CPU Product Line Lifecycles:
Econometric Duration Analysis using Parametric and Non-Parametric Estimators

by

Mischa Fisher
Bachelor of Arts, University of Victoria, 2007

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

MASTER OF ARTS

in the Department of Economics

Supervisory Committee

CPU Product Line Lifecycles:
Econometric Duration Analysis using Parametric and Non-Parametric Estimators

by

Mischa Fisher
BA, University of Victoria, 2007

Supervisory Committee

Dr. Kenneth Stewart, Supervisor
Department of Economics

Dr. David Giles, Departmental Member
Department of Economics

Abstract

This thesis provides a comprehensive history of the statistical background and uses of survival analysis, and then applies econometric duration analysis to examine the lifecycles of product lines within the microprocessor industry. Using data from Stanford University's CPUDB, covering Intel and AMD processors introduced between 1971 and 2014, the duration analysis uses both parametric and nonparametric estimators to construct survival and hazard functions for estimated product line lifetimes within microprocessor product families. The well-known and widely applied non-parametric Kaplan-Meier estimator is applied on both the entire sample as a whole, and segmented estimate that considers product line lifecycles of Intel and AMD separately, with median survival time of 456 days. The parametric duration analysis uses both the semi-parametric Cox proportional hazard model, and the fully parametric accelerated failure time model across the Weibull, Exponential and Log-Logistic distributions, which find modest association between higher clock speed and transistor count on diminishing expected time in the marketplace for microprocessors, while the number of cores and other attributes have no predictive power over expected survival times. It is expected that the transistor count and clock speed of a given processor's negative effect on expected duration, likely captures the co-trending of growth in transistor count with a larger marketplace and broader product categories.

Table of Contents

List of Tables

List of Figures

Acknowledgments

I must acknowledge the superb faculty at the University of Victoria's Department of Economics, as well as my thesis guides Dr. David Giles and Dr. Kenneth Stewart, and the outside reviewer Dr. Mary Lesperance. The entire academic team patiently put up with my bumbling attempts at matrix algebra, calculus, and statistics and I will be forever grateful for their guidance, humor, and wisdom. The department is home to some of the best professorial staff anywhere in academia. It is also necessary to acknowledge the supreme patience of Dr. Giles, who has steadfastly put up with my questions and delays. He's a true gentleman and scholar and while his retirement is well-earned for him, it's a big loss to future students who will be unable to share in his deep knowledge of statistics and econometrics.

1 Introduction

Duration analysis has been widely used in economics to model labor market characteristics, probabilities of firm survival upon market entry, and many other pressing areas of interest; its first widely cited use was modeling unemployment durations and probability of returning to work. Before it was used in economics, the statistical principles were found to have utility in both medicine and engineering to predict conditional survival rates and probability of critical failures, respectively.

This thesis uses econometric duration analysis to examine the lifecycle of product lines within the microprocessor industry. Using data from Stanford University's CPUDB, covering processors introduced between 1971 to 2014, the duration analysis uses both parametric and nonparametric estimators to construct survival and hazard functions for microprocessor product lines.

Microprocessors are an interesting area of research for four primary reasons that make the market unique. The first, of course, is the role they have played in transforming society, the economy, and even the nature of the human experience. One would be hard pressed to identify a similar single technology that has had such far-reaching consequences, even with the huge impacts devices like the telegram, railway, and air conditioner have had on our world; microprocessors touch every technology in every market.

The second, is that microprocessors are such a fast changing technology; advancement in materials science, combustion engines, pharmaceuticals, aerospace, and agriculture have not come close to the rapid rate of iterative improvement found in microprocessor

design and manufacturing. Materials science is an interesting corollary, where an analogous improvement would mean doubling material strength while halving weight every 18 months... an inconceivable level of improvement.

A third attribute that makes the marketplace for microprocessors so interesting is that this rapid changing, far-reaching technology has traditionally been dominated by a tiny handful of companies (the focus in this thesis being AMD and Intel who control 95% of the market).[1] Economic industrial organization has its own field of analysis and findings regarding just the oligopolistic characteristics of the marketplace (see Goettler and Gordon for example).[2]

And the fourth reason why this is such an interesting area to undertake a duration analysis is no published results of any such analysis currently exist. While there are certainly market analyses and forecasts available, estimating the duration of product line lifecycles using duration analysis is a novel approach to understanding the dynamics in the marketplace for CPUs.

Specifically, duration analysis for central processing unit (CPU) product lifecycle durations will answer the question of what is the expected product lifetime on the market, with what likelihood will a product be replaced by successor technology given its duration to date, and what characteristics predict shorter or longer product lifetimes. Using both parametric and nonparametric estimation techniques, it is found that neither clock speed, nor number of cores have any predicted effect on the conditional duration of CPU product-lines, which a mean survival time of 675.6 days. However, it is found that

[1] Goettler and Gordon 2014, 2.

[2] Goettler and Gordon 2011, 1141.

transistor count predicts a shorter lifespan for the product line, likely capturing the increasing intensity in the marketplace as time progresses.

2 Literature Review

The general statistical techniques that are used in this thesis have a long and storied history. Known by a variety of names in a variety of fields *survival analysis* in statistics and medicine, *reliability theory* in engineering, *event history analysis* in sociology, ***duration analysis*** (as it's known in economics) has its application roots in statistics and medicine. Its conceptual roots first appeared in the work of Dutch astronomer Christiaan Huygens in mapping out life expectancy data sent to him by his brother Lodewijk in 1669, the output of which closely resembles the output of a modern Kaplan-Meier estimator.[3] The more mathematically rigorous techniques, however, are a more modern development.

2.1 Before Economics

As the Huygens anecdote illustrates, the earliest attempted use of survival principles was in studying human mortality. While life tables constructed from large samples of population data were the main method for most of the last few centuries, they had shortcomings that required the onset of *"small sample theory and non-parametric methods*".[4]

In 1958, the statisticians Paul Meier and Edward Kaplan published their extremely influential paper *Nonparametric Estimation from Incomplete Observations*, the crux of which acknowledged that it is typically impossible to have complete measurements of all

[3] Wainer and Velleman 2000, 309-311.

[4] Kimball 1960, 505.

the members of a random sample. Their paper goes on to address this challenge by formulating *nonparametric* estimators, by which they mean, *"...the class of admissible distributions from which the best-fitting one is to be chosen is the class of all distributions."*[5] They then derive their product-limit estimator of the survival function for some set of observations, now widely known as the Kaplan-Meier estimator, with the familiar form of:

$$\widehat{S}_t = \prod_{t_i \leq t} \left[1 - \frac{d_i}{n_i}\right]$$

where S is the estimated probability of survival at time t, d_i is the change of states at time i, and n_i is remaining number of un state-changed observations. Kaplan and Meier also offer derivations of estimators for the mean and variance of the product-limit statistic. In general terms, the survival function being estimated is the fraction of observations still in an unchanged state, while the hazard function is the increasing slope that at any given time, an observation will change state during the next interval.

Writing 14 years later, Nelson offers a *"theory for the hazard plotting method*", noting the utility of hazard plots as providing "*estimates of the proportions of units failing by a given age, percentiles of the distribution, the behavior of the failure rate of the units as a function of their age, and conditional failure probabilities for units of any age."* [6]

One of the earliest widespread applied uses of the mathematical and statistical principles of duration analysis outside of mortality studies came from engineering and industrial production. Most prolifically, Romanian Engineer Joseph Jurin's *Quality*

[5] Kaplan and Meier 1958, 458-459.

[6] Nelson 1972, 945-951.

Control Handbook, which has been in continuous publication across 6 editions and nearly 60 years from 1951 to 2010. In the book, Jurin notes that examples of potential use of duration analysis in industrial production are “quantifying reliability… predicting warranty costs… deciding on the need for recall… [and/or] obtaining field failure information to help improve product design.″ [7]. Reliability analysis is so-called because the hazard function's "close relationship with failure processes, and maintenance strategies, reliability engineers often model time to failure in terms of it." [8] Of particular note, Jurin builds on why reliability engineers use this form of analysis by noting that "The traditional parameters of a statistical model (mean, standard deviation) are often not the primary interest in reliability studies. Instead, design engineers, reliability engineers, managers, and customers are interested in specific measures of product reliability or particular characteristics of the failure-time distribution." [9]

The 1970s saw further development and refinement of the general statistical tools as they applied to medical and engineering questions. *Regression Models and Life-Tables* (Cox 1972) introduced a number of new methods, most prominently developing a conditional likelihood function in a multivariate regression framework. Norman Breslow built on this with several papers, including a 1974 comparison of Cox's framework to older methods in studying childhood leukemia (Breslow 1974) and a 1975 paper, *Analysis of Survival Data under the Proportional Hazards Model*, that contributes to the pre-econometric methods by providing methodology to address censored survival data

[7] Jurin 1998, 48.2.

[88] ibid.

[9] Jurin 1998, 48.5-48.6.

where the variables of interest act multiplicatively on the hazard function[10]. This created the opportunity to understand additional quantitative tools for a number of uses, including time dependent covariates. Several cases of applied duration analysis were also published during this time frame, such as Prentice and Gloeckler's research on breast cancer survival, which used maximum likelihood estimation building off of Cox's model to understand the disparity of breast cancer survival rates between black and white patients.[11]

Another heavily cited text is Cox and Oakes (1984), which notes that "the time origin should be precisely defined for each individual" and that "often the scale for measuring time is clock time, although other possibilities certainly arise." [12] Cox and Oakes also provide meaningful insight on the use of various distributions. Noting that while "the exponential distribution was widely used in early work on reliability... [t]hat the distribution has only one adjustable parameter often means... that methods based on it are rather sensitive to even modest departures" and that more modern methods have had the goal of "making less stringent assumptions about distributional form."[13]

2.2 Econometric Duration Analysis

The first widely used model of duration analysis in economics[14] is Lancaster's (1979) *Econometric Methods for the Duration of Unemployment*, which was necessitated by the

[10] Breslow 1975, 49.

[11] Prentice and Gloeckler 1978.

[12] Cox and Oakes 1984, 3.

[13] Cox and Oakes 1984, 18.

[14] Burton 2003, 5.

absence of statistically thorough measurement of conditional unemployment spells.[15] Lancaster used real data of unemployed people in Britain to construct a more robust econometric method of study, notably, by breaking down the constituent parts of the estimator. Lancaster notes "Specification of θ has two components of which the first is functional form for the dependence on time - the time profile of the failure rate - and the second is the description of the way in which θ shifts, at a given point of time, between individuals pursuing optimum policies..."[16] Lancaster further outlines how "the implied duration distribution is known as the Weibull family." [17]

In the 1980s, duration analysis matured in economics and econometrics with a series of influential papers; starting with several that came from University of Chicago economist James Heckman. *Econometric Duration Analysis* (1984) targeted the specific nature of continuous time models, noting their importance because traditional econometric models lacked a time component, and in the event they did contain one, their discrete nature potentially failed to line up with the discrete timing of the observation intervals.[18] Heckman formulated a dynamic model of labor force participation, a one state model of search unemployment, and a dynamic model of a McFadden choice problem. The authors also raised some important criticisms of the older Kaplan-Meier method, principally, that it "is unlikely to prove successful in econometrics because (a) the available samples are small, especially after cross-classification by regressor variables, and (b) empirical modesty leads most analysts to admit that some determinants of any duration decision

[15] Lancaster 1979, 939.

[16] Lancaster 1979, 945.

[17] Lancaster 1979, 947.

[18] Heckman and Singer 1984, 63.

may be omitted from the data sets at their disposal. Failure to control for unobserved components leads to a well-known bias toward negative duration dependence."[19] Summarizing the paper's many contributions, it established a suite of non-parametric procedures to assess the structural hazard and mixing distributions of models (how complex must a model be and how best to further define it), because the existing models of biostatistics and reliability theory failed to consider the time and behavioral characteristics of economics. Another paper coauthored by Heckman, *New Methods for Analyzing Structural Models of Labor Force Dynamics* (1982) raised issues with the under-identification of traditional econometric models and presented an "appropriate" asymptotic treatment.

This decade of active research advancement was capped off with several summary textbooks capturing the developments of the 1980s, such as, *Econometric Analysis of Transition Data* (Lancaster 1990). Following the wave of influential papers in the 1970s and 1980s, econometric duration analysis has been used widely in economics outside of just labor market studies regarding employment. Examples follow a similar method of parametric estimation of the cumulative distribution function (CDF), the hazard function, and the survival function using the exponential, Weibull, or log-normal distributions; a non-parametric Kaplan-Meier estimator approach; or, some combination of the two. Several examples of this follow.

Gilbert (1992) used duration analysis as a roundabout way to get at improving automobile sales forecasts by providing a new baseline hazard function to develop

[19] Heckman and Singer 1984, 77.

predicted ownership spells[20]. She qualified the need for doing this by noting the shortcomings of traditional discrete panel data as failing to consider "the probability of adjusting from car *j* to car *k* may depend on how long car *j* has been owned," [21] and also that "the discrete time interval observations of the household's fleet may bear no relationship to the time frame the household uses in making its decisions." [22] She then further reiterated the common thread across most duration analysis, being that the "estimated relationship between the probability of buying car *k* and the attributes of car *k* and the characteristics of the household [are] not invariant to the length of the time interval that separates observations of a household's fleet", and the added issue of left and right censoring of the data.[23] Methodologically, Gilbert constructed the total hazard as the sum of three independent hazard functions, the hazard of vehicle replacement with a new vehicle, with a used vehicle, or the hazard of vehicle disposal without replacement. She used the Weibull distribution to model these hazards because of its ability to increase, decrease or stay constant with time.

Burton was one of the early users of duration analysis in studying technological adoption in agriculture; studying the UK (Burton and Rigby 2003) and Ethiopia (Burton and Dadi 2004). Burton offered several succinct summaries of the benefits of duration analysis for those purposes, noting: "it deals with both cross-section and time-series data... adoption and diffusion are studied together" because duration models allow for determinants to change across observations and within observations over time. This

[20] Gilbert 1992, 113.

