Link quality estimation for arbitrary packet sizes over wireless links using packet reception events

Martin Jacobsson | Christian Rohner

1Department of Biomedical Engineering and Health Systems, KTH Royal Institute of Technology, Huddinge, Sweden
2Department of Information Technology, Uppsala University, Uppsala, Sweden

Correspondence
Martin Jacobsson, Department of Biomedical Engineering and Health Systems, KTH Royal Institute of Technology, Hälsovägen 11C, Huddinge 141 57, Sweden.
Email: martin.jacobsson@sth.kth.se

Summary
The packet error rate of wireless links is known to increase with the length of packets. Yet, packet length is rarely taken into account in protocols and algorithms estimating the packet error rate. Still, it is an important factor that higher layer protocols need to be aware of. In this article, we systematically measure the relationship between packet length and packet error rate over a wide range of wireless links and technologies. On the basis of our measurements, we propose a simple empirical model that can capture this behavior. Using this model, multiple methods are proposed that can estimate the packet error rate for any packet length by sampling the link. We consider methods based on hello packets with controlled packet lengths as well as data packets, where the transmitted packet lengths cannot be controlled. We investigate the accuracy of the different estimation methods in various situations and show how they are able to predict the delivery ratio for different packet sizes.

KEYWORDS
bit error rate, hello protocol, link quality estimation, packet error rate

1 | INTRODUCTION

In most wireless networks, it is important to understand and measure the quality of the links. This information is needed in order to decide the best path through a network, the right level of forward error correction (FEC) coding, what transmission power to use, which transmission rate to use, which channel to use in a multichannel network, and much more. These are areas that are crucial to the performance of wireless systems, such as wireless sensor networks (WSN), ad hoc networks, mesh networks, and cellular networks. The process of identifying a link’s quality is known as link quality estimation (LQE) and has been a frequent topic of study.\(^1\)\(^-\)\(^8\)

For most higher layer functions, including routing, the packet delivery ratio (PDR) is the single most important information about a link’s quality. It is known that the PDR is dependent on the packet length.\(^9\)\(^,\)\(^10\) Until recently, the detailed empirical work on this topic was surprisingly thin. Most work concerning packet length effects are based on independent bit errors. The bit error rate (BER) is then translated into a packet error rate (PER) as follows:

\[
PER = 1 - (1 - BER)^l,
\]

where \(l\) is the packet length in bits, including all headers.
Recent publications have shown how the packet length affects the PDR with a systematic and extensive empirical study. The conclusion from this study is that Equation (1) does not describe this effect in sufficient detail. Furthermore, it is very common that LQEs ignore the packet length, and this affects the performance of higher layer protocols.

The extensive work on radio channel modeling often only explains the behavior either at packet level or at bit level, but rarely how they relate, with Gilbert being a notable exception. Furthermore, they tend to be too complex to be usable in the design of practical LQE methods. Either their models are too computationally complex to be carried out in a frequent manner or they have many model parameters that cannot easily be parameterized from typical available statistics about real links. Even the simplest models, such as the two-state Markov model from Gilbert cannot be parameterized without complex regression methods or maximum likelihood estimations (MLEs). Furthermore, they might not work well when the available information about a link is limited.

At the same time, many LQE methods are based on packet reception events, such as standard hello packet protocols. Hello packets are typically small packets periodically emitted by all nodes in a network, and their reception rate is used to estimate the PDR of each link. Instead of hello packets, some protocols use regular data packets or feedback from the automatic repeat request (ARQ) function to estimate the PDR. In either case, the packet length of interest might be different from the packet lengths used to sample the link. The length of data packets is determined by the applications and cannot be controlled. For hello packets, it is beneficial to keep them as short as possible. A shorter hello packet creates less overhead and also consumes less power to transmit, which is extra important for battery-powered nodes. To base an LQE on these types of packets, we need to understand the relationship between PDR and packet size. This article is based on packet reception events. However, if LQEs are based on other methods, such as observing chip errors, FEC corrections, signal to noise ratio, or soft decoding values, it is still necessary to account for the packet length effects, meaning that the work of this article still is highly relevant.

In this article, we extend our previous work in several ways. The frequency of packet losses indicates the quality of a link, but it is necessary to accumulate several reception results over a period of time before being able to reliably draw conclusions about a link’s true quality. Hence, we will study what effect different amount of reception events have on the accuracy for different packet length-aware LQE methods. We will also develop efficient ways to implement the LQE methods that works well also with limited information about a link as well as how to use data packets, where the packet sizes varies due to the applications being used. Furthermore, we take a more rigid mathematical approach in this article and also make sure that the developed methods can be implemented in practice, meaning that the per-packet computation is kept at a minimum and that we do not require the storage of large amounts of data. For the sake of completeness, we will first summarize the key results from our previous work.

The remainder of this article is organized as follows. Section 2 introduces our measurement setup, and Section 3 lists the main results from these measurements and introduces the LQE methods from our previous work. In Section 4, we look at the impact of reducing the amount of used packet reception events on the methods, and in Section 5, we develop and test estimation methods based on regular data packets. Section 6 contains related work, and Section 7 concludes this article.

## 2 Measurement Setup

To study packet reception-based LQEs, we created experiments to measure the packet loss for several different packet sizes. We focused on a single wireless link where one device sends packets to another device. Packets were sent continuously and with different sizes in random order (either laptops or wireless sensor devices) by the transmitting device. The receiving device did not transmit any packets during the experiments. Packets were considered received if and only if the receiver could correctly receive the packet using its wireless chip. Three different wireless technologies were tested: IEEE 802.11 (WiFi) at 2.4 GHz, IEEE 802.15.4 (used by ZigBee) at 2.4 GHz, and DASH7 (ISO/IEC 18000-7) at 433 MHz.

For WiFi, we used broadcast packets to avoid RTS/CTS and acknowledgements. We modified the driver to select broadcast data rates of 2 Mbps (default) or rates up to 54 Mbps. The transmission power was fixed to the maximum allowed value. We used standard off-the-shelf WiFi cards. For most experiments, we used a 3Com OfficeConnect 108Mb 11g PC Card together with Linux and the Madwifi driver 0.9.4 (now replaced by the ath5k driver). For our 54 Mbps IEEE 802.11g measurements, we used a WiFi card based on the Atheros AR2425 chip as transmitter and a Netgear WNA1100, which uses an Atheros AR9271 chip, as a receiver. Both transmitter and receiver ran Linux with the open source ath5k and ath9k_htc drivers, respectively.

The IEEE 802.15.4 measurements were done with t-mote sky, which is based on the CC2420 wireless chip from Texas Instruments. IEEE 802.15.4 is based on direct sequence spread spectrum (DSSS) and has a data rate of 250 kbps. The
standard on-board antenna of the t-mote sky was used (inverted F). For the software, we used TinyOS 2.1, with its built-in clear channel assessment (CCA) function. Also here, no retransmissions, acknowledgements, or other packets were sent.

The DASH7 measurements were done with Wizzimotes, which are based on the CC430F5137 System-on-chip solution from Texas Instruments. DASH7 uses frequency shift keying (FSK) with a channel size of 216 kHz, a data rate of 55.56 kbps, and PN9 encoding. We used a standard quarter-wave whip antenna. The software came from Wizzilab. Also here, no acknowledgements, error correction, or network functionalities were used. However, no CCA function was used.

The measurements were done in different static scenarios with different conditions and different distances. Most scenarios were set up in a typical office environment with several other interferers (both WiFi and others) and a few in a typical rural home environment with minimal interference. With the exception of the outdoor scenario and the measurements in the anechoic chamber, all links were non-line-of-sight (NLOS).

In each experiment, we generated 200 packets of each length using a Poisson arrival process during about 3.5 minutes. That meant that we utilized the wireless channel less than 5% of the time. Only one sender was used at a time, leading to no internal interference. For each scenario, we conducted 10 experiments in short succession with which we could calculate the 95% confidence interval. We sent packets with 15 or 16 different packet sizes, depending on the technology used. In total, 30 000 or 32 000 packets were generated over a time of about 35 minutes per scenario. Due to different transmission rates and packet sizes, transmission times differed slightly.

The selection of a Poisson arrival process for the experiment packets is because we want random observations of the link in our experiments. This also means that we look at the long-term behavior of the link and not changes from one packet to the next. Furthermore, all results of this article hold for any traffic pattern that a real user of the link may have when considered over a longer period and as long as there is no interplay between the link quality and the traffic pattern.

Both the DASH7 and the IEEE 802.15.4 links have a tendency to change over time. To counter this effect, sometimes, we had to measure over a longer period and extract the most stable subperiod and discard the rest. The extracted subperiods always contained the same amount of packets as the other experiments (2000 packets of each packet length).

The reported packet sizes exclude the physical layer (PHY) and medium access control (MAC) headers but included IP and higher layer headers. Both the used headers and the maximum allowed packet length differ between technologies. The maximum allowed packet length was 1500 bytes for WiFi compared with 114 bytes for IEEE 802.15.4 and 240 bytes for DASH7. The actual maximum is actually higher for the latter two if minimal headers can be used.

