On Estimation across Analog Erasure Links with and without Acknowledgements

Vijay Gupta

Abstract—Consider the problem of estimating a linear time-invariant process across communication links that erase data stochastically. We compare the stability and performance for two protocols: when the sensor transmits new data only on obtaining an acknowledgement for previously transmitted data (so-called Transmission Control Protocol (TCP)-like protocols) and when the source transmits new data at every time step (so-called User Datagram Protocol (UDP)-like protocols). We show that the effect of acknowledgements and retransmissions on performance is case dependent. When a single sensor transmits data across a single link, we prove that TCP-like protocols always perform worse than UDP-like protocols in the sense of minimizing error covariance. When either multiple sensors or multiple links are present, we prove that TCP-like protocols may perform better than the UDP-like protocols.

I. INTRODUCTION

Recently, significant attention has been directed towards networked control systems in which components communicate over communication networks (see, e.g., [1], [4] and the references therein). The estimation and control performance in such systems is severely affected by the properties of the communication channels. To counter such performance loss, two research directions have emerged:

- One block design: An estimator/controller at the output of the communication channel is designed to compensate for the imperfections introduced by the channel. Such designs have been considered for channels that introduce data loss (e.g., [19]) or delay (e.g., [15]).
- Two block design: Both an encoder at the input and a decoder (estimator/controller) at the output of the channel are designed. Such designs have been considered for digital noiseless channels (e.g., [17]), erasure channels (e.g., [8]), and additive noise channels (e.g., [3]).

In this work, we consider the two block design problem of estimation across communication links that exhibit data loss for two cases of network protocols. In the so-called TCP-like protocols, the estimator provides acknowledgements to the encoder for successfully transmitted packets. If an acknowledgement is not received for a packet, the packet is retransmitted. Thus, while every transmitted packet is successfully received, there might be a stochastic delay introduced. On the other hand, UDP-like protocols do not have a provision of acknowledgement and the encoder transmits a new packet at every time step. We show that the effect of acknowledgements and retransmissions on performance is case dependent. When a single sensor transmits data across a single link, TCP-like protocols always perform worse than UDP-like protocols in the sense of minimizing error covariance. When either multiple sensors or multiple links are present, we prove that the TCP-like protocols may perform better than the UDP-like protocols.

There is significant work in estimation and control across links that introduce erasure. Within the one block design framework, various approaches to compensate for the data loss (see, e.g., [7], [18], [14], [22], [2], [12], [19]) have been proposed. The two-block design paradigm has also been considered in works like [9], [8], [10], [13], [16].

One feature introduced by data transfer protocols for networks that erase data packets is the possibility of acknowledgement from the receiver, and subsequent retransmission. Examples of protocols providing such features are the Transmission Control Protocol (TCP) and Bluetooth. On the other hand, protocols such as the User Datagram Protocol (UDP) do not provide any possibility of acknowledgements. Whether acknowledgements and retransmissions are desirable for estimation and control has long been identified as an important research question. Experimentally, both the classes of protocols have been shown to be suitable for control ([20], [21] for UDP and [6] for Bluetooth). However, there has been limited work on an analytical understanding of the effect of acknowledgements and retransmissions on the problem.

Most of the works mentioned above ignore the possibility of acknowledgements from the estimator. For LQG control, acknowledgements over the controller-actuator link have been shown to be crucial for separation principle to hold [19] (thus, possibly achieving better performance). However, the effect of retransmissions has not been considered. On the other hand, works such as [5] mention that UDP packets should be used ‘to favor performance over data accuracy.’ As we show, there are cases in which such an intuitive understanding may not hold.

II. PROBLEM FORMULATION

Consider a process with state $x(k) \in \mathbb{R}^n$ that evolves as

$$x(k+1) = Ax(k) + w(k), \quad k \geq 0$$  \hspace{1cm} (1)

where $w(k)$ is process noise assumed to be white, Gaussian, zero mean with covariance $R_w > 0$. The initial state $x(0)$ is a zero mean Gaussian random variable with covariance matrix $P(0)$. We consider two cases. In the single sensor case, the process is observed using a sensor that generates measurements, or observations, of the form

$$y(k) = Cx(k) + v(k), \quad k \geq 0$$  \hspace{1cm} (2)
where \( y(k) \in \mathbb{R}^m \) and the measurement noise \( v(k) \) is white, Gaussian, zero mean with positive definite covariance matrix \( R_v > 0 \). We assume that the pair \((A, C)\) is observable. In the two sensor case, the process is observed using two sensors that generate measurements of the form

\[
y_i(k) = C_i x(k) + v_i(k), \quad k \geq 0, \quad i = 1, 2,
\]

where \( y_i(k) \in \mathbb{R}^{m_i} \) and the measurement noises \( v_i(k) \) are mutually independent, white, Gaussian, zero mean with positive definite covariance matrices \( R_v \) are of the following special form.