[21] Gilbert 1992, 97.

[22] Gilbert 1992, 98.

[23] Gilbert 1992, 99.

allows one, for example, to "estimate the probability that a farmer with given attributes will adopt organic practices in a particular year, given that adoption had not occurred by that time" and, as a result, "rather than focusing on the length of a spell, one can consider the probability of its end." [24] It has also been used to model exchange rate regimes, [25] duration of popular songs,[26] housing cycles in the OECD,[27] and survival of businesses in new or existing markets such as venture capital in France.[28]

2.3 Duration Analysis and CPUs

Despite duration analysis' now widespread use, the literature using duration analysis to model microprocessor product-line duration specifically is nearly non-existent. The closest research available was performed by Dr. Barry Bayus at UNC Chapel Hill in his 1998 paper *An Analysis of Product Lifetimes in a Technologically Dynamic Industry*. Bayus, in contrast to the analysis in this thesis, does not look at CPU lifetimes, but rather product lifetimes of personal computers; which, while built with constantly changing series of CPU technology, are their own distinct product category. Bayus notes explicitly the problem inherit in this thesis that he chooses to avoid, namely, that "directly analyzing product technology or product model lifetimes will not really be feasible because these lifecycles are relatively long." [29] Instead, Bayus, using sales estimates of

[24] Burton and Rigby 2003, 3-8.

[25] Giles and Shih, 2009.

[26] Giles, 2011.

[27] Bracke, 2011.

[28] Pommet, 2012.

[29] Bayus 1998, 769.

personal computers, calculates the duration as the year after product introduction in which sales are zero. [30]

Other research, while not in the nature of a duration analysis, is still relevant precursor knowledge in examining microprocessors. David Audretsch in several widely cited papers looked at manufacturing in the US based on the now outdated Standard Industrial Codes (SIC codes, since replaced by NAICS codes) to determine entry and survival using both traditional logit model (Audretsch 1995) and the Cox model for the hazard function approach (Audretsch and Mahmood 1995). The takeaways of this approach are that "the same industry structure characteristics that have been posited to pose a barrier to entry can be interpreted to pose... a barrier to survival." [31] This means that large capital costs and economies of scale - both characteristic of the microprocessor industry - are not necessarily limited to barriers to entry in industrial organization.

Since Intel is the dominant player in the CPU industry, and has the majority of observations in the CPUDB, it is also important to understand their internal behavior. Gawer and Henderson (2007) looked at Intel, specifically how the company approaches complimentary 'connector' markets, "which include chipsets and motherboards."[32] Their paper was not a rigorous piece of econometrics research, however, it did reinforce the complexity of the CPU market; where, for example, new product releases are tied to other factors. Noting a senior intel executive: "We got into the chipset business in a major way to accelerate platform transitions. To unleash the power of the Pentium, we had to

[30] Bayus 1998, 768.

[31] Audretsch 1995, 448.

[32] Gower and Henderson 2007, 11.

introduce the new PCI bus."[33] Further, their approach to sales is less conventional than basic Econ 101 microeconomics theory would suggest, as they attempted to "sell more microprocessors by partnering, not competing" with non-Intel entities.[34]

[33] Gower and Henderson 2007, 13.

[34] Gower and Henderson 2007, 24.

3 Data

The underlying subject matter of the data in question, microprocessors, can be convoluted; so it is important to establish first what it is we specifically mean by 'microprocessors'.

3.1 A (Brief) History of Processing Hardware

Numerous terms are used somewhat interchangeably: processor, microprocessor, CPU, semiconductor, integrated circuit, transistor, and microchip. Within those are sub-concepts such as cores, chips, sockets, processes, threads, etc; it's important to understand which of these is actually being studied.

Very early computers, such as the abacus and antikythera mechanism, used mechanical principles to perform computational tasks. However; what we really think of when we talk about early computers are the large, vacuum chamber machines of the mid 20th century, such as ENIAC, EDVAC, ORDVAC, and UNIVAC. These vacuum chamber, or vacuum tube, computers exploited the principles of a thermionic emission to control the flow of electrons through a gaseous medium within the device. In contrast, a semiconductor uses a solid-state material, most famously silicon, as its medium of electronic conduction. This is why processors and semiconductors are occasionally used as synonyms.

Transistors, which are built out of semiconductor material, are in some respect the basic building block of all electronic equipment. A transistor is in effect a control switch; when transistors and other components are integrated on a single chip they become an integrated circuit; a microprocessor is a type of integrated circuit. CPU itself is a more

general term, capturing where computation takes places; in the early days of computing, a CPU was not limited to a single microchip, however, for the last 40 years, the terms microprocessor and CPU have been used interchangeably.

In 1971, Intel released the 4004, the first commercially available microprocessor; since then, processor speed has increased as more and more transistors have been fit into individual CPUs resulting in the exponential improvement of computing power. This is a trend first predicted by Gordon Moore, then head of research at Fairchild Semiconductor and later chairman of Intel; in 1965 he noted that manufacturers were successfully doubling the components on integrated circuits at regular intervals, and that they would continue to do so for 'at least10 years'.[35] In the last decade or so, multi-core processors rather than transistor count and processor clock speed, have been the predominant characteristic of processor innovation.

The CPU as an integrated circuit / microprocessor, is the data captured by CPUDB and the subject of this analysis.

3.2 CPUDB

CPUDB (Central Processing Unit Data Base) is an "open and extensible database" collected and maintained by Stanford University's VLSI (Very Large-Scale Integration) research group.

The first chronological record in the database is the Intel 4004 processor; released in 1971 with a clock speed of 740 KHz running off an architecture of 2,300 transistors, it could deliver 60,000 instructions per second. The full dataset contains information on

[35] Schaller 1997, 54.

2,104 processors across a range of characteristics. Figure 1 (“Log of transistor count and time”) depicts the 643 observations for which there is date, core, transistor, and manufacturer information by plotting all four characteristics in a single space; it also illustrates the trend eponymous with Gordon Moore's previously mentioned prediction, and now dubbed ‘Moore’s law’, showing the exponential growth in transistor density approximately every 18 months.[36] Figure 2 provides a simple visual summary of Intel’s dominance within the marketplace, at least in terms of product line diversity, by showing distinct processor product lines by each manufacturer.

[36] Schaller 1997, 53.

Figure 1. Log of Transistor Count & Time

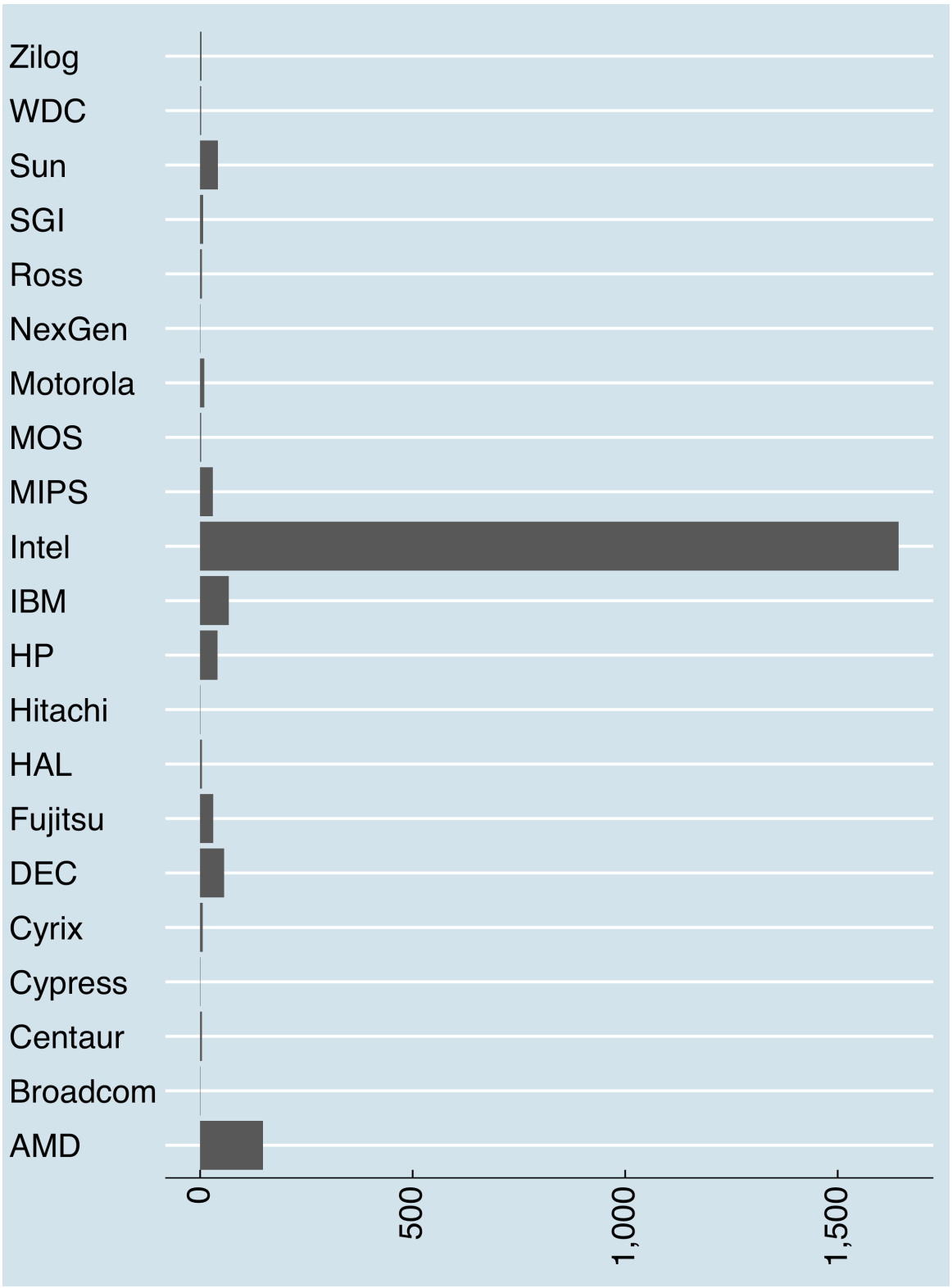

Figure 2. CPU Products by Manufacturer

3.3 Summary Statistics

Summary statistics of the major variables in the dataset are presented in Table 1.

Most importantly, the total date range covers a period of 43 years, from the introduction of Intel's 4004 to the modern Intel Core i-series. The most missing entry from the dataset is the maximum clock speed of the processor, for which there are only 502 observations. However, the most captured observation is the clock speed in megahertz and the number of threads per core (both more frequent even than date, which is the critical observation) so the missing maximum clock speeds are not a major concern. Transistor count, the subject of Moore's law, is the most interesting observation, as the min and max are so far apart and the mean suggests a heavily rightward skewed distribution, which we would expect from a series that grows exponentially.

VARIABLE	COUNT	MEAN	MEDIAN	MIN	MAX
Date	1,388	2008-02-21	2010-01-07	1971-11-01	2014-01-10
Clock (in MHz)	2,073	1,976	2,100	0.108	21,300
Max Clock	502	3,197	3,300	1,333	4,400
Threads Per Core	2,098	1.4	1	1	8
Number of Cores	2,100	2.8	2,000	1	61
Thermal Design Power in Watts (TDP)	1,769	61.4	55	0.5	300
Bus Width	547	56.4	64	4	64
Transistors	1,096	315,600,000	188	2,300	1,900,000,000
Die Size	1,123	175.7	135	1	1,215
Voltage Low (VDD Low)	1,047	1.5	1.075	.3	15
Voltage High (VDD High)	1,161	1.4	1.388	.9	15

Table 1. Summary Statistics

4 Methodology

Duration analysis is best understood in contrast to traditional Ordinary Least Squares (OLS) regression. Where OLS would find itself misspecified and producing distorted results given censored data, duration analysis allows for both. Similarly, time dependent covariates are a signature element of duration analysis, as the statistical underpinnings can be configured to allow the independent variables to change with time. A t-test or linear regression would ignore the censoring issue.

Given its expansion on basic, more widely known versions of regression analysis, a few important concepts are first best understood:

4.1 Duration Analysis Functions

The textbook version of parametric duration analysis requires understanding four distinct but related functions, each built on the assumption of ***T***, a non-negative random variable representing time of the duration ending, and ***t***, a given realization of that time.

1. CDF, the Cumulative Distribution Function of ***T***, known as ***F(t)***:

$$F(t) = \Pr(T \leq t)$$

Cumulative distribution functions are widely known, and they're an intuitive starting point for understanding duration analysis as it can be thought of as the probability that a randomly selected observed entity from the study population, will change states before time ***t***; or, similarly, the total proportion from that population that will change states before ***t***.

2. PDF, the Probability Density Function, is also widely understood as the derivative of a function's CDF. In the case of duration analysis, the PDF ***f(t)*** can be interpreted as the relative frequency of state-change times, given the time function. (Alternatively, one could view it as a histogram of state-change occurrences).

$$f(t) = \frac{\partial F(T)}{\partial t}$$

3. The Survival Function, is simply the complement of the CDF, so while the CDF captures the probability that something will change states before time ***t***, the Survival function captures the probability that it will not.

$$S(t) = \Pr(T > t)$$

4. Finally, the hazard function, which is the ratio of the PDF and the Survival Function, captures the propensity to change state in the next interval, given the absence of change up to that particular time ***t***.

$$\lambda(t) = \frac{f(t)}{1 - F(t)}$$

Each of these functions is shown in Table 2:

Function	CDF	PDF	Survival Function	Hazard Function
Notation	F(t)	f(t)	S(t)	λ (t)
Relationship to T	$F(t) = \Pr(T \leq t)$	$f(t) = \frac{\partial F(t)}{\partial t}$	$S(t) = \Pr(T > t)$	$\lambda(t) = \frac{f(t)}{1-F(t)}$

Table 2. Relationship of Duration Functions

4.2 Common Distributions

The normal distribution is widely known in and out of statistics. However, it is seldom useful for duration analysis because it captures values below zero, which cannot work in a time duration capacity.[37]

The three most common distributions that are used in duration analysis are the exponential, Weibull, and variations of either the log-normal or log-logistic distributions.[38] However, other distributions are also used, such as the gamma, Gompertz-Makeham, compound exponential, orthogonal polynomial, generalized F, inverse Gaussian, translation, scale family, and proportional family.[39]

Figure 3 compares the exponential, Weibull, and log-normal distributions. The exponential distribution's rate parameter is set to 2, the Weibull distribution's shape parameter is set to 2 and its rate parameter is set to 1, and the log-logistic distribution's location parameter is set to 0 and its scale parameter is set to 2. The figure provides an illustration of the various distributional forms survival trends can assume; it is worth noting that the exponential distribution is a special case of the Weibull distribution if the shape parameter is set to 1 (if the Weibull and exponential distributions' shape parameters were set to 1 on Figure 3, the two lines would overlap perfectly).

