# Considerations for Obsolescence Prediction

Emily Nystrom & Anthony Leclerc
Naval Information Warfare Center Atlantic

**Abstract**

Our research considers statistical and mathematical techniques that demonstrate potential for improving the predictive capabilities used by the Navy in modeling hardware obsolescence. In Section 1, we define the type of obsolescence considered in our research (Section 1.1) and discuss common predictive methods in the obsolescence literature (Section 1.2). In Section 3, our experiments comparing the performances of random forests and random survival forests for modeling data with right-censored responses are described.

## 1 Background

### 1.1 Defining Obsolescence

Obsolescence is a broad term that encompasses a range of types (e.g., style obsolescence versus functional obsolescence) or reasons for some change in the useability of an item. In this report, we focus on hardware obsolescence that results when a component is discontinued by the original equipment manufacturer (OEM). Obsolescence considered in this report fits within the umbrella of *diminishing manufacturing sources and material shortages* (DMSMS), which is defined as "the loss, or impending loss, of manufacturers or suppliers of items, raw materials, or software" [1]. Software obsolescence is an area for future investigation, and extending our results for hardware obsolescence to software obsolescence would require additional research and validation on relevant data. Underlying mathematical and statistical assumptions should be considered. Some of the challenges (e.g., missing data, censored data, and data development over time) encountered in dealing with hardware obsolescence data may also be encountered in the context of software obsolescence.

### 1.2 Predictive Methods in the Obsolescence Literature

Strategies for handling obsolescence events are typically grouped into two main categories: reactive strategies, which respond to obsolescence events (e.g., announcements of impending or past obsolescence), and proactive strategies, which seek to anticipate obsolescence events and consider possible solutions prior to the onset of obsolescence [2]. Analytic tools, such as the methods described in this report, may be used to support proactive strategies for obsolescence management. Table 1 provides a sample of some questions of interest for quantifying and forecasting obsolescence.

The increased integration of commercial off the shelf (COTS) products in government systems (e.g., in line with the 1994 "Perry Memo"; [3]) has heightened the necessity of leveraging proactive obsolescence management strategies within the government [4]. As the number of COTS parts increases, the ability of government customers to guarantee component availability from the original source (the OEMs) throughout the system's intended operational lifecyle decreases.

#### 1.2.1 Using Market Demand Data and the Normal Curve to Model Lifecycle

Early research for obsolescence prediction considered using market data to leverage the supply-demand driver of obsolescence [13]. [13] modeled the lifecycle as a normal distribution, where the lifecycle stages are marked by the number of standard deviations from peak demand (Fig. 1). In [13], the maturity phase is the region within one standard devation ($\sigma$) of the mean ($\mu$) (i.e., $(\mu - \sigma, \mu + \sigma)$), and the obsolescence stage (or "zone of obsolescence") occurs 2.5 to 3.5 standard deviations[1] above the mean (i.e., $(\mu + 2.5\sigma,\ \mu + 3.5\sigma)$). In [13], values for $\mu$ and $\sigma$ are estimated from historical market data for the specific hardware component under consideration, whereas some more recent machine learning approaches (e.g., [5]) shift reliance away from component-specific data toward data sources that are more generic in terms of the components and timeframe used for collection and maintenance. Generalizations of [13]'s concept, such as additional distributions for the demand curve, have also been considered.

Relevant Questions for Quantifying and Forecasting Obsolescence

1. **Predict Obsolescence Date**: *When will a component become obsolete?* [5–7]
2. **Predict Current or Future Lifecycle Stage**:
   *What is the current lifecycle stage of a component?* [5]
   *What will the component's lifecycle stage be at a future point in time?*
3. **Estimate Future Obsolescence Risk**:
   *What is the likelihood that obsolescence will occur prior to a specified time?* [8]
   *What is the likelihood that obsolescence will occur during a specific time interval?*
4. **Estimate Obsolescence Cost**:
   *What is the expected cost of procurement based on various obsolescence dates and alternative sources?*
   *What is the optimal system design, in terms of selection of components and in light of obsolescence risk?* [9]
5. **Estimate the Tradeoff between Component Obsolescence and System Replacement**:
   *Under which conditions will the expected cost of obsolescence at the component level exceed the expected cost of system replacement?* [10]
6. **Estimate Risk of Counterfeit**:
   *What is the relationship between obsolescence risk and counterfeit susceptibility?* [11]
   *What is the likelihood that a component procured from an alternative source is counterfeit?*
7. **Estimate Replacement Technology Lineage and Timeline**:
   *What part will replace the current part?*
   *What is the "gap" length, during which time period neither the old nor the new components are available?*
   *How are different versions of components related?* [12]

Table 1: Sample questions that motivate and focus the application of predictive techniques to obsolescence data with the goal of building models that facilitate a better understand the future obsolescence environment.

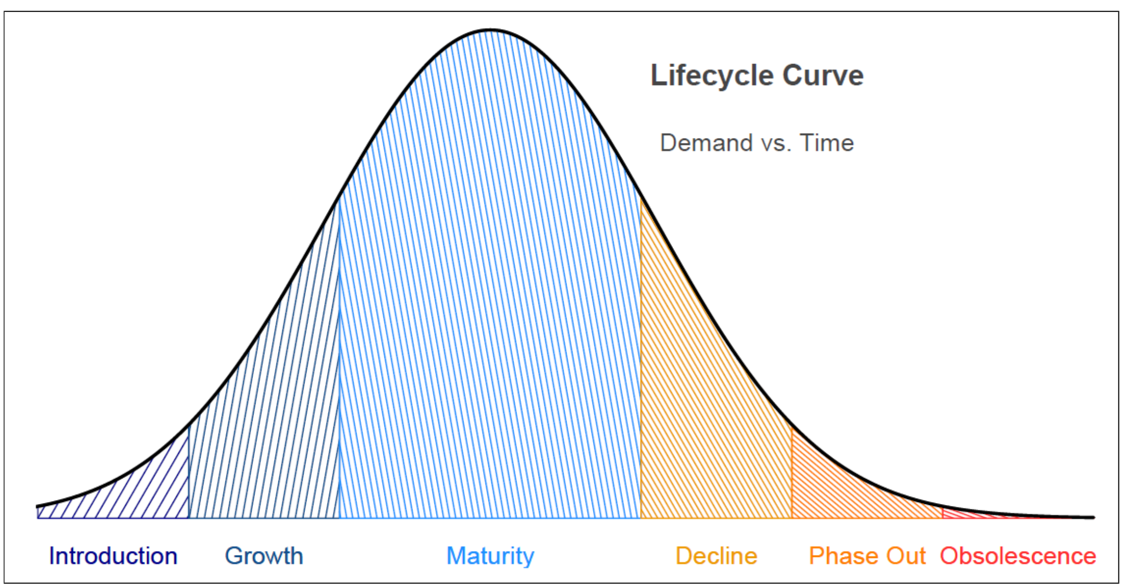



Figure 1: The lifecycle curve (as in [13]) depicts demand versus lifecycle phase, where the start and end points for each lifecyle phase are determined by the distance (in standard deviations) from the demand peak.

### 1.2.2 Linear Models for Relating Lifecycle Length and Introduction Date

A discussion of the dependence of lifecycle length on part introduction year (the "cone of obsolescence" [6]) is included in [4]. [4] includes separate least squares (LS) estimated regression equations for introduction date ($Y_1$) and obsolescence date ($Y_2$) based on the peak demand ($X$); these equations can then be used to derive the LS estimated regression coefficients for a model of lifecycle length ($L$). [4] does not provide detail about any statistical analysis performed.

In Equations 1 to 5, we provide a framework for comparing intermediate models for introduction date and obsolescence date (Eq.s 1 and 2, respectively), from which the difference (Eq. 5) is derived, to a directly-defined model for lifecycle length (Eq. 3). Our contribution frames models that appear in the obsolescence prediction literature (Eq.s 1, 2, and 5 below, expanded from [4]) in terms of well-established statistical techniques and methods:

| Response Name | Linear Model | Error Structure | |
|---|---|---|---|
| Introduction Date: | $Y_{1i} = \beta_0 + \beta_1 X_i + \epsilon_{1i},$ | $\epsilon_{1i} \overset{\text{iid}}{\sim} N\left(0, \sigma_1^2\right).$ | (1) |
| Obsolescence Date: | $Y_{2i} = \gamma_0 + \gamma_1 X_i + \epsilon_{2i},$ | $\epsilon_{2i} \overset{\text{iid}}{\sim} N(0, \sigma_2^2).$ | (2) |
| Lifecycle Length: | $L_i = \alpha_0 + \alpha_1 X_i + \epsilon_{3i},$ | $\epsilon_{3i} \overset{\text{iid}}{\sim} N\left(0, \sigma_3^2\right).$ | (3) |
| Difference: | $D_i = Y_{2i} - Y_{1i}$ | | (4) |

$$= \underbrace{\gamma_0 - \beta_0}_{\alpha_0} + (\underbrace{\gamma_1 - \beta_1}_{\alpha_1}) X_i + \underbrace{\epsilon_{2i} - \epsilon_{1i}}_{\epsilon_{3i}}, \quad \epsilon_{3i} \overset{\text{iid}}{\sim} N\left(0, \sigma_3^2 = \sigma_1^2 + \sigma_2^2 - 2\sigma_{12}\right),$$

where $\sigma_{12}$ denotes the covariance between $Y_1$ and $Y_2$. We note that the responses in Equations 1 to 5 have non-negative support sets, making these normal distributions *approximations* only. Other statistical distributions may be considered if the enforcement of the support set becomes particularly relevant.