**Lemma 2.1 (Proposition III.1 from [10]):** We can convert the process (1) and sensors (3) to a state-space representation with the structure (1)-(3), where the matrices \( A \in \mathbb{R}^{n \times n} \), \( C_1 \in \mathbb{R}^{m_1 \times n} \) and \( C_2 \in \mathbb{R}^{m_2 \times n} \) are of the form

\[
A = \begin{bmatrix} A_{1,1} & A_{1,2} & A_{1,3} \\ 0^{n_2 \times n_1} & A_{2,2} & A_{2,3} \\ 0^{n_3 \times n_1} & 0^{n_3 \times n_2} & A_{3,3} \end{bmatrix},
\]

\[
C_1 = \begin{bmatrix} C_{1,1} \\ 0^{m_1 \times n_1} \\ C_{1,2} \end{bmatrix},
\]

\[
C_2 = \begin{bmatrix} C_{2,1} \\ 0^{m_2 \times n_1} \\ C_{2,2} \end{bmatrix},
\]

where \( A_{i,j} \in \mathbb{R}^{n_i \times n_j} \), \( A_{i,j} \in \mathbb{R}^{m_i \times n_j} \) and \( n_1 + n_2 + n_3 = n \).

In the above representation, \( A_{1,1} \) describes the dynamics of the state subspace that is not observable from \( y_1(k) \), while the modes that are not observable by \( y_2(k) \) are dictated by the dynamics of \( A_{2,2} \). \( A_{3,3} \) specifies the dynamics of the modes that are observable by both \( y_1(k) \) and \( y_2(k) \). If there are no such modes, then \( A_{3,3} \) is an empty matrix (\( n_3 = 0 \)).

We also partition

\[
x(k) = \begin{bmatrix} x_1(k)^{n_1 \times 1} \\ x_2(k)^{n_2 \times 1} \\ x_3(k)^{n_3 \times 1} \end{bmatrix},
\]

\[
v_1(k) = \begin{bmatrix} v_{1,1}(k)^{n_1 \times 1} \\ v_{1,2}(k)^{n_2 \times 1} \\ v_{1,3}(k)^{n_3 \times 1} \end{bmatrix},
\]

\[
v_2(k) = \begin{bmatrix} v_{2,1}(k)^{n_1 \times 1} \\ v_{2,2}(k)^{n_2 \times 1} \\ v_{2,3}(k)^{n_3 \times 1} \end{bmatrix}.
\]

Finally, we define two sensors \( H_1 \) and \( H_2 \) that respectively generate observations of the form

\[
h_1(k) = \begin{bmatrix} C_{1,1} \\ C_{1,2} \\ C_{1,3} \end{bmatrix} x(k) + \begin{bmatrix} v_{1,1}(k) \\ v_{1,2}(k) \\ v_{1,3}(k) \end{bmatrix},
\]

\[
h_2(k) = \begin{bmatrix} C_{2,1} \\ C_{2,2} \end{bmatrix} x(k) + \begin{bmatrix} v_{2,1}(k) \\ v_{2,2}(k) \\ v_{2,3}(k) \end{bmatrix}.
\]

The process (1) is observable from either \( H_1 \) or \( H_2 \).

We consider three arrangements. In the single sensor, single link case, the sensor communicates with an encoder across an analog erasure communication link. The analog erasure communication link supports input a finite-dimensional real vector \( s(k) \) at every time step \( k \). The output of the link is described by a vector \( d(k) \) that equals \( s(k) \) with probability \( 1 - p \) and equals the special symbol \( \emptyset \) otherwise, where \( \emptyset \) denotes the event that an erasure occurred and no data was received. We will refer to an erasure event for a transmission at time \( k \) as \( r(k) = 0 \), and a successful transmission as \( r(k) = 1 \). Note that while for simplicity we have assumed no delay to be introduced by the link, a constant delay to account for the time that the receiver waits before declaring an erasure can easily be considered. The probability \( p \) is called the erasure probability. For a TCP-like protocol, the decoder receives \( d(k) \) and transmits the link state \( r(k) \) at every time \( k \) to the encoder. The encoder maintains the variable \( a(k) \) that equals \( r(k) \) with probability \( 1 - p_{\text{ack}} \) at any time step and is \( \emptyset \) otherwise. By convention, \( a(-1) = r(-1) = 1 \). The probability \( p_{\text{ack}} \) is called the acknowledgement loss probability.