---

[37] Jurin 1999, 48.

[38] Hahn and Shapiro 1968, 133-134.

[39] Cox and Oakes 1984, 17.

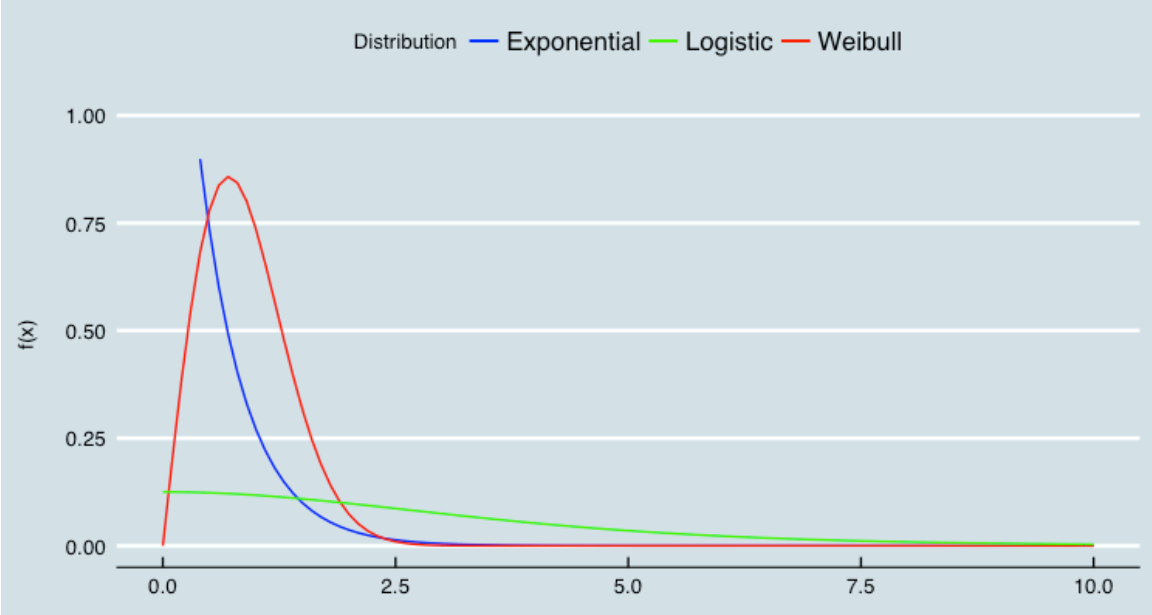

Figure 3. Common Survival Analysis Distributions

4.3 Estimators

The Kaplan-Meier estimator is a straightforward, simple, non-parametric estimator of the survival function. In the notation below, it is assumed that t_i are ordered, d_i is the number of product lines observed ending, and Y_i is the number of product lines still active. Then, the Kaplan-Meier estimator of the survival function is:

$$\hat{S}(t) = \begin{cases} 1 & \text{if } t < t_1 \\ \prod_{t_i \leq t} \left[1 - \frac{d_i}{Y_i}\right] & \text{if } t_1 \leq t \end{cases}$$

In addition to estimating the survival function, the mean survival time is a key statistic of interest and can be obtained as:

$$\hat{\mu} = \int_0^\infty \hat{S}(t)dt$$

In contrast to the Kaplan-Meier estimator, the Cox proportional hazards model fits the duration data, as well as its covariates, to an estimated hazard function, where z is a

vector of observations on the covariates, and β is the corresponding vector of coefficients; in this analysis the covariates are the technical attributes (transistors, clock speed, hardware cores, die size) associated with a given CPU product-line, and h_0 is the baseline hazard (the hazard if z is zero). The baseline hazard $\boldsymbol{h_0(t)}$ here is unspecified, which increases the utility of the Cox model and reveals why it's known as a semi-parametric estimator:

$$h(t|z) = h_0(t) \exp\left\{\beta' z\right\}$$

In addition to the non-parametric and semi-parametric estimators above, duration analysis can also include accelerated failure time models that start with a baseline mean hazard, μ, and W, a term that captures the distribution of the error term (typically the exponential, Weibull, log-normal, or log-logistic). Written in a log-linear format with failure time X on the left side below, is the accelerated failure time model:

$$\log X = \mu - \theta' Z + \sigma W$$

4.4 Reshaping Data

The date data captured in the CPU DB are limited to a single introduction date, as visualized in Figure 4, since a microprocessor product line technically never ceases to exist; this creates a sizable challenge in measuring the end of a product line. An additional challenge is that sales figures are typically unavailable for most chipset lines, so the method used by others of defining some arbitrary low-sales figure date is likewise not an option. Given these limitations, a separate formulation is required.

Duration existence is instead defined as the duration for which the same product line is being updated within the same processor family. For example, if ***n*** CPUs exist in microprocessor family ***x***, as seen in Figure 6, then the duration of CPU family ***x*** is defined as the difference between the date of introduction of the first CPU in that family and the date of introduction of CPU ***n***. Microprocessor families that were still being produced at the end of 2014, or for those families with only a single processor in them, are right-censored in the dataset at 2014. The distribution of these durations by manufacturer are displayed in Figure 7.

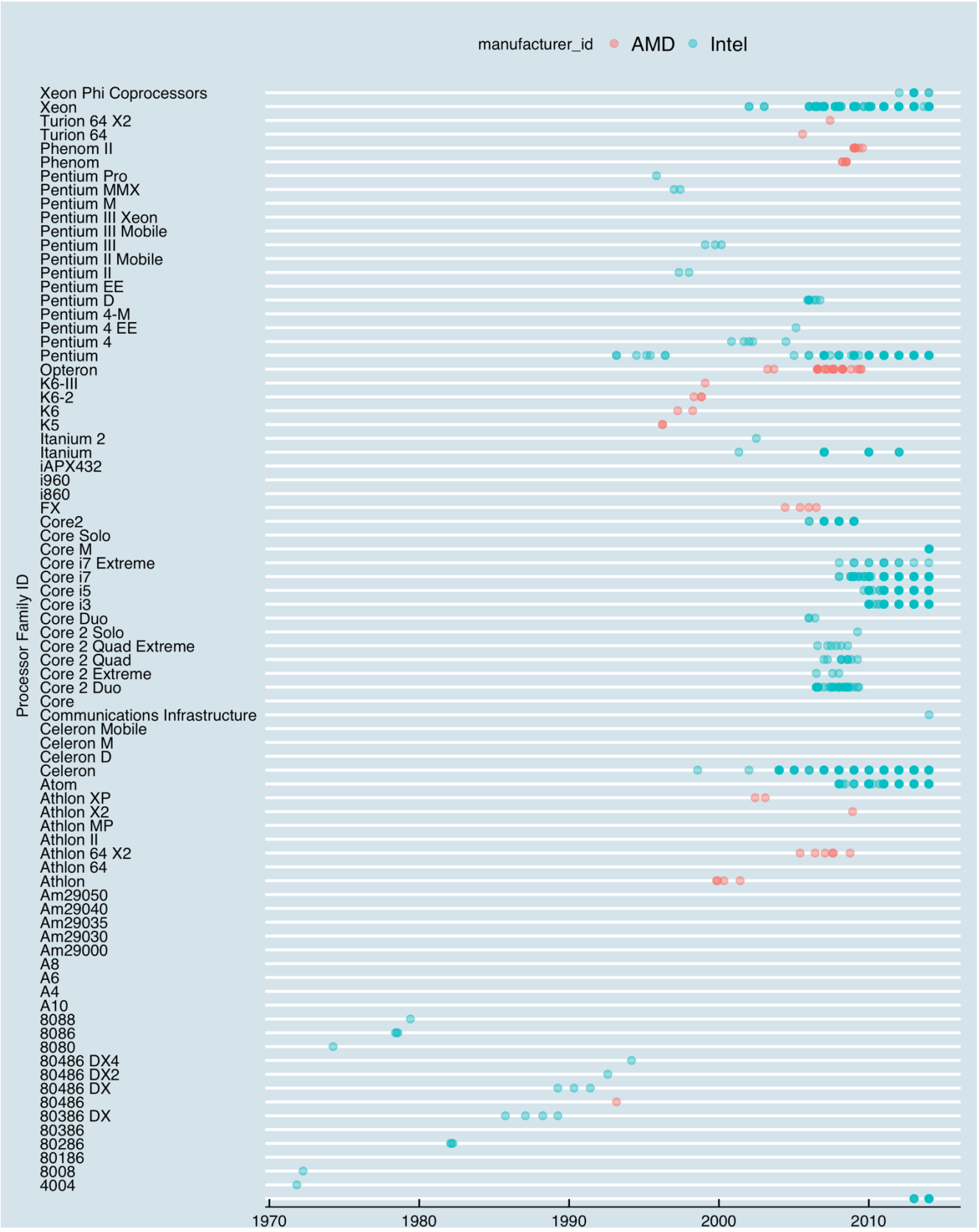

Figure 4. Processor introduction dates

The described methodology represents a trade-off between two imperfect ways of measuring CPU lifecycles. This method, looking at the first and last observations in a given family of microprocessors, is most succinct way to define durations, since examining each individual CPU introduced becomes arbitrarily difficult when new products are introduced in the same day or same week. The drawback of this method, is that the later high-profile branded Intel microprocessor families – the Pentium, the Xeon, the Core-i series – all have exceptionally long lifecycles as Intel migrated to a system of maintaining consistent branding across product lines. Therefore, for these three families of processors, durations were instead defined as the persistence of a given

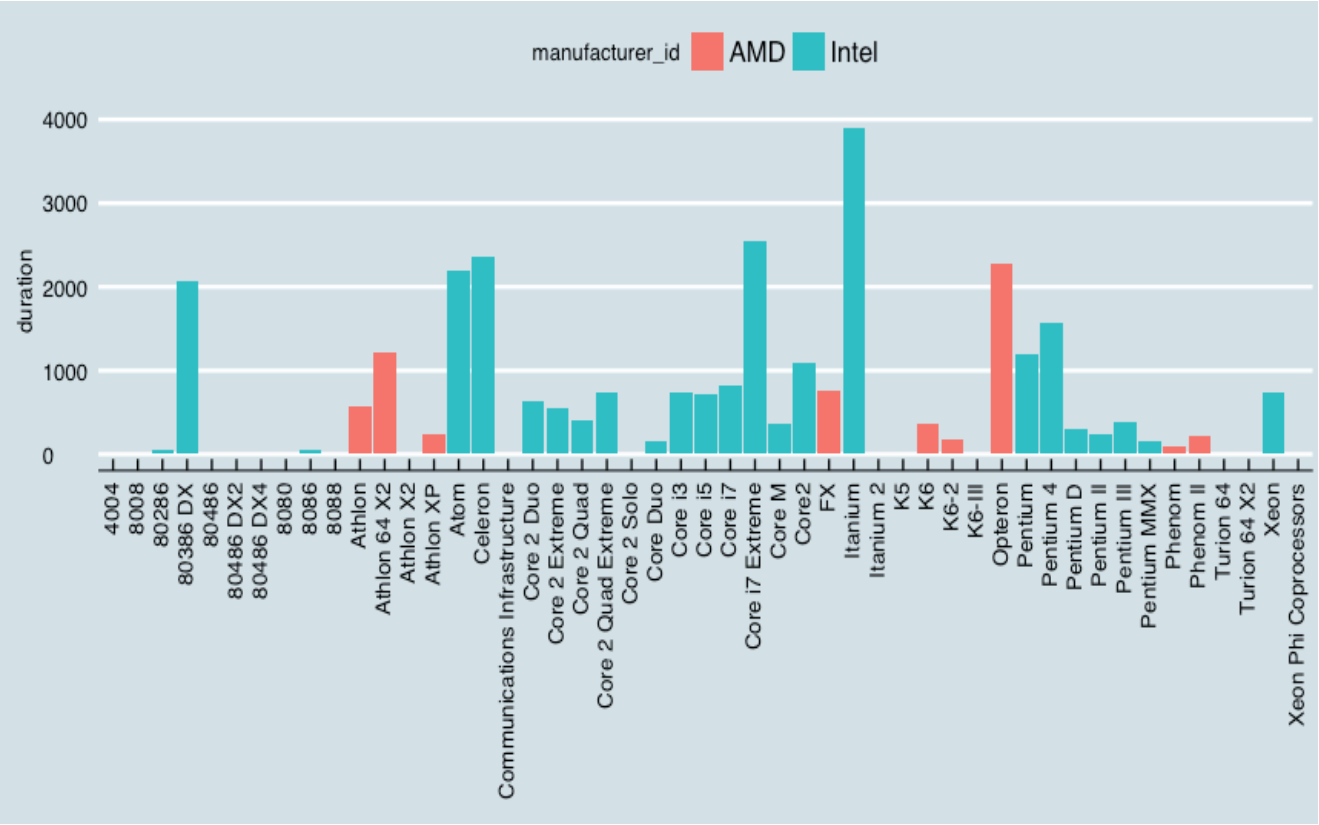

Figure 5. Processor Family Durations

microarchitecture within the broader Pentium, Xeon, or Core-i series by calculating their time durations within these relevant sub families.

