Estimating the Rate of Web Page Updates

Sanasam Ranbir Singh
Indian Institute of Technology Madras
Department of Computer Science and Engineering
Chennai-36, India
ranbir@lantana.tenet.res.in

Abstract

Estimating the rate of Web page updates helps in improving the Web crawler's scheduling policy. But, most of the Web sources are autonomous and updated independently. Clients like Web crawlers are not aware of when and how often the sources change. Unlike other studies, the process of Web page updates is modeled as non-homogeneous Poisson process and focus on determining localized rate of updates. Then various rate estimators are discussed, showing experimentally how precise they are. This paper explores two classes of problems. Firstly the localized rate of updates is estimated by dividing the given sequence of independent and inconsistent update points into consistent windows. From various experimental comparisons, the proposed Weibull estimator outperforms Duane plot(another proposed estimator) and other estimators proposed by Cho et la. and Norman Matloff in 91.5%(90.6%) of the whole windows for synthetic(real Web) datasets. Secondly, the future update points are predicted based on most recent window and it is found that Weibull estimator has higher precision compared to other estimators.

1 Introduction

Most of the information available on the Web are autonomous and are updated independently of the clients. Estimating the rate at which Web pages are updated, is important for Web clients like Web crawlers, Web caching etc. to improve their performance. Since the sources are updated on their own, such a client does not know exactly when and how often the sources change. Web crawlers need to revisit the sources repeatedly to keep track of the changes and maintain the repository upto-date. Further, if a source is revisited before it is changed, then it will be a wastage of resources such as CPU time, file system and network bandwidth for both client and server. Such a visit is defined as a *premature* visit. For a client like a Web crawler with a collection of thousands of millions of such sources, it is important to schedule its visits optimally such that number of unobserved changes between the subsequent visits and the number of premature visits are minimum. Therefore, study of the changing behavior of web pages and estimating their changing rate can benefit in improving Web crawler's scheduling policy, Web page popularity measure, preprocessing the accessed schedules etc. This is the main motivation of this work. Another question that is attempted to answer in this paper is the localized rate of changes of the sources. For example, *At what rate was a Web page updated during the month of January last year? How does the rate of updates change during the month of January for the last five years?* This paper focuses mainly on estimating localized rate of updates rather than global update rates. Since the rate of updates of the sources may vary with time, localized estimation provides more detail, useful and accurate information.

Contrast to the previous studies [Cho and Garcia-Molina, 2003; Matloff, 2005], the process of Web page updates is modeled as *time-varying Poisson process*. It is because of the fact that sources are updated by its owner autonomously and independently whenever there are new developments and the rate of updates may vary with time. Therefore, it is more appropriate to assume the update process as a *time varying Poisson process* rather than *time homogeneous Poisson process*. The advantage of modeling the problem as *non-homogeneous Poisson process* is its wide range of flexibilities. It can be noted that *homogeneous Poisson process* is a special case of *non-homogeneous Poisson process*. A major difficulty in modeling the problem as non-homogeneous Poisson process is that it has infinitely many parameters. In particular, it is parameterized by its arrival-rate $\lambda(t)$, which varies with time. To simplify the problem, the attention is restricted to certain assumptions of $\lambda(t)$ i.e., *constant*, *increasing* or *decreasing*. Such assumption help in reducing the number of parameters. This assumption may not be correct over a long period of observations (say, over years of page update records), but reasonable over appropriate subintervals (say, over few weeks). Therefore, the whole observation space is divided into a sequence of windows of consistent behavior (see Section 4). The above assumption is reasonable as this paper is focussed more on the *instanteneous page update rates*.

1.1 Accessing the Web pages

One major problem in modeling the changing behavior of Web page is that, the complete information of the sequence of updates is not available. Web pages are accessed repeatedly through normal activities such as periodic crawling of web pages. It is only possible to determine, whether a page

had changed, but not possible to determine the number of changes between two subsequent accesses, which results in loss of information. The accuracy of the estimation of the rate of changes depends on how small the number of unobserved updates is.