For the two sensor, two link problem, each sensor transmits data to an estimator over an erasure link. The link states \( r_1(k) \) and \( r_2(k) \) are assumed to be independent of each other. Denote the erasure probability for the link joining sensor \( i \) to the estimator by \( p_i \). For a TCP-like protocol, the decoder transmits the link states \( (r_1(k), r_2(k)) \) to the two encoders. With an abuse of notation, we denote the variable maintained by the encoder to be \( a(k) \). The variable is updated as

\[
a(k) = \begin{cases} (r_1(k), r_2(k)) & \text{with prob } 1 - p_{\text{ack}} \\ \emptyset & \text{otherwise.} \end{cases}
\]

In the single sensor, line network problem, the sensor transmits data to the estimator across a network with analog erasure communication links in series. There are \( N + 1 \) nodes including the sensor (node 1) and the estimator (node \( N + 1 \)). Every node \( i \) transmits a vector \( s_{i,i+1}(k) \) to node \( i + 1 \) across an erasure link. The erasure events in various links are mutually independent. The link state for the link between nodes \( i \) and \( i + 1 \) is denoted by \( r_{i,i+1}(k) \), and the corresponding erasure probability is \( p_{i,i+1} \). Under a TCP-like protocol, node \( i \) transmits the link state information for link from node \( i \) to \( i + 1 \) to the node \( i \).

We assume that all sources of randomness - the process noise, measurement noises, initial condition, erasure events, and the acknowledgement erasures - are mutually independent. We consider the two block design problem. For the single sensor, single link problem, for a UDP-like protocol, the encoder at time \( k \) computes and transmits a finite-dimensional real vector \( s(k) = f(k, \{y(j)\}_{j=0}^k) \). For the TCP-like protocol, in addition to these measurements, the encoder also has access to the received acknowledgments. At every time \( k \), there are two possibilities:

- If \( a(k-1) = 1 \) and \( r(k-1) = 1 \), \( s(k) = g(k, \{y(j)\}_{j=0}^k \cdot \{a(j)\}_{j=0}^{k-1} \).
- If \( a(k-1) = \emptyset \) or \( r(k-1) = 0 \), \( s(k) = s(k-1) \).

Thus, if the previous transmission was successful, and an acknowledgement was received, the encoder transmits a new packet. Otherwise, the previous packet is retransmitted. For the two sensor, two link problem, for a UDP-like protocol, the \( i \)-th encoder at time \( k \) computes and transmits a causal function of its own measurements \( y_i(0), \ldots, y_i(k) \). For
the TCP-like protocol, the function can also depend on the received acknowledgements \(a(0), \cdots, a(k-1)\).

- If \(a(k-1) = (r_1(k-1), r_2(k-1)) = (1, 1)\), \(s_i(k) = g_i(k, \{y_j(j)\}_{j=0}^k, \{a(j)\}_{j=0}^{k-1})\).
- If \(a(k-1) = 0\) or \(r(k-1) \neq (1, 1)\), \(s_i(k) = s_{i+1}(k-1)\).

For the single sensor, line network problem, the encoder is present at the sensor and has the same structural constraints as in the single sensor, single link problem. For the UDP-like protocol, every intermediate node transmits to node \(i+1\) at time \(k\) the quantity received from node \(i-1\) at time \(k-1\). If no quantity was received due to erasure on the link between \(i-1\) and \(i\), the node \(i\) does not transmit anything. For a TCP-like protocol, if \(r_{i,i+1}(k-1) = 0\) or \(a_{i+1,i}(k-1) = 0\), the node \(i\) retransmits \(s_{i,i+1}(k-1)\) at time \(k\). Otherwise, if \(r_{i,i+1}(k-1) = a_{i+1,i}(k-1) = 1\), then node \(i\) transmits whatever quantity was last received from node \(i-1\). We assume that the vector transmitted by any encoder is always time-stamped.