Durations using the above described methodology are as seen in Figure 5.

Specifically, following this methodology provides 97 observations, 15 from AMD and 82 for Intel:

Group	n	Median	Mean	se(rmean)
Intel	82	1219	5038	1437
AMD	15	374	2234	481

Table 3. Summary duration statistics

As noted in Cox and Oakes (1984) other measures of usage besides time are useful, such as "operating time of a system, mileage of car, or some measure of cumulative load" [40] but these are more appropriate for industrial reliability applications than they are for econometric analysis. For microprocessor life-cycles, standard calendar time is used as the measurement variable, the major challenge of doing this as noted above, is that all processor families then by default contain right-censored *n*th observations. One can still go out today and buy an unused Intel 4004 CPU if so inclined.

[40] Cox and Oates 1984, 3.

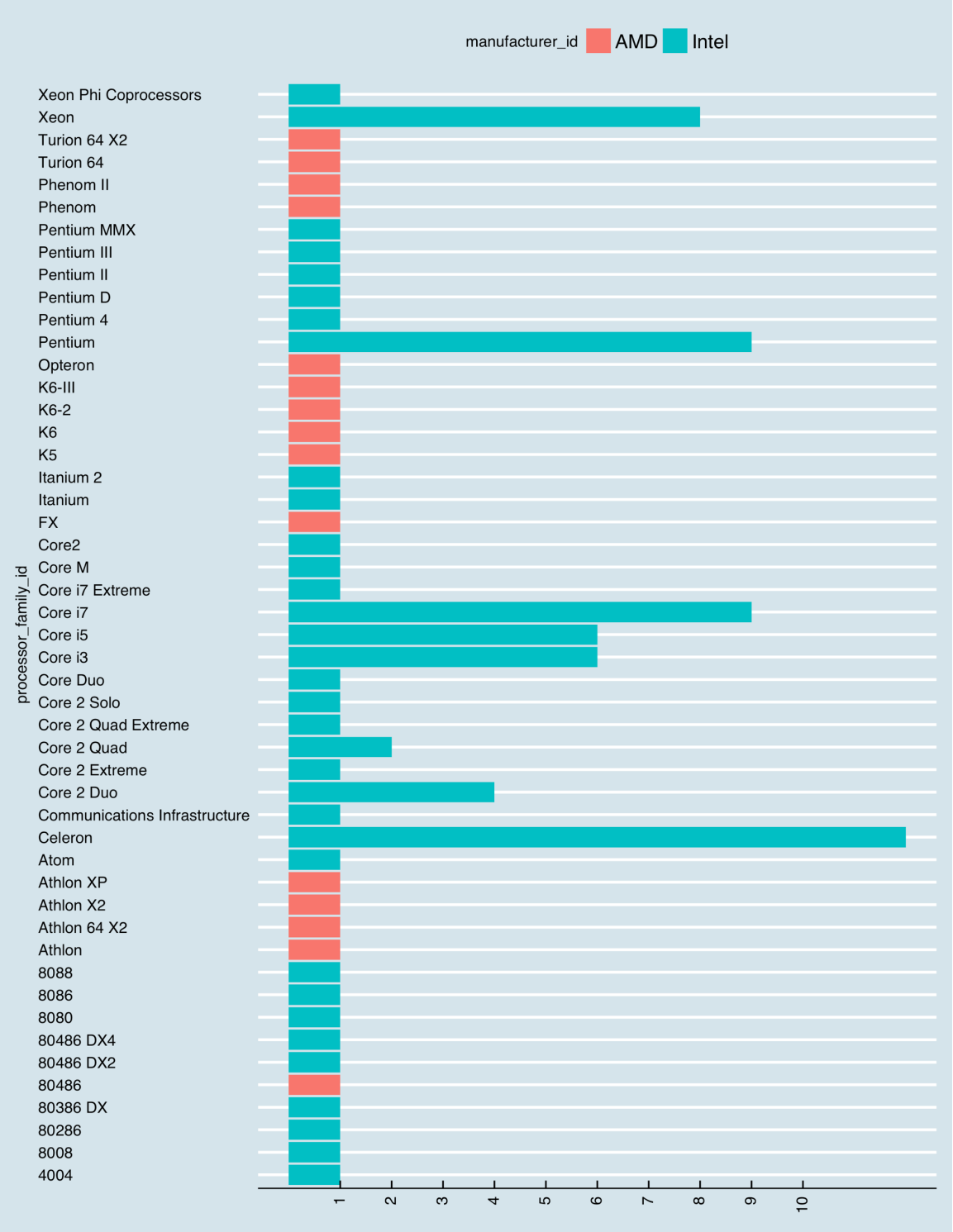

Figure 6. Number of Products within each family

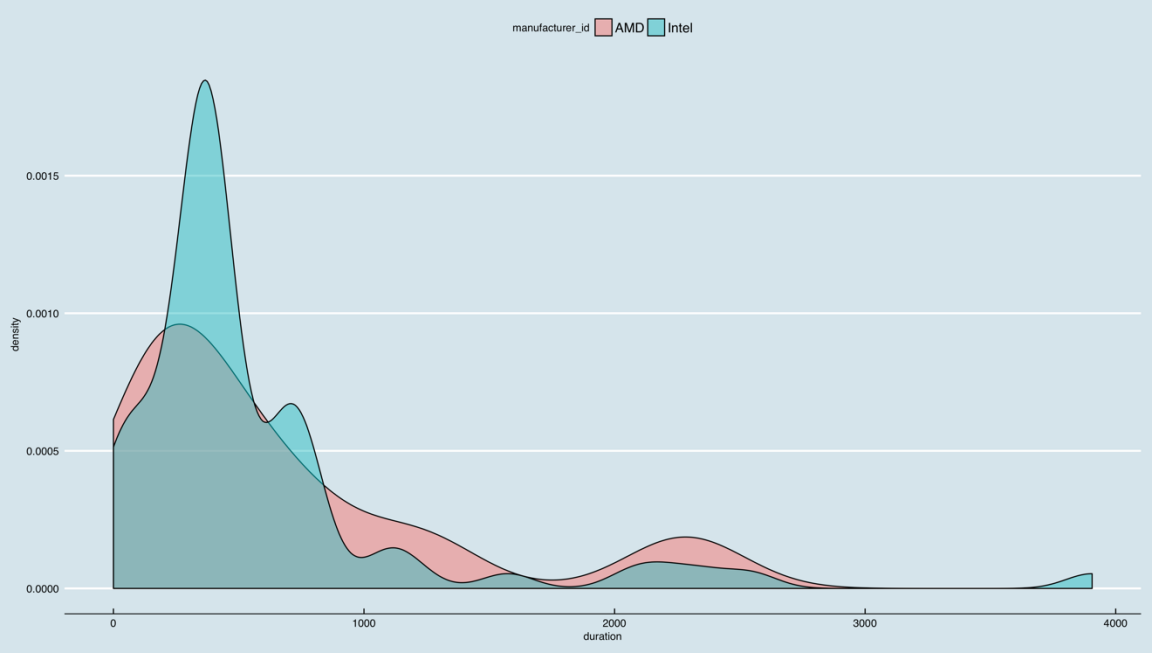

Figure 7. Density of duration times, by manufacturer

4.5 Code and Packages

Data loading, reshaping, and statistical analysis were all performed using the R programming language in the RStudio Integrated Development Environment (IDE). Reshaping, loading, and most visualization were performed using the R packages dplyr, lubridate, and ggplot2. The duration analysis itself was performed using the survival package authored by MAYO Clinic statistician Terry Therneau and available on the Comprehensive R Archive Network (CRAN). The package conveniently features a number of useful functions, including surv() which creates objects to serve as response variables in regression formulas, survfit() which creates survival curves given the unique structure of the output data, and coxph() which fits the semi-parametric Cox proportional hazards regression model to the data. Full project code can be found in Appendix A.

5 Results

5.1 Simple Non-Parametric Kaplan-Meier

The Kaplan-Meier estimator captures the decay in product lifecycles and censoring with a sharp drop off in survival following the one-year mark, which is what one would expect from a market defined by constant seasonal updates.

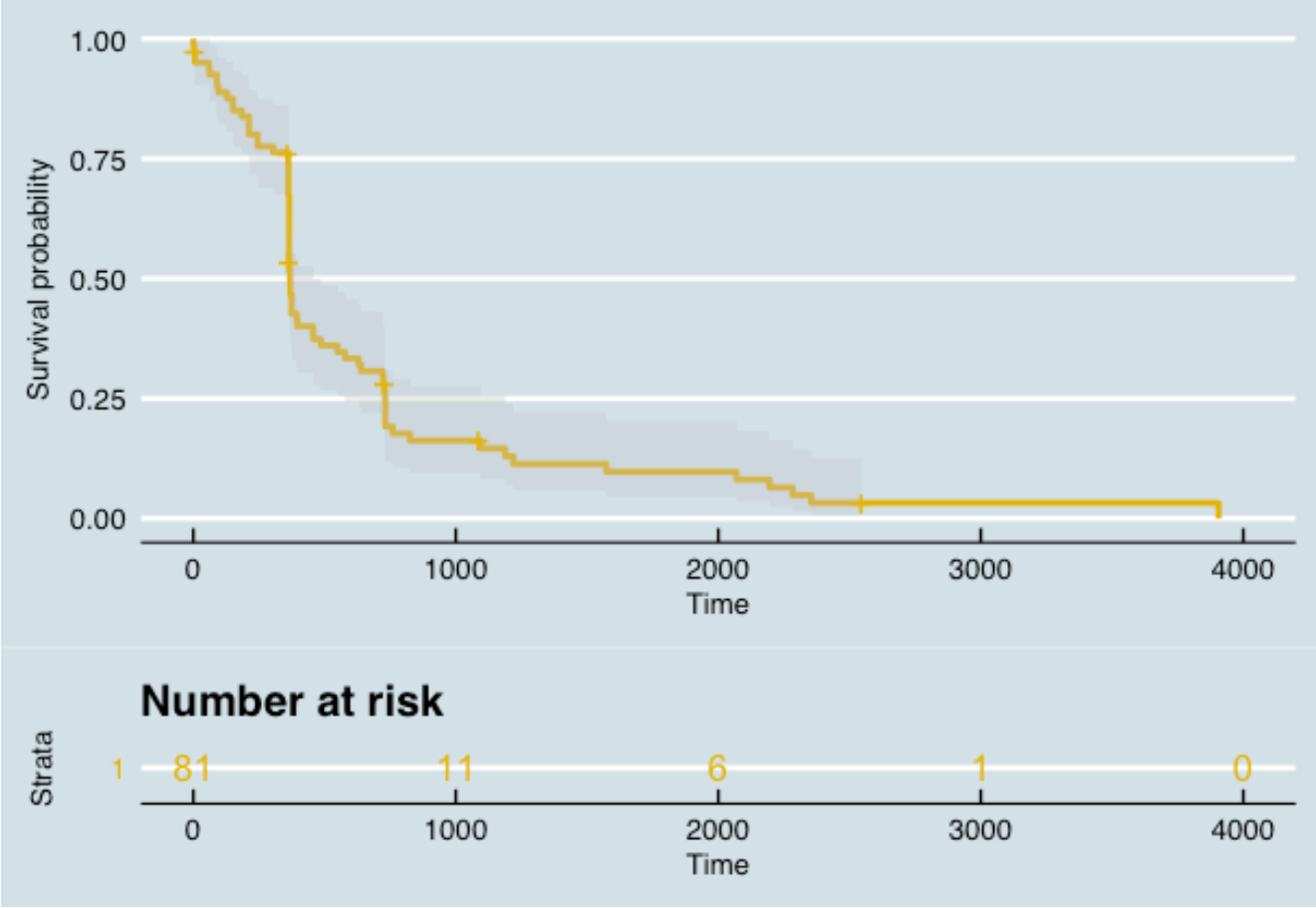

Figure 8. Kaplan-Meier Survival Curve for Combined Intel & AMD Observations

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0	81	1	0.9877	0.0123	0.96390	1.000
3	80	1	0.9753	0.0172	0.94209	1.000
6	78	2	0.9503	0.0242	0.90398	0.999
59	76	1	0.9378	0.0269	0.88645	0.992
61	75	1	0.9253	0.0293	0.86954	0.985
90	74	2	0.9003	0.0335	0.83704	0.968
96	72	1	0.8878	0.0352	0.82131	0.960
128	71	1	0.8753	0.0369	0.80585	0.951
151	70	2	0.8503	0.0399	0.77562	0.932
184	68	1	0.8378	0.0412	0.76080	0.923
212	67	3	0.8003	0.0447	0.71731	0.893
245	64	2	0.7752	0.0467	0.68900	0.872
304	62	1	0.7627	0.0475	0.67502	0.862
362	60	7	0.6738	0.0526	0.57822	0.785
365	53	11	0.5339	0.0561	0.43460	0.656
366	40	1	0.5206	0.0562	0.42124	0.643
368	39	2	0.4939	0.0564	0.39478	0.618
369	37	2	0.4672	0.0564	0.36867	0.592
374	35	3	0.4271	0.0561	0.33012	0.553
394	32	1	0.4138	0.0560	0.31745	0.539
396	31	1	0.4004	0.0557	0.30486	0.526
456	30	1	0.3871	0.0554	0.29235	0.513
457	29	1	0.3737	0.0551	0.27994	0.499
486	28	1	0.3604	0.0547	0.26761	0.485
549	27	1	0.3470	0.0543	0.25538	0.472
578	26	1	0.3337	0.0538	0.24324	0.458
629	25	1	0.3204	0.0533	0.23119	0.444
639	24	1	0.3070	0.0527	0.21924	0.430
722	23	1	0.2937	0.0521	0.20740	0.416
723	22	1	0.2803	0.0514	0.19566	0.402
728	19	1	0.2656	0.0508	0.18255	0.386
730	18	1	0.2508	0.0501	0.16960	0.371
731	17	4	0.1918	0.0462	0.11966	0.307
760	13	1	0.1770	0.0449	0.10768	0.291
825	12	1	0.1623	0.0435	0.09594	0.275
1096	10	1	0.1461	0.0421	0.08303	0.257
1188	9	1	0.1298	0.0404	0.07053	0.239
1219	8	1	0.1136	0.0385	0.05848	0.221
1573	7	1	0.0974	0.0362	0.04694	0.202
2069	6	1	0.0811	0.0336	0.03600	0.183
2195	5	1	0.0649	0.0306	0.02578	0.163
2283	4	1	0.0487	0.0269	0.01649	0.144
2354	3	1	0.0325	0.0223	0.00844	0.125
3906	1	1	0.0000	NaN	NA	NA

Table 4. Kaplan Meier Survival Table for AMD & Intel

The simple Kaplan-Meier estimator has a mean survival time of 675.6 days (s.e. = 94.5) and a median survival time of 368 days.