The vector of estimated regression coefficients, denoted by $\widehat{\boldsymbol{\alpha}} = (\widehat{\alpha}_0, \widehat{\alpha}_1)^T$, follows a normal distribution:

$$\widehat{\boldsymbol{\alpha}} = \widehat{\boldsymbol{\gamma}} - \widehat{\boldsymbol{\beta}} = \left[\mathrm{X}^T \mathrm{X}\right]^{-1} \mathrm{X}^T \left(\mathbf{Y}_2 - \mathbf{Y}_1\right) \sim N\left(\boldsymbol{\mu}_{\widehat{\boldsymbol{\alpha}}} = \boldsymbol{\alpha}, \Sigma_{\widehat{\boldsymbol{\alpha}}} = \sigma_3^2 \left[\mathrm{X}^T \mathrm{X}\right]^{-1}\right), \tag{5}$$

where $\Sigma_{\widehat{\boldsymbol{\alpha}}}$ denotes the covariance matrix for $\widehat{\boldsymbol{\gamma}}$, $\mathbf{Y}_j = (Y_{j1}, Y_{j2}, ..., Y_{jn})^T$ denotes the $j^{\text{th}}$ response vector ($j = 1, 2$), $\mathbf{X} = (X_1, X_2, ..., X_n)^T$ denotes the predictor vector (here, the peak demand), and $\mathrm{X} = [\mathbf{1}, \mathbf{X}]$ denotes the $n$ by 2 design matrix.

Thus, the LS estimated regression coefficients ($\widehat{\alpha}_0$ and $\widehat{\alpha}_1$) obtained from the estimated regression equation for lifecycle length (Eq. 3) are equal to the values obtained by differencing the LS estimated regression coefficients from the separate regression equations for obsolescence date (Eq. 2; $\widehat{\gamma}_0$ and $\widehat{\gamma}_1$) and the introduction date (Eq. 1; $\widehat{\beta}_0$ and $\widehat{\beta}_1$):

$$\widehat{\alpha}_j = \widehat{\gamma}_i - \widehat{\beta}_j, \text{ for } j = 0, 1.$$

The vector of standard errors for $\widehat{\boldsymbol{\alpha}}$, denoted by $SE_{\widehat{\boldsymbol{\alpha}}} = (SE_{\widehat{\alpha}_0}, SE_{\widehat{\alpha}_1})^T$, can be obtained indirectly, without fitting the model for lifecycle length (Eq. 9) *if* the response vectors ($\mathbf{Y}_1$ and $\mathbf{Y}_2$), the estimated response vectors ($\widehat{\mathbf{Y}}_1$ and $\widehat{\mathbf{Y}}_2$) from the individual models, and the design matrix (X) are available:

$$\Sigma_{\widehat{\boldsymbol{\alpha}}} = \sigma_3^2 \left[\mathrm{X}^T \mathrm{X}\right]^{-1}, \text{ where } \sigma_3^2 \text{ denotes the variance of } \epsilon_{3i}. \tag{6}$$

$$s_3^2 = \frac{\sum_i \left(L_i - \widehat{L}_i\right)^2}{n-2} \tag{7}$$

$$= \underbrace{\frac{\sum_i \left(Y_{1i} - \widehat{Y}_{1i}\right)^2}{n-2}}_{s_1^2} + \underbrace{\frac{\sum_i \left(Y_{2i} - \widehat{Y}_{2i}\right)^2}{n-2}}_{s_2^2} - 2 \underbrace{\frac{\sum_i \left(Y_{2i} - \widehat{Y}_{2i}\right)\left(Y_{1i} - \widehat{Y}_{1i}\right)}{n-2}}_{s_{12}}. \tag{8}$$

$$SE_{\widehat{\boldsymbol{\alpha}}} = s_3 * \text{Diag}\left\{\left[\mathrm{X}^T \mathrm{X}\right]^{-1}\right\}^{1/2}. \tag{9}$$

However, calculating $\widehat{\boldsymbol{\alpha}}$ directly (by estimating the lifecycle model) requires less information (only $\mathbf{X}$ and $\mathbf{Y}_1 - \mathbf{Y}_2$) than the indirect method (Eq.s 6-9).

Like [7] and [6], [14] analyzed data extracted from the SiliconExpert database. [14]'s methods include a discussion of "failure time analysis" for analyzing lifecycle length. Included in [14] is a graph that illustrates the inverse relationship between the number of parts that are not obsolete at a given introduction date and the fraction of parts that are not obsolete at a given introduction date. The estimated regression equation reported in [14] (Eq. 10) for predicting lifecycle length ($L$) using market introduction date ($t_{\text{intro}}$) is based on a dataset of linear regulators introduced between 1990 and 2005 (anlayzed around 2010). The estimated regression equation (Eq. 11) reported in [15] has the same underlying model/format and appears[2] to have been fitted to data with introduction dates ranging from 1993 to 2007:

$$\widehat{L} = 0.021089\, t_{\text{intro}}^2 - 85.2217\, t_{\text{intro}} + 86095.7\,. \quad (10)$$

$$\widehat{L} = 0.0789\, t_{\text{intro}}^2 - 316.93\, t_{\text{intro}} + 318133\,. \quad (11)$$

As [14] explains, the useability of statistical methods like the framework described in [14] are restricted to predicting small future changes, and extrapolated predictions may fail when major changes, such as unprecidented design changes, technological breakthroughs, or spikes in market demand due to additional demand sources (e.g., new systems or repurposes for existing technology resulting in a noteable change and/or change in the direction of market demand, especially when not counterbalanced by additional supply sources and/or expansion of production capabilities), are encountered.

#### 1.2.3 Viewing Obsolescence at the Piece Part Level

In some instances, obsolescence of a part or system is driven by obsolescence of individual component(s) or subsystem(s) – the lifecyle mismatch problem (e.g., [13]). Strategies like those implemented by the Sunset Supply Base (SSB) [16] and implemented in Q-Star drill down to identify the lowest-level components ("piece parts") that are driving obsolescence of larger systems with the idea that in some cases, handling piece part obsolescence is enough to mitigate obsolescence of the larger system. In our report, we focus on the obsolescence of individual components. On the other hand, [9], for example, focuses on system-level obsolescence.

#### 1.2.4 Machine Learning for Obsolescence Prediction

[5] discusses using machine learning to predict obsolescence date and obsolescence status (e.g., active or obsolete) for parts. The article includes empirical results for data with different types of cell phones and cameras. Techniques applied included basic implementations of random forests, artificial neural networks[3], and support vector machines. In [5], results are separated into two sections, which the authors named *Obsolescence Risk Machine Learning* (ORML) and *Lifecycle Machine Learning* (LCML). ORML classifies parts by obsolescence statuses: active or obsolete; LCML predicts obsolescence date[4]. Of particular relevance for our research, [5] reports that the ORML implementation of random forests yielded superior experimental results (compared to neural networks and support vector machines) for accuracy, interpretability, and maintainability/flexibility[5]; but the random forests algorithm was the slowest[6] algorithm implemented in [5].

## 2 Considerations for Modeling Obsolescence Data

### 2.1 Censoring in the Context of Obsolescence Data

[14] discusses the procurement lifecyle length in light of survival analysis (or "failure time analysis"). In context of survival analysis, the term *right censored* describes data for which only a lower bound value is known. Among other reasons, data may be right-censored because a terminal event (e.g., obsolescence) has not occured by the time of data collection; that is, the dataset as a whole is only partially-developed, containing a mixture of complete and incomplete records. Suppose Part H was introduced in 2010 and had not become obsolete at the end of 2017 when a particular dataset was updated. In this case, the obsolescence date is know to be sometime in the future. Instead of leaving the obsolescence date blank, the obsolescence date can be recorded as $2017^+$ to indicate that the date is right-censored (denoted by the + superscript) and that the true obsolescence date is yet future. Left-censoring is similarly defined and occurs in data for which only an upper bound is known (e.g., for a part is known to have been introduced by 2015, $2015^-$ indicates that 2015 is an upper bound for the intro date).