Assuming that the Web pages are accessed periodically with a small interval τ, let $m(i)$ be the number of page updates between the $(i-1)^{th}$ and i^{th} accesses and $\{t_{(i,1)}, t_{(i,2)}, t_{(i,3)}, ..., t_{(i,m(i))}\}$ be the time sequence of corresponding updates. The changes can be observed in the following two ways:

-**Last date of change:** Some Web servers provide the information about when the page was last modified. It is not possible to obtain the complete update information between accesses. In such case, we only observe $t_{(i,m(i))}$ and $\{t_{(i,1)}, t_{(i,2)}, t_{(i,3)}, ..., t_{(i,m(i)-1)}\}$ are left unobserved. There is no way to obtain the unobserved information. Assuming that $m(i)$ is small, we ignore the unobserved updates and count only one update during $((i-1)\tau, i\tau]$ if changes occur and consider $t_{(i,m(i))}$ as the renewal point.

-**Existence of change:** Some Web servers do not provide such information *i.e., last date of change*. In such case, it is only possible to determine whether the page has changed between accesses comparing with the old image of the page. We can not even determine $t_{(i,m(i))}$. It could have changed any number of times anywhere between $(i-1)^{th}$ access and i^{th} access, $i > 0$. If changes occur, it is counted as one.

1.2 Formulating the Process

In the studies [Cho and Garcia-Molina, 2003; Matloff, 2005], the two classes of observing updates are handled separately. Unlike these studies, the observed information is preprocessed and a model is formulated to handle both the cases identically simplifying the problem. For the case of *last date of change*, the $t_{(k,m(k))}$ is considered as the update point of the k^{th} interval and the process of update time is formulated as $\{t(i) \mid t(i) = t_{(k,m(k))}, i = N(k\tau), i, k > 0\}$, where $N(t)$ is the number of updates during the period $(0, t]$.

For the case of *existence of change*, the $t_{(k,m(k))}$ is not available. Assuming that sources are accessed periodically with an optimal interval, we can approximate $t(i)$, say the middle point or the end point of the period or a point determined from an appropriate distribution. Here, the $t(i)$ is approximated as the middle point and the process of update time is formulated as $\{t(i) \mid t(i) = (k-1)\tau + \tau/2, i = N(k\tau), i, k > 0\}$, where $N(t)$ is the number of observed updates during the period $(0, t]$.

Thus, the time sequence $\{t(i) : i = 0, 1, 2, ...\}$ represents the sequence of the update points for both the cases satisfying the properties $t(0) = 0$ and $t(i) < t(i+1)$, where $t(i)$ is the time of the i^{th} update. Since the client does not know exactly when and how often the Web page updates, the sequence $\{t(i) : i = 0, 1, 2, ...\}$ is an independent and identically distributed random process, which is often modeled as a Poisson process in several studies [Brewington and Cybenko, 2000; Cho and Garcia-Molina, 2003; Matloff, 2005]. This study also assumes the model as a Poisson process, but as a non-homogeneous Poisson process. One of the advantages of modeling the process as a non-homogeneous Poisson process includes the ability to determine the instantaneous update rate i.e., $\lambda(t)$. In summary, this paper makes the following contributions:

-handles the *last date of change* and *existence of change* cases identically.

-models the system as a *non-homogeneous Poisson process*.

-proposes two $\lambda(t)$ estimators namely Weibull estimator and Duane plot estimator and compares the estimates using various datasets with the estimators proposed in existing literatures.

The rest of the paper is organized as follows. Section 2 discusses non-homogeneous Poisson distribution. In Section 3, we discuss different estimators of rate of updates. In Section 4, we divide the whole observation space into windows of consistent behavior. Experimental verifications are discussed in Section 5. Then we conclude the paper in Section 6.

2 Non-homogeneous Poisson Process

A non-homogeneous Poisson process over $[0, \infty]$ is determined by a rate function $\lambda(t) \geq 0$, known as *intensity density* of the process (see Chapter 6, Trivedi [Trivedi, 1982]). Let $N(T)$ be the number of updates during the time period $(0, T]$. Then, the *intensity function* of the process, i.e. mean number of updates during the period $(0, T]$ is defined as follows.

$$\mu(T) = E(N(T)) = \int_0^T \lambda(t)dt, \quad \text{for } \int_0^\infty \lambda(t)dt = \infty$$