### 3 Measurement Results and Analysis

Figure 1 shows the measurement results with each line denoting one link. Figure 1A shows eight different static links with different conditions and different distances using WiFi with a data rate of 2 Mbps. Six links were set up in a typical office environment with several other interferers (both WiFi and others) and two in a typical rural home environment with minimal interference (links 2 and 6). The links were all non-line-of-sight (NLOS). The remaining subfigures show similar measurements in different environments and with different configurations and technologies. The dashed lines in Figure 1B show measurements done in an anechoic chamber.

#### 3.1 Packet loss analysis

From the results in Figure 1, we can see that the PDR indeed decreases with increasing packet length, but that there is a large variation among the different scenarios. Sometimes, as demonstrated by link 7 in Figure 1, this difference can be quite dramatic; going from 46% PDR for 50 bytes packets down to only 6.5% for 1500 bytes packets. We can also see that it is not possible to accurately estimate the packet loss for all packet lengths if we only generate hello packets of one packet length, since we cannot know the slope of the curve. Links 5, 6, and 7 of Figure 1 demonstrate this case as they all have roughly the same delivery ratio for 50 bytes packets but radically different PDRs for long packets. This is the reason to extend Equation (1) with a length-independent delivery component \( p_0 \) as follows:

\[
    p_L = p_0 \cdot p^L,
\]

where \( p_L \) denotes the packet delivery probability for a packet of length \( L \) and \( p^L \) is the length-dependent delivery component with \( p \) as its parameter. For a list of these notations and other mathematical notations used throughout this article, see Table 1. More results and details about these results can be found in our previous work.11
To motivate this model, we may reason as follows. There are both packet length-independent factors and packet length-dependent factors that affect the chance of packet loss. For instance, we may assume that a successful packet reception involves two events: (a) the receiver detects the packet transmission and is able to tune onto the transmitter (this event is independent of the packet length) and (b) the receiver can correctly receive all bytes in the packet (this event is length dependent). However, these two events are of course not independent either, but as we will show later, this empirical model is accurate enough for our purposes.

Under the assumption that byte errors are independent, you may think of $p$ as the payload byte delivery probability. We prefer to write the equation based on PDR and bytes, instead of PER, BER, and bits, since one byte is the smallest transmittable unit in all the wireless protocols in this study and most other wireless protocols. Note that we can translate between the two forms, since $1 - p$ is linked to the BER.
### Table 1
Mathematical notation frequently used throughout this article

<table>
<thead>
<tr>
<th>Notation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Λ</td>
<td>Packet length of the large hello packets (bytes)</td>
</tr>
<tr>
<td>λ</td>
<td>Packet length of the small hello packets (bytes)</td>
</tr>
<tr>
<td>L</td>
<td>Packet length of the sought PDR (bytes)</td>
</tr>
<tr>
<td>px</td>
<td>PDR of packet length x</td>
</tr>
<tr>
<td>̂px</td>
<td>An estimate of px</td>
</tr>
<tr>
<td>p</td>
<td>Packet-length dependent parameter of model in Equation (2)</td>
</tr>
<tr>
<td>p0</td>
<td>Packet-length independent parameter of model in Equation (2)</td>
</tr>
<tr>
<td>M</td>
<td>Maximum packet length (bytes)</td>
</tr>
<tr>
<td>h</td>
<td>Header length (bytes)</td>
</tr>
<tr>
<td>n</td>
<td>The number of hello packet reception events used in an estimation</td>
</tr>
<tr>
<td>S, B</td>
<td>The set of used data packet reception events used in an estimation divided into two sets (small and big)</td>
</tr>
<tr>
<td>d</td>
<td>The packet length threshold used to divide packets into S or B (bytes)</td>
</tr>
<tr>
<td>PDRX</td>
<td>The measured packet delivery ratio for the set X of packet reception events</td>
</tr>
</tbody>
</table>

Abbreviation: PDR, packet delivery ratio.

We also note that Equation (2) is inline with the channel model by Gilbert.\(^\text{12}\) Gilbert states that for a link following the Gilbert channel model, the relationship between PDR and packet length is as follows:

\[
p_L = ap^x + (1 - a)q^x,
\]

(3)

where \(\alpha\), \(p\), and \(q\) are model parameters (Gilbert uses \(A\), \(J\), and \(L\)). By choosing \(q = 0\), we obtain Equation (2). Hence, the Gilbert model is a generalization of our model. However, we will show that Equation (2) is practically good enough for LQEs for real systems and that wireless communication links follow the model of Equation (2) quite well regardless of whether the link follows Gilbert’s two states channel model or not.

### 3.2 Estimating the packet delivery probability

How can we find the PDR of a particular packet length when we know the PDR of another length? In this section, we will develop some methods for this problem. We base these on the assumption that PDRs for one or two packet sizes are known and that we need the PDR of a third. We will use \(\lambda\) and \(\Lambda\) for the packet sizes with the known PDRs and \(L\) for the packet size of the PDR that we want to know. Without loss of generality, we assume \(\lambda < \Lambda\). We use \(p_x\) to denote the PDR of packets with length \(x\) bytes. Hence, the problem is to find \(p_L\) given \(p_{\lambda}\) and/or \(p_{\Lambda}\).

In the first method, which is frequently used by higher layer protocols, such as routing protocols, no packet length effect is assumed, i.e., that \(p_L = p_x\) for all valid sizes \(x\). We call this method the flat method, and it is enough to know only one PDR in this method (e.g., \(p_{\Lambda}\)).

In a second method, we still assume only one known PDR value but use the model of Equation (1). Assume that the header length \(h\) is excluded from the packet lengths and that bit errors indeed are independent. Then, we can obtain \(p = (1 - BER)^8\) as the byte delivery ratio. Converting from bits to bytes and from error rates to delivery rates in Equation (1), we obtain the following equation system for the two packet sizes of interest (namely, \(\Lambda\) and \(L\)):

\[
\begin{align*}
p_{\Lambda} &= p^{h+\Lambda} \\
p_L &= p^{h+L}
\end{align*}
\]

which solves to:

\[
p_L = p^{(h+L)/(h+\Lambda)}.
\]

(4)

We can now estimate \(p_L\) if we know \(p_{\Lambda}\). We call this the BER header method. However, as observed in our previous work,\(^\text{11}\) the most accurate estimations are not obtained with \(h\) set to the actual header length in Equation (4), but with a significantly larger \(h\). This is because we are no longer just predicting the header but also the rest of the length-independent reception event. However, in this method, we only have the PDR for one packet size, which means that \(h\) must be a fixed parameter for all links for a given technology. The best approach is to use the \(h\) that minimizes the estimation errors for the average behavior of typical links. We call this the BER optimal method.
If a network designer chooses to collect the PDRs for two different packet sizes, such as using two different hello packet sizes, a better estimation can be made. In this method, which we call the two-packet lengths method (two sizes), we can calculate both \( p \) and \( p_0 \) in Equation (2) by the following equation system:

\[
\begin{align*}
 p_\Lambda &= p_0 p^\Lambda, \\
 p_\Lambda &= p_0 p^\Lambda, \\
 p &= (p_\Lambda/p_\lambda)^{1/(\Lambda - \lambda)}, \\
 p_0 &= p_\lambda p^{-\lambda},
\end{align*}
\]

which solves to:

\[
\begin{align*}
 p &= (p_\Lambda/p_\lambda)^{1/(\Lambda - \lambda)}, \\
 p_0 &= p_\lambda p^{-\lambda},
\end{align*}
\]  

(5)

where \( p_\Lambda \) is the delivery probability for hello packets of length \( \Lambda \), \( p_\lambda \) is the delivery probability for length \( \lambda \), and \( \Lambda \neq \lambda \). The solution can be inserted into Equation (2) to calculate an accurate estimation for the PDR of packet length \( L \) as follows:

\[
p_L = p_\lambda \left( \frac{p_\Lambda}{p_\lambda} \right)^{\frac{1}{\Lambda - \lambda}}.
\]

(6)

We can of course also imagine networks where more than two packet sizes are used. Perhaps hello packets contain a varied amount of data whose length cannot be controlled (e.g., a neighbor list). In such situations, we may use a method based on least squares or MLE. However, we should keep in mind that these methods already are complex for per-packet calculations, and, furthermore, more packet sizes lead to even more complex the computations. Another area where we may be forced to consider many different packet sizes is when using data packets. Section 5 presents solutions for that.

Nevertheless, we define and test two methods using more than two packet sizes but only as theoretical benchmarks. The first one, we refer to as the least squares (L-S) method and is based on Equation (2) and ordinary least squares of the logarithm of all the \( p_x \) values between 50 and 500 (six PDR values). This method has an additional problem when applied in real LQEs, we also need to deal with the situation when \( p_x \) values are estimated to be 0, which is not possible due to the logarithm.

The second method is called Gilbert, because it is based on Gilbert's model,\(^\text{12}\) which is found in Equation (3). We parameterize this model based on all the \( p_x \) values between 50 and 500. To find the optimal parameters, we had to use two standard search algorithms, namely, the L-BFGS-B optimization algorithm\(^\text{22}\) and nonlinear least squares based on the Trust Region Reflective algorithm. Both algorithms are only suitable for offline analysis and hence are not practical for real-time per-packet parameterization. For each link, we chose the search result with the lowest absolute error for the six PDR values after removing search results with nonvalid parameters.

