Question 1 (Sockets):

For basic EWMA, the timeout occurs after the first iteration - (0.92 sec) and then it is asymptotically converges to 1.8 secs from below. For the Jacobson/Karels algorithm, the timeout occurs after the second iteration (1.08s). The maximum value of the timeout is 1.63s. It eventually converges to 0.9s (from above).

Question 2 (Internet Checksum):

(a) Let \( \{a_i\} \) denote the set of 16-bit words over which the checksum is being calculated, suppose \( j \) is the word which has a corrupted bit and let \( b_j \) denote the value of that corrupted word. The checksum stored in the packet will be \( c = \sum_i a_i \), where sum is taken to be the ones complement sum. Ones complement addition commutes, so \( c = \sum_i a_i = \sum_{i\neq j} a_i + a_j \). Similarly, the value that the receiver will calculate on the corrupted packet is \( d = \sum_{i\neq j} a_i + b_j \). But the error is not detected only when \( c \) and \( d \) are equal, which only happens if \( a_j \) and \( b_j \) are equal, and this is not the case. This holds for any \( j \) since transposition of bits does not affect the checksum, so all 1-bit errors are detected.

(b) The simplest checksum algorithm to detect one bit errors is a parity bit – a 1-bit sum of every single bit of the packet. If any bit is corrupted (including the checksum bit itself), the parity check will fail. This checksum is not used in UDP and TCP because it will fail to detect errors involving more than one bit with 50% probability (while the regular checksum is very likely to detect multi-bit errors).

(c) In order to guarantee that that the whole packet is not corrupted, the number of checksum bits needs to be equal in length to the data in the packet. Just sending all the data as the protected checksum is the best you can do. So now we have to prove it.

If the length of the payload is \( x \), then there are \( 2^x \) possible payloads of this length. If fewer than \( x \) bits are used in the checksum, let’s say \( y \) bits, then we would have \( 2^y \) possible checksums. By an application of the pigeonhole principle, we can see that there will be at least two payloads with the same checksum. So if one payload gets sent and bits get flipped and it converts into the other payload with the same checksum, the receiver will not be able to discern the correct from the incorrect.
Question 3 (TCP modeling)

(a) Let us consider a “cycle” of the TCP sawtooth to be the period when the congestion window increases from $\frac{W}{2}$ to $W$ packets - and then a loss occurs, then the total number of packets transmitted during this cycle would be:

$$\frac{W}{2} + \left(\frac{W}{2} + 1\right) + \left(\frac{W}{2} + 1 + 1\right) + .. + W$$

$$= \left(\frac{W}{2} + 1\right) \frac{W}{2} + \sum_{i=0}^{\frac{W}{2}} i$$

$$= \frac{3}{8}W^2 + \frac{3}{4}W$$

Since we lose one packet per cycle, the loss rate is just the number of lost packets over the number of transmitted packets, is just the reciprocal of the quantity above.

(b) We are given the average throughput to be $\frac{0.75W}{RTT}$

$$= \frac{0.75W \times MSS}{RTT} \text{ bytes per sec.}$$

We substitute for $W$, using the relation derived in part (a).

$$\frac{3}{8}W^2 + \frac{3}{4}W = \frac{1}{L}$$

for large values of $W$, the above relation can be approximated to yield

$$\frac{3}{8}W^2 \approx \frac{1}{L}$$

$$= W \approx \sqrt{\frac{8}{3L}}$$

So substituting for $W$, we get throughput to be

$$\approx \frac{1.22 \times MSS}{RTT \sqrt{L}}$$
Question 4 (TCP Congestion Control):

(a) Identify the intervals of time where TCP slow start is operating. *Slow-start is operating in rounds [1, 6] and [23, 26]*

(b) Identify the intervals of time when additive increase is operating. *[6, 16] [17, 22]*

(c) After the 16th transmission round, is packet loss detected by a triple duplicate ACK or by a timeout? *Triple Duplicate ACK.*

(d) After the 22nd transmission round, is packet loss detected by a triple duplicate ACK or a timeout? *Timeout - the congestion window is down to 1*

(e) What is the value of *ssthresh* at the first transmission round? *32*

(f) What is the value of *ssthresh* at the 18th transmission round? *21*

(g) What is the value of *ssthresh* at the 24th transmission round? *13*

(h) During what transmission round is the 70th packet sent? *Round 7*

(i) Assuming a packet loss is detected is after the 26th round by the receipt of a triple-duplicate ACK, what will the value of *CWin* and *ssthresh* be? *CWin = ssthresh = 4*

(j) (Editor’s note: this subpart was confusingly written, and has many interpretations. The intended answer was ssthresh = 21, cwnd = 1.)

(k) (Same note as (j). Intended: 1 + 2 + 4 + 8 + 16 + 21 = 52.)