At every time step \(k\), the estimator generates an estimate \(\hat{x}_{\text{dec}}(k+1)\) of the process state \(x(k+1)\) to minimize the estimation error covariance \(P(k+1) = E[(x(k+1) - \hat{x}_{\text{dec}}(k+1))(x(k+1) - \hat{x}_{\text{dec}}(k+1))^T]\), where the expectation is taken over the initial condition, and the process and measurement noises. Due to the stochastic data loss introduced by the channel, \(P(k)\) is a random variable. We will concentrate on the expected value of \(P(k)\), although the techniques we use can easily be extended to obtain arbitrary moments. In particular, if the term \(\lim_{k \to \infty} E[P(k)]\) is bounded, we say that the estimate error covariance is stable. It may be noted that our version of a TCP-like protocol is intended to be only a simple abstraction, and thus does not consider many issues important in practical implementations.

**Notation:** Define by \(M(k+1), M_1(k+1),\) and \(M_2(k+1)\), the error covariance of the estimate \(\hat{x}(k + 1)\) of state \(x(k+1)\) based on measurements \(\{y_j(j)\}_{j=0}^k, \{h_1(j)\}_{j=0}^k, \{h_2(j)\}_{j=0}^k\) respectively. Due to observability, all these matrices converge geometrically to steady state values. Denote by \(f_{C_1}^n(M)\) the Riccati recursion on matrix \(M\) based on sensor 1 carried out \(n\) times:

\[
\begin{align*}
f_{C_1}(M) &= A M A^T + R_w \\
f_{C_1}^{n+1}(M) &= f_{C_1}(f_{C_1}(\cdots f_{C_1}(M) \cdots)),
\end{align*}
\]

with the convention that \(f_{C_1}^n(M) = M\). Define \(f_{C_2}(M)\) and \(f_{C_2}^n(M)\) similarly, but with sensor 2 replacing sensor 1. Let \(A_1\) denote the unobservable part of \(A\) when the pair \((A, C_1)\) is converted to the observer canonical form. Finally, denote by \(\rho(T)\) the spectral radius of matrix \(T\).

### III. Main Results

**A. Single Sensor, Single Link Case**

1) **Optimal Encoder and Estimator:** For this case, we can use the results from [8], [9] to obtain the optimal encoder design for the estimation problem. The optimal encoder design, irrespective of the availability of acknowledgement to the encoder, is to transmit at every time step \(k\) is the estimate \(\hat{x}(k+1)\) of the state \(x(k+1)\) based on measurements \(y(0), \cdots, y(k)\). Thus, for a UDP-like protocol, the estimator transmits the estimate \(\hat{x}(k+1)\) at every time step, i.e., \(s(k) = \hat{x}(k+1)\). On the other hand, for a TCP-like protocol, a new estimate is transmitted only if the previous transmission was successful and the corresponding acknowledgement successfully received. Thus, for a TCP-like protocol,

\[
s(k) = \begin{cases} 
\hat{x}(k+1) & \text{if } a(k-1) = r(k-1) = 1 \\
\hat{x}(k+1) & \text{if } a(k-1) = 0 \text{ or } r(k-1) = 0.
\end{cases}
\]

Note that for a TCP-like protocol, estimates \(\hat{x}(j)\) corresponding to some times \(j\) are never transmitted. This is necessary so that the queue length at the encoder remains stable.

With either protocol, the estimator has access to some estimates \(\hat{x}(j)\) for \(0 \leq j \leq k + 1\). At every time \(k\), define \(t_s(k)\) as the maximal value of time \(j\) such that \(\hat{x}(j)\) was received by the estimator till time \(k\). The optimal estimator design is to compute the estimate as \(\hat{x}_{\text{dec}}(k+1) = A^{k+1-t_s(k)} \hat{x}(t_s(k))\). If no packet has been received till time \(k\), we define \(t_s(k) = 0\). Note that the estimator needs to store only one estimate in its memory.

2) **Analysis for a UDP-like Protocol:** For a UDP-like protocol, the time stamp \(t_s(k)\) is given by \(t_s(k) = \max\{j + 1| j \leq k, r(j) \neq 0\}\). In other words, the time stamp is determined by the last time \(j \leq k\) such that transmission was successful at time \(j\). If no packet has been received till time \(k\), \(t_s(k) = 0\). Since any packet is received with a probability \(1 - p\) independent of other packets, the estimate evolves as

\[
\hat{x}_{\text{dec}}(k+1) = \begin{cases} 
\hat{x}(k+1) & \text{with prob. } 1 - p \\
A \hat{x}_{\text{dec}}(k) & \text{otherwise}.
\end{cases}
\]

Thus, the expected error covariance satisfies the discrete Lyapunov recursion

\[
E[P_d^{U, DP}(k+1)] = (1 - p)M(k+1) + p AE[P_d^{U, DP}(k)]A^T + R_w,
\]

with the initial condition \(P(0)\). The steady state covariance satisfies the corresponding Lyapunov equation and exists (i.e., the estimation error is stable) iff \(p \rho(A)^2 < 1\).