### 3.3 Comparison of methods

Here, we numerically evaluate the different packet loss estimation methods by including all the 59 links that we measured, including the ones from Figure 1. Figure 1 does not include all our links, since some scenarios and links were omitted due to space limitations and similarity with other links. Especially indoor 2 also include five more links from a rural residential location in addition to the ones shown in Figure 1. Since the different technologies have different maximum packet sizes, we also used different values for \( \lambda \) and \( \Lambda \) for the different technologies. Table 2 lists all the used parameters for the different technologies.

We tested the six different estimation methods and calculated the absolute error compared with the actual PDR of all packet sizes. Then, the mean absolute errors (MAE) were calculated for all packet sizes and links per technology. That is, for each scenario \( S \) and estimation method \( e \), we calculated the MAE as follows:

### Table 2

<table>
<thead>
<tr>
<th>Tech.</th>
<th>( \lambda )</th>
<th>( \Lambda )</th>
<th>Max length (M)</th>
<th>Header length (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>WiFi</td>
<td>50</td>
<td>500</td>
<td>1500</td>
<td>~63</td>
</tr>
<tr>
<td>802.15.4</td>
<td>4</td>
<td>44</td>
<td>114</td>
<td>17</td>
</tr>
<tr>
<td>DASH7</td>
<td>5</td>
<td>80</td>
<td>240</td>
<td>10</td>
</tr>
</tbody>
</table>

*Note. All values are in bytes. The header size is on top of the reported packet length. The header size for WiFi is an estimate based on the timings of the preamble. It also varies a bit with data rate.*
The optimal $h$ so that the methods can be used for every packet reception event, even on resource-constrained devices. It is not enough to only have simple models with good enough accuracy. For an LQE to be of practical use, it also needs to be easy to compute in an efficient way. In this section, we focus on keeping the required computation and storage down without hardware support for floating-point arithmetics, we note that it is possible to find fixed point arithmetic implementations of all the methods. For the BER optimal method and Equation (4), we want to make the exponent a small integer by choosing $\Lambda$ in clever ways. This would simplify the formula to a few simple multiplications. For instance, if we choose $\Lambda = 103$, this would yield an exponent of $\frac{h+L}{h+\Lambda} = 2$, which would radically simplify Equation (4) and only require a single multiplication.

3.4 Per-packet handling

It is not enough to only have simple models with good enough accuracy. For an LQE to be of practical use, it also needs to be easy to compute in an efficient way. In this section, we focus on keeping the required computation and storage down so that the methods can be used for every packet reception event, even on resource-constrained devices.

The first to consider is how to obtain and store the packet reception events and how to estimate $p_x$ and $p_\Lambda$. Obviously, we need the computation per-packet reception event to be small and at the same time reduce the memory allocation for the collected data, since LQE mechanisms often will be handled in low level operating system drivers, in the firmware of network interface cards, or on simple embedded microcontrollers. Furthermore, it is important that an online algorithm is used, meaning that when a new packet reception event occurs, the computation is incremental and based on previous calculations instead of being forced to recalculate almost everything. This can keep the per-packet computation very low. All the presented methods allow for this if the $p_x$ and/or $p_\Lambda$ can also be obtained without recalculation every time.

The estimation of $p_x$ and/or $p_\Lambda$ can easily be obtained from some of the last packet reception events by a simple moving average. Such an average can be calculated in an online fashion without a full summation for each new packet reception event. In the remainder of this article, we will assume such a solution. However, this solution requires the last $n$ packet receptions to be stored at the receiver. For accuracy reason, $n$ needs to be large enough, which we will further discuss in Section 4. However, there are alternatives that removes the storage requirement, such as the use of an exponentially weighted moving average (EWMA), which can be considered in many LQE implementations.

To simplify even further for resource-constrained systems, such as embedded systems based on micro-controllers without hardware support for floating-point arithmetics, we note that it is possible to find fixed point arithmetic implementations of all the methods. For the BER optimal method and Equation (4), we want to make the exponent a small integer by choosing $\Lambda$ in clever ways. This would simplify the formula to a few simple multiplications. For instance, if we are considering $L = 1500$, we may choose a hello packet size of $\Lambda = 103$. This would yield an exponent of $\frac{h+L}{h+\Lambda} = 2$, which would radically simplify Equation (4) and only require a single multiplication.

In a similar way, we may select $\lambda = 48$ and $\Lambda = 411$ for the two-packet lengths method and Equation (6). In this case, the exponent becomes 4, and we can estimate the PDR for 1500 bytes packets with just one division and three multiplications.

### Abbreviations

- BER, bit error rate
- MAE, mean absolute error

### Table 3

MAE (in percent) of all packet sizes for every link for all three technologies

<table>
<thead>
<tr>
<th>Technology</th>
<th>Flat $\lambda$</th>
<th>BER Header $\lambda$</th>
<th>BER Optimal $\lambda$</th>
<th>2 Sizes $\lambda &amp; \Lambda$</th>
<th>L-S $\lambda - \Lambda$</th>
<th>Gilbert $\lambda - \Lambda$</th>
</tr>
</thead>
<tbody>
<tr>
<td>WiFi</td>
<td>8.6</td>
<td>4.2</td>
<td>26.7</td>
<td>6.7</td>
<td>2.9</td>
<td>1.7</td>
</tr>
<tr>
<td>802.15.4</td>
<td>12.9</td>
<td>5.4</td>
<td>6.5</td>
<td>2.7</td>
<td>1.2</td>
<td>1.1</td>
</tr>
<tr>
<td>DASH7</td>
<td>25.2</td>
<td>11.3</td>
<td>6.9</td>
<td>3.7</td>
<td>2.8</td>
<td>2.2</td>
</tr>
</tbody>
</table>

$\text{MAE}(S, e) = \frac{1}{|S|} \sum_{s \in S} \frac{1}{|L|} \sum_{L \in L_s} |e_L(p_x^s, p_\Lambda^s) - p_L^s|,$

where $S$ is the scenario expressed as a set of links, $L$ is the set of packet sizes, $p_x^s$ is the measured PDR of packet length $x$ on link $s$, and $e_L()$ is the estimation method trying to estimate $p_L$, such as Equation (4) or (6).

The results are shown in Table 3 in units of a percent. For the BER optimal method, we calculated and used the optimal $h$ for each individual technology but over all the measured links. The optimal $h$ was found by a scalar minimization algorithm of the MAE, and we found $h = 1294$ for WiFi, $h = 38$ for 802.15.4, and $h = 20$ for DASH7 to be the optimal values.

Table 3 shows that errors are roughly declining when going from left to right, ie, going from simple methods towards more complex and better methods. For methods only using one single $p_x$ value as input, we measured using either $p_x$ or $p_\Lambda$ as input data. Going from the flat method and the BER header method with the actual header length to the BER optimal method with a better $h$ parameter, we can immediately see a significant error reduction. Also, using longer hello packets improves the accuracy. When using the two-packet lengths method (2 Sizes), the errors reduce even further. However, the benefit of using the least squares method or the Gilbert model with search algorithm parameterization is small, even though all the six PDR values between $\lambda$ and $\Lambda$ are being used.

In the remainder of this article, we will focus on only WiFi. However, it must be pointed out that the same approaches apply to the other technologies as well and that the results are similar.
4 | HELLO PACKET SAMPLING

With a hello protocol, we sample a link's quality by injecting packets on a periodic basis. The quality of the LQE is dependent on the availability of fresh hello packet reception events. For a new link, there is only a few number of reception events, and this will affect the accuracy of the estimate in the beginning. As more hello packets are injected and used, the accuracy becomes better if the link is stable. For unstable links, we do not want to depend on too old reception events, but we discard them after a while. This means that the number of packets that can be used is reduced due to freshness requirements. To still get a good estimate of dynamic links, we need to increase the amount of hello packets per second.

The methods defined in Section 3.2 will work well when the in-data (ie, the estimates of $p_2$ and $p_L$) is very good and in all the evaluations so far, we have had a total of 2000 packets sent per-packet length on stable links. However, what happens to the accuracy if we only use a limited number of packets? To answer this, we will model the variables $p_2$, $p_L$, and $L$ stochastically using errors-in-variables models and assume that the links follow our loss model in Equation (2).

In the following, we let $n$ denote the total available packet reception events. If two packet sizes are used, we assume equal amount of reception events per-packet size (ie, $n/2$). We define $\hat{p}_L$ and $\hat{p}_\Lambda$ as the measured values of $p_L$ and $p_\Lambda$ respectively. Since packet receptions can be considered as i.i.d. Bernoulli events, we have that $\hat{p}_L \sim B(n, p_L)/n$ and $\hat{p}_\Lambda \sim B(n, p_\Lambda)/n$, where $B(n, p)$ is the Binomial distribution with the parameters $n$ and $p$.