Now, the probability that the number of updates during the period $(0, T]$ equals to k, is given by

$$Pr[N(T) = k] = \frac{[\mu(T)]^k}{k!} e^{-\mu(T)}$$

for $k = 0, 1, 2, ...$ The Time homogeneous Poisson process is a special case, where $\lambda(t) = \lambda$ for all time instances $t \geq 0$. Now, the probability of $N(T)$ becoming zero is equal to $Pr[N(T) = 0] = e^{-\mu(T)}$. Thus we can write its cumulative distribution function (*cdf*) as $F(T) = 1 - e^{-\mu(T)}$ and its probability density function (*pdf*) as $f(T) = \mu'(T)e^{-\mu(T)} = \lambda(t)e^{-\int_0^T \lambda(t)dt}$. Using the expression of $F(t)$ and $f(t)$ above, the instantaneous $\lambda(t)$ can be expressed as

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

3 Estimating $\lambda(t)$

As stated in Section 1, the whole observation space is divided into a sequence of windows of consistent behavior. A window is a sequence of update points satisfying the property that $t(i) > t(i-1)$, where $t(i)$ is the time of the i^{th} update. Since the sequence of update points are independent and identically distributed and satisfies the memoryless property, each window can be processed independently to determine the $\lambda(t)$.

Definition 1 *A window is said to be consistent, iff either one of the followings is true.*
1. $\lambda(t(i)) < \lambda(t(j)), \forall i < j, j > 0$ *i.e., increased*
2. $\lambda(t(i)) > \lambda(t(j)), \forall i < j, j > 0$ *i.e., decreased*
3. $\lambda(t(i)) = \lambda(t(j)), \forall i < j, j > 0$ *i.e., constant*

3.1 Intuitive Measure: $N(t)/t$

If $N(t)$ is the number of updates during the period $(0, t]$, the rate of updates at the time t is $\lambda(t) = N(t)/t$ and instantaneous *mean time between updates (MTBU)* at time t, $MTBU(t) = t/N(t)$. This estimation is suitable when the window satisfies the third consistency property i.e., the rate of update is *constant*. It is the biased estimation of $\lambda(t)$ in a homogeneous Poisson process [Cho and Garcia-Molina, 2003].

3.2 Estimation using Weibull Distribution

If the rate of update is either *increasing* or *decreasing* or *constant*, the Weibull distribution is a good mechanism to model the process. The two parameter Weibull *pdf* is given by

$$f(t) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta-1} e^{-(t/\eta)^{\beta}}, \tag{2}$$

where β is known as *shape parameter* and η is known as *scale parameter* and $\beta, \eta > 0$ (see Chapter 2, Patrick [O'Connor, 2002], [Lieblein, 1955]). Then its *cdf* is given by $F(t) = 1 - e^{-(t/\eta)^{\beta}}$. Substituting equation 2 and $F(t)$ in equation 1, we get

$$\lambda(t) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta-1} \tag{3}$$

From the study [Lieblein, 1955], we have (1) if $\beta = 1$, then $\lambda(t) = 1/\eta$, a constant (*the process reduces to the homogeneous Poisson process*), (2) if $\beta < 1$, then $\lambda(t)$ decreases as t increases and (3) if $\beta > 1$, then $\lambda(t)$ increases as t increases. A window is always in one of the above three states. Therefore, Weibull's distribution is obviously a good choice of estimating $\lambda(t)$.

In [Tsokos, 1995], the author found the relationship between $MTBU(t)$ and $\lambda(t)$ as follows.

$$MTBU(t_i) = \begin{cases} = \frac{1}{\lambda(t_i)} & \text{for } \beta = 1 \\ > \frac{1}{\lambda(t_i)} & \text{for } \beta < 1 \\ < \frac{1}{\lambda(t_i)} & \text{for } \beta > 1 \end{cases}$$

Thus, $1/\lambda(t)$ defines the bounds of $MTBU(t)$. Now our task is to estimate the parameters β and η. We estimate these two parameters using *maximum likelihood estimator* as given below. The conditional *pdf* of the i^{th} event given that the $(i-1)^{th}$ event occurred at t_{i-1} is given by $f(t_i \mid t_{i-1}) = \frac{\beta}{\eta}\left(\frac{t_i}{\eta}\right)^{\beta-1} e^{-\frac{1}{\eta^{\beta}}(t_i^{\beta} - t_{i-1}^{\beta})}, t_{i-1} < t_i$. Now, the likelihood function can be defined as

$$L(\beta, \eta) = \prod_{i=1}^{n} f(t_i \mid t_{i-1}) = \left(\frac{1}{\eta}\right)^{n\beta} \beta^n e^{-(\frac{t_n}{\eta})^{\beta}} \prod_{i=1}^{n} t_i^{\beta-1}$$