As described in [14], the length of time allowed for a dataset to develop may have a censoring effect on the data available for analysis. In a plot of lifecycle length versus introduction date, the lack of datapoints in the upper triangular region (top right corner) is an artifact of right censoring in cases where the range of introduction dates shown (e.g., the horizontal axis range) nears the date of data collection. The maximum lifecycle length ($L^*_{\max}$)

observable at the time of analysis ($t_{\text{analysis}}$) provides an upper bound on the observable lifecycle length ($L$) and is given by

$$L^*_{\max} = t_{\text{analysis}} - t_{\text{intro}}. \tag{12}$$

Figure 2 illustrates one possible realization of a fully-developed dataset simulated from a given partially-developed dataset. Data were simulated to illustrate development of obsolescence data over time: as of 2018 (left); as of 2035 (right). Each plot displays market introduction date (year) versus procurement lifecycle length. The apparent relationship between introduction year and lifecycle length may be biased if the maximum observable lifecycle length at the date of analysis (or the last data update) is not considered. At present, data are only partially developed (e.g., some components are not yet obsolete, so their true lifecycle length is unknown) and will continue to develop into the future (e.g., more components will become obsolete and render final lifecycle lengths). The apparent strength of linearity between introduction date and lifecycle length observed in this realization of the fully-developed data (future) is minimal, compared to the naive/initial (non-conditional) perceived relationship gathered from the partially-developed (present) data. [14] refers to the maximum observable lifecycle length (blue line) as the "analysis

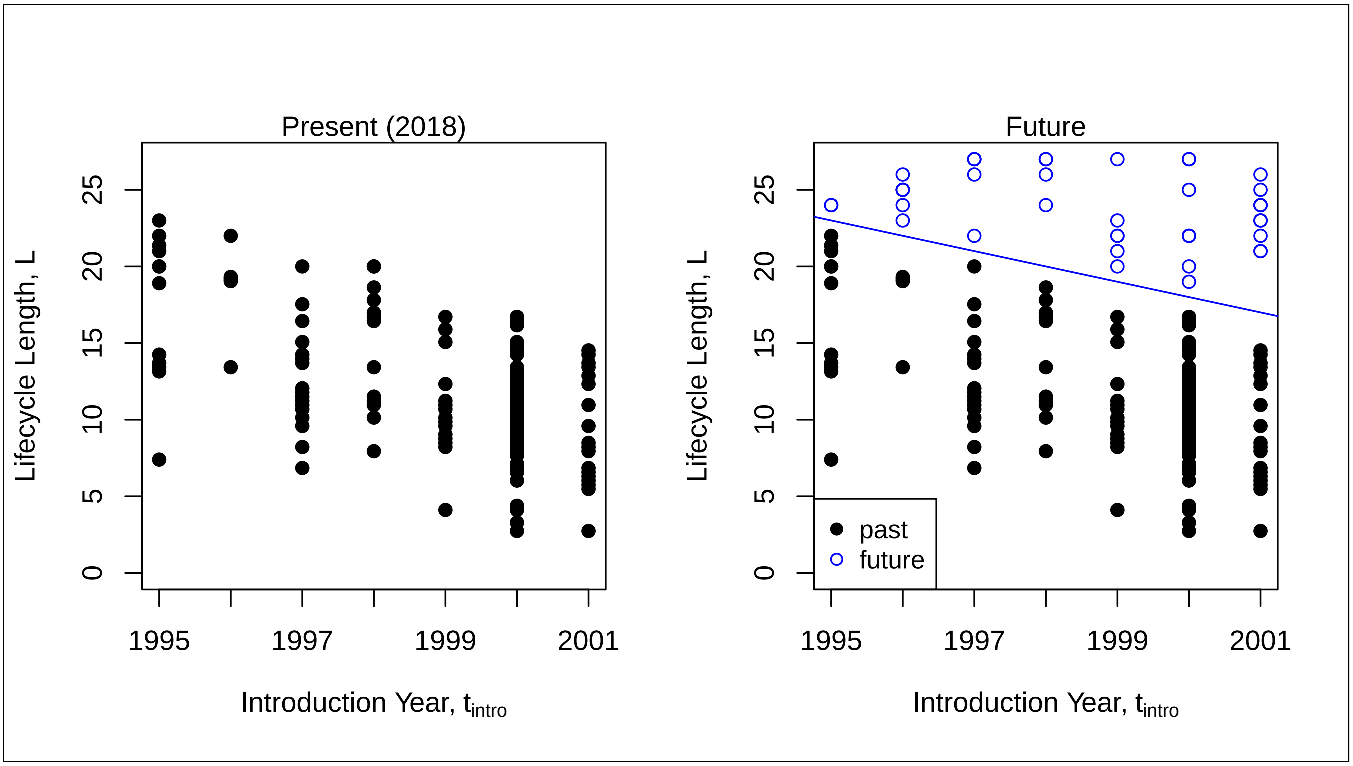



Figure 2: An Illustration of the Development of Obsolescence Data over Time.
Data were simulated to illustrate development over time: as of 2018 (left); as of 2035 (right). The position of each data point is determined by the introduction year and the lifecycle length ($L = t_{\text{obsolescence}} - t_{\text{intro}}$). The color of each data point indicates whether the full lifecycle length ($L$) was observed in the past (black) or the future (blue) (i.e., whether or not the part is obsolete at the time of analysis). The blue line indicates the maximum lifecycle length that can by observed up to the time of analysis (i.e., $L^*_{\max} = t_{\text{analysis}} - t_{\text{intro}}$).

boundary" and advises that

> "the bottom boundary of the data .... is the key to forecasting the worst case procurement life. ... [Procurement age] has [an] effect on the procurement life [if and only if] the bottom boundary of the data set [a boundary which is based on the minimum observed lifecycle lengths at each given introduction date] .... [has a] nonzero slope"[7].

## 2.2 Overlap between Obsolescence and Reliability Timelines

The procurement lifecycle often overlaps in time with the reliability lifetime, which refers to the time between installation $t_{\text{in}}$ and replacement $t_{\text{out}}$ for a specific instance of a component (Fig. 3).

Although the timelines overlap, data records are often maintained by separate entities. Another noteable difference– the start and endpoints for the reliability lifetime vary by instance, whereas all instances of a part have identical procurement lifecycles (e.g., have identical introduction dates and obsolescence dates). This distinction between the reliability lifetime and the procureability lifecycle motivates consideration of different techniques between the separate spaces. For example, for a single component, experiments for studying the reliability lifetime can be designed to use multiple replicates by recording data from multiple instances of the same component.

An understanding of the reliability timeline can allow for a better understanding of the impact of obsolescence. For example, estimating the number of spares to buy upon recieving a last time buy notice requires an understanding of the expected reliability lifetime. Specifically, the estimation formula in [8] (p.13) defines the probability of an obsolescence issue as the product of the probability of obsolescence and the probability of spare inventory depletion.

## 2.3 Modeling Manufacturer Survey Responses

This section suggests leveraging finite state stochastic processes, such as Markov chains, to build a foundational framework for modeling the manufacterer survey process. Definitions for Markov chains are provided in Section B.1; see [17] for a full introduction to Markov chains. In this section, we motivate the use of stochastic processes with visualizations using stochasic processes for the manufacturer survey process. Communications with the OEM are

| | G | O | R |
|---|---|---|---|
| G | $p_{G,G}$ | $p_{G,O}$ | $p_{G,R}$ |
| O | 0 | $p_{O,O}$ | $p_{O,R}$ |
| R | 0 | 0 | $p_{R,R}$ |

Table 2: Transition Probabilities for the OEM Survey Process modeling the Vendor Part Number (VPN). Rows indicate the source state ($i$), and columns indicate the destination state ($j$). The entry in row $i$ column $j$ corresponds to $p_{i,j}$, the probability of moving from state $i$ to state $j$ in one step (i.e., $P\left(X_{k+1} = j \middle| X_k = i\right)$).

used by some obsolescence management teams as part of a proactive strategy for anticipating obsolescence. Periodic surveys sent to OEMs by the obsolescence management teams are one such communication tool. The obsolescence status can be viewed as a stochastic process. Althought Markov chains are specifically mentioned here, the Markov property may not hold for the specific problem under consideration. We discuss survey responses in terms of the statespace and transition diagrams commonly used for Markov chains, but a non-Markovian stochastic process with a finite statespace may also satisfy our purposes. See [17] for more information about stochastic processes.

Two common types of Markov chains, the continuous time Markov chain (CTMC) and a discrete time Markov chain (DTMC), are distinguished by their handling of time (continous versus discrete). The CTMC monitors the state transitions without breaks (Fig. 4(a)), whereas the DTMC monitors the state transitions at discrete time points, such as the transition times (e.g., times at which the state changes) or the times at which the OEM is surveyed (Fig. 4(b); Fig. 5). The CTMC described here aligns with our desire to predict the lifecycle length or the time of obsolescence. The DTMC monitored at survey timepoints may also provide insight into the lifecycle length but because of the discrete nature of results from such a DTMC will not align as closely with results from methods for predicting obsolescence date in a continuous range. A strength of the survey DTMC is its potential usefulness in investigateing the expected impact of the timing and/or regularity of surveys with respect to the usefulness of gaining actionable information from the surveys.

| **State** | **VPN State Description** |
|---|---|
| Red | The original OEM component is not available, and no alternates are available. |
| Orange | A Life Time Buy (LTB) alert has been issued, but the part is still procurable. |
| Green | The approved VPN is available. |

Table 3: Vendor Part Number (VPN) State Descriptions (based on [18]).

# 3 Our Consideration of Random Forests

## 3.1 Random Forests (RF) and Random Survival Forests (RSF)

Introduced by [19] in 2001, the random forests (RF) algorithm is an ensemble machine learning method that uses randomization to effectively leverage a forest of decision trees in a fashion that reduces dependence among trees.