4.1 | Bias and consistency

The first questions are whether our estimation methods are unbiased and consistent. It can be shown that all our estimators (flat, BER hdr., BER opt., and two-packet lengths) are consistent. Hence, if we increase the amount of packet reception events (ie, let $n \to \infty$), then the estimator will get better and converge (in probability) to the correct value of $p_L$. However, only the flat estimation method is unbiased, meaning that for small values of $n$, the other methods have a small bias. They all tend to overestimate more often than they underestimate. This follows from the fact that all the methods use the exponential function $x^a$, which is strictly convex for $a > 1$. Due to Jensen’s inequality, we can conclude that the estimators are upwardly biased except for the trivial cases of $\hat{p}_L = 1$ or $\hat{p}_\Lambda = 0$. Nevertheless, it is possible to calculate the amount of bias. Below is the bias formula for the BER method with any value of $h$ and packet sizes such that $L > \Lambda$.

$$
\text{Bias}_{p_L}[\hat{p}_\Lambda] = E[(\hat{p}_\Lambda)^a] - p_L
$$

$$
E[(\hat{p}_\Lambda)^a] = \sum_{k=0}^{n} \binom{n}{k} a \binom{n}{k} p_\Lambda^k(1-p_\Lambda)^{n-k},
$$

(7)

where $a = (h + L)/(h + \Lambda)$ and $a \geq 1$. This bias is strongly dependent on $n$, $p_\Lambda$, and the exponent $a$. Unfortunately, it is difficult to calculate exactly, but the bias reduces when $n \to \infty$ (the estimator is consistent) or $a \to 1$ (it becomes the flat method) as well as when $p_\Lambda \to 0$ or $p_\Lambda \to 1$.

Since we can quantify the bias, it may be tempting to compensate for this in the estimator to achieve an unbiased estimator, and this can be done using numerical calculations of Equation (7) or by finding good approximations. However, in general, subtracting the bias from an estimator does not necessarily lead to smaller estimation errors and that also holds here. Numerical calculations that we did indicate that the MAE of the estimator does not reduce and even increases if we introduce bias compensation. Hence, we discourage from the use of bias compensation for these estimates.

4.2 | Two-packet lengths method

In the two-packet lengths method, we need to divide the packet reception events into the two different packet sizes. This means more variance since we have half as many events per in-data estimate ($\hat{p}_L$ and $\hat{p}_2$). Furthermore, we have two variates, both with Binomial distributions as before, namely, $\hat{p}_2 \sim B(n/2, p_2)/(n/2)$ and $\hat{p}_L \sim B(n/2, p_L)/(n/2)$. Add to this this the fact that the estimator is upwardly biased. Hence, we need to quantify the errors this estimate has when $n$ is small.

First, due to insufficient sampling, we may end up with situations when the standard two-packet lengths method (Equation 6) cannot be used or will obviously estimate wrong due to too big variance in the input. There are in particular two cases that we need to take special care of. Namely, if either $\hat{p}_2 = 0$ or $\hat{p}_L = 0$, and if $\hat{p}_2 < \hat{p}_L$. To deal with this, we redefine the two-packet sizes estimation function as follows:...
In the remaining of this article, we will use this estimation function for the two-packet lengths method. It can be shown that this modified estimation function is consistent, since the in-data errors will reduce when \( n \to \infty \) and only the last case will be used.

Using a Monte Carlo simulation approach, we numerically calculated the absolute error of the two-packet lengths method as follows:

\[
\varepsilon_L(\hat{p}_\lambda, \hat{p}_\Lambda) = \begin{cases} 
0 & \text{if } \hat{p}_\lambda = 0 \text{ or } \hat{p}_\Lambda = 0 \\
(\hat{p}_\lambda + \hat{p}_\Lambda)/2 & \text{if } \hat{p}_\lambda < \hat{p}_\Lambda \\
\hat{p}_\lambda (\frac{p_\lambda}{p_\Lambda}) \frac{L}{\lambda} & \text{otherwise}
\end{cases}
\]

and calculated the MAE by repeatedly using all the links from all 44 WiFi scenarios. Note that we use \( \varepsilon_L(p_\lambda, p_\Lambda) \) rather than the actual \( p_L \) as benchmark. This is because we here only want to study the errors introduced by insufficient sampling. Later, we will look at the complete errors.

As before, we used \( \lambda = 50, \Lambda = 500, \) and \( L = 1500, \) and took \( p_{50} \) and \( p_{500} \) from all measured WiFi links. For each link, we simulated packet reception events for the used hello packet sizes according to a binomial distribution divided by the number of simulated packets. The resulting estimates \( \hat{p}_{50} \) and/or \( \hat{p}_{500} \) were then fed to the different estimation methods, and the MAE was calculated. Each link was repeated enough times for the MAE results to become statistically significant (99% confidence interval within ±0.01%). Figure 2A shows the results. The max curve shows the link yielding the worst MAE; i.e., the MAE of all links fall within the gray area.

From Figure 2A, we can see that the MAE quickly decreases as we use more packets. At \( n = 200 \) packets (100 packets per length), we get an MAE of 6%. At \( n = 400 \) packets, this number drops to 4.4%. These MAE values should be compared with the errors introduced by the estimation methods, which are shown in Table 3. It is clear that the sampling errors are significantly bigger than the method errors when considering hello protocols with small \( n \). These results are important as the amount of available packet reception events might be limited for many links. Maybe it is too expensive to often generate hello packets, or a link shows a dynamic behavior meaning that information from old packets must be discarded as they are outdated.

In Figure 2B, we further study the errors due to insufficient sampling for the two-packet lengths method by showing the CDF for \( n = 20, 200, \) and \( 2000. \) Due to the fact that some links have more room for variance and thereby may give raise to bigger estimation errors, we have divided all our WiFi links into three equal-sized sets (Weak, Average, and Good). This was done by sorting the links according to their average PDR for all packet sizes and equally divided them into the three sets. The figure shows the CDF curves for each link set and each of the three \( n \) values. We can see that the links with average link quality have larger errors than both the good and the weak ones. This is not surprising given that the average links have more room for variance of \( p_L. \) For the good and weak links, \( p_L \) is bounded by 100% and 0%, respectively. Hence, a large portion of the problems with estimation accuracy stems from links with in-between quality. It is here we will see the biggest errors, and this effect is aggravated when \( n \) is small.

**FIGURE 2** Errors for the two-packet lengths estimation method (50&500) due to insufficient hello packet sampling.
4.3 | Comparison of methods when $n$ is small

We know that insufficient sampling leads to bigger errors for the two-packet lengths method, but how does this compare with the other methods? Figure 3 combines the sampling errors and the method errors. It shows the errors compared with the actual $p_{1500}$ values of each link, ie, $|e_L(\hat{p_x}, \hat{p_\lambda}) - p_L|$. The figure shows the MAE from all links in all WiFi scenarios for the three different estimation methods. The results of Figure 3 reflects the actual errors to be expected in practice given $n$ hello packet samplings for each of the methods.

In Figure 3, we can see how the two-packet lengths method reduces its estimation error as the number of packets are increased as we showed previously. It is also apparent that the methods based on a single packet length do improve the estimations quicker. Since we know that the errors due to the model are smaller for the two-packet length method, the only explanation to this is that the model is worse in dealing with errors in the input estimates due to the insufficient sampling. Hence, if it is necessary to keep the number of hello packets down (eg, dynamic links), a single packet length method is preferred. When the amount of packets increase, the two-packet lengths method becomes more accurate, outperforming the BER optimal method using 50 bytes packets already at 40 packet receptions. The BER optimal method using 500 bytes packets is outperformed from around 700 packet receptions. However, we need to remember that the results for the BER optimal method are optimistic as we are using the best possible $h$ for the tested links and real systems will never be able to do that without extensive measurements.

In Figure 3, we also included the result of using extensive search algorithms based on Gilbert’s model (ie, Equation 3). While it is more accurate with good input values, we can see that this method, if it was computationally practical, would still converge slowly as $n$ increases; ie, using Gilbert’s model causes problems when dealing with limited information. The least-squares method could not be implemented as it cannot deal well with $p_\lambda = 0$ values, which are very common when $n$ is small.

Since the errors in Figure 3 include both model errors and errors due to limited sampling, we can use it to predict what the estimation errors will be in real LQE implementations. As such, it is not always the case that the better model of the two-packet length method always is better. In the case of few samplings, such as when dealing with a dynamic link, single packet lengths methods will achieve lower errors.

4.4 | Impact of the used hello packet sizes

Another aspect affecting the estimation errors is the used hello packet sizes, ie, $\Lambda$ and $\lambda$. In this section, we try different hello packet sizes in combination with different amount of hello packet samplings to find out how the estimation errors are affected. Figure 4 shows the results for all estimation methods. In all curves, we look at the mean absolute estimation error of all packet sizes.

Figure 4A shows the single packet size methods, such as the flat method, the BER method using the actual header length, and the BER method using the optimal $h$ value. The solid curves show the MAE when $n$ is large, while the dashed curves show the same, but with a hello packet sampling of only $n = 100$ (50 small packets and 50 large packets). Since we look at the errors of estimating all packet sizes, a hello packet length close to the middle (ie, around $M/2 = 750$) does indeed give us the best results. However, we can also see how the accuracy improves as we start using larger hello packets.
The effect on estimation errors when estimating all packet sizes using different values of $\Lambda$ and $\lambda$ up until $\Lambda = 800$. From there onwards, there is no improvement or even a degradation. The degradation can be explained by the fact that it becomes harder to correctly estimate small packet sizes using large hello packets.