Taking log on both sides we get

$$\log L(\beta, \eta) = n\log(1/\eta^{\beta}) + n\log\beta - (\frac{t_n}{\eta})^{\beta} + (\beta-1)\sum_{i=1}^{n} t_i$$

Assuming $c = 1/\eta^{\beta}$, we can write

$$\log L(\beta, c) = n\log(c) + n\log\beta - ct_n^{\beta} + (\beta-1)\sum_{i=1}^{n} t_i$$

Taking partial derivative w.r.t. c and equating to zero, we get

$$\hat{\eta} = \frac{t_n}{n^{1/\beta}}. \tag{4}$$

Again, taking partial derivative w.r.t. β and equating to zero, we get

$$\hat{\beta} = \frac{n}{n\ln t_n - \sum_{i=1}^{n}\ln t_i} = \frac{n}{\sum_{i=1}^{n}\ln\frac{t_n}{t_i}} \tag{5}$$

Lemma 1 *$\hat{\beta}$ is biased for small value of n and equal to $\frac{n}{n-2}\beta$*

Proof As reported in [Calabria *et al.*, 1988], $2n\beta/\hat{\beta} \sim Z$, where Z is a chi-square random variable with $2(n-1)$ degree of freedom. Then we can write $\hat{\beta} = \frac{2n\beta}{Z} = \frac{2n\beta}{\Gamma(n-1)2^{n-1}} t^{n-2}e^{-t/2}$. Now, the expectation of $\hat{\beta}$ can be derived as $E[\hat{\beta}] = \frac{2n\beta}{\Gamma(n-1)2^{n-1}}\int_0^{\infty} t^{n-3}e^{-t/2}dt = \frac{2n\beta}{(n-2)\Gamma(n-2)2^{n-1}}\Gamma(n-2)2^{n-2} = \frac{n}{n-2}\beta$. If n is small, then $\hat{\beta}$ is biased. □

Now, Lemma 1 suggests correcting the ML by a factor of $(n-2)/n$ and thus obtain

$$\tilde{\beta} = \frac{n-2}{n}\hat{\beta} = \frac{n-2}{\sum_{i=1}^{n}\ln\frac{t_n}{t_i}}.$$

We can easily prove that $E[\tilde{\beta}]$ is unbiased i.e., $E[\tilde{\beta}] = E[\frac{n-2}{n}\hat{\beta}] = \frac{n-2}{n}E[\hat{\beta}] = \beta$. Thus we get $\tilde{\eta} = \frac{t_n}{n^{1/\beta}}$. Substituting the value of $\tilde{\beta}$ and $\tilde{\eta}$ in equation 3, we determine the value of $\tilde{\lambda}(t) = n\tilde{\beta}t^{\tilde{\beta}-1}/t_n^{\tilde{\beta}}$. In [Calabria *et al.*, 1988], it is proved that $\tilde{\lambda}(t)$ is less biased i.e., $E[\tilde{\lambda}(t)] = \lambda(t)(n-2)/(n-3)$. Again, we correct $\tilde{\lambda}(t)$ by a factor $(n-3)/(n-2)$ to get an unbiased estimation and thus obtain

$$\check{\lambda}(t) = \frac{n-3}{n-2}\tilde{\lambda}(t) = \frac{(n-3)n}{n-2}\tilde{\beta}t^{\tilde{\beta}-1}/t_n^{\tilde{\beta}}.$$

It can easily be proved that $E[\check{\lambda}(t)]$ is unbiased i.e., $E[\check{\lambda}] = E[\frac{n-3}{n-2}\tilde{\lambda(t)}] = \frac{n-3}{n-2}E[\tilde{\lambda}(t)] = \lambda(t)$. We use this estimator in our experiment. Another unbiased estimate of $\lambda(t)$ is $\bar{\lambda}(t) = (n-3)\hat{\beta}t^{\hat{\beta}-1}/t_n^{\hat{\beta}}$, which is obtained by correcting $\hat{\lambda}(t)$ by a factor of $(n-3)/2$ [Calabria *et al.*, 1988]. Once we get windows of consistent behavior (see Section 4), we can thus estimate $\lambda(t)$.

3.3 Estimation using Duane Plots

In this Subsection, we discuss another $\lambda(t)$ estimator using Duane plot [Duane, 1964].