3) **Analysis for a TCP-like Protocol:** With a TCP-like protocol, at time \(k\), the encoder may retransmit the estimate \(\hat{x}(j + 1)\) where \(j \neq k\). For notational convenience, denote the estimate transmitted by the encoder at time \(k\) by \(\hat{x}(k(j))\). Define the error \(e(k+1) = A^{k+1-k(j)} \hat{x}(k(j)) - x(k+1)\) with covariance \(M_e(k+1)\) and the error at the estimator \(e_{\text{dec}}(k+1) = \hat{x}_{\text{dec}}(k+1) - x(k+1)\) with covariance \(P_{d,TCP}^{T}(k+1)\). Given the erasure events at time \(k\), the two
error terms evolve as
\[
\begin{bmatrix}
e(k+2)
eg(k+1) \\
\end{bmatrix} =
\begin{bmatrix}
A & 0 \\
0 & A \\
w(k+1) & e(k+1) \\
w(k) & e_{dec}(k) \\
\end{bmatrix}
\begin{bmatrix}
e(k+1) \\
e_{dec}(k) \\
\end{bmatrix}
\]
+ \begin{bmatrix}
A & 0 \\
0 & A \\
w(k+1) & e(k+1) \\
w(k) & e_{dec}(k) \\
\end{bmatrix}
\begin{bmatrix}
e(k+1) \\
e_{dec}(k) \\
\end{bmatrix}
\]
with prob \(p\)
+ \begin{bmatrix}
0 & 0 \\
I & 0 \\
m(k+2) & e(k+1) \\
0 & e_{dec}(k) \\
\end{bmatrix}
\begin{bmatrix}
e(k+2) \\
e_{dec}(k+1) \\
\end{bmatrix}
\]
with prob \((1-p)p_{ack}\)
+ \begin{bmatrix}
0 & 0 \\
I & 0 \\
m(k+2) & e(k+1) \\
0 & e_{dec}(k) \\
\end{bmatrix}
\begin{bmatrix}
e(k+1) \\
e_{dec}(k) \\
\end{bmatrix}
\]
with prob \((1-p)(1-p_{ack})\)
\tag{5}
\end{equation}

where \(m(k+2)\) is the estimation error for \(x(k+2)\) given all the measurements \(y(0), y(1), \ldots, y(k+1)\). For simplicity of notation, define \(q = p + (1-p)p_{ack}\). From (5),
\begin{align}
E[P_{d TC}^T(k+1)] &= p \left( AE[P_{d TC}^T(k)]A^T + R_w \right) \\
&\quad + (1-p) \left( E[M_{e}(k+1)] \right) \\
E[M_{e}(k+1)] &= (1-q)M(k+1) + qAE[M_{e}(k)]A^T + R_w. \\
\end{align}
\tag{6}

Since \(q \geq p\), the necessary and sufficient condition for stability of the estimate error covariance is \(q \rho(A)^2 < 1\).

4) Comparison of UDP-like and TCP-like Protocols:
Since \(q \geq p\), the stability region with TCP-like protocols is, in general, smaller than the stability region obtained when UDP-like protocols are being used. However, if acknowledgements are not erased \(p_{ack} = 0\), \(q = p\) and the stability conditions are identical. However, the performance of the two protocols is quite different. Since \(M(j)\) is the error covariance in estimating \(x(j+1)\) based on all measurements till time \(j\) and \(E[M_{e}(j)]\) is the error covariance in estimating \(x(j+1)\) based on a subset of those measurements, \(M(j) \leq E[M_{e}(j)]\). Thus equations (6) and (4) imply that for any process and erasure probability, \(E[P_{d UDP}^T(k)] \leq E[P_{d TC}^T(k)]\).