In Figure 4B, we show the results for the two-packet lengths method. As we now have two different hello packet sizes, we get many more curves. However, we omitted many of them to reduce clutter in the graph. Each curve represents a different size of the smaller hello packet (i.e., $\Lambda$). The $x$-axis is the difference between the large and the small hello packet size (i.e., $\lambda - \Lambda$) in bytes. For a large $\Lambda$, the difference cannot be too large, because then the larger hello packet would be larger than the maximum packet size. That is why the curve for $\Lambda = 900$ bytes stops at 600 bytes on the $x$-axis. Also here, we show the results for when $n$ is large and when we use a limited hello packet sampling of $n = 100$.

From Figure 4B, we can see that the best choice of $\Lambda$ is 500 bytes, and this is the best choice also among the omitted curves. However, any $\Lambda$ in the range 300 to 700 bytes are also quite good and give similar estimation errors. Furthermore, the two used packet sizes must not be too similar in size. The results suggest that $\lambda - \Lambda$ should be at least 200 bytes, but even larger differences are better. However, when $n$ is large, we can see a slight decrease in estimation accuracy as the packet length difference is increased beyond around 900 bytes. The same cannot be seen when $n$ is small, instead a larger difference between the two used hello packet sizes is almost always better.

5 USING DATA PACKETS

Regular data packets can also be used to estimate the link quality instead of dedicated hello packets. In order to do so, it must be known which packets are lost, which can be achieved by sequence numbers, for example. Unfortunately, the lengths of the data packets are determined by the applications, and they can have any length. While this can give more complete estimations, computations get more complex, in particular as the number of different packet sizes increases. In this section, we will develop computationally efficient estimation methods that use regular data packets as the in-data. As previously, an online algorithm solution is needed that can reduce the amount of computation per packet.

If the packet length mix generated by the applications over a link is dominated by either one or two particular packet lengths, this problem becomes trivial; we can use any of the previous methods and just discard packets of other lengths. However, if this is not the case, we need new methods. It is also possible that the packet length mix changes over time, making the problem even harder. To deal with this, we may try standard statistical methods, such as nonlinear least squares or MLE. However, both these methods are very computationally intensive as they require a large amount of calculations. It is also hard to find good online algorithms for these. In addition, they are not necessarily unbiased and consistent. For instance, taking the logarithm of the PDR values and using a linear least squares method will minimize in log-space only and will not generate the optimal parameters of the original error model. Hence, we will develop dedicated methods for estimating using data packets instead. We first estimate the model parameters $p$ and $p_0$ and then look at the final accuracy of estimating $p_{1500}$. 

![FIGURE 4](image-url)
5.1 Estimating $p$

A simple way to find an estimate is to divide the packets into two \textit{bins}, one for small packets and one for big packets. Let us assume the division is made at $d = M/2$, where $M$ is the maximum packet size (ie, $M = 1500$ for WiFi). We define two disjoint sets of packets $S = \{i | L_i \leq M/2 \}$ and $B = \{i | L_i > M/2 \}$, one for each bin. $L_i$ represents the packet length of the $i$th packet reception event, and if we have $n$ events, then $|S| + |B| = n$. In the remaining of the article, we make the assumption that $|S| \approx |B|$. Furthermore, let $L$ be a stochastic variable representing the packet length distribution. Then we can define $L_S$ to be a stochastic variable representing the packet length distribution in the bin of small packets and likewise $L_B$ will be the distribution for the large bin. For instance, if $L$ is a uniform distribution between 1 and $M$, then $L_S$ is uniform between 1 and $M/2$ and $L_B$ between $M/2 + 1$ and $M$. For any distribution $L$, we can calculate the average packet length and the total PDR per bin as follows:

\begin{align*}
\bar{S} &= \frac{1}{|S|} \sum_{i \in S} L_i \\
PDR_S &= \frac{1}{|S|} \sum_{i \in S} y_i \\
\bar{B} &= \frac{1}{|B|} \sum_{i \in B} L_i \\
PDR_B &= \frac{1}{|B|} \sum_{i \in B} y_i,
\end{align*}

(8)

where $y_i$ is 1 if packet $i$ was correctly received or 0 if it was lost or corrupted.

The two points $\bar{S}, PDR_S$ and $\bar{B}, PDR_B$ allow us to reduce the problem to the two-packet lengths method for calculating $p$ and $p_0$, ie, using Equation (5). However, if we assume that packet lengths are uniformly distributed between 1 and $M$, that packet losses exactly follow our error model in Equation (2) and that there are large amounts of packets in both bins, we will still not predict the correct values; ie, $PDR_S$ does not predict $p_S$ and the same for $PDR_B$.

**Theorem 1.** The estimators $PDR_S$ and $PDR_B$ of $p_S$ and $p_B$ respectively, are both asymptotically upwardly biased.

**Proof.** We prove this theorem using Jensen’s inequality. Our model is exponential, which is a convex function. Hence, for any type of packet length distribution, it follows from Jensen’s inequality that:

$$E[PDR_B] = E[p_0 p_1] = p_0 E[p_1] \geq p_0 p M^{3/2}$$

in the large bin. The results are the same for the small bin.

**Corollary 1.** $PDR_S$ and $PDR_B$ are not consistent estimators of $p_S$ and $p_B$ (does not converge in probability).

**Proof.** See Appendix A.

To further explain this, assume that packet lengths are uniformly distributed (between 1 and $M$). This means that the packet lengths $L_i$ in each of the sets $S$ and $B$ also are uniformly distributed. Hence, we get $\bar{S}$ and $\bar{B}$ exactly in the middle of their respective bins.

The consequence of Theorem 1 and Corollary 1 is that Equation (8) overestimates the PDR, and this overestimation is systematic and independent of the sample size used for the prediction, that is, the estimator is not consistent. The only exception would be if the packet size distribution is strongly bimodal, with one mode in each bin. Hence, $PDR_S \geq p_S$ and $PDR_B \geq p_B$. However, it is also important to quantify this inequality since a small difference may be ignored. We can calculate the expected values for each of the four values of Equation (8) as follows:

\begin{align*}
E[\bar{B}] &= \frac{3}{4} M + \frac{1}{2} \\
E[\bar{S}] &= \frac{1}{4} M + \frac{1}{2} \\
\lim_{|B| \to \infty} E[PDR_B] &= E[p_0 p_1] = \frac{p_0}{M/2} \sum_{k=M/2+1}^{M} p_k = \frac{p_0}{M/2} \left( \sum_{k=0}^{M} p_k - \frac{M}{2} \sum_{k=0}^{M/2} p_k \right) \\
&= \frac{p_0}{M/2} \left( \frac{1-p}{1-p} - \frac{1-p^{M/2+1}}{1-p} \right) = p_0 p 2 M \left( \frac{1-p^{M/2}}{1-p} \right) p^{M/2}
\end{align*}

(9)
FIGURE 5 Bias for the PDR$_S$ and PDR$_B$ estimators for $M = 1500$ and uniformly distributed packet sizes distribution for large samples sizes (Equations 11 and 12). We assume $p_0 = 1$ and a cut-off between $S$ and $B$ at $M/2$

$$\lim_{|S| \to \infty} E[\text{PDR}_S] = p_0 p \frac{2}{M} \left( \frac{1 - p^{M/2}}{1 - p} \right),$$  

when $p < 1$. The asymptotic bias is as follows:

$$\lim_{|B| \to \infty} \text{Bias}[\text{PDR}_B] = \lim_{|B| \to \infty} E[\text{PDR}_B] - p_0 |B| = p_0 \frac{2}{M} \left( \frac{1 - p^{M/2}}{1 - p} \right) p^{M/2} - p_0 p^{3M/4+1/2}$$

$$= p_0 \left( \frac{2p}{M} \cdot \frac{1 - p^{M/2}}{1 - p} - p^{M/4+1/2} \right) p^{M/2}$$  

(11)

$$\lim_{|S| \to \infty} \text{Bias}[\text{PDR}_S] = p_0 \left( \frac{2p}{M} \cdot \frac{1 - p^{M/2}}{1 - p} - p^{M/4+1/2} \right).$$  

(12)

Since Equations (11) and (12) are difficult to grasp, we plot them in Figure 5. We can note that the error is much bigger for the estimator PDR$_S$ compared with PDR$_B$, and this is because the slope of the curve is steeper for the smaller packet sizes (see for instance Figure (1). These errors will have impact on the accuracy for any estimator if used this way and increasing the sample size will not improve. We have exemplified with uniform packet size distribution, but this will hold true for most other distributions, except a bimodal distribution. Despite all these issues, we can still define an estimator $\hat{p}$ for $p$ by using the two points from Equation (8) and the two-packet lengths method, ie, Equation (5) as follows:

$$\hat{p} = \left( \frac{\text{PDR}_B}{\text{PDR}_S} \right)^{1/(\bar{B} - \bar{S})},$$  

(13)

when $PDR_S > 0$ and $PDR_S \geq PDR_B$.