5.2 Non-Parametric Kaplan-Meier Across Companies

Using the same Kaplan-Meier estimators to look at AMD and Intel separately shows the different outcomes in product lifecycles between the two manufacturers. AMD processor families show a more rapid probability of their duration ending before the one-year mark, while Intel faces a steeper probability of processor family ending starting at the one-year mark and persisting through time.

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
96	9	1	0.889	0.105	0.7056	1.000
184	8	1	0.778	0.139	0.5485	1.000
212	7	1	0.667	0.157	0.4200	1.000
245	6	1	0.556	0.166	0.3097	0.997
365	5	1	0.444	0.166	0.2141	0.923
578	4	1	0.333	0.157	0.1323	0.840
760	3	1	0.222	0.139	0.0655	0.754
1219	2	1	0.111	0.105	0.0175	0.705
2283	1	1	0.000	NaN	NA	NA

Table 5. Kaplan Meier table for AMD

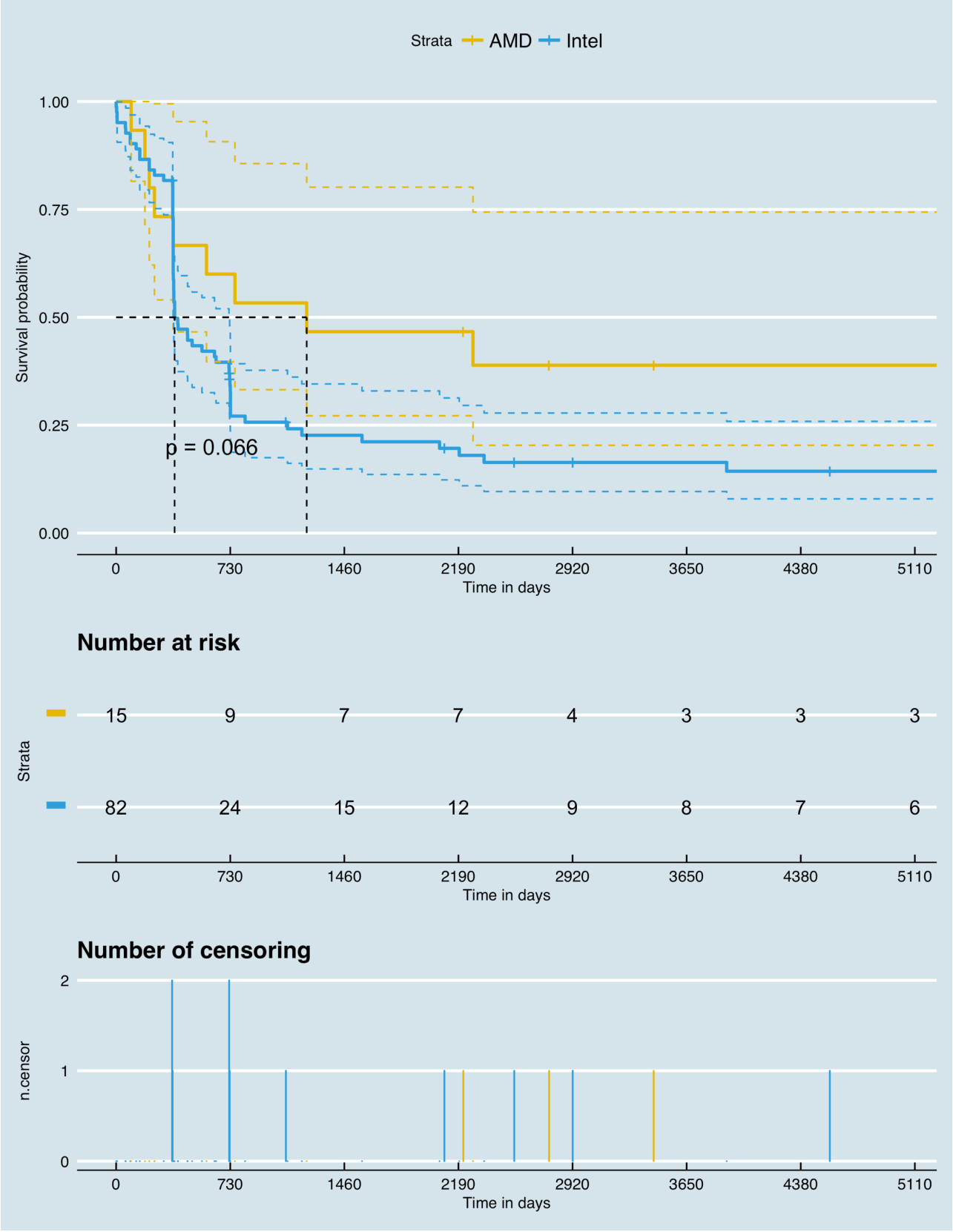

Figure 8. Kaplan Meier curve for Intel & AMD

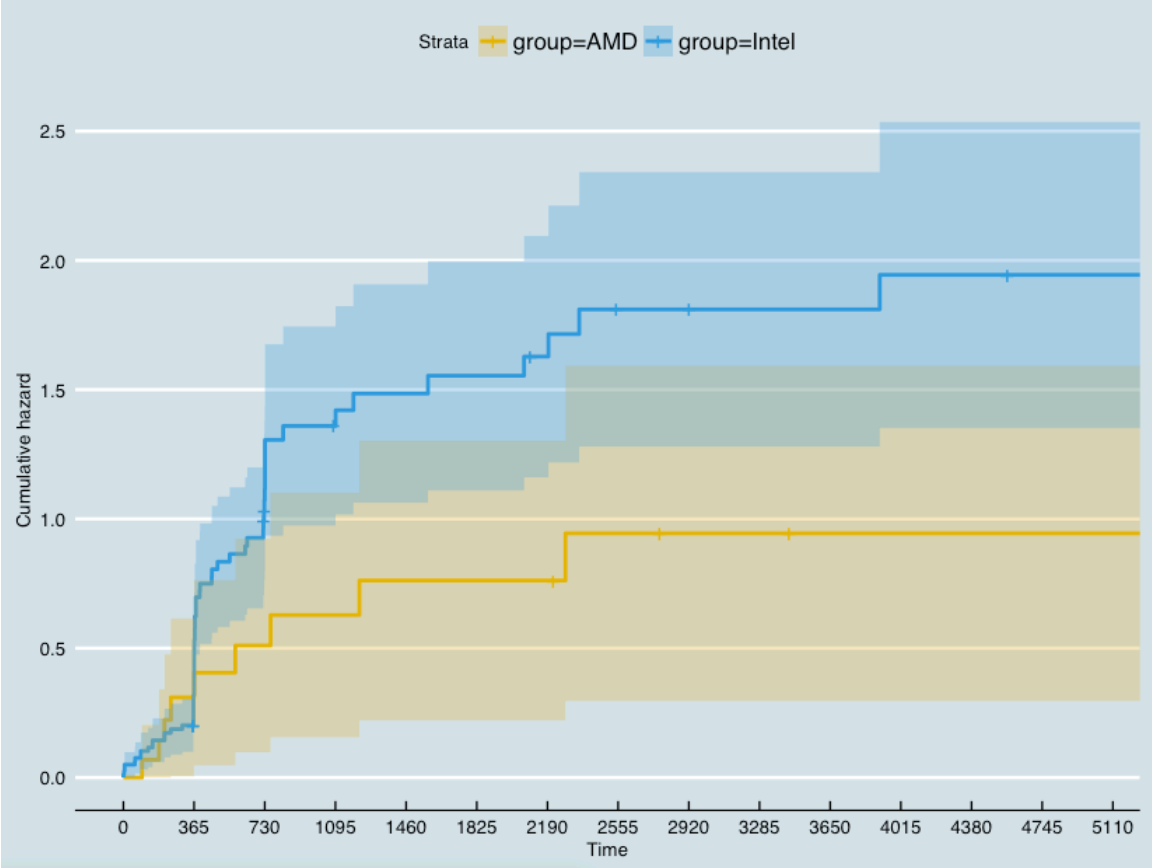

Figure 9. Cumulative KM Hazard Function for AMD & Intel

time	n.risk	n.event	survival	std.err	lower 95% CI	upper 95% CI
0	72	1	0.9861	0.0138	0.9594	1.000
3	71	1	0.9722	0.0194	0.9350	1.000
6	69	2	0.9440	0.0272	0.8922	0.999
59	67	1	0.9300	0.0302	0.8726	0.991
61	66	1	0.9159	0.0329	0.8536	0.983
90	65	2	0.8877	0.0374	0.8173	0.964
128	63	1	0.8736	0.0394	0.7997	0.954
151	62	2	0.8454	0.0429	0.7654	0.934
212	60	2	0.8172	0.0458	0.7322	0.912
245	58	1	0.8031	0.0472	0.7158	0.901
304	57	1	0.7890	0.0484	0.6997	0.890
362	55	7	0.6886	0.0551	0.5886	0.806
365	48	10	0.5452	0.0595	0.4402	0.675
366	36	1	0.5300	0.0597	0.4250	0.661
368	35	2	0.4997	0.0600	0.3949	0.632
369	33	2	0.4694	0.0601	0.3653	0.603
374	31	3	0.4240	0.0597	0.3217	0.559
394	28	1	0.4089	0.0595	0.3075	0.544
396	27	1	0.3937	0.0592	0.2933	0.529
456	26	1	0.3786	0.0588	0.2792	0.513
457	25	1	0.3634	0.0584	0.2653	0.498
486	24	1	0.3483	0.0579	0.2515	0.482
549	23	1	0.3332	0.0573	0.2378	0.467
629	22	1	0.3180	0.0567	0.2243	0.451
639	21	1	0.3029	0.0559	0.2109	0.435
722	20	1	0.2877	0.0552	0.1976	0.419
723	19	1	0.2726	0.0543	0.1845	0.403
728	16	1	0.2555	0.0535	0.1695	0.385
730	15	1	0.2385	0.0526	0.1548	0.367
731	14	4	0.1704	0.0473	0.0988	0.294
825	10	1	0.1533	0.0456	0.0856	0.274
1096	8	1	0.1342	0.0437	0.0708	0.254
1188	7	1	0.1150	0.0415	0.0567	0.233
1573	6	1	0.0958	0.0387	0.0434	0.212
2069	5	1	0.0767	0.0354	0.0310	0.190
2195	4	1	0.0575	0.0313	0.0198	0.167
2354	3	1	0.0383	0.0261	0.0101	0.146
3906	1	1	0.0000	NaN	NA	NA

Table 6. Kaplan Meier Survival Table for Intel Product Lines

5.3 Basic Parametric Model with Covariates

This section contains the results for the semi-parametric Cox proportional hazards model results looking at the predicted effects of clock speed, transistor count, and number of cores. The Z-test p-values are for the two-sided alternative, but in this case even halving them for the one-sided alternative yields little predictive power with any degree of statistical significance; the one exception is transistor count, which in certain scenarios predicts a great probability of shorter durations.

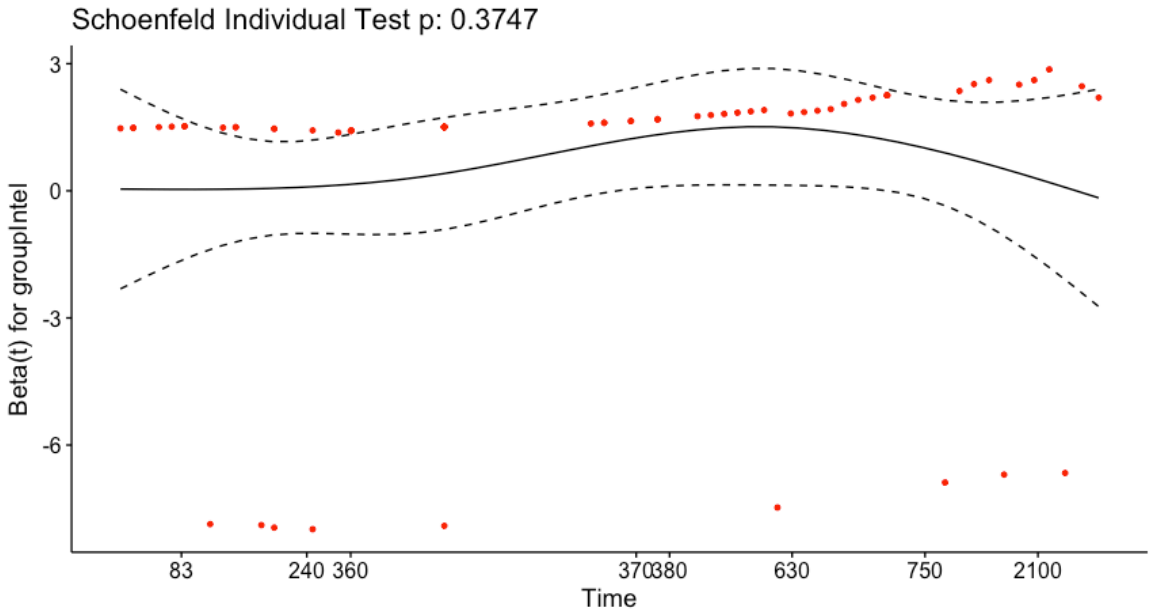

Figure 10. Schoenfield Individual Test for Intel/AMD Proportional Hazards Model

	Dependent variable:							
	time							
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
groupIntel	0.640*							
	(0.359)							
hw_ncores		−0.014			0.014		0.665**	0.233
		(0.024)			(0.094)		(0.297)	(0.224)
clock_in_ghz			0.424***		0.538**	0.442**		0.336*
			(0.120)		(0.223)	(0.208)		(0.186)
transistors_in_billions:clock_in_ghz						−0.826		
						(0.684)		
transistors_in_billions:hw_ncores							−1.049*	
							(0.564)	
transistors_in_billions				1.307**		2.332	2.690*	−0.638
				(0.536)		(1.777)	(1.607)	(1.193)
hw_ncores:clock_in_ghz					−0.038			
					(0.072)			
Observations	96	96	96	69	96	69	69	69
R^2	0.038	0.005	0.129	0.077	0.141	0.138	0.135	0.133
Max. Possible R^2	0.997	0.998	0.998	0.997	0.998	0.997	0.997	0.997
Log Likelihood	−285.050	−288.169	−281.798	−195.462	−281.145	−193.095	−193.214	−193.293
Wald Test	3.170* (df = 1)	0.370 (df = 1)	12.430*** (df = 1)	5.960** (df = 1)	12.720*** (df = 3)	8.850** (df = 3)	10.250** (df = 3)	9.890** (df = 3)
LR Test	3.697* (df = 1)	0.489 (df = 1)	13.230*** (df = 1)	5.504** (df = 1)	14.537*** (df = 3)	10.239** (df = 3)	10.001** (df = 3)	9.842** (df = 3)
Score (Logrank) Test	3.279* (df = 1)	0.389 (df = 1)	12.810*** (df = 1)	6.096** (df = 1)	13.477*** (df = 3)	9.426** (df = 3)	10.846** (df = 3)	10.366** (df = 3)
Note:								*p<0.1; **p<0.05; ***p<0.01