B. The Two Sensor, Two Link Case

1) Encoder and Estimator Design: When multiple sensors are present, optimal encoder designs are known only if perfect acknowledgements are available from the estimator. For the UDP-like protocol, we use the following sub-optimal encoder structure that was shown in [10] to achieve the maximal stability region over all causal encoder structures. At each time step \(k\), each sensor transmits the estimate of all modes that it can observe based on its own measurements till time \(k\). Thus, sensor 1 transmits the estimate \(\hat{x}_2(k)\) of \(x_2(k)\) and \(\hat{x}_{3,1}(k)\) of modes \(x_3(k)\) based on \(y_1(0), \ldots, y_1(k)\). Similarly the sensor 2 transmits the estimate \(\hat{x}_1(k)\) of \(x_1(k)\) and \(\hat{x}_{3,2}(k)\) of modes \(x_3(k)\) based on \(y_2(0), \ldots, y_2(k)\). The estimator initializes \(\hat{x}_{dec}(-1) = 0\) and updates its estimate as follows:
- If \((r_1(k), r_2(k)) = (1, 1)\),
  \[
  \hat{x}_{dec}(k+1) = A \begin{bmatrix}
  \hat{x}_1(k) \\
  \hat{x}_2(k) \\
  \hat{x}_{3,1}(k) \\
  \end{bmatrix}
  \]
- If \((r_1(k), r_2(k)) = (1, 0)\), set
  \[
  \hat{x}_{dec}(k+1) = A \begin{bmatrix}
  \hat{x}_{dec,1}(k) \\
  \hat{x}_2(k) \\
  \hat{x}_{3,1}(k) \\
  \end{bmatrix}
  \]
- If \((r_1(k), r_2(k)) = (0, 1)\), set
  \[
  \hat{x}_{dec}(k+1) = A \begin{bmatrix}
  \hat{x}_{dec,2}(k) \\
  \hat{x}_1(k) \\
  \hat{x}_{3,2}(k) \\
  \end{bmatrix}
  \]
- If \((r_1(k), r_2(k)) = (0, 0)\), set
  \[
  \hat{x}_{dec}(k+1) = A \begin{bmatrix}
  \hat{x}_{dec,1}(k) \\
  \hat{x}_{dec,2}(k) \\
  \hat{x}_{dec,3}(k) \\
  \end{bmatrix}
  \]

In the above equations, \(\hat{x}_{dec,i}(k)\) denotes the estimate at the decoder of the modes \(x_i(k)\).

For the TCP-like protocol, for simplicity we shall assume that \(p_{ack} = 0\). When all the acknowledgements are successfully received, the optimal encoder and estimator algorithm was given in [10]. Due to space constraints, we refer the reader to that work for details on the algorithm. We will simply identify that each encoder \(i\) can calculate a quantity \(I_i,k(k)\) based on its own measurements that enables the calculation of the estimate \(\hat{x}(k+1)\) that is identical to the estimate of state \(x(k+1)\) based on measurements \(y_1(0), \ldots, y_1(k), y_2(0), \ldots, y_2(k)\). Then, the encoder \(i\) transmits the following vector at each time \(k\):
\[
s_i(k) = \begin{cases} 
I_{i,k}(k) & \text{if } r_1(k-1) = r_2(k-1) = 1 \\
s_i(k-1) & \text{otherwise}
\end{cases}
\]

The estimator calculates \(\hat{x}(j)\) using the most recently received vectors \(I_{1,j,j}(j)\) and \(I_{2,j,j}(j)\). It then generates \(\hat{x}_{dec}(k+1)\) through time-updating by a suitable number of steps.

2) Analysis for a UDP-like Protocol: The expected error covariance for a UDP-like protocol at any time \(k\) can be calculated by conditioning on the success of transmissions from sensors 1 and 2. There are four possibilities:
- \((r_1(k), r_2(k)) = (0, 0)\): In this case, the estimate \(\hat{x}_{dec}(k)\) is time updated.
- \((r_1(k), r_2(k)) = (1, 1)\): In this case, the estimate is identical to the estimate based on measurements \(h_1(0), \ldots, h_1(k)\).
- \((r_1(k), r_2(k)) = (1, 0)\): In this case, the estimate is obtained by further conditioning on the last time step \(j\) at which transmission from sensor 2 was possible. The estimate for modes \(x_2(k+1)\) and \(x_3(k+1)\) is based on
measurements $y_1(0), \ldots, y_1(k)$; while the estimate for $x_1(k+1)$ is calculated based on measurements $y_2(0), \ldots, y_2(j)$. Since the modes $x_1(k)$ are unobservable from sensor 1, the estimate for $x_1(k+1)$ can thus be seen to be based on measurements $h_1(0), h_1(1), \ldots, h_1(j), y_1(j+1), \ldots, y_1(k)$.

