Analysis. In *2010 10th IEEE Working Conference on Source Code Analysis and Manipulation*, pages 65–74, Timosoara, Romania, sep 2010. IEEE.

[9] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Supplement 1):5228–35, apr 2004.

[10] B. Grün and K. Hornik. topicmodels: An R Package for Fitting Topic Models. *Journal of Statistical Software*, 40(13):1–30, 2011.

[11] J. Hayes, A. Dekhtyar, and J. Osborne. Improving requirements tracing via information retrieval. In *Proceedings of the 11th IEEE International Requirements Engineering Conference*, pages 138–147, Monterey, CA, 2003. IEEE Comput. Soc.

[12] A. Hindle, C. Bird, T. Zimmermann, and N. Nagappan. Do topics make sense to managers and developers? *Empirical Software Engineering*, 20(2):479–515, jun 2014.

[13] G. J. Holzmann. Landing a Spacecraft on Mars. *IEEE Software*, 30(2):83–86, mar 2013.

[14] L. Layman, M. Zelkowitz, V. Basili, and A. P. Nikora. Toward Baselining Software Anomalies in NASA Missions. In *2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops*, pages 13–14, Dallas, Texas, USA, nov 2012. IEEE.

[15] E. Linstead and P. Baldi. Mining the coherence of GNOME bug reports with statistical topic models. In *2009 6th IEEE International Working Conference on Mining Software Repositories*, pages 99–102, Vancouver, BC, may 2009. IEEE.

[16] E. Linstead, C. Lopes, and P. Baldi. An Application of Latent Dirichlet Allocation to Analyzing Software Evolution. In *2008 Seventh International Conference on Machine Learning and Applications*, pages 813–818, San Diego, CA, 2008. IEEE.

[17] S. K. Lukins, N. A. Kraft, and L. H. Etzkorn. Source Code Retrieval for Bug Localization Using Latent Dirichlet Allocation. In *2008 15th Working Conference on Reverse Engineering*, pages 155–164, Antwerp, Belgium, oct 2008. IEEE.

[18] S. Mani, K. Sankaranarayanan, V. S. Sinha, and P. Devanbu. Panning requirement nuggets in stream of software maintenance tickets. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2014*, pages 678–688, Hong Kong, nov 2014. ACM Press.

[19] L. Martie, V. K. Palepu, H. Sajnani, and C. Lopes. Trendy bugs: topic trends in the Android bug reports. In *Proceedings of the 9th IEEE Working Conference on Mining Software Repositories*, pages 120–123, Zurich, Switzerland, jun 2012. IEEE Press.

[20] T. Menzies and A. Marcus. Automated severity assessment of software defect reports. In *2008 IEEE International Conference on Software Maintenance*, pages 346–355. IEEE, sep 2008.

[21] D. Mimno, H. M. Wallach, E. Talley, M. Leenders, and A. McCallum. Optimizing semantic coherence in topic models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '11)*, pages 262–272. Association for Computational Linguistics, jul 2011.

[22] NASA. Procedural Handbook for NASA Program and Project Management of Problems, Nonconformances, and Anomalies. Technical Report NASA Office of Safety and Missions Assurance, NASA, 2008.

[23] S. Neuhaus and T. Zimmermann. Security Trend Analysis with CVE Topic Models. In *2010 IEEE 21st International Symposium on Software Reliability Engineering*, pages 111–120, San Jose, CA, nov 2010. IEEE.

[24] D. Newman, Y. Noh, E. Talley, S. Karimi, and T. Baldwin. Evaluating topic models for digital libraries. In *Proceedings of the 10th Annual Joint Conference on Digital Libraries (JCDL '10)*, pages 215–224, Gold Coast, Australia, jun 2010. ACM Press.

[25] A. T. Nguyen, T. T. Nguyen, J. Al-Kofahi, H. V. Nguyen, and T. N. Nguyen. A topic-based approach for narrowing the search space of buggy files from a bug report. In *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*, pages 263–272, Lawrence, KS, nov 2011. IEEE.