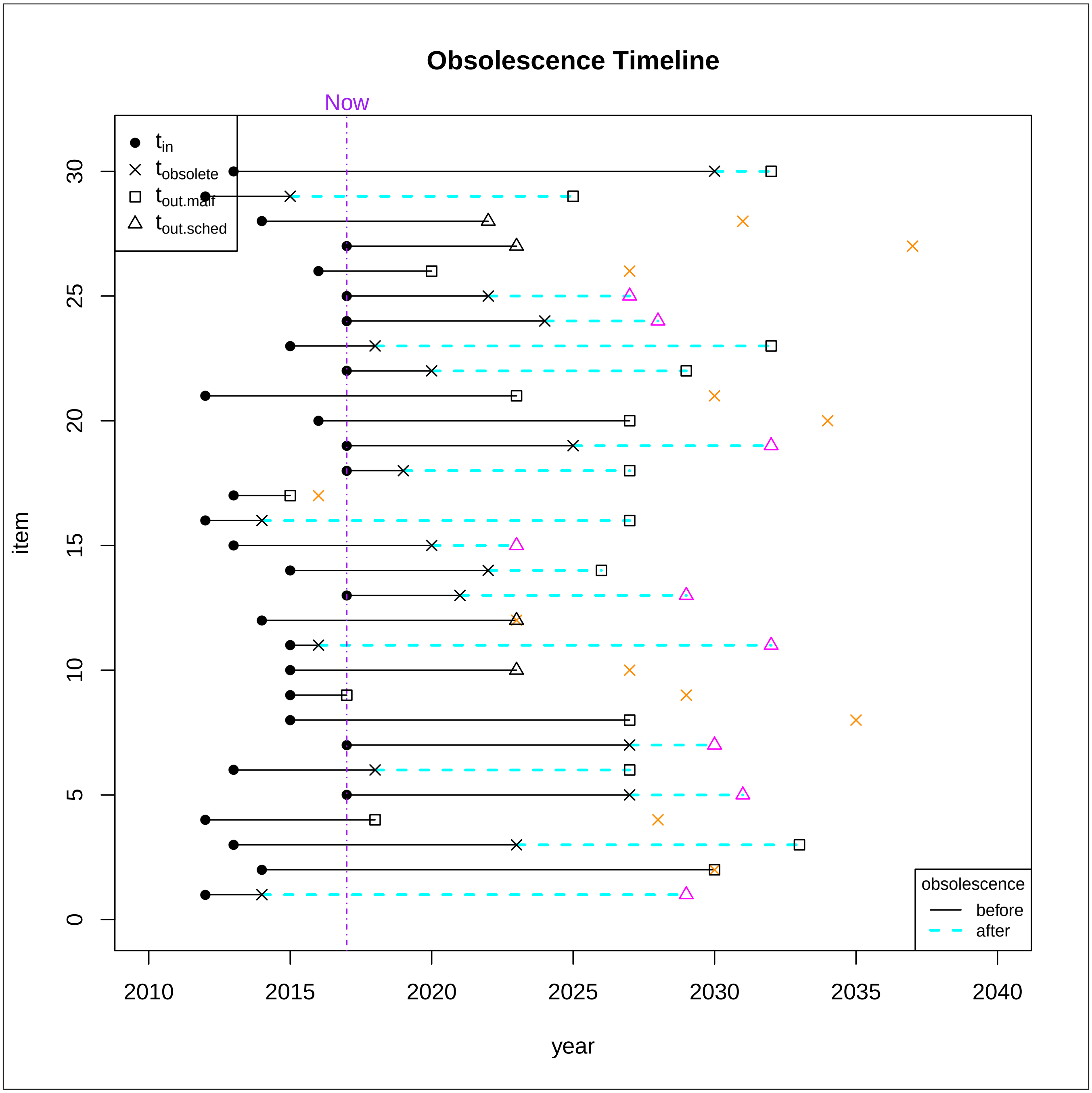
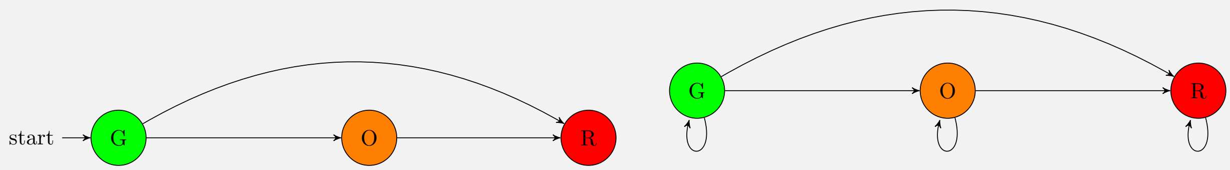



Figure 3: **An Illustration of the Overlap between Obsolescence and Reliability Timelines.** For each point, the type of event (e.g., install, removal, and obsolescence) is indicated by the shape. Circles indicate install times ($t_{\text{in}}$), and x's indicate obsolescence dates ($t_{\text{obsolete}}$). Squares (□) and triangles (△) indicate removal times ($t_{\text{out}}$), at which items (one particular instance of a component) were or will be removed due to either malfunction ($t_{\text{out.malf}}$) or scheduled maintenance ($t_{\text{out.sched}}$), respectively. For horizontal line segments, line color and line type distinguish connections between pre- versus post-obsolescence events. Solid black lines connect $t_{\text{in}}$ to the earlier of $t_{\text{out}}$ and $t_{\text{obsolete}}$. Dotted turquoise lines connect $t_{\text{obsolete}}$ to $t_{\text{out}}$ when obsolescence preceeds removal. Point color corresponds to logical considerations derived from the order of events (within an item). When connected by a black line, the start and end points are colored black. When obsolescence preceeds scheduled maintenance (pink △), scheduled maintenance may be delayed or cancelled due to reduced availability of replacement parts. When scheduled maintenance preceeds obsolescence (orange △), obsolescence is inconsequential for the specific item. Part lifetime (in the sense of reliability) is the sum of the black and teal lines, and lifecycle length (in the sense of procurability/obsolescence ) is the span from the part introduction date (which is not shown here but is some date prior to the installation date, $t_{\text{in}}$) to the obsolescence date (△). The dashed, vertical purple line indicates the present time ("now"), after which (actual, non-simulated) data would not yet be available.

**Transition Diagrams for Vendor Part Number**




(a) Transition Diagram for a Continuous Time Markov Chain (CTMC), where states are monitored continuously and nontrivial transitions (i.e., movements between states) are recorded.

(b) Transition Diagram for a Discrete Time Markov Chain (DTMC), where states are recorded at a discrete sequence of surveys to the original equipment manufacturer (OEM).

Figure 4: Transition Diagrams for (a) Continous and (b) Discrete Time Markov Chains.
The diagrams shown here describe the transition of a given Vendor Part Number (VPN) among possible states (green, orange, and red), which correspond to changes in the obsolescence status.

**Counting Manufacturer Surveys**

Notation:

- Let $t_{\text{intro}}$ denote the market introduction date, at which the VPN status turns green.
- Let $t_{\text{LTB}}$ denote the time at which the Last Time Buy notice is sent out and the VPN status turns orange.
- Let $t_{\text{obsolete}}$ denote the obsolescence date, at which point the VPN status turns red.
- Let $T_s$ denote the time of the $s^{th}$ survey.
- Let $N(t)$ denote the number of surveys that have occurred at or before time $t$ (i.e., $N(t) = \sum_s I(T_s \leq t) = \max\{s : T_s \leq t\}$).
  It follows that $N(T_s) = s$ and $N(T_s^-) = s - 1$, where the negative sign superscript denotes the time instantly prior (to the left of) $T_s$. Assume $N(t_{\text{intro}}) = 0$ (e.g., $N(0) = 0$ if time is aligned with the origina; that is, if $t_{\text{intro}} = 0$) unless stated otherwise.

The number of surveys that occur in each state is given by the following table (Table 4; *assuming that surveys reflect current information as of the survey time/no lag time between CTMC jumps and OEM awareness, which makes sense b/c the OEM is probably the party initiating the announcement VPN status changes).

| state | number of surveys | time elapsed |
|---|---|---|
| Green | $N(t_{\text{LTB}}^-) - N(t_{\text{intro}}^-)$ | $T_{N(t_{\text{LTB}}^-)} - T_{N(t_{\text{intro}}^-)}$ |
| Orange | $N(t_{\text{obsolete}}^-) - N(t_{\text{LTB}}^-)$ | $T_{N(t_{\text{obsolete}}^-)} - T_{N(t_{\text{LTB}}^-)}$ |
| Red | $N(t) - N(t_{\text{obsolete}}^-)$ | $T_{N(t)} - T_{N(t_{\text{obsolete}}^-)}$, where $t \geq t_{\text{obsolete}}$ |

Table 4: Time Elapsed and Number of Surveys conducted, by VPN obsolescence status.

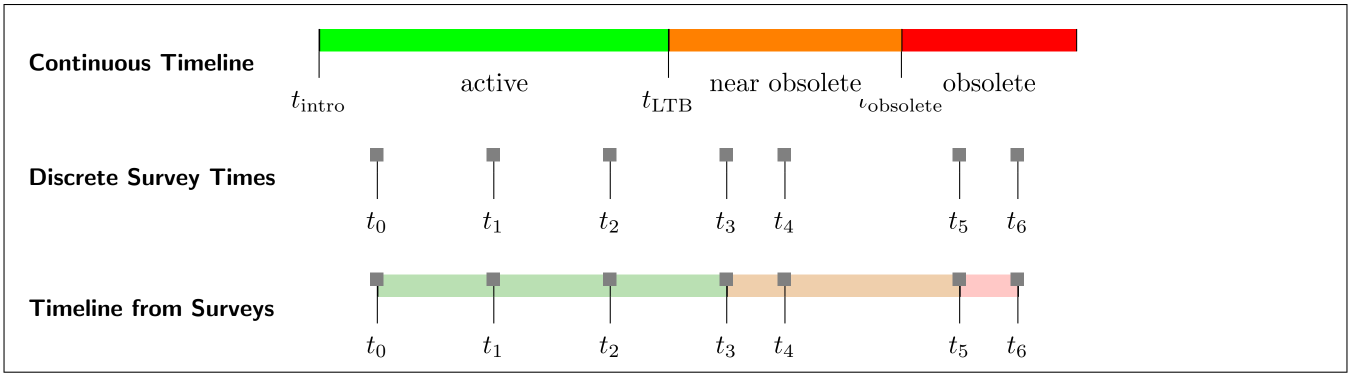



Figure 5: Obsolescence and Survey Timelines.
Although obsolescence has a **continous timeline**, surveys to the manufacturer occur as discrete time points. When the true values of the obsolescence timeline (e.g., $t_{\text{intro}}$, $t_{\text{LTB}}$, and $t_{\text{obsolescence}}$) are unavailable, the statuses collected from **discrete time surveys** can be used to construct a **timeline from surveys**, in which the obsolescence status is assumed (for lack of information) to remain constant until time of the next survey.
*Considerations:* When reconstructing a timeline from surveys, the time-between-surveys is an important factor in driving the expected ability of to reconstruct a timeline that closely matches the continous (true) obsolescence timeline. Another consideration – allowed movements between statuses are also important in determining the appropriateness of a discrete reconstruction. For example, if a process can go from red to green and back, a discrete reconstruction showing a constant green at consecutive survey points is less informative than consecutive greens pulled from a process that cannot degrade into previous statuses (e.g., a process that cannot return to lower states; a process that cannot return to green once it has passed into orange).