- $(r_1(k), r_2(k)) = (0, 1)$: By a similar argument as above, the estimate for $x_1(k+1)$ can be seen to be based on measurements $h_2(0), h_2(1), \ldots, h_2(j), y_2(j+1), \ldots, y_2(k)$.

Based on the above arguments, the error covariance at time $k+1$ is given by

$$E[P_d^{UDP}(k+1)] = p_1p_2 \left( AE[P_d^{UDP}(k)]AT + R_w \right) + (1 - p_1)(1 - p_2)M_1(k+1) + (1 - p_1)p_2((1 - p_2) + \sum_{j=0}^{k-1} p_2^j f_{C_1}^{j+1}(M_1(k-j)) + p_2^k f_{C_1}^{k+1}(P(0))) + (1 - p_2)p_1((1 - p_1) + \sum_{j=0}^{k-1} p_1^j f_{C_2}^{j+1}(M_2(k-j)) + p_1^k f_{C_2}^{k+1}(P(0)))$$

with $E[P_d^{UDP}(0)] = P(0)$. Necessary and stability conditions are thus defined by the three inequalities $p_1p_2\rho(A)^2 < 1, p_2\rho(A_1)^2 < 1$, and $p_1\rho(A_2)^2 < 1$.

3) Analysis for a TCP-like Protocol: For a TCP-like protocol, the encoders transmit vectors of the form $I_{i,j,l}(j)$ at every time $k$. If the transmission from at least one encoder fails, the estimator time updates the previous estimate. If transmission from both encoders is successful, the estimate $(k+1)$ is the same as the estimate of $x(k+1)$ calculated based on measurements $y_1(0), \ldots, y_1(j), y_2(0), \ldots, y_2(j)$. Thus, similar to the single sensor case for TCP-like protocol, the expected error covariance can be evaluated to be

$$E[P_d^{TCP}(k+1)] = p_{12} \left( AE[P_d^{TCP}(k)]AT + R_w \right) + (1 - p_{12})E[M_{12}(k+1)] = (1 - p_{12})M(k+1) + p_{12} \left( AE[M_{12,e}(k)]AT + R_w \right)$$

where $M(k)$ is the error covariance of $x(k)$ given measurements from both sensors till time $k-1$ and $p_{12} = 1 - (1 - p_1)(1 - p_2)$. The necessary and sufficient condition for stability is given by $(1 - (1 - p_1)(1 - p_2))\rho(A)^2 < 1$.

4) Comparison of UDP-like and TCP-like Protocols: Since $p_{12} = p_1p_2 + p_1(1 - p_2) + p_2(1 - p_1)$, the stability condition for TCP-like protocol is more restrictive than the first condition obtained for UDP-like protocol. Moreover, since $\rho(A_1) \leq \rho(A)$ and $\rho(A_2) \leq \rho(A)$, this condition is more restrictive than the second and third stability conditions for the UDP-like protocol. Thus, unlike the single sensor case, stability with TCP-like protocol is more restricted than with UDP-like protocol if multiple sensors are present. This is due to the fact that for UDP-like protocol, the estimator utilizes a packet from either of the sensors even if the transmission from the other sensor does not succeed at that time step. On the other hand, for the case of TCP-like protocol, transmission from both the sensors has to succeed at the same time step for any data to be used.

However, an interesting observation is that unlike the single sensor case, for multiple sensors, the performance with TCP-like protocol can be better than with UDP-like protocol. As an example, consider a scalar system with $A = 2$. Let $C_1 = C_2 = 1$, and the noise covariances be $R_u = R_{v,1} = 1$, $R_{v,2} = 2$. For the case when $p_1 = p_2 = 0.01$, the steady state error covariance with UDP-like protocol is given by 4.2675, and with TCP-like protocol is given by 3.6821. Thus, the performance with TCP-like protocol may be better than that with UDP-like protocol. Intuitively, this is due to the fact that the presence of acknowledgements in the TCP-like case permits an encoding design that leads to an estimate that optimally utilizes all measurements from both sensors. On the other hand, such algorithms when acknowledgements are not available are not known. Although for the UDP-like protocol we have considered a specific encoder and estimator design where the estimator chooses estimates for modes $x_3(k)$ as the estimate from one of the sensors, any other estimator design (e.g., based on combining the estimates from the two sensors optimally) would still be sub-optimal. Thus, for some values of the parameters, the performance with the TCP-like protocol will be better for such designs as well.