Table 7. Cox Proportional Hazard Results

Since the Cox proportional hazards model is predicting the conditional death rate, the positive coefficients predict shorter time in the market for a given processor line. Figure 10 shows the results of the Schoenfeld residuals test, which verifies the independence between time and the residuals of the Cox model. In this case, the residuals are not independent of time, challenging the idea that Intel and AMD have separate hazards associated with their processor lines, and remaining consistent with our other results. Table 7 lists 8 models selected to cover a range of possible statistical relationships. Models 1-4 were selected to test the individual predictors treating Intel as a binary factor variable, and the three characteristic variables as individual predictors. All models including transistor count have fewer available observations and thus a total sample size of 69 instead of 97 for the other models. Model 8 was selected to test the simultaneous effect of all three characteristic capturing variables, and the remaining models were selected to test for any potential interaction effects; for example, does clock speed affect duration only on processors with certain numbers of cores (model 5), or does transistor count effect the expected based on overall clockspeed (model 6).

5.4 Fully Parametric Model with Covariates

Using fully parametric approaches means specifying a few different distributional options. Where Kaplan-Meier approach does not require assuming any underlying distributional form to the data, the accelerated failure time estimators are maximum likelihood estimators for the parameters of the Weibull, exponential, or log-logistic distributions (see Figure 3). The log-logistic and log-normal produce similar but distinct

results, with the log-logistic fitting marginally better than the log-normal in the case of transistors specifically.

5.4.1 Single Predictors

Being informed by the Cox proportional hazards model, the first AFT models look at transistors and clock speed separately, both of which have statistically significant effects. Since the AFT model is looking at time to failure, these negative coefficients are consistent with the Cox proportional hazards' positive coefficients; namely, that a higher transistor count and faster clock speed both predict shorter expected durations of the processor family. In this case, the coefficients suggest that for every billion transistors, the product spends 3 fewer days on the market. Similarly, the results in Table 9 tell the same story for clock speed, which also trends with transistor count as the two are highly correlated.

5.4.2 Testing Parameter Restrictions of the Weibull and Exponential

As noted in section 4.2, the exponential is a special case of the Weibull, where the shape parameter in the Weibull is set to 1. The fully parametric, single predictor transistor Weibull model predicts a shape parameter of 0.7143 [41]; the difference between a parametric value of 1 (the special case equal to the exponential), the predicted shape parameter value of 0.7143, and a Weibull with the shape parameter of 1 but the scale

[41] The survreg output from the R package technically predicts a 'scale' parameter of 1.4; however, the package embeds the ouput in a general location-scale family; so the shape parameter is 1/scale from the summary output.

adjusted to 0.9 (to show contrast with the exponential) are visualized in Figure 11. Performing an analysis of variance test between the Weibull curve with the shape restricted to 1 and that at 0.714. yields a statistically significant variation (table 8), supporting the use of the general case of the Weibull in adding the additional shape parameter to the accelerated failure time models.

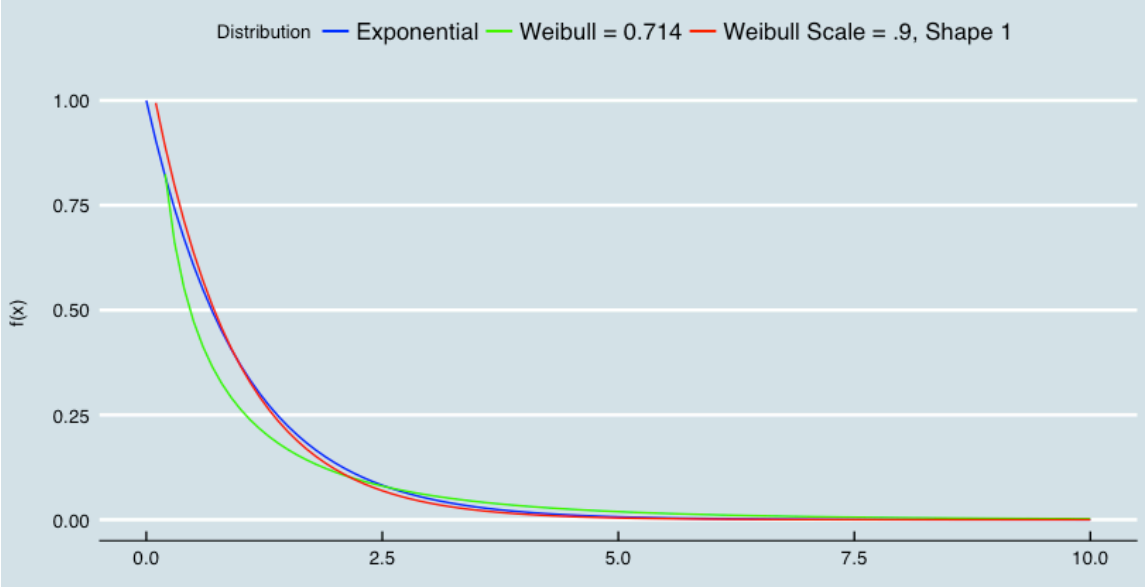

Figure 11. Visualization of Weibull Parameter Values

Statistic	N	Mean	St. Dev.	Min	Max
Resid. Df	2	66.500	0.707	66	67
-2*LL	2	882.210	7.571	876.857	887.564
Df	1	−1.000		−1	−1
Deviance	1	−10.707		−10.707	−10.707
Pr(>Chi)	1	0.001		0.001	0.001

Table 8. Testing Variance of Weibull Shape Parameters

	Dependent variable:			
	time			
	exponential	*Weibull*	*survreg: loglogistic*	*survreg: lognormal*
	(1)	(2)	(3)	(4)
transistors_in_billions	−3.204***	−3.323***	−2.212**	−2.411**
	(0.458)	(0.800)	(0.914)	(0.989)
Constant	8.401***	8.300***	7.168***	7.288***
	(0.185)	(0.313)	(0.340)	(0.346)
Observations	69	69	69	69
Log Likelihood	−460.086	−447.373	−441.785	−443.448
χ^2 (df = 1)	37.816***	13.090***	5.760**	5.756**
Note:				*p<0.1; **p<0.05; ***p<0.01

Table 9. Fully parametric AFT results for transistors

	Dependent variable:			
	time			
	exponential	*Weibull*	*survreg: loglogistic*	*survreg: lognormal*
	(1)	(2)	(3)	(4)
clock_in_ghz	−0.889***	−0.922***	−0.639***	−0.661***
	(0.102)	(0.137)	(0.172)	(0.176)
Constant	8.934***	8.964***	7.809***	7.842***
	(0.235)	(0.313)	(0.388)	(0.376)
Observations	96	96	96	96
Log Likelihood	−594.851	−590.622	−588.836	−592.489
χ^2 (df = 1)	82.261***	37.366***	13.488***	13.652***
Note:				*p<0.1; **p<0.05; ***p<0.01

Table 10. Fully parametric AFT results for clock speed

5.4.3 Multiple Predictors

Tables 8 (and its complement in table 11 showing log of transistor count) and 9 show that clock speed and transistor count both predict shorter product family lifetimes when tested in isolation; interestingly enough, putting all the major processor attributes into a single model eliminates the effect of transistor count. Because more transistors allow for more operations per second, it is mathematically linked directly to processor clock speed, with some degree of variance depending on the specific aspects of microarchitecture designs. This fundamental link between the two supports the suggestion that when transistors are in a standalone model its effect is a proxy for clock speed. The reason this is relevant is because clock speed itself captures some underlying trend for market competitiveness. As time progressed from 1971 onward, microprocessors went from narrowly used components included in a tiny personal computer market, to being installed in the billions of phones, laptops, tablets, personal computers, gaming consoles, and servers that are in use today. The lack of significance of the hardware cores variable could be a result of multiple hardware cores being in use consistently over time, and thus not capturing this market competitiveness element, or it could be due to the little underlying variance in the data, with most processors only having 1 core, and when they have 2 or 4 cores, it genuinely does not predict any shorter lifespan for the product line.

	Dependent variable:			
	time			
	exponential	*Weibull*	*survreg: loglogistic*	*survreg: lognormal*
	(1)	(2)	(3)	(4)
transistors_in_billions	1.119	1.287	0.962	1.542
	(1.318)	(1.857)	(1.953)	(2.224)
hw_ncores	−0.290	−0.386	−0.438	−0.541
	(0.251)	(0.353)	(0.377)	(0.428)
clock_in_ghz	−0.885***	−0.900***	−0.545**	−0.573*
	(0.179)	(0.258)	(0.270)	(0.298)
Constant	9.229***	9.330***	8.174***	8.277***
	(0.341)	(0.499)	(0.578)	(0.595)
Observations	69	69	69	69
Log Likelihood	−448.506	−441.599	−439.292	−441.017
χ^2 (df = 3)	60.974***	24.637***	10.745**	10.618**
Note:				*p<0.1; **p<0.05; ***p<0.01

Table 11. Fully parametric AFT results for all variables

	Dependent variable:		
	time		
	exponential	*Weibull*	*survreg: lognormal*
	(1)	(2)	(3)
log(transistors)	−0.349***	−0.381***	−0.335***
	(0.058)	(0.080)	(0.074)
Constant	8.825***	8.903***	8.064***
	(0.301)	(0.413)	(0.387)
Observations	69	69	69
Log Likelihood	−443.782	−438.428	−436.229
χ^2 (df = 1)	70.423***	30.979***	16.871***
Note:			*p<0.1; **p<0.05; ***p<0.01

Table 12. Log of Transistors in AFT Model

6 Summary

Following the standard duration analysis practice of examining both parametric and nonparametric estimates of CPU lifecycle durations at AMD and Intel, finds that the hedonic covariates of transistor count, die size, cores, and clock speed have little meaningful effect on the lifecycle of the microprocessor product lines. Specifically, it is found that the main attributes of a given processor (clock speed, transistor count, number of cores) have little statistically significant predicted effect on the conditional duration of CPU product-lines, which have a mean survival time of 456 days. Clock speed and transistor count both predict shorter durations under a variety of models, however, it stands to reason that this could be capturing some unknown other phenomena such as market competitiveness and faster iterative hardware cycles that came in later years. Within the parametric models, study was limited to testing the three most commonly used distributions in duration analysis, the exponential, Weibull, and log-logistic. This is a result of the assumption that the absence of any form of statistical significance within these common distributions would render more obscure distributions to find a marginally better fit irrelevant. Furthermore, the processor characteristics are time-invariant, so their changing does not inherently have to be considered within the Cox proportional hazards or the accelerated time failure models.

The principal implication of these results is that there is a natural, relatively unquantitative arbitrariness to CPU product lines, based on marketing and broader market dynamics, rather than persistent long-running characteristics of CPUs being marketed under a given product line.

6.1 Shortcomings & Future Research

Intel's, and to a lesser degree AMD's, tendency for opaque and byzantine naming schemes of its architecture is the single largest limiting factor in terms of analysis. Some of this is likely attributable to the difficult nature of designing and manufacturing semiconductors, since tricking rocks into thinking is no small task. However, the longevity of their flagship product lines has more to do with consistent branding than it does with any underlying similarity between the current architecture and the nature of the processors at introduction. Unfortunately, this opaque naming scheme, combined with the rigid and predictable 1 to 2 years update cycles, likely render much further duration analysis of this sort unfruitful given the data limitations. Absent these data concerns, a lifecycle duration analysis on a per individual CPU model-number, with life-cycle defined based on some fixed percent reduction in gross sales would be a very enlightening avenue of research given the potential dynamics surrounding sales of a specific CPU.

References

Audretsch, David B. "Innovation, growth and survival." International Journal of Industrial Organization 13, no. 4 (1995): 441-57. doi:10.1016/0167-7187(95)00499-8.

Audretsch, David B., and Talat Mahmood. "New Firm Survival: New Results Using a Hazard Function." The Review of Economics and Statistics 77, no. 1 (1995): 97. doi:10.2307/2109995.