Property 1 *Let $MTBU(t_i) = \frac{t_i}{i}, i > 0$ be the cumulative mean time between updates at i^{th} update. If the sequence of $MTBU(t_i)$ is either increasing or decreasing or constant as i increases and if we plot a graph between $MTBU(t_i)$ along y-axis and t_i along x-axis on* log-log *graph paper, the points tend to line up following a straight line. This plot is known as Duane Plot.*

The slope of the straight line passing through points is known as *Duane plot slope*. From the straight line, we can write $\log MTBU(t) = \log \alpha + \gamma \log t$, where γ is the *Duane plot slope* and α is the intercept when $t = 1$. Equating $MTBU$ to its expected value, we get $MTBU(t) = \alpha t^{\gamma}$. Similarly, cumulative $\lambda(t)$ can be written as $\lambda(t) = \frac{1}{\alpha} t^{-\gamma}$. Now, the slope γ can be calculated as

$$\gamma = \frac{\log MTBU(t_i) - \log MTBU(t_j)}{\log t_i - \log t_j}. \quad (6)$$

As we consider the $\lambda(t)$ estimation only within the consistent windows, the Duane plot can be applied to estimate the $\lambda(t)$. This estimation works fine as long as both the points $(MTBU(t_i), t_i)$ and $(MTBU(t_j), t_j)$ are on the best straight line through the points. But, it is not the case in reality. Therefore, the straight line which is closest to all the observed points is considered and the γ is the slope of the obtained straight line. The simple linear regression is applied to find the best straight line closest to all the observed points.

3.4 Estimation using Linear Regression

If we consider a consistent window with either *constant* or *increasing* or *decreasing* $\lambda(t)$, linear regression is an effective mechanism to estimate $\lambda(t)$. Under such conditions, the relationship between $\lambda(t)$ and t can be defined by a straight line. Linear regression defines a straight line closest to the data points. We can define the equation of the closest straight line as $y' = a + bx$, where a and b are regression coefficients and defined as $a = \overline{y} - b\overline{x}$ and $b = \frac{\sum (x_i - \overline{x})(y_i - \overline{y})}{\sum (x_i - \overline{x})^2}$. We apply simple linear regression to get the best straight line closest to all the observed points in the *Duane plot*. If we consider $x = \log t_i$ and $y = \log MTBU_c(t_i)$, then b is equal to the slope of the *Duane plots* γ and a is equal to the intercept α.

3.5 Measuring error

An error at a point i is defined as the difference between the observed value and predicted value at that point i i.e., $\bigtriangledown(i) = f_{observed}(i) - f_{predicted}(i)$. The error in measuring $\lambda(t)$ is minimum if error in measuring $MTBU(t)$ is minimum and vice versa. Again, overall error in $MTBU(t)$ is minimum when $\sum_{i=1}^{N(t)} \bigtriangledown(i) = \sum_{i=1}^{N(t)} \big(MTBU(t(i)) - (t(i) - t(i-1))\big)$ is minimum. Thus, $\sum_{i=1}^{N(t)} \bigtriangledown(i)$ can be used as a measure to compare different estimations of $\lambda(t)$. The $\lambda(t)$ with the minimum $\sum_{i=1}^{N(t)} \bigtriangledown(i)$ is considered as the best estimate. But, in the case of linear regression mechanism $\sum_i \bigtriangledown(i) = 0$. To avoid this problem, we can use mean of square error $MSE = \sum_{i=1}^{n} \bigtriangledown(i)^2/n$ as a measure to compare different estimators.

4 Generating Consistent Window

In reality, most of the Web page update period sequence is inconsistent. It is found to be a sequence of *constant*, *increasing* and *decreasing* over subsequent period of times in any order. In this Section, a procedure to generate the consistent windows (as define in Definition 1) out of a sequence of update points is discussed. First the localized $\lambda(t)$ is defined. If W be the set of update points, then the average number of update points within the window W can be defined as $\mu(W) = \int_W \lambda(x)dx < \infty$, where $\lambda(x)$ is locally integrable. Now the localized $\lambda(x)$ can be defined as $\lambda(x) = \lim_{W \to x} \frac{Pr(N(W)=1)}{|W|}$, where $N(W)$ denotes the number of updates within W and x is an update point. Thus, it is necessary for the window W to be consistent so as to be able to predict $Pr(N(W) = 1)$.