**Corollary 2.** When packet sizes are uniformly distributed and $p > 0$, $\hat{p}$ as defined in Equation (13) is a consistent estimator of $p$ (converges in probability).

**Proof.** When we combine Equation (9) and (10) with Equation (13), we get:

$$\lim_{|S\cup B| \to \infty} E[\hat{p}] = \lim_{|S\cup B| \to \infty} E \left[ \left( \frac{\text{PDR}_B}{\text{PDR}_S} \right)^{1/(\bar{B} - \bar{S})} \right] = \left( \frac{p_0 p \frac{2}{M} \left( \frac{1 - p^{M/2}}{1 - p} \right) p^{M/2}}{p_0 p \frac{2}{M} \left( \frac{1 - p^{M/2}}{1 - p} \right)} \right)^{1/(M/2)} = (p^{M/2})^{1/(M/2)} = p$$

since all values are bounded, and all the functions are continuous within those bounds. This shows that $\hat{p}$ is asymptotically unbiased. Since $\text{Var}[\hat{p}] \to 0$ as $n \to 0$, we can conclude that $\hat{p}$ also is consistent. This follows from Chebyshev’s inequality. See Appendix A for details.

Hence, despite the overestimation of $PDR_S$ and $PDR_B$, $\hat{p}$ is a consistent estimator for $p$ and hence will work well as long as the sample size is not too small. This is an effect of $PDR_S$ and $PDR_B$ having errors that are canceled out in the estimation.
of $p$. Note that this estimation works quite well even when packet lengths are not uniformly distributed as long as both the $S$ and $B$ sets contain sufficient amount of packets. Therefore, depending on the packet length mix, we may choose another cut off threshold between the $S$ and $B$ sets than $M/2$ to make sure both bins contain sufficient amount of packets.

5.2 Estimating $p_0$ using the two-bins method

While the method of dividing packets into two bins works well for estimating $p$, it does not estimate $p_0$ accurately due to the overestimation discussed in the previous section. Even so, we may still try to use it and, as we will see later, it will actually work quite well in many scenarios. To find $p_0$ using this method, which we will call the two-bins method, we use Equations (9) and (10) with the bottom equation of Equation (5), and then we obtain these two options:

$$
\hat{p}_0 = \frac{PDR_S}{p^S} \quad \text{or} \quad \hat{p}_0 = \frac{PDR_B}{p^B}.
$$

(14)

To get an even better estimate in the case of a low number of packet reception events, we choose to take the arithmetic average of the two options of Equation (14).

However, there are other ways of estimating $p_0$. One might attempt to divide into more than two bins and use a linear least square method of the logarithm of the bin PDRs. However, increasing the number of bins beyond two will not improve the estimation. As long as we divide packets into bins, we will still overestimate the curve exactly as before and such estimators will remain asymptotically upwardly biased and inconsistent. Hence, it is sufficient to use just two bins.

Another way is to compensate for the bias, which we tried without success in Section 4.1. However, as we will soon see, in this case, bias compensation works better. From Equations (11) and (12), we know that we can quantify this bias if the packet length distribution is uniform. The bias simply depends on $p$ and $M$. To do this, we define a $\beta$ as follows:

$$
p_0 \beta = \lim_{|S| \to \infty} \text{Bias}[\text{PDR}_S].
$$

Using this $\beta$, we can get new estimators for $p_0$ as follows:

$$
\hat{p}_0 = \frac{PDR_S}{p^S + \beta} \quad \text{or} \quad \hat{p}_0 = \frac{PDR_B}{p^B + \beta p^M/2} \quad \text{where} \quad \beta = \frac{2p}{M} \cdot \frac{1 - p^{M/2}}{1 - p} - p^{M+1/2}.
$$

(15)

Using this method, which we call the two-bins+$\beta$ method, we can find $p_0$ in an accurate way if the packet length distribution is uniform or approximately uniform. However, it must be noted that if the packet length distribution is not uniform, the accuracy may disappear. For instance, if the distribution is bimodal with one mode in each bin, there is no error, and we should have that $\beta = 0$. Instead, using this method, we would underestimate.

To simplify the two-bins+$\beta$ method, we could approximate the calculation of $\beta$ in Equation (15) with a simple polynomial function that approximates well in the valid area of $p$, ie. $0.998 < p \leq 1$. All our WiFi scenarios have $p$ values within those bounds. With polynomials of degree 3, we can efficiently approximate both $\beta$ and $\beta \cdot p^{M/2}$ and thereby avoid calculating the exponentiations of Equation (15) if needed.

5.3 A consistent estimator of $p_0$ in the general case

To find an unbiased and consistent estimation method of $p_0$, we need to leave the bin approach. The following method, which we call the transform method, is based on a coordinate system transformation. We start by assuming that we know $p$, which we may have obtained from the two-bins method in Equation (13). Then, we use the following coordinate transformation:

$$
\begin{cases}
x'_i &= x_i \\
y'_i &= y_i + p_0 - p_0 \cdot p^S
\end{cases},
$$

where $x_i$ is the packet length of the $i$:th packet, $y_i$ is 0 or 1 depending on if packet $i$ arrived or not, and $y'_i$ is the transformed version of $y_i$. The transformation makes the PDR curve, which normally is an exponential decay function, into a horizontal line at $y' = p_0$. This happens since we simply subtract the actual PDR curve (with the correct $p$ value) from the original $y_i$ or PDR values. As a consequence, the average of the transformed values $y'_i$ should be a consistent estimator of $p_0$ as
Numerical results

follows:

\[
\frac{1}{n} \sum_{i=1}^{n} y_i' \to p_0 \quad \text{when} \quad n \to \infty,
\]

assuming that we have a total of \( n = |S| + |B| \) packet reception events. From this, we get:

\[
np_0 = \sum_{i=1}^{n} (y_i + p_0 - p_0 \cdot p_i^0) = \sum_{i=1}^{n} y_i + np_0 - p_0 \sum_{i=1}^{n} p_i^0,
\]

(16)

when \( n \to \infty \). Now, we can find an estimator of \( p_0 \), by rewriting Equation (16) as follows:

\[
\hat{p}_0 \approx \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} p_i^0} = \frac{|S| \cdot \text{PDR}_S + |B| \cdot \text{PDR}_B}{\sum_{i=1}^{n} p_i^0}.
\]

(17)

To estimate both \( p \) and \( p_0 \), we need to calculate the sums of Equation (8) for both bins. Then, we obtain \( \hat{p} \) from Equation (13) and \( \hat{p}_0 \) from Equation (17). For most parts, this method contains an extra exponent in the denominator of Equation (17). However, the biggest challenge is when \( \hat{p} \) changes, because then we need to recalculate the denominator of Equation (17). Essentially, we have to recalculate the entire sum, which means calculating a lot of exponents. Hence, this is not an online algorithm. However, in Appendix B, we develop an approximation based on Jensen’s inequality (\( \hat{p}_i^0 \) is a convex function) and Hölder’s defect, which is a quantification of how much inequality there is in Jensen’s inequality. When we use this approximation, we call it the online transform method.

The next question is whether these two estimators are unbiased and consistent. The estimator \( p \) of Equation (13) is only asymptotically unbiased but biased for any \( n < \infty \). This means that \( \hat{p}_0 \) of Equation (17) must be biased if it is based on the estimator \( p \) of Equation (13). However, \( \hat{p}_0 \) of Equation (17) is consistent when combined with a consistent estimator \( \hat{p} \), such as Equation (13), as shown by the following theorem.

**Theorem 2.** The estimator \( \hat{p}_0 \) of Equation (17) is consistent of \( p_0 \) under the assumption that the estimator \( \hat{p} \) is consistent.

**Proof.** We start by showing that \( \hat{p}_0 \) is asymptotically unbiased; i.e., that \( E[\hat{p}_0] \to p_0 \) when \( |S|, |B| \to \infty \):

\[
E[\hat{p}_0] = E \left[ \frac{\sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} p_i^0} \right] = E \left[ \frac{\sum_{i \in S} y_i + \sum_{i \in B} y_i}{\sum_{i \in S} \hat{p}_i^0 + \sum_{i \in B} \hat{p}_i^0} \right] = p_0 E \left[ \frac{|S| \cdot \text{PDR}_S + |B| \cdot \text{PDR}_B}{\sum_{i \in S} \hat{p}_i^0 + \sum_{i \in B} \hat{p}_i^0} \right] \to p_0,
\]

(18)

when \( |S|, |B| \to \infty \). The last step holds true since \( \hat{p} \to p \) (is consistent). Finally, we can conclude the proof in the same way as in Corollary 1 and Appendix A, since we trivially can see that \( \text{Var}[\hat{p}_0] \to 0 \) when \( |S|, |B| \to \infty \).

Under the assumption that \( \hat{p} \) is a perfect estimator (i.e., \( \hat{p} = p \)), numerical results suggest that \( \hat{p}_0 \) of Equation (17) is not only asymptotically unbiased but also unbiased for all \( n \). However, since this does not hold true for the estimator \( \hat{p} \) of Equation (13), this is mostly of theoretical interest. Nevertheless, Equation (17) does not add its own bias when \( n \) is small.

