Network Analysis and Simulation - Homework 4

Michele Polese, 1100877

June 11, 2015

Exercise 1 - A discrete event queue simulator in MATLAB

The discrete event queue simulator implemented for this homework is based on the simulator proposed by Law in [1]. It is a simplified version of a more general DE simulator since it has to handle just one server and the events may be only arrivals or departures. The objective is to measure the delay that users suffer when entering in the system for different utilization factors ρ and for different arrival and service processes. The simulator is designed to be as more general as possible, given that the code is written in MATLAB.
The simulator uses the following counters:

- `clock` the current system time
- `next_arr` the time of the next arrival. It is updated whenever there is an arrival, by extracting a random interarrival time according to the specifics of the queue
- `next_dep` the time of the next departure. It is updated by extracting a random service time if an user arrives and finds the queue empty or if a user departs and there is at least one other user waiting in the queue. Instead, when the system is empty the `next_dep` has no meaning and it is set to `next_arr` + 1.
- `time_of_last_event` is the time at which the previous event happened
- `server_status` set to 0 if the server is free, 1 if it is busy
- `number_in_queue` number of users in the queue (= number of users in the system - 1)
- `times_of_arrival` a FIFO queue that collects the time of arrival of each user
- `number_of_events` and `renewal_instants` are 2 counters that are used to stopping conditions on the simulation. Generic simulations stop when the number of events (either arrivals or departures) has reached a given maximum. Instead, if the queue has independent arrivals (as it is for each of the cases that will be considered later), the simulator counts a renewal instant when an arrival finds the queue empty, and the simulation stops when this counter reaches a given threshold, which is lower than the maximum number of events. Indeed a renewal interval is completely representative of the statistics of the queue and since for small utilization factors ρ these intervals are short, then in order to compute a reliable estimate of the quantities of interest in the queue it is sufficient a smaller number of events than the maximum allowed. Instead for ρ approaching 1 it may happen that the queue never empties and therefore a complete renewal interval is never observed, this explains the need of a maximum number of events to stop the simulation.

The simulation begins with an initial phase of initialization, with the extraction of the first interarrival time (and the first departure time set to the previous plus 1), and the counters all to 0. Then at each step the simulator checks whether the next event is an arrival or a departure by comparing `next_arr` with `next_dep`. This explains why when the system is empty `next_dep` is set to `next_arr` + 1: the next event will be surely an arrival.
If the event is an arrival, the simulator extracts the next interarrival time t_a and stores `clock` + t_a in `next_arr`. It checks if the queue is empty, in this case it schedules immediately a departure for the user just arrived. If not, the user is backlogged in the queue. In both cases the arrival time (the current clock value) is stored in `times_of_arrival`.
If the event is a departure, the simulators performs the following operations. At first it checks if there is someone left in the queue. If not, it sets the next departure time as at the beginning at `next_arr` + 1, instead if a user is ready enters in service it computes a random service time. Since a user has just left, the head of the queue `times_of_arrival` is read and popped out. The delay for that user is computed as the difference between current clock value and its arrival time, and added to a cumulative metric D_t. The number of departed users u_d is increased by one.

The interarrival and service times are generated with 2 dedicated functions, that can be easily extended in order to support any kind of slotted or continuous interevent time. As for now, they can handle

- Poisson arrivals/departures, with exponential interevent times
- Arrivals/departures with probability b in each slot, i.e. interevent times are geometric of mean $1/b$
- Interevent times of 1 or 2 slot with probability 0.5 each

The slot length can be specified.
There are some other counters implemented in the simulator that collects statistics:

- D_t and u_d as previously mentioned collect a cumulative delay and the number of departed users and are updated each time a user leaves, as specified. Then for each simulation the average delay will be

$$\hat{d} = \frac{D_t}{u_d} \tag{1}$$

- $\int Q(t)$, where $Q(t)$ measures the number of users in the queue at time t and it is represented by the counter `number_in_queue`. $\int Q(t)$ instead measures the total time spent in the queue by users which are actually queued.
- $\int B(t)$, where $B(t)$ signals if the server is free or busy at time t and corresponds to the variable `server_status` at each event. $\int B(t)$ instead measures the total amount of time in which the server was busy. Then, given the value T of the counter `clock` at the end of each simulation it is possible to estimate the utilization factor as

$$\hat{\rho} = \frac{\int_0^T B(t)}{T} \tag{2}$$

The simulated scenarios are three: an M/M/1 queue, an M/G/1 queue and finally a queue with geometric arrivals and departures (let's call it geo/geo/1 from now on).
The simulations are called from an external script that for each ρ dynamically adjusts the number of simulations needed to reach a required confidence level on the estimation of the delay. In particular at least 2 iterations are required in order to compute a sample variance $\hat{s}$. Then the width of the confidence interval at iteration r is computed as $\eta\sqrt{\hat{s}}/\sqrt{r}$ with $\eta = 1.96$. Note that unless r is big enough, the C.I. computed with the Gaussian approximation may not be the real confidence interval. Therefore a minimum number of 10 iterations is required.
The width of confidence interval just computed is then checked against a threshold related to the sample mean $\hat{m}$, computed at each iteration as $\hat{m}/30$. If the C.I. is below this threshold the simulator stops and proceeds with another value of ρ, if needed. In this way the confidence interval is related to the value of the delay that the system tries to estimate. There is also another parameter, which is a maximum number of iterations allowed for each ρ which is set to 500. Indeed, for ρ close to 1, an entire renewal interval may not be observed in each simulation, yielding results with very high variance and large C.I. and to shrink the width of the C.I. below the required threshold may be very costly in terms of time. Note instead that for ρ going to 0 also a single run of the simulation, of a proper length, returns an estimation of the delay which is reliable since the number of observed renewal intervals is very high, therefore in this cases the simulator will surely stop after the first 10 iterations with a confidence interval whose width is much smaller than $\hat{m}/30$.
In order to assess that the simulator is well-written and performs in the correct way, the results are checked against theoretical results or measures obtained in the previous homework for the geo/geo/1 queue. The maximum number of renewal intervals is set to 10^4, while the maximum number of events to 10^5.
The M/M/1 queue has exponential interarrival and service times. The service rate μ is assumed normalized to 1 and for each ρ then $\lambda = \rho\mu = \rho$. With this assumption the average delay m_d (in this case it is equal to the time spent in the system) is as in [2]

$$m_d = \frac{1}{\mu - \lambda} = \frac{1}{1 - \rho} \tag{3}$$

The results of the simulations and the comparison with the theoretical results are in Figure 1. The estimated value of $\hat{\rho}$ as in (2) is plotted in Figure 2 against the theoretical ρ.
The M/G/1 queue under analysis has Poisson arrivals and service time in 1 or 2 slots with the same probability. Given the duration l_{slot} of a slot the mean length of a service time is then $m_y = 1.5 l_{slot}$, while the variance is $\sigma_y^2 = 0.25 l_{slot}^2$. Then the rate of arrivals is $\lambda = \rho/m_y$. The theoretical average delay from [2] is

$$m_d = \frac{m_y}{2} + \frac{m_y + \lambda\sigma_y^2}{2(1 - \lambda m_y)} \tag{4}$$