C. Single Sensor, Line Network Case

1) Optimal Encoder and Estimator: As for the single sensor, single link case, the optimal encoder design, irrespective of the availability of acknowledgements to the encoder, is to transmit at every time step $k$ the estimate $\hat{x}(k+1)$ of the state $x(k+1)$ based on measurements $y(0), y(1), \ldots, y(k)$, where $y(k)$ is the received packet. The intermediate nodes are not allowed any processing. The estimator generates the estimate $\hat{x}_{dec}(k+1)$ through an appropriate time-updating of the last estimate it received.

2) Analysis for a UDP-like Protocol: The analysis for a UDP-like protocol is similar to the single sensor, single link case. The equivalent erasure event for the line network is if any of the $N$ links suffers an erasure. Thus, the equivalent erasure probability is $p_l = 1 - \prod_{i=1}^{N} (1 - p_{l,i+1})$. Moreover, even if a packet is received by the estimator, the error covariance is given by $M(k)$, which is the error covariance of the estimate of $x(k+1)$ given measurements till time $k+1 - N$. Thus, the expected error covariance satisfies the discrete Lyapunov recursion

$$E[P_d^{UDP}(k+1)] = (1 - p_l)M(k+1) + \rho(A)[AE[P_d^{UDP}(k)]AT + R_w],$$

with the initial condition $P(0)$. The steady state covariance satisfies the corresponding Lyapunov equation and exists (i.e., the estimation error is stable) iff $p_l\rho(A)^2 < 1$.

Note that performance with TCP-like protocol is not always better. As an example, when $p_1 = p_2 = 0.1$, the expected error covariances with UDP-like and TCP-like protocol are given by 4.6 and 40.0 respectively.
3) Analysis for a TCP-like Protocol: The analysis for a TCP-like protocol follows by concatenating the analysis for a single sensor, single link $N$ times. For node $i$, denote the estimate transmitted to node $i + 1$ at time $k$ by $\hat{x}(k_{i+1}(j))$ and the last estimate received from node $i - 1$ till time $k$ by $\hat{x}(k_{i-1}(j))$. Also define the error covariances of estimate $x(k_{i+1})$ based on these estimates by $P_{i,\text{out}}^i(k+1)$ and $P_{i,\text{in}}^i(k+1)$ respectively. Following equations (6) and (7), the error covariance terms evolve as

$$E[P_{i,\text{out}}(k+1)] = p_{i,i+1}(AEP_{i,\text{in}}(k)+A^T + R_w) + (1-p_{i,i+1})E[P_{i,\text{out}}(k+1)],$$

$$E[P_{i,\text{in}}(k+1)] = (1-q_{i,i+1})E[P_{i,\text{in}}(k+1)] + q_{i,i+1}(AE[P_{i,\text{out}}(k)+A^T + R_w]),$$

where $E[P_{i,\text{in}}(k)] = M(k)$, $E[P_{N,\text{in}}(k)] = E[P_{d,\text{TCP}}(k)]$ and $q_{i,i+1} = p_{i,i+1} + (1-p_{i,i+1})p_{\text{ack},i}$. Using this value, we can evaluate $E[P_{d,\text{TCP}}(k)]$ explicitly. Moreover, we note from the above discrete Lyapunov recursions that the necessary and sufficient condition for stability of the estimate error covariance is $q_{i,i+1}p(A)^2 < 1$, for $i = 1, \cdots, N$.

4) Comparison of UDP-like and TCP-like Protocols: For this case we see that TCP-like protocols can have a larger stability region that UDP-like protocols, and hence a much better performance. For simplicity, assume that $p_{i,i+1} = p$ and acknowledgements are not erased. Then, the stability condition with the UDP-like protocol is $(1-(1-p)^N)p(A)^2 < 1$, while for TCP-like protocol is $pp(A)^2 < 1$. Thus, as $N$ increases, the error covariance becomes unstable for UDP-like protocols, while it may be stable for TCP-like protocols. Thus, TCP-like protocols can perform better.

IV. Conclusions

We considered the problem of estimation across analog erasure links under two different protocols. In the TCP-like protocol, the estimator acknowledges any received packet, and the erased packets are retransmitted. In the UDP-like scenario, no acknowledgements (and hence retransmissions) are permitted. We show that the effect of acknowledgements and retransmissions on performance is case dependent. When a single sensor transmits data across a single link, we prove that TCP-like protocols always perform worse than UDP-like protocols in the sense of minimizing error covariance. When either multiple sensors or multiple links are present, we prove that the TCP-like protocols may perform better than the UDP-like protocols.

References