Due to the approximation used in the Hölder’s defect, the online transform method is both biased and asymptotically biased and hence not consistent. How these estimators behave for different amounts of packet reception events \( n \) will be demonstrated numerically in the following section. We will see that this bias is within reasonable bounds.

### 5.4 Numerical results

The remaining question is how good the different methods are in practice. In this section, we first used the 44 WiFi links to find representative values for \( p \) and \( p_0 \). Then, we used those values to simulate how the different methods perform. For each link, the \( p_0 \) and \( p \) were found by an optimization algorithm that minimized the absolute errors between Equation (2) and all \( p_x \) for that link.

With all 44 values of \( p \) and \( p_0 \), we again resorted to a Monte Carlo simulation, where we simulated \( n = 2000 \) data packet reception events per link. For each packet, the simulator generated a random packet length according to a certain packet length distribution, calculated the expected PDR for that packet length based on Equation (2) and the \( p_0 \) and values \( p \) of
the simulated WiFi link. Finally, the successful or failed delivery of the packet was simulated by a Bernoulli trial using the calculated PDR. The split between the two bins was \( d = M/2 \), ie, 750 bytes for our WiFi tests.

In the first comparison, the data packet-based estimation methods had access to the actual \( p \) value of the link, and the task was only to estimate \( p_0 \). Table 4 shows the results for all the estimation methods introduced in the previous sections; 2 Bins is the two-bins method, 2 Bins+\( \beta \) is the two-bins method with bias compensation, Tf is the transform method, oTf is the online transform-method, and MLE is an MLE based on the following likelihood function:

\[
\mathcal{L}(p_0, p; R \cup L) = \prod_{i \in R} p_0 p_i^{p_i} \prod_{i \in L} (1 - p_0 p_i^{p_i}),
\]

(18)

where \( R \) is the set of packets received and \( L \) is the set of packets lost in a given sample \( R \cup L \). We maximized the logarithm of Equation (18) using the L-BFGS-B optimization algorithm, which is a bounded version of the limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm. This method is only for comparison due to its complexity.

The table shows the MAE between \( p_0 \) and the estimated \( \hat{p}_0 \) for all 44 WiFi links repeated 5000 times. The confidence interval is neglectable. As shown in the table, several different packet length distributions were used. First, we tested uniform packet size distribution where all valid packet sizes are equally probable. The bimodal distribution selected only packet lengths of either 375 bytes or 1125 bytes, ie, \( M/4 \) and \( 3M/4 \). The trimodal used three different packet lengths (50, 775, and 1500 bytes). In both bimodal and trimodal, all used packet lengths were equally probable. We also tried integer versions of the exponential and normal distributions that were truncated to between 1 and 1500 bytes. Finally, we tested the simple and complete IMIX distributions, which are used in standardized performance tests and are designed to mimic the real packet size distribution on the Internet. IMIX Simple is a trimodal distribution with different packet sizes and unequal probability between the packet sizes. IMIX Complete is based on IMIX Simple with a uniform packet length distribution as base.

From the results in Table 4, we can see that there is very little difference between the methods. Hence, in practice, we can choose a simpler method, such as the two-bins method. Regarding the two-bins+\( \beta \) method, we can see an improvement for the uniformly distributed packet lengths, which is expected. However, for the other extreme, the bimodal distribution, the error actually increases slightly. This shows that this method may degrade the accuracy. The conclusion from this is that the two-bins method is the simplest method, and only the significantly more complex transformation method gives better results. In the remainder of this section, we only focus on the two-bins method and the online transform method.

In Figure 6, we show the importance of \( d \), ie, where the split between the two bins is, which is used by both the two-bins method and the (online) transform method. These results were obtained using the same Monte Carlo simulation approach as above and repeated enough times to make the confidence intervals very small. A uniform packet length distribution was used. The figure shows the MAE when using the complete methods, ie, predicting both \( p \) and \( p_0 \) and using those to predict the actual \( p_{1500} \). Both the two-bins method and the online transform methods are shown. From the figure, we can see that the used \( d \) value has little impact as long as it is not approaching extreme values. We can also see that \( d = M/2 \) is not the optimal split but a slightly smaller value. As shown in the figure, the result is similar for both methods as well as for different \( n \), the amount of data packet reception events used. We repeated the experiments with other packet length distributions, but the results were similar.

Finally, in Figure 7, we compare the complete two-bins method versus the online transform method when estimating \( p_{1500} \) using different amounts of data packets (eg, different \( n \)). We have also included the flat method based on data packets and an MLE method of Equation (18) as comparison. We also attempted an MLE variant of Gilbert’s model, but that was too difficult to implement in a robust manner.

<table>
<thead>
<tr>
<th>Packet Size Distribution</th>
<th>MLE</th>
<th>2 Bins</th>
<th>2 Bins+( \beta )</th>
<th>Tf</th>
<th>oTf</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniform (1-1500)</td>
<td>1.4</td>
<td>1.1</td>
<td>0.94</td>
<td>0.85</td>
<td>0.97</td>
</tr>
<tr>
<td>Bimodal (375 + 1125)</td>
<td>0.93</td>
<td>0.8</td>
<td>0.86</td>
<td>0.74</td>
<td>0.76</td>
</tr>
<tr>
<td>Trimodal (50, 775, 1500)</td>
<td>1.05</td>
<td>1.3</td>
<td>0.97</td>
<td>0.82</td>
<td>1.15</td>
</tr>
<tr>
<td>Exponential (1/375)</td>
<td>0.99</td>
<td>1.6</td>
<td>1.5</td>
<td>0.73</td>
<td>0.78</td>
</tr>
<tr>
<td>Normal (( \mu = 775, \sigma = 375 ))</td>
<td>1.7</td>
<td>1.1</td>
<td>0.94</td>
<td>0.88</td>
<td>0.96</td>
</tr>
<tr>
<td>IMIX Simple</td>
<td>0.82</td>
<td>2.3</td>
<td>2.0</td>
<td>0.71</td>
<td>0.75</td>
</tr>
<tr>
<td>IMIX Complete</td>
<td>0.83</td>
<td>1.6</td>
<td>1.3</td>
<td>0.73</td>
<td>0.82</td>
</tr>
</tbody>
</table>

Abbreviations: MAE, mean absolute errors; MLE, maximum likelihood estimation.
FIGURE 6  Mean absolute errors (MEA) of $p_{1500}$ for the two-bins method and the online transform method when trying different values of $d$, i.e., the packet size where we split the two bins. Uniform packet length distribution was used.

FIGURE 7  Mean absolute errors (MAE) for the two-bins and online transform methods compared with the flat method and a maximum likelihood estimation (MLE) variant when varying the amount of sampling and the packet size distributions and estimating $p_{1500}$.

From the figure, we can see that the two-bins method outperforms the flat method and performs well for both distributions and that the errors are almost identical. Similar results are obtained from other packet length distributions. When comparing Figure 7 with Figure 3, we find that the two-bins method follows the two-packet lengths method with the two-bins method having a lower error. This is due to the longer packets being used in the two-bins estimation. We can also see that both the two-bins method, and the online transform methods are not far behind the MLE method. This means that these methods should be expected to perform close to what is possible, while still offering a feasible solution in practice for most LQE use cases.

6 | RELATED WORK

Previous research has shown that packet length has an impact on the PDR with longer packets having a larger chance of getting corrupted. Unfortunately, too many LQEs ignore this and simply assume the same delivery ratio among all packet lengths. Some authors, such as Fonseca et al., have acknowledged this and propose to use regular packets or protocol packets, such as acknowledgments and signaling packets.

The current empirical work on packet length effects is surprisingly thin. Most works concerning these effects are based solely on theoretical approaches. Quite often, a Raleigh or Rician small-scale fading model is used to find the BER. The BER is usually translated into a PER using Equation (1). This means that independent bit errors are assumed, and this is used without any empirical validation.

In other works, it is common to see a model that describes either the BER or the PER but very rarely their relationship. However, Gilbert does this for his two-state Markov chain model. Unfortunately, even Gilbert’s simple model contains three parameters that needs to be found by an LQE, making it an unsuitable choice for an LQE. This also means...
that more complex models based on Markov chains\textsuperscript{13-16} also are unsuitable for LQE even though they may be better at describing the behavior of the links.

Most of the work on packet size effects on PDR is within packet length optimization or packet size adaptation\textsuperscript{24,29,31-33}, ie, how to find the optimal packet length for a link. Most base their work on theoretical models or simulations and develop mechanisms to improve throughput, reduce energy consumption, and/or minimize delay. Only very few of them, such as Lettieri and Srivastava\textsuperscript{31}, did measurements, but then not of the actual impact of packet length on PDR.