Breslow, N. E. "Analysis of Survival Data under the Proportional Hazards Model." International Statistical Review / Revue Internationale de Statistique 43, no. 1 (1975): 45. doi:10.2307/1402659.

Breslow, N. E. "Covariance Analysis of Censored Survival Data." Biometrics 30, no. 1 (1974): 89. doi:10.2307/2529620.

Burton, Michael, Dan Rigby, and Trevor Young. "Modelling the Adoption of Organic Horticultural Technology in the UK using Duration Analysis." The Australian Journal of Agricultural and Resource Economics 47, no. 1 (2003): 29-54. doi:10.1111/1467-8489.00202.

Cox, David R. "Regression Models and Life-Tables." Journal of the Royal Statistical Society 34, no. 2 (1972): 187-220. doi:10.1007/978-1-4612-4380-9-37.

Cox, David R., and D. Oakes. Analysis of Survival Data. Cambridge: Chapman and Hall, 1984.

Flinn, Christopher J., and James J. Heckman. New Methods for Analyzing Structural Models of Labor Force Dynamics. Cambridge, MA, 1982.

Gawer, Annabelle, and Rebecca Henderson. "Platform Owner Entry and Innovation in Complementary Markets: Evidence from Intel." Journal of Economics and Management Strategy 16, no. 1 (2007). doi:10.1111/j.1530-9134.2007.00130.x.

Goettler, Ronald L., and Brett R. Gordon. "Does AMD Spur Intel to Innovate More?" Journal of Political Economy 119, no. 6 (2011): 1141-200. doi:10.1086/664615.

Goettler, Ronald L., and Brett R. Gordon. "Competition and Product Innovation in Dynamic Oligopoly." Quantitative Marketing and Economics 12, no. 1 (March 2014): 1-42. doi:10.2139/ssrn.1944933.

Hahn, Gerald J., and Samuel S. Shapiro. Statistical Models in Engineering. New York: Wiley, 1968.

Juran, J. M., and A. Blanton. Godfrey. Juran's Quality Handbook. New York: McGraw Hill, 1999.

Kaplan, E. L., and Paul Meier. "Nonparametric Estimation from Incomplete Observations." Journal of the American Statistical Association 53, no. 282 (1958): 457. doi:10.2307/2281868.

Kimball, A. W. "Estimation of Mortality Intensities in Animal Experiments." Biometrics 16, no. 4 (1960): 505. doi:10.2307/2527758.

Lancaster, Tony. "Econometric Methods for the Duration of Unemployment." Econometrica 47.4 (1979): 939. Web.

Lancaster, Tony. The Econometric Analysis of Transition Data. Cambridge: Cambridge University Press, 1990.

Nelson, Wayne. "Theory and Applications of Hazard Plotting for Censored Failure Data." Technometrics 14, no. 4 (1972): 945. doi:10.2307/1267144.

Prentice, R. L., and L. A. Gloeckler. "Regression Analysis of Grouped Survival Data with Application to Breast Cancer Data." Biometrics 34, no. 1 (1978): 57. doi:10.2307/2529588.

Schaller, R.R. "Moore’s Law: Past, Present and Future." IEEE Spectrum 34, no. 6 (1997): 52-59. doi:10.1109/6.591665.

Wainer, Howard, and Paul F. Velleman. "Statistical Graphics: Mapping The Pathways Of Science." ETS Research Report Series 2000, no. 2 (2000): I-66. doi:10.1002/j.2333-8504.2000.tb01840.x.