Now, the total observation space is divided into windows W_i of consistent $MTBU$ i.e. either *constant* or *increasing* or *decreasing* in nature. the length of a window $\mid W_i(k) \mid$ is defined by $(t_{j+k} - t_j), j > 0$, where $W_i(k)$ starts just after j^{th} update and k is the number of update points within the window. The accuracy of the estimation depends on the nature of the update points within the window. So, *how do we decide the window size?* The important factor in deciding the window size i.e., k is the number of consistent points within the window, not $\mid W_i(k) \mid$.

Consistent window: The initial window consists of the last (first) three update points and extended backward (forward). A window is always in one of the three states, either *constant* or *increasing* or *decreasing* state. The window extends if the next point at t_j is consistent with the state of the current window. We often do not find consistent windows of large size. To increase the size of the windows, we allow few inconsistent update points within the window with a limit on the amount of discrepancies. Even if an update point t_j is not exactly consistent with the current state, we consider the t_j as a consistent point if the amount discrepancy is below some *threshold value*, otherwise we create the next window.

DataSet	min	max	mean	Std. De.	#Pages
Real	4	57	27	11.46	26972
Synthetic	4	201	103	45.85	299324

Table 1: Characteristics of the datasets. (min:Minimum number of updates in a page, max: maximum number of updates in a page, mean: mean number of updates and Std. De.: Standard Deviation)

5 Experimental Evaluation

The comparisons of different estimators are based on both synthetic datasets (*collected from the UCLA WebArchive project data available at http://webarchive.cs.ucla.edu/*) and real Web datasets. Synthetic datasets consist of 3,00,000 pages' records and each record contains 200 access units. The real Web datasets were collected locally. The 27034 number of frequently access Web pages were identified from our local proxy server log file. These pages were crawled every day for two months (12^{th} *April, 2006 to* 12^{th} *June, 2006*) to keep track of the changes.

Table1 shows the characteristics of the two datasets. The pages having less than four update points during the observation period are ignored. The proposed estimators are compared with the estimators proposed in [Cho and Garcia-Molina, 2003] and [Matloff, 2005]. The λ proposed by Cho

DataSet	Estimator	Total	Const	Incr	Decr
Synthetic	λ_{cho}	3.849	0.639	13.094	5.19
	λ_{norman}	3.925	0.745	13.174	5.248
	Weibull	2.363	0.2488	7.955	3.274
	Duane	2.956	0.339	10.307	4.059
Real	λ_{cho}	2.379	0.642	5.9	4.23
	λ_{norman}	2.471	0.753	5.979	4.296
	Weibull	1.363	0.245	4.069	2.513
	Duane	1.802	0.348	4.633	3.359

Table 2: Average MSE for different estimators over different window types. Windows were generated with a *Threshold* value of 3.

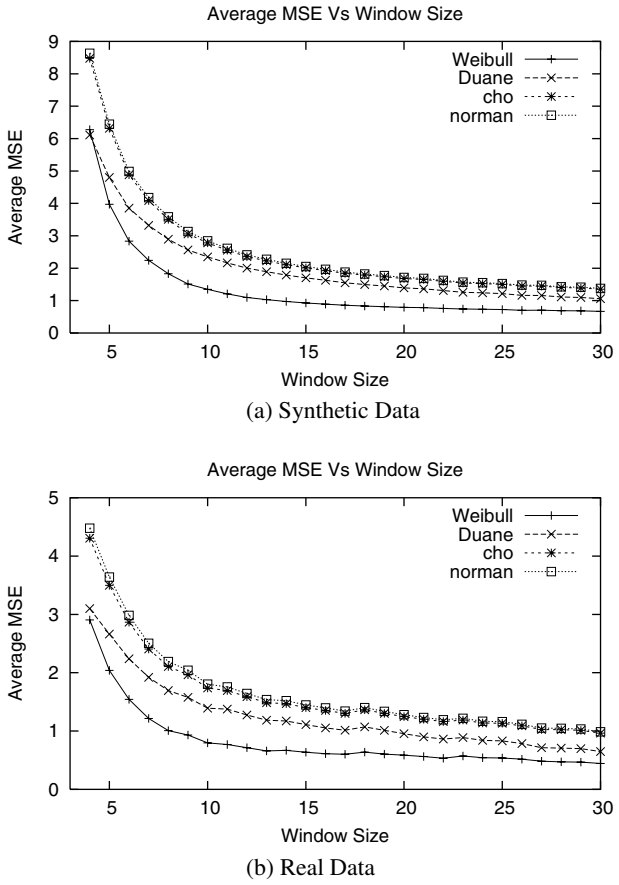
et al. can be stated as $\lambda_{cho} = -\frac{1}{\tau}log(\frac{n-X+0.5}{n+0.5})$, where n is the number of intervals of τ and X is the number of intervals with at least one update point. Again, the λ proposed by Matloff N. can be stated as $\lambda_{norman} = -\frac{1}{\tau}log(1-\frac{M_n}{n})$, where n is the number of intervals of τ and M_n is the number of intervals with at least one unobserved update point. To get the best result out of the available information, τ is set to 1.