Also outside the area of packet length optimization, very few studies have been done of actual measurements. One exception is Nguyen et al\textsuperscript{9} which showed that the packet loss increases exponentially with the packet length and that the packet loss doubles for each additional 300 bytes on a wireless link without FEC coding. Another one is Chakeres and Belding-Royer\textsuperscript{10} where the authors study the use of hello packets in the Ad hoc On-Demand Distance Vector Routing (AODV) routing protocol and try different parameter combinations. One of their conclusions is that when using hello packets of the same length as the data packets, end-to-end performance in the network increases from 60.7\% to 80.8\% delivery ratio. Dong et al\textsuperscript{32} included one graph of one link based on IEEE 802.15.4 where packet loss for various packet sizes are shown and Srinivasan\textsuperscript{34} looks at the PDR for both data packets and acknowledgments and concludes a significant difference.

Perhaps the most complete measurement of the packet length effect on PDR is the work by De Couto et al.\textsuperscript{1} There, the authors extracted measurements from their own testbed and plot the packet length vs PDR for some of the links. They use the results to explain why they experience a lower accuracy for their LQE. However, no further analysis or modeling of the packet length effect is done.

## 7 Conclusion

The PDR of wireless links depends, besides channel conditions, on the packet length. Derived from measurements, we introduced a simple loss model that captures this behavior. Using this model, we proposed several ways of estimating the packet delivery probability of longer data packets based on sampling the link using shorter packets.

We showed the estimation accuracy of these methods when estimating delivery ratios of different packet lengths. By sampling using two different packet lengths and then applying some few low-complexity calculations, the estimation of the packet delivery probability for arbitrary packet lengths was significantly improved. We also presented how to do this efficiently and what impact a small sample size has on the estimators.

Finally, the use of data packet reception events as input for the estimators was discussed. Several efficient estimation methods were developed that could use reception events for data packets where the used packet lengths are dictated by the applications. We showed how an online algorithm for this could be constructed. Lastly, a comparison study of the accuracy of all the proposed methods were done, showing improved accuracy for our proposed link quality estimators. Hence, we believe that all future link quality estimators must consider the effects of packet length.

## Acknowledgments

This work has partly been carried out within a project (project number 2015-02536) funded by the Swedish strategic innovation program on Internet of Things, which is a joint effort between VINNOVA, Formas, and the Swedish Energy Agency. The authors would like to thank Armin Halilovic, KTH for his mathematical contributions to this article.

## References


APPENDIX A: PROOF COROLLARY 1

Proof. We prove the case for \( PDR_s \) and concludes that the proof for \( PDR_b \) is identical.

We know that \( PDR_s \) is asymptotically upwardly biased. Assume that \( PDR_s \rightarrow a \) when \( \mid S \mid \rightarrow \infty \). Then, we know that \( a > p_S \). We also know that \( Var[PDR_s] \rightarrow 0 \) as \( \mid S \mid \rightarrow \infty \) since \( Var[PDR_s] = Var[y_i/\mid S \mid] \), which follows from the Bienaymé formula. We now show that \( PDR_s \rightarrow a \) (converges in probability). Hence, we need to show that for all \( \epsilon \):

\[
\lim_{\mid S \mid \rightarrow \infty} \Pr(\mid PDR_S - a \mid > \epsilon) = 0.
\]  \hspace{1cm} (A1)

We choose a \( n_0 \) that is large enough so that for all \( \mid S \mid > n_0 \):

\[
|E[PDR_S] - a| < \frac{\epsilon}{2}.
\]

If \( \mid PDR_S - a \mid \geq \epsilon \), then in accordance with reverse triangle inequality:

\[
|PDR_S - E[PDR_S]| = |PDR_S - a - (E[PDR_S] - a)| \geq |\mid PDR_S - a\mid - |E[PDR_S] - a| |
\geq |PDR_S - a| - |E[PDR_S] - a| \geq \epsilon - \frac{\epsilon}{2} = \frac{\epsilon}{2},
\]

and this means that we can conclude the following implication:

\[
|PDR_S - a| \geq \epsilon \quad \Rightarrow \quad |PDR_S - E[PDR_S]| \geq \frac{\epsilon}{2}
\]

which, in turn, implies

\[
P(|PDR_S - a| \geq \epsilon) \leq P(|PDR_S - E[PDR_S]| \geq \frac{\epsilon}{2}), \hspace{1cm} (A2)
\]

and according to Chevyshev’s inequality, we have

\[
P(|PDR_S - E[PDR_S]| \geq \frac{\epsilon}{2}) \leq \frac{Var[PDR_S]}{\left(\frac{\epsilon}{2}\right)^2} \rightarrow 0 \hspace{1cm} (A3)
\]

as \( \mid S \mid \rightarrow \infty \). Equations (A2) and (A3) together mean that Equation (A1) holds. Hence, \( PDR_s \rightarrow a \) and \( PDR_s \) cannot be consistent for \( p_S \) at the same time.

\( \square \)

APPENDIX B: HÖLDER’S DEFECT

In this appendix, we will use Hölder’s defect (see Steele\textsuperscript{36}, ch. 6) to find an online calculation method for the denominator of Equation (17). We will assume WiFi with \( 0 \leq L_i \leq 1500 \) and \( 0.998 \leq \hat{p} \leq 1 \), which holds for all the measured WiFi links.
in this article. The same approach can be used for other technologies as well. We know from Jensen’s inequality that:

\[ \hat{p}^F \leq \frac{1}{n} \sum_{i=1}^{n} \hat{p}^{Li} \leq 1, \]  

(B4)

where \( \bar{L} = \frac{1}{n} \sum_{i=1}^{n} L_i \). Hölder’s defect quantifies the error in Jensen’s inequality (Steele\(^{35}\), ch. 6) and should be able to find tighter bounds than Equation (B4).

The convex function is \( \varphi(x) = \hat{p}^x \) and its second derivative is \( \varphi''(x) = (\ln \hat{p})^2 \hat{p}^x \). Hölder’s defect applied on Equation (B4) states that:

\[ \frac{1}{n} \sum_{i=1}^{n} \hat{p}^{Li} = \hat{p}^F + \frac{1}{2} \cdot \mu \cdot \text{Var}[L], \]

where \( \mu \in [m, M] \) and

\[ m = \inf \{ \varphi''(x) : x \in [0, 1500] \} = (\ln \hat{p})^2 \hat{p}^{1500} \]

\[ M = \sup \{ \varphi''(x) : x \in [0, 1500] \} = (\ln \hat{p})^2. \]

To find a good estimate, we can set \( \mu \) to the average between \( m \) and \( M \), but this will be too low (again according to Jensen’s inequality). Instead, we use the average packet length \( \bar{L} \) as follows:

\[ \bar{\mu} = (\ln \hat{p})^2 \hat{p}^\bar{L}. \]

Then, we get the following approximation for the denominator of Equation (17):

\[ \sum_{i=1}^{n} \hat{p}^{Li} \approx n \hat{p}^{\bar{L}} \left( 1 + \frac{1}{2} (\ln \hat{p})^2 \text{Var}[L] \right). \]  

(B5)

This approximation looks more complex but is easier to compute in incremental steps. Note that there are good online algorithms for the calculation of the sample mean.\(^{36}\) The variance can also be calculated in an online fashion by, for instance, keeping running sums for the following identity: \( \text{Var}[X] = E[X^2] - (E[X])^2 \). This makes it possible to find a completely online algorithm for the approximation in Equation (B5).

That Equation (B5) is a good estimate can be shown numerically with a Monte Carlo simulation. It is clear that the estimate will be worse for some packet length distributions than others and for different values of \( \hat{p} \). To put this to a test, we calculated the relative error for some different options of \( \hat{p} \) and common packet length distributions that can be imagined or has been proposed. We used the same distributions as in Table 4.

Table B1 shows the results from the simulation. It is clear that the maximum relative error is found for small \( \hat{p} \). The maximum relative error from the tested distributions is found to be 7.6 %, which should be an approximate upper bound for the approximation in Equation (B5). Given all the other errors that arise in the estimations of PDRs, this approximation is still reasonable.

<table>
<thead>
<tr>
<th>Packet Size Distribution</th>
<th>Relative Error (%)</th>
<th>( \hat{p} = 0.998 )</th>
<th>( \hat{p} = 0.999 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniform (1 - 1500)</td>
<td>3.14</td>
<td>0.244</td>
<td></td>
</tr>
<tr>
<td>Bimodal (375, 1125)</td>
<td>0.92</td>
<td>0.068</td>
<td></td>
</tr>
<tr>
<td>Trimodal (50, 775, 1500)</td>
<td>7.21</td>
<td>0.661</td>
<td></td>
</tr>
<tr>
<td>Exponential (1/375)</td>
<td>3.85</td>
<td>0.707</td>
<td></td>
</tr>
<tr>
<td>Normal (( \mu = 775, \sigma = 375 ))</td>
<td>2.44</td>
<td>0.187</td>
<td></td>
</tr>
<tr>
<td>IMIX Simple</td>
<td>6.78</td>
<td>1.342</td>
<td></td>
</tr>
<tr>
<td>IMIX Complete</td>
<td>7.60</td>
<td>1.694</td>
<td></td>
</tr>
</tbody>
</table>

Note. The maximum 99% confidence interval was \( \pm 0.0074\% \) for \( \hat{p} = 0.998 \) and \( \pm 0.0006\% \) for \( \hat{p} = 0.999 \).