Appendix A. Code

```
options(scipen = 999999)
library("dplyr")
library("tidyr")
library("ggplot2")
library("lubridate")
library("ggrepel")
library("ggthemes")
library("survival")
library("scales")
library("survminer")
library("stargazer")
setwd("/Volumes/SSD/Users/mischafisher/Dropbox/-- UVic/Thesis/Data")

##### GGSURV ######
ggsurv <- function(s, CI = 'def', plot.cens = T, surv.col = 'gg.def',
                   cens.col = 'red', lty.est = 1, lty.ci = 2,
                   cens.shape = 3, back.white = F, xlab = 'Time',
                   ylab = 'Survival', main = ''){

 library(ggplot2)
 strata <- ifelse(is.null(s$strata) ==T, 1, length(s$strata))
 stopifnot(length(surv.col) == 1 | length(surv.col) == strata)
 stopifnot(length(lty.est) == 1 | length(lty.est) == strata)

 ggsurv.s <- function(s, CI = 'def', plot.cens = T, surv.col = 'gg.def',
                      cens.col = 'red', lty.est = 1, lty.ci = 2,
                      cens.shape = 3, back.white = F, xlab = 'Time',
                      ylab = 'Survival', main = ''){

   dat <- data.frame(time = c(0, s$time),
                     surv = c(1, s$surv),
                     up = c(1, s$upper),
                     low = c(1, s$lower),
                     cens = c(0, s$n.censor))
   dat.cens <- subset(dat, cens != 0)

   col <- ifelse(surv.col == 'gg.def', 'black', surv.col)

   pl <- ggplot(dat, aes(x = time, y = surv)) +
     xlab(xlab) + ylab(ylab) + ggtitle(main) +
     geom_step(col = col, lty = lty.est)

   pl <- if(CI == T | CI == 'def') {
     pl + geom_step(aes(y = up), color = col, lty = lty.ci) +
       geom_step(aes(y = low), color = col, lty = lty.ci)
   } else (pl)

   pl <- if(plot.cens == T & length(dat.cens) > 0){
     pl + geom_point(data = dat.cens, aes(y = surv), shape = cens.shape,
                     col = cens.col)
   } else if (plot.cens == T & length(dat.cens) == 0){
```

```
    stop ('There are no censored observations')
  } else(pl)

  pl <- if(back.white == T) {pl + theme_bw()
  } else (pl)
  pl
}

ggsurv.m <- function(s, CI = 'def', plot.cens = T, surv.col = 'gg.def',
                     cens.col = 'red', lty.est = 1, lty.ci = 2,
                     cens.shape = 3, back.white = F, xlab = 'Time',
                     ylab = 'Survival', main = '') {
  n <- s$strata

  groups <- factor(unlist(strsplit(names
                                   (s$strata), '='))[seq(2, 2*strata, by = 2)])
  gr.name <-  unlist(strsplit(names(s$strata), '='))[1]
  gr.df <- vector('list', strata)
  ind <- vector('list', strata)
  n.ind <- c(0,n); n.ind <- cumsum(n.ind)
  for(i in 1:strata) ind[[i]] <- (n.ind[i]+1):n.ind[i+1]

  for(i in 1:strata){
    gr.df[[i]] <- data.frame(
      time = c(0, s$time[ ind[[i]] ]),
      surv = c(1, s$surv[ ind[[i]] ]),
      up = c(1, s$upper[ ind[[i]] ]),
      low = c(1, s$lower[ ind[[i]] ]),
      cens = c(0, s$n.censor[ ind[[i]] ]),
      group = rep(groups[i], n[i] + 1))
  }

  dat <- do.call(rbind, gr.df)
  dat.cens <- subset(dat, cens != 0)

  pl <- ggplot(dat, aes(x = time, y = surv, group = group)) +
    xlab(xlab) + ylab(ylab) + ggtitle(main) +
    geom_step(aes(col = group, lty = group))

  col <- if(length(surv.col == 1)){
    scale_colour_manual(name = gr.name, values = rep(surv.col, strata))
  } else{
    scale_colour_manual(name = gr.name, values = surv.col)
  }

  pl <- if(surv.col[1] != 'gg.def'){
    pl + col
  } else {pl + scale_colour_discrete(name = gr.name)}

  line <- if(length(lty.est) == 1){
    scale_linetype_manual(name = gr.name, values = rep(lty.est, strata))
  } else {scale_linetype_manual(name = gr.name, values = lty.est)}

  pl <- pl + line

  pl <- if(CI == T) {
```

```
    if(length(surv.col) > 1 && length(lty.est) > 1){
      stop('Either surv.col or lty.est should be of length 1 in order
           to plot 95% CI with multiple strata')
    }else if((length(surv.col) > 1 | surv.col == 'gg.def')[1]){
      pl + geom_step(aes(y = up, color = group), lty = lty.ci) +
        geom_step(aes(y = low, color = group), lty = lty.ci)
    } else{pl +  geom_step(aes(y = up, lty = group), col = surv.col) +
        geom_step(aes(y = low,lty = group), col = surv.col)}
  } else {pl}


  pl <- if(plot.cens == T & length(dat.cens) > 0){
    pl + geom_point(data = dat.cens, aes(y = surv), shape = cens.shape,
                    col = cens.col)
  } else if (plot.cens == T & length(dat.cens) == 0){
    stop ('There are no censored observations')
  } else(pl)

  pl <- if(back.white == T) {pl + theme_bw()
  } else (pl)
  pl
}
pl <- if(strata == 1) {ggsurv.s(s, CI , plot.cens, surv.col ,
                                cens.col, lty.est, lty.ci,
                                cens.shape, back.white, xlab,
                                ylab, main)
} else {ggsurv.m(s, CI, plot.cens, surv.col ,
                 cens.col, lty.est, lty.ci,
                 cens.shape, back.white, xlab,
                 ylab, main)}
 pl
}


################################################
################### THESIS #####################
################################################

d  <- read.csv("./cpudb/processor.csv", stringsAsFactors = FALSE)

#CLOCK SPEED IS IN MhZ
#Transistors are in millions 2300 for Intel 4004

d <- select(d, -created_at, -updated_at)
d$date <- ymd(d$date)
d$manufacturer_id[d$manufacturer_id == 1] <- "AMD"
d$manufacturer_id[d$manufacturer_id == 2] <- "Cypress"
d$manufacturer_id[d$manufacturer_id == 3] <- "DEC"
d$manufacturer_id[d$manufacturer_id == 4] <- "Fujitsu"
d$manufacturer_id[d$manufacturer_id == 5] <- "Hitachi"
d$manufacturer_id[d$manufacturer_id == 6] <- "HP"
d$manufacturer_id[d$manufacturer_id == 7] <- "IBM"
d$manufacturer_id[d$manufacturer_id == 8] <- "IDT"
d$manufacturer_id[d$manufacturer_id == 9] <- "Intel"
d$manufacturer_id[d$manufacturer_id == 10] <- "Motorola"
d$manufacturer_id[d$manufacturer_id == 11] <- "NEC"
```

```
d$manufacturer_id[d$manufacturer_id == 12] <- "Samsung"
d$manufacturer_id[d$manufacturer_id == 13] <- "TI"
d$manufacturer_id[d$manufacturer_id == 14] <- "Toshiba"
d$manufacturer_id[d$manufacturer_id == 15] <- "unnamed"
d$manufacturer_id[d$manufacturer_id == 16] <- "TSMC"
d$manufacturer_id[d$manufacturer_id == 17] <- "MIPS"
d$manufacturer_id[d$manufacturer_id == 18] <- "SGI"
d$manufacturer_id[d$manufacturer_id == 19] <- "Sun"
d$manufacturer_id[d$manufacturer_id == 20] <- "Broadcom"
d$manufacturer_id[d$manufacturer_id == 21] <- "Cyrix"
d$manufacturer_id[d$manufacturer_id == 22] <- "HAL"
d$manufacturer_id[d$manufacturer_id == 23] <- "MOS"
d$manufacturer_id[d$manufacturer_id == 24] <- "NexGen"
d$manufacturer_id[d$manufacturer_id == 25] <- "Ross"
d$manufacturer_id[d$manufacturer_id == 26] <- "Zilog"
d$manufacturer_id[d$manufacturer_id == 27] <- "WDC"
d$manufacturer_id[d$manufacturer_id == 28] <- "Centaur"
d$manufacturer_id <- as.factor(d$manufacturer_id)

d_codename <- read.csv("./cpudb/code_name.csv", stringsAsFactors = FALSE)
d$code_name_id <- d_codename$name[match(d$code_name_id, d_codename$id)]
d$code_name_id <- as.factor(d$code_name_id)

d_procfam <- read.csv("./cpudb/processor_familie.csv", stringsAsFactors = FALSE)
d$processor_family_id <- d_procfam$name[match(d$processor_family_id, d_procfam$id)]
d$processor_family_id <- as.factor(d$processor_family_id)

d_microid <- read.csv("./cpudb/microarchitecture.csv", stringsAsFactors = FALSE)
d$microarchitecture_id <- d_microid$name[match(d$microarchitecture_id, d_microid$id)]

d_techid <- read.csv("./cpudb/technologie.csv", stringsAsFactors = FALSE)
d$technology_id <- d_techid$name[match(d$technology_id, d_techid$id)]
d$technology_id <- as.factor(d$technology_id)

d$model <- as.factor(d$model)

## DISTRIBUTIONS
x_lower <- 0
x_upper <- 10
max_height2 <- max(dexp(x_lower:x_upper, rate = 1, log = FALSE),
                   dweibull(x_lower:x_upper, shape = 1, log = FALSE),
                   dlogis(x_lower:x_upper, scale = 1, log = FALSE))
ggplot(data.frame(x = c(x_lower, x_upper)), aes(x = x)) + xlim(x_lower, x_upper) +
  ylim(0, max_height2) +
  stat_function(fun = dexp, args = list(rate = 2), aes(colour = "Exponential")) +
  stat_function(fun = dweibull, args = list(shape = 2), aes(colour = "Weibull")) +
  stat_function(fun = dlogis, args = list(scale = 2), aes(colour = "Logistic")) +
  scale_color_manual("Distribution", values = c("blue", "green", "red")) +
  labs(x = "\n x", y = "f(x) \n",
       title = "Common Survival Analysis Distribution Density Plots \n") +
  theme(plot.title = element_text(hjust = 0.5),
        axis.title.x = element_text(face="bold", colour="blue", size = 12),
        axis.title.y = element_text(face="bold", colour="blue", size = 12),
        legend.title = element_text(face="bold", size = 10),
        legend.position = "top") + theme_economist()
```

```
#FULL DATA PLOT
ggplot(d, aes(date, log(transistors), color=manufacturer_id, size=hw_ncores)) +
  geom_point(alpha = 0.5) +
  xlab("Date") + ylab("Log of Transistor Count") +
  theme_economist()
ggsave("imgFullData.pdf", path="../LaTeX/img/", width=24, height=12, units="cm")
ggsave("imgFullData.png", path="../LaTeX/img/", width=8.5, height=11, units="in")


#MANUFACTURER SUMS
ggplot(d, aes(manufacturer_id)) + geom_bar() + theme_economist() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust=0.4),axis.title.x=element_blank()) +
  scale_y_continuous(labels=scales::comma) + ylab("Processor Count") + coord_flip() +
  theme(text = element_text(size=20))
ggsave("imgManuCount.pdf", path="../LaTeX/img/", width=24, height=48, units="cm")
ggsave("imgManuCount.png", path="../LaTeX/img/", width=8.5, height=11, units="in")


#### Microarchitecture ID ####

ggplot(filter(d, manufacturer_id == "Intel"), aes(date, microarchitecture_id)) + geom_point()
ggplot(filter(d, manufacturer_id == "AMD"), aes(date, microarchitecture_id)) + geom_point()


#### Processor Family ID ####

ggplot(filter(d, manufacturer_id == "Intel"), aes(date, processor_family_id)) + geom_point(size=2)
ggplot(filter(d, manufacturer_id == "AMD"), aes(date, processor_family_id)) + geom_point(size=2)
dtemp <- filter(d, manufacturer_id == "AMD" | manufacturer_id =="Intel")
names(dtemp)
ggplot(dtemp, aes(date, processor_family_id, color=manufacturer_id)) + geom_point(size=2, alpha=0.5)
+
  ylab("Processor Family ID") + xlab("Year") + theme_economist()
ggsave("imgProcessorFamilyDates.pdf", path="../LaTeX/img/", width=25.5, height=33, units="cm")
ggsave("imgProcessorFamilyDates.png", path="../LaTeX/img/", width=8.5, height=11, units="in")

##### CREATE DURATIONS ######

d3 <- d %>% filter(manufacturer_id=="Intel" | manufacturer_id=="AMD") %>% droplevels()
d3 <- filter(d3, !is.na(date))
d3 = d3 %>% group_by(processor_family_id) %>%
  mutate(duration = c(diff(date), NA_real_),
         last.date = if_else(date==max(date), max(date), as.Date(NA))) %>%
  arrange(processor_family_id, date)
d3 <- arrange(d3, processor_family_id, date)
d3 <- d3 %>% filter(processor_family_id != "")

d3 = d3 %>% group_by(processor_family_id) %>%
  mutate(duration = c(diff(max(date)), NA_real_),
         last.date = if_else(date==max(date), max(date), as.Date(NA))) %>%
  arrange(processor_family_id, date)

d_intel <- d %>% filter(manufacturer_id=="Intel")
write.csv(d_intel, "intel.csv")
d_AMD <- d %>% filter(manufacturer_id=="AMD")
```

```
d_AMD <- filter(d_AMD, !is.na(date))
d_AMD <- filter(d_AMD, processor_family_id != "")
write.csv(d_AMD, "amd.csv")

d_AMD_durations <- read.csv("AMD-durations.csv", stringsAsFactors = FALSE)
d_AMD_durations <- filter(d_AMD_durations, !is.na(duration) | censor=="yes")
d_AMD_durations$event <- 1
d_AMD_durations$event[d_AMD_durations$censor=="yes"] <- 0
d_AMD_durations$processor_family_id <- as.factor(d_AMD_durations$processor_family_id)
attach(d_AMD_durations)
time <- duration
event <- event
group <- hw_ncores
X <- cbind(clock, transistors) #add back hw_ncores

#########################################################
############## FULL DURATION ANALYSIS ###############
#########################################################

d_AMD_durations <- read.csv("AMD-durations.csv", stringsAsFactors = FALSE)
d_INTEL_durations <- read.csv("intel-durations.csv", stringsAsFactors = FALSE)
d_durations <- rbind(d_AMD_durations, d_INTEL_durations)
d_durations$date <- mdy(d_durations$date)
d_durations <- filter(d_durations, censor=="yes" | censor=="no")
d_durations$event <- 2
d_durations$event[d_durations$censor=="yes"] <- 1
d_durations$end_time <- ymd("2014-12-31")
d_durations$time <- interval(d_durations$date, d_durations$end_time) %>%as.duration()
d_durations$time <-  as.numeric(d_durations$time/(60*60*24))
d_durations$time[d_durations$censor=="no"] <-
as.numeric(d_durations$duration[d_durations$censor=="no"])
d_final <- select(d_durations, manufacturer_id, processor_family_id,clock, hw_ncores, transistors,
die_size,event,time)
d_final$group <- as.factor(d_final$manufacturer_id)

##Plot Number of processors in each family
ggplot(d_final, aes(processor_family_id, fill=manufacturer_id)) + geom_bar() + theme_economist() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust=0.4),axis.title.x=element_blank()) +
  scale_y_continuous(labels=scales::comma, breaks=seq(1:10)) + ylab("Processor Family Counts") +
  coord_flip()
ggsave("imgProcessorFamilyCount.png", path="../LaTeX/img/", width=8.5, height=11, units="in")


#KMsurvivalSIMPLE

kmsurvival <- survfit(Surv(time,event)~group, data=d_final)
summary(kmsurvival)
plot(kmsurvival, xlab="Time", ylab="Survival Probability")
ggsurv(kmsurvival) + guides(linetype=F) + theme_economist() + geom_step(size=2)
ggsave("imgKMsimple.png", path="../LaTeX/img/", width=15, height=8.5, units="in")

imgggsurvplot <- ggsurvplot(
  kmsurvival,                 # survfit object with calculated statistics.
  data = d_final,           # data used to fit survival curves.
  risk.table = TRUE,       # show risk table.
  pval = TRUE,             # show p-value of log-rank test.
```

```
  conf.int = TRUE,         # show confidence intervals for
  # point estimates of survival curves.
  palette = c("#E7B800", "#2E9FDF"),
  xlim = c(0,5000),         # present narrower X axis, but not affect
  # survival estimates.
  xlab = "Time in days",   # customize X axis label.
  break.time.by = 730,     # break X axis in time intervals by 500.
  ggtheme = theme_economist(), # customize plot and risk table with a theme.
  risk.table.y.text.col = T,# colour risk table text annotations.
  risk.table.height = 0.25, # the height of the risk table
  risk.table.y.text = FALSE,# show bars instead of names in text annotations
  # in legend of risk table.
  ncensor.plot = TRUE,      # plot the number of censored subjects at time t
  ncensor.plot.height = 0.25,
  conf.int.style = "step",  # customize style of confidence intervals
  surv.median.line = "hv",  # add the median survival pointer.
  legend.labs =
    c("AMD", "Intel")    # change legend labels.
)
ggsave(file = "ggsurv.pdf", print(survp))
ggsave(file="imgKMsimplefull.png", print(imgggsurvplot), path="../LaTeX/img/", width=8.5,
height=11, units="in")

print(kmsurvival, print.rmean=TRUE)

ggsurvplot(kmsurvival,
           conf.int = TRUE,
           risk.table.col = "strata", # Change risk table color by groups
           ggtheme = theme_economist(), # Change ggplot2 theme
           palette = c("#E7B800", "#2E9FDF"),
           fun = "cumhaz", xlim = c(0,5000),
           break.time.by = 365)

## Cox Proportional hazard Univariate

coxph.uni <- coxph(Surv(time, event)~group, data=d_final)
summary(coxph.uni)
ggsurvplot(survfit(coxph.uni, data=d_final), color = "#2E9FDF",xlim = c(0,5000),break.time.by = 365,
           ggtheme = theme_economist())


##Multivariate Cox

coxph.trans <- coxph(Surv(time,event)~ transistors, data=d_final, method="breslow")
summary(coxph.trans)

coxph.core <- coxph(Surv(time,event)~ hw_ncores, data=d_final, method="breslow")
summary(coxph.core)

coxph.clock <- coxph(Surv(time,event)~ clock, data=d_final, method="breslow")
summary(coxph.clock)

coxph.transclock <- coxph(Surv(time,event)~ transistors*clock, data=d_final, method="breslow")
summary(coxph.transclock)

coxph.transcore <- coxph(Surv(time,event)~ transistors*hw_ncores, data=d_final, method="breslow")
```

```
summary(coxph.transcore)

coxph.coreclock <- coxph(Surv(time,event)~ hw_ncores*clock, data=d_final, method="breslow")
summary(coxph.coreclock)

coxph.3var <- coxph(Surv(time,event)~ clock + transistors + hw_ncores, data=d_final,
method="breslow")
summary(coxph.3var)
stargazer(coxph.3var, title="Cox Proportional Hazard Results", type="html")
stargazer(coxph.uni,coxph.core, coxph.clock,coxph.trans, coxph.coreclock, coxph.transclock,
coxph.transcore, coxph.3var)

## parametric single transistors

d_final <- filter(d_final, time!=0)
exponential <- survreg(Surv(time,event) ~ transistors, data=d_final, dist="exponential")
summary(exponential)
weibull <- survreg(Surv(time,event) ~ transistors, data=d_final, dist="weibull")
summary(weibull)
loglogistic <- survreg(Surv(time,event) ~ transistors, data=d_final, dist="loglogistic")
summary(loglogistic)
stargazer(exponential, weibull, loglogistic)

## parametric multivariable

exponential <- survreg(Surv(time,event) ~ transistors + hw_ncores + clock, data=d_final,
dist="exponential")
weibull <- survreg(Surv(time,event) ~ transistors + hw_ncores + clock, data=d_final, dist="weibull")
loglogistic <- survreg(Surv(time,event) ~ transistors + hw_ncores + clock, data=d_final,
dist="loglogistic")
stargazer(exponential, weibull, loglogistic)

##clock speed

exponential <- survreg(Surv(time,event) ~ clock, data=d_final, dist="exponential")
weibull <- survreg(Surv(time,event) ~ clock, data=d_final, dist="weibull")
loglogistic <- survreg(Surv(time,event) ~ clock, data=d_final, dist="loglogistic")
stargazer(exponential, weibull, loglogistic)

## parametric multivariable interaction

exponential <- survreg(Surv(time,event) ~ transistors * clock+ hw_ncores + clock, data=d_final,
dist="exponential")
weibull <- survreg(Surv(time,event) ~ transistors * clock + hw_ncores + clock, data=d_final,
dist="weibull")
loglogistic <- survreg(Surv(time,event) ~ transistors *clock,  + hw_ncores + clock, data=d_final,
dist="loglogistic")
stargazer(exponential, weibull, loglogistic)

##Bootstrap
library("boot")

d_boot <- na.omit(d_final)
data(d_boot)
boot.fun <- function(data) {
  surv <- survfit(Surv(time, event) ~ group, data = d_boot)
```

```
  out <- NULL
  st <- 1
  for (s in 1:length(surv$strata)) {
    inds <- st:(st + surv$strata[s]-1)
    md <- min(surv$time[inds[1-surv$surv[inds] >= 0.5]])
    st <- st + surv$strata[s]
    out <- c(out, md)
  }
  out
}
boot.case <- censboot(d_boot, boot.fun, R = 499, strata = d_boot$group)

# Now we will look at the same statistic using the conditional
# bootstrap and the weird bootstrap.  For the conditional bootstrap
# the survival distribution is stratified but the censoring
# distribution is not.

boot.s1 <- survfit(Surv(time, event) ~ group, data = d_boot)
boot.s2 <- survfit(Surv(time-0.001*event, 1-event) ~ 1, data = d_boot)
boot.cond <- censboot(d_boot, boot.fun, R = 499, strata = d_boot$group,
                      F.surv = boot.s1, G.surv = boot.s2, sim = "ordinary")




## ADDITIONAL SUMMARY STATS AND GRAPHS ##

ggplot(d_durations, aes(processor_family_id)) + geom_bar() + theme_economist() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust=0.4),axis.title.x=element_blank()) +
  scale_y_continuous(labels=scales::comma) + ylab("Processor Count") + coord_flip() +
  theme(text = element_text(size=20))

ggplot(d_durations, aes(reorder(id, -duration), duration, fill=manufacturer_id)) +
  geom_bar(stat='identity', position='dodge') + theme_economist() +
  theme(axis.title.x=element_blank(),axis.text.x=element_blank(), axis.title.y=element_blank()) +
  coord_flip()

ggplot(d_durations, aes(processor_family_id, time, fill=manufacturer_id)) +
  geom_bar(stat='identity', position='dodge') + theme_economist() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, vjust=0.4),axis.title.x=element_blank())
ggsave("imgProcessorFamilyDurations.png", path="../LaTeX/img/", width=15, height=8.5, units="in")

ggplot(d_durations, aes(duration, fill=manufacturer_id)) + geom_density(alpha=0.5) +
theme_economist()
ggsave("imgDurationDistributions.png", path="../LaTeX/img/", width=15, height=8.5, units="in")

aggregate(d_durations$duration, by=list(Category=d_durations$manufacturer_id), FUN=mean)
aggregate(d_durations$duration, by=list(Category=d_durations$manufacturer_id), FUN=median)
aggregate(d_durations$duration, by=list(Category=d_durations$manufacturer_id), FUN=max)

d_durations %>% filter(manufacturer_id == "AMD") %>% summary()
```

```
## Alternative Formulation of Kaplan Hazard Function

H.hat <- -log(kmsurvival$surv)
H.hat <- c(H.hat, tail(H.hat, 1))
h.sort.of <- kmsurvival$n.event / kmsurvival$n.risk
H.tilde <- cumsum(h.sort.of)
H.tilde <- c(H.tilde, tail(H.tilde, 1))

plot(c(kmsurvival$time, 250), H.hat, type="s")
points(c(kmsurvival$time, 250), H.tilde, lty=2, type="s")
legend("topleft", legend=c("H.hat","H.tilde"), lty=1:2)
```