The experimental evaluations consists of two parts. In the first part, the localized $\lambda(t)$ is estimated using the estimators discussed in Section 3 from the sequence of data points in the observation space. The sequence of consistent windows are generated using the procedure discussed in Section 4. The consistent windows with a minimum size of four update points are considered. If the size of the window is less than four, then one half is merged with the left window and another half with the right window in the sequence introducing some discrepancies. For each window, the $\lambda(t)$ is estimated at every update point t using different estimators. For simplicity, $MTBU(t)$ is approximated as $MTBU(t) \sim 1/\lambda(t)$ and then regenerate the whole observation space again using the estimated $MTBU(t)$. The difference between actual update point and estimated update point is considered as an error. The errors in every estimated update points are recorded and the average MSE for each window is determined to compare the accuracy of different estimators.

		Rest	cho	nor	Wei	Dua
Syn	cho	0.002%		100%	1.2%	2.3%
	nor	0%	0%		1%	1.4%
	Wei	91.5%	98.8%	99%		91.5%
	Dua	8.5%	97.6%	98.5%	8.5%	
Real	cho	0.002%		100%	1.2%	1.6%
	nor	0%	0%		1%	0.92%
	Wei	90.6%	98.8%	99%		90.6%
	Dua	9.4%	98.4%	99.1%	9.4%	

Table 3: Estimator in the row outperforms estimator in the column. Windows were generated with a *Threshold* value of 3. (Syn :Synthetic, Wei:Weibull, Dua: Duane, cho:λ_{cho}, nor:λ_{norman})

Table 2 compares the average MSE of different estimators in terms of different window classes. The proposed Weibull



(a) Synthetic Data

(b) Real Data

Figure 1: Window size versus MSE between different estimators. Windows were generated with a *Threshold* value of 3.

estimator outperforms other estimators for all the cases. For the case of *constant state* windows, all these four estimators performs equally well with very small average MSE. But, Weibull estimator outperforms the rest with significantly smaller average MSE for both increasing and decreasing state windows. In Table 3, the percentage of the number of windows out of the whole windows, in which row estimator outperforms the column estimator, is shown. It shows that Weibull estimator outperforms the rest in 91.5% of the whole windows for synthetic datasets and in 90.6% of the whole windows for real Web datasets. Whereas the previous estimators λ_{cho} and λ_{norman} outperforms the rest almost in 0% for both synthetic and real datasets. Again we compare different estimators in terms of the average MSE for different window sizes as shown in Figure 1. It clearly shows that error decreases as we increase the size of window. For the larger size windows, all estimators perform equally well (still Weibull outperforms by smaller extents), but for the smaller size windows, Weibull outperforms the other estimators by almost double.

The main motivation behind investigating localized $\lambda(t)$, rather than global λ is to use the piecewise information to improve the prediction of future update points. For example, *the*

trend in $\lambda(t)$ during the month of January for last five years can be used (or combined with recent trend) to predict future $\lambda(t)$ of next January. Due to not having enough datasets to make use of such information, such study is left as the future work.