Randomization appears in two aspects of the algorithms. For each tree, a subset of observations is randomly selected from the full dataset and used to construct the tree. For each node within a tree, a subset of features is randomly selected, from which the "optimal" node-splitting feature is determined.

In 2008, [20] introduced the random survival forests (RSF) algorithm, which is based on [19]'s RF algorithm and is tailored for handling right-censored response data from a survival analysis perspective. Compared to the traditional RF algorithm, primary distinctions of the RSF algorithm include differences in node splitting rules, terminal node conditions (e.g., counting uncensored observations instead of total observations in a node), and the type of outcome that is predicted (e.g., predicted mortality, instead of a predicted response). By default, the RSF algorithm predicts mortality, whereas the RF algorithm predicts response. Because mortality and response are not directly comparable, we developed an algorithm to derive predicted responses from the predict mortalities output by the RSF algorithm. Our transformation algorithm leverages the area under the survival curve (see Section B.2) to allow comparison between responses predicted by the RF and RSF algorithms.

## 3.2 Experimenting with Random Forests

### 3.2.1 Design for Comparing Forests

In this section, we provide an experimental setup for comparing the performances of traditional random forests and random survival forests.

**Experimental Setup**

1. Identify a dataset and a continuous response to use in the experiment.
   - The dataset may contain real and/or simulated data.
   - The size of the dataset should be taken into consideration. Both the number of features and the number of observations should be "reasonably large" to allow for a useable implementation of the RF algorithm. In practice, the usefulness of the RF algorithm may be limited for some applications because of limitations in the availability of datasets of sufficient size, in terms of the number of features or observations.
   - Values recorded in the dataset will be assumed to be "true," fully-observed values for the purpose of this experiment. Missing data should be handled, and the resulting dataset will be treated as the "true"/"original" dataset. In general, missing data deserves careful consideration. Our missing data handling allows us to isolate censoring in experimental data (without adding possible missingness) and does not reflect a recommendation for general missing data handling.

2. Induce right censoring into the response data.
In this step, some responses will be right censored (i.e., the observed value will be less than the true value), while other responses will remain unchanged (i.e., the observed value will be equal to the true value). Record binary censored status ($\delta$) accordingly.

3. Train random forests and random survival forests.
   - Train a random forest using the original dataset (from 1.), in which all responses are uncensored ($Y$).
   - Train a random forest using observed responses ($Y^*$; from 2.), in which some responses are right-censored.
   - Train a random forest using observed responses ($Y^*$) and include censoring status ($\delta$) as an additional *predictor*.
   - Train a random survival forest using observed responses ($Y^*$) along with censoring statuses ($\delta$); the observed response and censoring status are paired in the survival object (the response input into the random survival forest).

| **Status** | $\delta$ | $Y^*$ | **Observed Time** | **Observed Length**, $L^*$ |
|---|---|---|---|---|
| censored | 0 | $C$ | $t_{\text{analysis}}$ | $L^*_{\text{max}}$ |
| uncensored | 1 | $Y$ | $t_{\text{obsolescence}}$ | $L$ |

Table 5: Example observed time and censoring status, in the context of obsolescence and procurement lifecycle length. $Y^*$ denotes the observed response value, which can be either the true response value ($Y$) or a right-censored value ($C$). The status of the observation is denoted by $\delta$. According to conventional notation used in survival analysis, $\delta$ equals zero when the true value of the response has not been observed and one when the true response has been observed. In this table, parallels are given for observed time and observed length (from Section 2.1, Eq. 12).

4. Compare predicted responses ("testing" dataset) from the forests.
   (a) Compare results from original data versus observed data.
      - Fix the algorithm to be used (RF vs. RSF).
      - Vary the response data used (original versus observed; $Y$ vs. $Y^*$) for the training data.
      - Vary the response data used (original versus observed; $Y$ vs. $Y^*$) for the testing data.
      - Fix the features to be provided to the algorithm.
      - Compare predicted responses from the random forest fit to the original (all uncensored) responses ($Y$) with the predicted responses from the random forest fit to the observed (some censored) responses ($Y^*$).

   (b) Compare results fit to the observed (some censored) responses ($Y^*$).
      - Vary the algorithm used (RF vs. RSF).
      - Fix the response to be used (observed, $Y^*$) for the training data.
      - Vary the responses to be used (original versus observed; $Y$ vs. $Y^*$) for the testing data.
      - Fix the features to be provided to the algorithm, except for censor status.
      - Vary the provision of censor status ($\delta$), as appropriate. Censor status must be included with the response input into the RSF algorithm; thus the role of censor status is fixed in the RSF algorithm. We can, however, consider adding censor status ($\delta$) as a feature in the RF.
      - Using the observed (some censored) responses ($Y^*$) to train the algorithm, compare responses predicted by the RF fit to responses predicted from the RSF.
      By default, the RSF algorithm predicts mortality, wherease the RF algorithm predicts the response value. In order to compare predictions from the RF and the RSF algorithms, we created an algorithm that predict response values based on survival curves, which can be returned by the RSF algorithm. (See Section B.2 for more detail about deriving estimated responses from estimated survival curves.)

   (c) Compare results, focusing on one particular censor status (censored vs. uncensored).
      - For experiments 4a and 4b, investigate the effectiveness of the implementations, by censor status.

- Consider whether the methods tend to perform better for censored responses, uncensored responses, or neither.

**Experimental Factors**

- RF variant: traditional or survival
- Training data: true or censored
- Testing data: true or censored

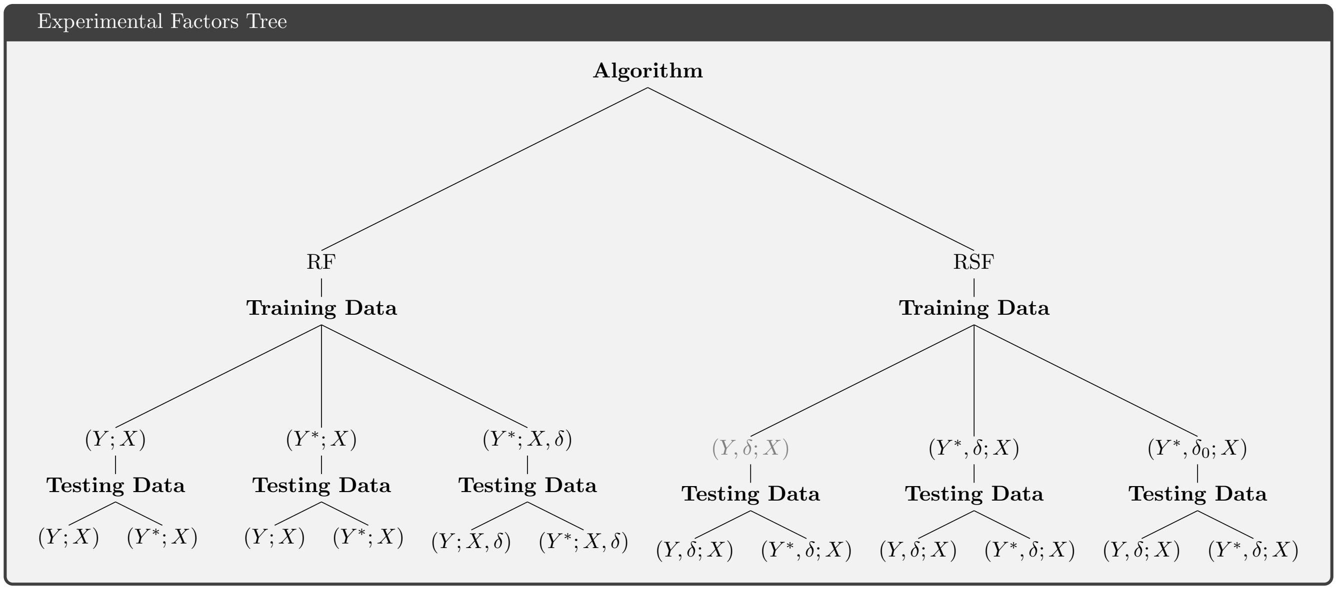



Figure 6: Experimental Factors Tree for Random Forest and Random Survival Forest Experiments.
Factors include algorithm, training data, and testing data. Notation: $Y$ denotes true responses, $Y^*$ denotes observed responses (some censored), $X$ denotes features (other than censor status), $\delta$ denotes censor status, and $\delta_0$ denotes the randomly generated "placebo" censor status. The semi-colon (;) is used to separate inputs treated as part of the response from features (e.g., (response; features) or (response; features, another feature).