[26] A. T. Nguyen, T. T. Nguyen, T. N. Nguyen, D. Lo, and C. Sun. Duplicate bug report detection with a combination of information retrieval and topic modeling. In *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering - ASE 2012*, pages 70–79, New York, New York, USA, 2012. ACM Press.

[27] B. Nuseibeh. Ariane 5: Who Dunnit? *IEEE Software*, 14(3):15–16, 1997.

[28] A. Panichella, B. Dit, R. Oliveto, M. Di Penta, D. Poshyvanyk, and A. De Lucia. How to effectively use topic models for software engineering tasks? an approach based on genetic algorithms. In *Proceedings of the 2013 International Conference on Software Engineering (ICSE '13)*, pages 522–531, San Francisco, CA, may 2013. IEEE Press.

[29] D. L. Parnas. Software aging. In *Proceedings of the*

16th International Conference on Software Engineering, pages 279–287, Sorrento, Italy, may 1994.

[30] N. Pingclasai, H. Hata, and K.-i. Matsumoto. Classifying Bug Reports to Bugs and Other Requests Using Topic Modeling. In *2013 20th Asia-Pacific Software Engineering Conference (APSEC)*, pages 13–18, Bangkok, Thailand, dec 2013. IEEE.

[31] P. Runeson, M. Alexandersson, and O. Nyholm. Detection of Duplicate Defect Reports Using Natural Language Processing. In *29th International Conference on Software Engineering (ICSE'07)*, pages 499–510, Minneapolis, MN, may 2007. IEEE.

[32] R. K. Saha, M. Lease, S. Khurshid, and D. E. Perry. Improving bug localization using structured information retrieval. In *2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 345–355, Palo Alto, CA, nov 2013. IEEE.

[33] B. J. Sauser, R. R. Reilly, and A. J. Shenhar. Why projects fail? How contingency theory can provide new insights - A comparative analysis of NASA's Mars Climate Orbiter loss. *International Journal of Project Management*, 27(7):665–679, oct 2009.

[34] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 101(476):1566–1581, dec 2006.

[35] S. W. Thomas, B. Adams, A. E. Hassan, and D. Blostein. Validating the Use of Topic Models for Software Evolution. In *2010 10th IEEE Working Conference on Source Code Analysis and Manipulation*, pages 55–64, Timisoara, Romania, sep 2010. IEEE.

[36] X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun. An approach to detecting duplicate bug reports using natural language and execution information. In *Proceedings of the 13th international conference on Software engineering - ICSE '08*, page 461, Leipzig, Germany, may 2008. ACM Press.

[37] X. Xia, D. Lo, X. Wang, and B. Zhou. Accurate developer recommendation for bug resolution. In *2013 20th Working Conference on Reverse Engineering (WCRE)*, pages 72–81, Kingston, ON, oct 2013. IEEE.

[38] X. Xie, W. Zhang, Y. Yang, and Q. Wang. DRETOM: Developer recommendation based on topic models for bug resolution. In *Proceedings of the 8th International Conference on Predictive Models in Software Engineering - PROMISE '12*, pages 19–28, Lund, Sweden, sep 2012. ACM Press.

[39] S. Yadla, J. H. Hayes, and A. Dekhtyar. Tracing requirements to defect reports: an application of information retrieval techniques. *Innovations in Systems and Software Engineering*, 1(2):116–124, jul 2005.

[40] W. Zhao, J. J. Chen, R. Perkins, Z. Liu, W. Ge, Y. Ding, and W. Zou. A heuristic approach to determine an appropriate number of topics in topic modeling. *BMC bioinformatics*, 16 Suppl 1:S8, jan 2015.

[41] J. Zou, L. Xu, W. Guo, M. Yan, D. Yang, and X. Zhang. Which Non-functional Requirements Do Developers Focus On? An Empirical Study on Stack Overflow Using Topic Analysis. In *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*, pages 446–449, Florence, Italy, may 2015. IEEE.