However in the second part of the experiment, *the effectiveness of different estimators* is investigated by predicting the future update points from the past information running a simulation program. Simulation starts with first consistent window in the observation space and based on this window, the next update point is predicted. Then the window is extended to include the newly predicted point and again predict the next update point and so on. As pointed out in Section 1, an efficient estimator should be able to detect good amount of update points with minimum amount of resource consumption. Each premature visit is a wastage of resources such as CPU time, file system, network bandwidth etc. (see Section 1). To compare different estimators , a measure of *precision* is defined and stated as follows.

$$precision = \frac{\#Observed\ Update\ points}{\#Total\ Visits}$$

The precision of an estimator represents the probability of detecting an update per visit. It is reasonable to conclude that estimator with the highest precision is the best estimator, with the highest probability of detecting update points. There are 744385 number of update points in real dataset and 30931883 in synthetic dataset. Table 4 shows the comparisons among the four estimators in terms of the percentage of updates detected, premature visits, precision and number of visits. Though the λ_{cho} and λ_{norman} estimators detect the highest number of the update points, it is very expensive, almost 42% (36%) of the total visits over real (synthetic) dataset are premature visits. Moreover, the number of visits scheduled by both λ_{cho} and λ_{norman} estimators are very large compared to the number of visits scheduled by Weibull estimator. Whereas Weibull estimator has the highest precision i.e. the probability of detecting updates per visit, almost 0.70 (0.76) for real (synthetic) datasets and minimum number of visits. If the number of visits is normalized, Weibull estimator detects the highest number of update points, next followed by Duane, then λ_{norman} and λ_{cho}. If the number of visits is normalized to the number of visits of Weibull estimator, both the estimotors λ_{cho} and λ_{norman} detect only around 62% (60%) and Duane estimator detects around 68% (66%) for synthetic (real) datasets. Note that the best policy is the one that detects the highest number of update points with minimum resource consumption, in other word highest number of update points detected from the same number of visits. Therefore, Weibul estimator outperforms other estimators.

		%Obs	%Prem	%Prec	%Visits
Syn	λ_{cho}	94.9%	36%	0.64	154%
	λ_{norman}	95.2%	36.3	0.637	153%
	Weibull	74.1%	23.8%	0.762	100%
	Duane	91.2%	29.5%	0.7	129%
Real	λ_{cho}	95.1%	41.5%	0.58	155%
	λ_{norman}	95.5%	42%	0.58	157%
	Weibull	72.3%	30.9%	0.69	100%
	Duane	88.6%	36.4%	0.64	133%

Table 4: Comparison of % of observed updates, % of premature visits, precision, % of visits over Weibull estimator. (Syn: Synthetic, Obs: Observed update points, Prem: Premature visits, Prec: Precision). Windows were generated with a *Threshold* value of 3

6 Conclusions

This paper models the process of web page updates as non-homogeneous Poisson process. In a scenario with inconsistent rate of updates, piecewise estimation provides more detail and useful information compared to global estimation. In the study, the whole observation space is divided into independent consistent windows to estimate localized rate of updates. Two update rate estimation mechanisms namely *Weibull* and *Duane plot* are proposed. Tests on both synthetic and real datasets confirm that Weibull's estimator outperforms *Duane plot* and both the existing estimators λ_{cho} and λ_{norman} while estimating localized rate of updates. While investigating the effectiveness of using these estimators to predict future update points, Weibull estimator has highest probability of detecting update points.

References

[Brewington and Cybenko, 2000] Brian E. Brewington and George Cybenko. How dynamic is the web? In *Proceeding of WWW2000*, March 2000.

[Calabria *et al.*, 1988] R. Calabria, M. Guida, and G. Pulcini. Some modified maximum likelihood estimator for the weibull process. *Reliability Engineering and System Safety*, 23, 1988.

[Cho and Garcia-Molina, 2003] Junghoo Cho and Hector Garcia-Molina. Estimating frequency of change. *ACM Transaction on Internet Technology*, 3:256–290, August 2003.

[Duane, 1964] J. T. Duane. Learning curve approach to reliability monitoring. *IEEE Transactions on Aerospace*, 2, April 1964.

[Lieblein, 1955] Julius Lieblein. On moments of order statistics from the weibull distribution. *Annals of Mathematical Statistics*, 26, June 1955.

[Matloff, 2005] Norman Matloff. Estimation of internet file-access/modification rates from indirect data. *ACM Transactions on Modeling and Computer Simulation*, 15:233–253, July 2005.

[O'Connor, 2002] Patrick D. T. O'Connor. *Practical Reliability Engineering*. John Wiley, forth edition, 2002.

[Trivedi, 1982] Kishor S. Trivedi. *Probability and Statistics with Reliability, Queuing and Computer Science Application*. Prentice Hall, 1982.

[Tsokos, 1995] Chris P. Tsokos. *Reliability growth: nonhomogeneous poisson process*. CRC Press, 1995.