**Additional Factors for Future Consideration**

- $p_{\text{censored}}$: percentage of observations for which the response is right-censored
- RF rule: splitting rule used for the RF algorithm
- RSF rule: splitting rule used for the RSF algorithm
- $n$: sample size; total number of observations
- evaluation method:
  - Method A (*Subset Data*): Partition observations in disjoint "testing" and "training" subsets. Observations in the training data are used to build the forest. Then used the entire forest (all trees) to predict response values *for observations in the testing data.*
    Method A is commonly used for various machine learning methods.
  - Method B (*Subset Trees*): Use the full dataset to train a random forest. Then, for each observation, use *trees for which the observation was "out-of-bag" (OOB)* to predict the response value.
    Method B is available for RSFs and RFs as a consequence of way observations are sampled from the full dataset and input into each tree in the forest.

**Additional Experiments and Future Directions**

- Compare results by censor status generation method – using a randomly-generated ("placebo") censor status ($\delta_0$) versus the actual censor status ($\delta$).
- Consider extensions to the RSF algorithm to allow for left-censored or interval-censored responses.
- Consider training an RSF with the original response ($Y$) data (Table 6, $(Y, \delta; X)$).

**Suggested Statistics for Quantifying Predictive Performance**

1. Compare response predictions ($\widehat{Y}$) to truth ($Y$).
   - Percentage of predicted responses within $k$ units (e.g., 1 year) of the corresponding true responses:
   $$\frac{\sum_{i=1}^{n} I\left(\left|Y_i - \widehat{Y}_i^{\text{algo}}\right| \leq k\right)}{n}, \text{ for algo } \in \{\text{RF}, \text{RSF}\}.$$
   - scaled sums of squared errors or mean squared prediction error ($MSPE$):
   $$MSPE^{\text{algo}} = \frac{SSE^{\text{algo}}}{n} = \frac{\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i^{\text{algo}}\right)^2}{n}, \text{ for algo } \in \{\text{RF}, \text{RSF}\}.$$
2. Compare predictions between algorithms (e.g., RF vs. RSF).
   - Percentages of predictions within $k$ units between algorithms:
   $$\frac{\sum_{i=1}^{n} I\left(\left|\widehat{Y}_i^{\text{RF}} - \widehat{Y}_i^{\text{RSF}}\right| \leq k\right)}{n}.$$
     - This quantity considers precision of predictions between algorithms.
     - *Are predictions from one algorithm similar to predictions from another algorithm?*
   - Scaled sum of squared differences between algorithms:
   $$\frac{SS(\text{RF}, \text{RSF})}{n} = \frac{\sum_{i=1}^{n}\left(\widehat{Y}_i^{\text{RF}} - \widehat{Y}_i^{\text{RSF}}\right)^2}{n}.$$
3. Compare predictions by censor status ($\delta$).
   Similarly to 1 and 2, additionally group results by predictor censor status to investigate the impact of censor status on algorithm performance. More specifically, use censor status as the possible confounder and check whether Simpson's paradox is observed. *Does the algorithm that performs better overall also perform better for each censor status?*

### 3.2.2 Results for Comparing Forests

Experiments were conducted using R version 3.4.4. R code for our experiments is available upon written request to the authors.

Select summary results for using train/test partitioned data are provided in Table 6. Select summary results evaluated for out-of-bag (OOB) data are provided in Table 7. (See [21] for a discussion of out-of-bag estimation versus train/test partitioning for random forests.) Full results from our experiments are saved as files .rds files and are available upon written request from the authors.

In our experiments, random forests were much slower than random survival forests. For each of our experimental runs, the RF training time was less than 0.2 seconds and the RSF training time was between four and ten seconds. Additional investigation can be conducted to determine how the ratio of the RF verses RSF runtime scales when additional experimental factors are considered (e.g., number of features, number of observations, proportion of observations that are censored).

In our experiments, the training time was faster for experimental runs that used test/train partitioned data. The authors expect that the increase in runtime for the non-partitioned data is most likely a result of the increased number of observations when the in-bag data is drawn from the full dataset, rather than restricting the in-bag data to be drawn from the training dataset only.

| experiment | algorithm | train | test | time | $MSPE$ | $MSPE_{\text{test},\delta=0}$ | $MSPE_{\text{test},\delta=1}$ |
|---|---|---|---|---|---|---|---|
| 1 | RF | $(Y;X)$ | $(Y;X)$ | 0.10 | 55.53 | | 55.53 |
| 2 | RF | $(Y^*;X)$ | $(Y;X)$ | 0.12 | 385.64 | | 385.64 |
| 3 | RF | $(Y^*;X,\delta)$ | $(Y;X,\delta)$ | 0.12 | 64.43 | | 64.43 |
| 4 | RSF | $(Y^*;X,\delta)$ | $(Y;X,\delta)$ | 4.43 | 56.99 | | 56.99 |
| 5 | RF | $(Y;X)$ | $(Y^*;X)$ | 0.10 | 1040.56 | 5213.08 | 58.79 |
| 6 | RF | $(Y^*;X)$ | $(Y^*;X)$ | 0.10 | 961.04 | 3424.70 | 381.35 |
| 7 | RF | $(Y^*;X,\delta)$ | $(Y^*;X,\delta)$ | 0.10 | 323.08 | 1410.75 | 67.16 |
| 8 | RSF | $(Y^*;X,\delta)$ | $(Y^*;X,\delta)$ | 4.40 | 1027.73 | 5145.65 | 58.81 |

Table 6: Select summary results for evaluating forest performance on testing dataset (when data was partitioned into test and training sets).

| experiment | algorithm | data | time | $MSPE_{\text{oob}}$ | $MSPE_{\text{oob},\delta=0}$ | $MSPE_{\text{oob},\delta=1}$ |
|---|---|---|---|---|---|---|
| 9 | RF | $(Y;X)$ | 0.17 | 71.97 | | 71.97 |
| 10 | RF | $(Y^*;X)$ | 0.19 | 1392.79 | 5258.94 | 426.25 |
| 11 | RF | $(Y^*;X,\delta)$ | 0.17 | 667.52 | 2992.12 | 86.37 |
| 12 | RSF | $(Y^*;X,\delta)$ | 8.90 | 1479.29 | 7098.91 | 74.38 |

Table 7: Select summary results for evaluating forest performance on out-of-bag data.

The mean squared prediction error (MSPE) for the response was comparable for RFs trained on uncensored data and RSFs trained on a mixture of censored and un-censored observations. When testing data included a mixture of censored and uncensored responses, the order (e.g., ranking smallest to largest) of the MSPEs aggregated by censoring differs by censor status. Overall, the algorithms that performed better for uncensored data performed worse for censored data.

As expected, the best performance (with respect to both runtime and accuracy) was observed when the random forest was trained and tested on uncensored responses. When censored responses were used in the training data, the addition of censor status as a training feature in the RF resulted in increased accuracy over the traditional RF. When training responses included both censored and uncensored responses but testing responses were all uncensored, the RSF implementation achieved better accuracy than all other RF implementations considered. But when tested on censored responses only, the RSF performed worse than RF implementations. In all cases, the results from the RF trained on uncensored responses (best-case scenario data) are closest to RSF results.

Our results indicate that the presence or absence of censored responses in the testing data is a relevant factor when identifying the method that optimizes the prediction accuracy (e.g., minimizes the MSPE). When censored responses are included in training data, the predictive ability of the trained RF may be impacted – a result of practical importance for applications, like obsolescence.

### 3.3 Challenges and Considerations

One of the main challenges of deriving data-driven predictions for obsolescence data is obtaining data. The number of features and the number of observations available in the data may be a limiting factor in determining appropriate methods. Additional challenges arise due to missing values, censored values, and/or errors in the data. Content differences among data sources is another practical characteristic that may limit the effectiveness of data aggregation across sources.

## 4 Concluding Remarks

In conclusion, our work explored connections among methods in the existing literature as well as pioneered consideration of additional methods, particularly random survival forests and Markov chains, for applications in proactive obsolescence analytics. For random forests specifically, our results indicate that the presence or absence of censored responses in the testing data is important in identifying the method that optimizes the prediction accuracy.

## Acknowledgements

- Fiscal Year 2018 funding for this project was awarded through the Navy Innovative Science and Engineering Program at the Naval Information Warfare Center Atlantic.
- We appreciate the vision, insight, and support provided by the Obsolescence Management Information System research team at the Naval Undersea Warfare Center Keyport Division: Mr. Dennis Summers, Dr. Jennifer Williams, Mr. Jordan Love, and Mr. Connor Bradley.

## References


[1] Defense Standardization Program Office, "SD-22 – Diminishing sources and material shortages: A guidebook of best practices for implementing a robust DMSMS management program," tech. rep., Fort Belvoir, Jan 2016.

[2] D. Swenson, "Obsolescence 101 within the HM&E Navy," *DMSMS Conference 2015*, Dec 2015.

[3] J. Gansler and W. Lucyshyn, "Commercial-off-the-shelf (COTS): Doing it right," Master's thesis, University of Maryland, Center for Public Policy and Private Enterprise, 2008.

[4] P. Singh, P. Sandborn, T. Geiser, and D. Lorenson, "Electronic part obsolescence driven product redesign planning," *International Journal of Advanced Manufacturing Systems*, vol. 7, no. 2, 2004. This article describes using the left tail of a normal distribution to estimate the "zone of obsolescence." Statistical results supporting decrease in lifecycle length over time is also provided.

[5] C. Jennings, D. Wu, and J. Terpenny, "Forecasting obsolescence risk and product life cycle with machine learning," *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, vol. 6, Sept 2016. This article demonstrates how varies machine learning algorithms (e.g., random forests and support vector machines) can be applied to predict obsolescence date and obsolescence status.

[6] D. Rosson, "Predicting part lifecycles utilizing machine learning," Master's thesis, University of Washington, 2016.

[7] J. Williams, D. Summers, D. Rosson, C. Bradley, K. DeReuiter, J. Love, and K. Stageberg, "A machine learning approach for part type identification and predicting obsolescence date." Working Paper, 2017.

[8] F. Rojo, R. Roy, E. Shehab, K. Cheruvu, and P. Mason, "A cost estimating framework for electronic, electrical and electromechanical (eee) components obsolescence within the use-oriented product-service systems contracts," *Proceedings of the Institute of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 226, pp. 154–166, January 2012. Estimating the cost of obsolescence.

[9] F. Geng, G. Dubos, and J. Saleh, "Spacecraft obsolescence: Modeling, value analysis, and implications for design and acquisition," *2016 IEEE Aerospace Conference*, pp. 1–13, 2016.

[10] T. Wendling, "Obsolescence risk and the systematic destruction of wealth," tech. rep., April 2012.

[11] Aerospace Industries Association, "Counterfeit parts: Increasing awareness and developing countermeasures," tech. rep., Arlington, March 2011.

[12] M. Bradley and R. Dawson, "An analysis of obsolescence risk in it systems," *Software Quality Journal*, vol. 7, pp. 123–130, 1998.

[13] R. Solomon, P. Sandborn, and M. Pecht, "Electronic part life cycle concepts and obsolescence forecasting," *IEEE Transactions on Components and Packaging Technologies*, pp. 707–717, Dec 2000. This article uses a normal-like distribution lifecycle curve to predict obsolescence using demand data.

[14] P. Sandborn, V. Prabhakar, and O. Ahmad, "Forecasting electronic part procurement lifetimes to enable the management of DMSMS obsolescence," *Microelectronics Reliability*, vol. 51, pp. 392–399, 2011.

[15] K. Arney, C. Mastrangelo, and D. Summers, "Utilization of reliability theory to predict part obsolescence," Dec 2016. [slides].

[16] M. Barkenhagen and M. Murphy, "The sunset supply base: Long term COTS supportability, implementing affordable methods and processes," Master's thesis, Naval Postgraduate School, 2003.

[17] S. Ross, *Chapter 4: Markov Chains*, pp. 183–274. Elsevier, 11 ed., 2014.

[18] J. Williams, "RE: Touching base." [email], 2018.

[19] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[20] H. Ishwaran, U. Kogalur, E. Blackstone, and M. Lauer *Annals fo Applied Statistics*, vol. 2, no. 3, pp. 841–860, 2008.

[21] L. Breiman, "Out-of-bag estimation," tech. rep., 1996.

[22] M. Delgado, E. Cernandas, and S. Barro, "Do we need hundreds of classifiers to solve real world classification problems?," *Journal of Machine Learning Research*, vol. 15, pp. 3133–3181, 2014.

[23] C. Jennings, "R code referenced in [5]," 2016. [R code].

[24] G. Rodriguez, *Chapter 7: Survival Models*. [lecture notes], 2007. This chapter rovides a resource for learning survival analysis and censoring methods.

# Appendix

## A Remarks

**Remarks about Literature Referenced**

[1] From [13]: In our opinion, it makes more sense to define the obsolescence stage as anything above 2.5 standard deviations above the mean (*What comes after* $\mu + 3.5\sigma$*?*), with the knowledge that the probability of being above 3.5 standard deviations above the mean is negligible for observations following the normal distribution (i.e., $P(X > \mu + 3.5\sigma)$ is small when $X \sim \text{Normal}(\mu, \sigma)$).

[2] [15]'s slides provide minimal detail about the dataset used to estimate Equation 11 ( [15] slide 14).

[3] [5]'s treatment of neural networks is trivial. Due to limitations of the datasets used in [5], the neural network effectively resulted in taking an average. The full potential of neural networks are not discussed or leveraged in [5].

[4] In [5], *lifecycle forecasting* refers to obsolescence date prediction, rather than lifecycle length prediction. Given the introduction date, the lifecycle length can be derived from the obsolescence date (i.e., $L = t_{\text{obsolescence}} - t_{\text{intro}}$). However, when the introduction date is not included in the dataset or is expected to be subject to missingness, uncertainty, or other errors, the lifecycle length is not directly determinable from obsolescence date. In fact, introduction date is often missing in the SiliconExpert dataset that we encountered.

[5] Comparison of characteristics such as interpretability and maintainability/flexibility is somewhat subjective being qualitative as opposed to quantitative characteristics such as accuracy. [5]'s conclusions agree with those reported in other sources (e.g., [22]). Though subjective, interpretability is important to practitioners, some of whom favor transparant and/or familiar techniques over less understood and/or less explainable (e.g., "black box") methods.

[6] Computational runtime for training the machine learning algorithms is not a major concern for our target application (per conversation with obsolescence SMEs). Based on Jennings's supporting R code [23], it appears that [5] uses "evaluation speed" to refer to the computational time required to training the machine learning algorithm.

[7] We re-arranged the quote into an iff statement for conciseness. See page 7 of [14] for the original version.

Table 8: Our remarks about details in the obsolescence literature.

## B Definitions

### B.1 Definitions for Markov Chains

A Markov chain is a stochastic process, denoted by $X$, that satisfies the **markov property** that can be summarized as follows: "Given the present, the future is independent of the past." Processes that satisfy the Markov property are characterized by a lack of dependence on historical data other than the current state. The exact length or information embedded in the "present" or current state may vary (e.g., the current state may be a concatenation of the three day precipitation history – YNY would represent that it rained yesterday and three days ago, but not today.), but at some point having a longer track record will add predictive value for processes that satisfy the markov property. A more technical definition is given in Equations 13 to 14.

A **discrete time Markov chain** (DTMC) is a Markov chain in which the transitions between states occur at discret stepping times. A DTMC may record movement times only (e.g., step 1, step 2, ..., step $N$) without accounting for the elapsed time between movements; or the current state of a process may be recorded at fixed times (e.g., the first day of each month), in which case "trivial" movements (e.g., moving from state $i$ to state $i$, which is actually not moving) may also be recorded when the current state of the process is the same for consecutive checkin times.

A **continuous time Markov chain** (CTMC) is a Markov chain in which the transitions are measured in continous time. The continuous temporal record realized in the CTMC provides more information than the DTMC. For example, the exact length of time elapsed between transitions can be derived from the CTMC but cannot be determined from the DTMC. In fact, DTMCs can be extracted from the CTMC (but not vice versa).

As a matter of notation, let $X_k$ denotes the state of a DTMC at step $k$, and let $X(t)$ denotes the state of a CTMC at time $t$. While the temporal record is discrete for DTMCs versus continous for CTMCs, the statespace is (often assumed to be) discrete or discretized in some fashion for both DTMCs and CTMCs.

Using this notation, the Markov property can be more formally defined as follows:

$$\text{For DTMCs, } P\left(X_{k+1} = j \middle| X_0 = i_0, X_1 = i_1, ...., X_k = i\right) = P\left(X_{k+1} = j \middle| X_k = i\right). \tag{13}$$

$$\text{For CTMCs, } P\left(X(t+s) = j \middle| X(0) = i_0, X(t_1) = i_1, ...., X(t_{k-1}) = i_{k-1}, X(t) = i\right) = P\left(X(t+s) = j \middle| X(t) = i\right), \tag{14}$$

where $t_0 < t_1 < t_2 < ... < t_{k-1} < t$ and $s > 0$.

The statespace, denoted by $S$, is the set of states that may be visited by the Markov chain. For example, a possible statespace for a Markov chain with states that represent the type of precipiation observed on the previous day is given by $S = \{\text{rain}, \text{snow}, \text{sleet}, \text{hail}, \text{none}\}$. The probability that a DTMC ($X$) transitions from state $i$ to state $j$ in consecutive steps is given by

$$p_{ij} = P\left(X_{k+1} = j \,|X_k = i\right),$$

and the probability that $X$ moves from $i$ to $j$ in $c$ steps is given by

$$p_{ij}^{(c)} = P\left(X_{k+c} = j \,|X_k = i\right).$$

$$\begin{aligned} \text{Then } p_{ij}^{(1)} &= p_{ij} \\ p_{ij}^{(2)} &= \sum_k p_{ik}p_{kj} \\ \vdots \quad & \quad \vdots \\ p_{ij}^{(c)} &= \sum_{j_1} p_{ij_1} \sum_{i_2,\, j_2} \dots \sum_{i_{c-1},\, j_{c-1}} \prod_{k=2}^{c-1} p_{i_k j_k} \sum_{i_c} p_{i_c j}. \end{aligned}$$

**Aside: Paths from $i$ to $j$ versus First visit from $i$ to $j$**

Note that $p_{ij}^{(c)}$ does not restrict attention to a path that stops as soon as it reaches $j$; that is, this probability counts paths that visit $j$ $k$ times for integer values of $k$ as long as the path starts at $i$ and ends at $j$. For example, consider a path that moves within a statespace of size $k$, starting at state $i$ on step 0 and ending at state $j$ on step 3.

| **step** | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| **general state** | $i$ | $k_1$ | $k_2$ | j |

| | **Examples** | | | | **Conditions** |
|---|---|---|---|---|---|
| Example 1 | $i$ | $j$ | $i$ | $j$ | $p_{i,j},\, p_{j,i} > 0.$ |
| Example 2 | $i$ | $i$ | $i$ | $j$ | $p_{i,i},\, p_{i,j} > 0.$ |

The number of paths meeting these criteria is bounded above by $k^2$ (i.e., $k$ choices for each of states $k_1$ and $k_2$). The number of allowable paths may be further decreased based on the following considerations:

- Self-loops may or may not be counted as steps (e.g., depending on the definition of "transition") (Disallowing self-loops pulls the upper bound down to $(k-1)(k-2)$ since $k_1$ cannot be $i$ and $k_2$ can be neither $k_1$ nor $j$.) In this report, we allow for both trivial (i.e., $X_k = X_{k+1}$) and nontrivial (i.e., $X_k \neq X_{k+1}$) in our use of the term "step"; that is, we count a self-loop (e.g., "moving" from state $i$ to state $i$) as a step even though no change has occurred. We choose this definition because of the intended applications (Section **??**). Other conventions may restrict their definitions of "step" strictly to include only non-trivial transitions/jumps of movements (e.g., $X_k \neq X_{k+1}$).
- Only count paths for which each step has a nonzero probability; that is consecutive states in the path "communicate"/are connected by edges with nonzero weights. (e.g., the number of paths is given by $\left|\left\{(k_1, k_2) \,:\, p_{i,k_1},\, p_{k_1,k_2},\, p_{k_2,j} > 0\right\}\right|$).

The statespace transition diagram is a graph, $G = (V = S, E)$, where the vertex set ($V$) is defined by the statespace and the edge set ($E$) contains all paths corresponding to steps with nonzero probabilities (i.e., $E = \{e_{ij} \,:\, p_{ij} > 0,\ \text{where } i, j \in S\}$).

## B.2 Deriving an estimated response for Random Survival Forests

As used in survival analysis or reliability theory, the survival function (also called the "survivor function"), $S_T$, represents the probability that the lifetime, $T$, exceeds a given time point, $t$:

$$S_T(t) = 1 - F_T(t) = P(T \geq t),$$

where $F_T$ denotes the cumulative density function of the random variable $T$. In other words, $S_T(t)$ describes the probability of surviving past time $t$, whereas $F_T(t)$ denotes the probability of failing/dying/or of an event of interest occuring by time $t$, where the random variable $T$ represents the lifetime of the unit under consideration.

In general, the expected value of continuous random variable $X$ is given by the following well-known formula:

$$E(X) = \int_{\mathcal{X}} x f_X(x) dx,$$

where $f_X$ denotes the probability density function of $X$ and $\mathcal{X}$ denotes the support set of $X$.
For a proper random variable $X$ (i.e.,$S_X(\infty) = 0$) with nonnegative support set (e.g., $\mathcal{X} \subseteq (0, \infty)$), the following integral equals zero:

$$\int_0^\infty \left[x f_X(x) - S_X(x)\right] dx = 0.$$

Thus (as claimed in [24], pp.3-4), the expected value is equal to the area under the survival curve:

$$E(X) = \int_0^\infty S_X(x) dx. \tag{15}$$

Given a list of points $(x, S_X(x))$, approximation methods for numerical integration, such as Simpson's rule and the trapezoid rule, can be used to estimate the area under the survival curve. This estimated area under the survival curve provides an estimate for the expected value of $X$, based on the relationship in Equation 15.

# C Additional Figures

Similar to Figure 2 (Section 2.1), Figure 7 displays a possible future realization of obsolescence data, including a depiction of the common development cutoff

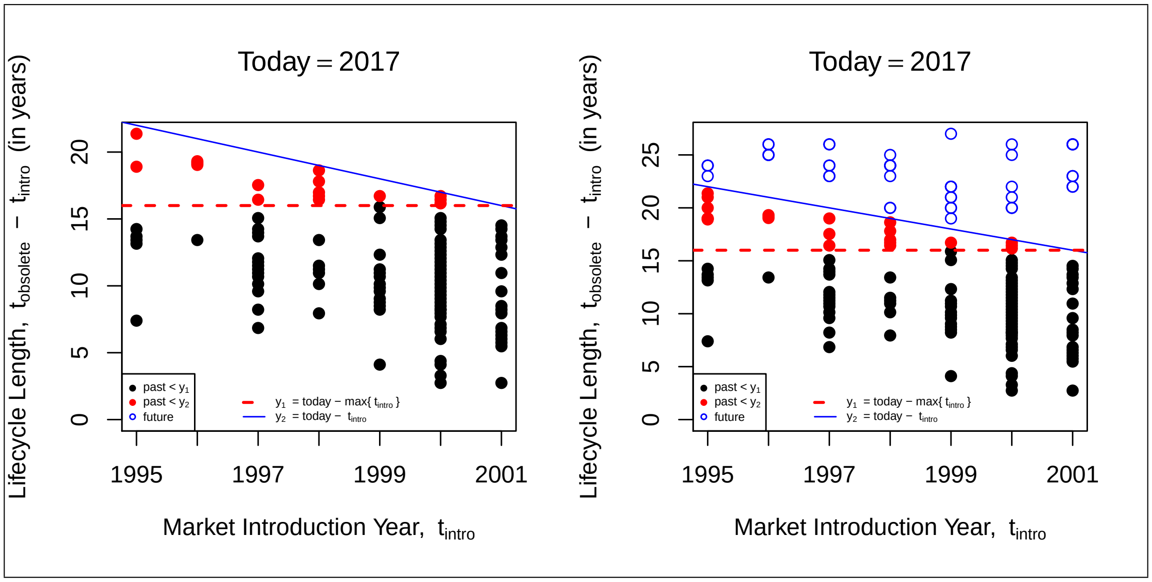



Figure 7: Illustration of the Impact of Partial Development on the Perceived Relationship Between Introduction Date and Lifecycle Length.
(Left): Data as it appears when it is only partially developed (at present);
(Right): a possible realization for the fully-developed data (future) in which the apparent strength of linearity between introduction date and lifecycle length is minimal, compared to the naive/initial (non-conditional) perceived relationship gathered from the partially-developed data.
The solid blue line indicates the maximum lifecycle length that can by observed up to the time of analysis (i.e., $L_{\max} = t_{\text{obsolescence}} - t_{\text{analysis}}$).
The dotted red line indicates the minimum value among the maximum observable lifecycle lengths (i.e., min $L_{\max} = \min_{t_{\text{intro}}} (t_{\text{analysis}} - t_{\text{intro}}) = t_{\text{analysis}} - \max\{t_{\text{intro}}\}$) – a line of "shared common development."

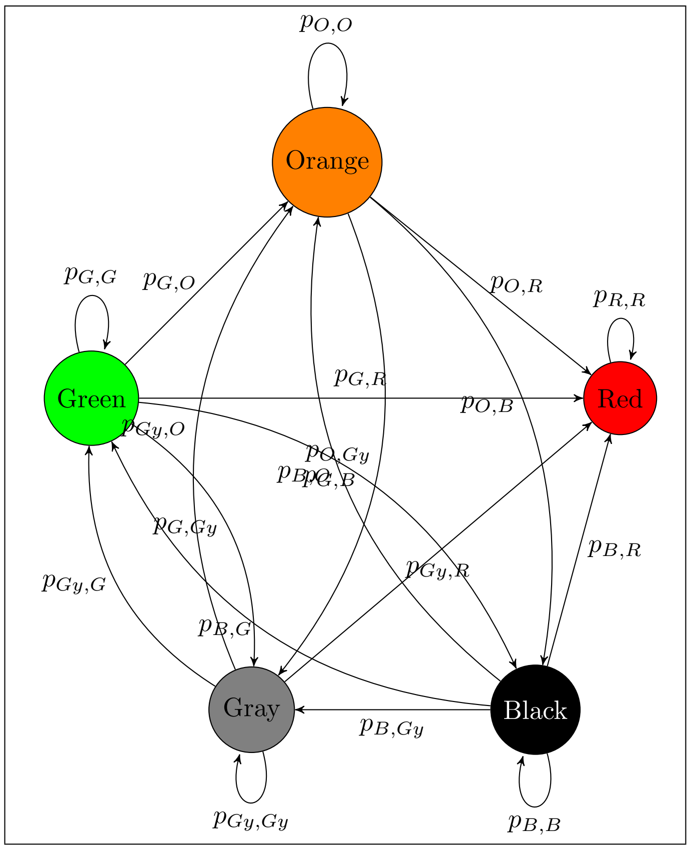



Figure 8: An Example of a Transition Diagram for a Discrete Time Markov Chain (DTMC) with Five States. In addition to the three states for survey responsiveness (represented by red, orange, green; Fig. 4) the expanded state space, $S^*$, includes states that account for possible non-responsiveness of vendors to surveys (represented by gray and black): $S^* = \{\text{green}, \text{orange}, \text{red}, \text{gray}, \text{black}\}$. Edges in this diagram are weighted by transition probabilities; that is, the weight of the edge from node $i$ to node $j$ is denoted by $p_{i,j}$ and represents the probability that the DTMC will transition from state $i$ to state $j$ in the next step (i.e., $p_{i,j} = P\left(X_{t+1} = j \middle| X_t = i\right)$).