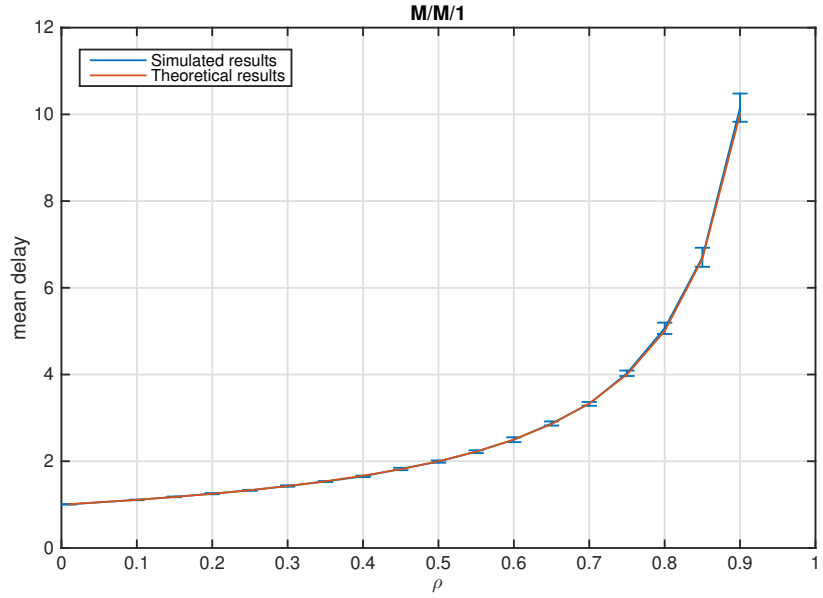


Figure 1: Delay vs. ρ for an M/M/1 queue, simulated and theoretical results by assuming $\mu = 1$

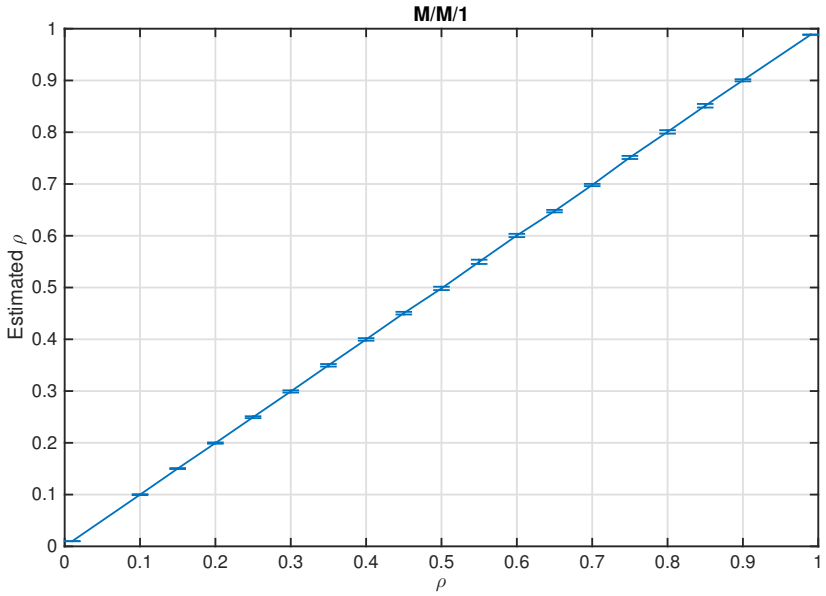


Figure 2: $\hat{\rho}$ vs. ρ for an M/M/1 queue

The results of the simulation and comparison with the theoretical curve is in Figure 3. The slot length is normalized to 1, therefore $l_{slot} = 1$. Figure 4 shows the estimated utilization factor against the theoretical one.
Finally the simulation of the geo/geo/1 queue is compared with simulation of the same queue, in a discrete time fashion, which was part of the previous homework and with the theoretical analysis that is developed below. In each slot (of length 1) there is an arrival with probability $b_a = 0.5$ and a departure (if the queue is not empty) with probability $b_s \in [0.5, 1]$. Note that for these values of b_s the minimum utilization factor is $\rho = 0.5$. From now on, for simplicity, b_s will be b.
The theoretical analysis starts from the considering the system as a Markov Chain, characterized by the number of users which are in the system at the end of each slot. In particular, by assuming that users that arrive in one slot cannot be served in the same slot, then for state $X_k > 0$ (i.e. when there is at least one user the system) there can be only three

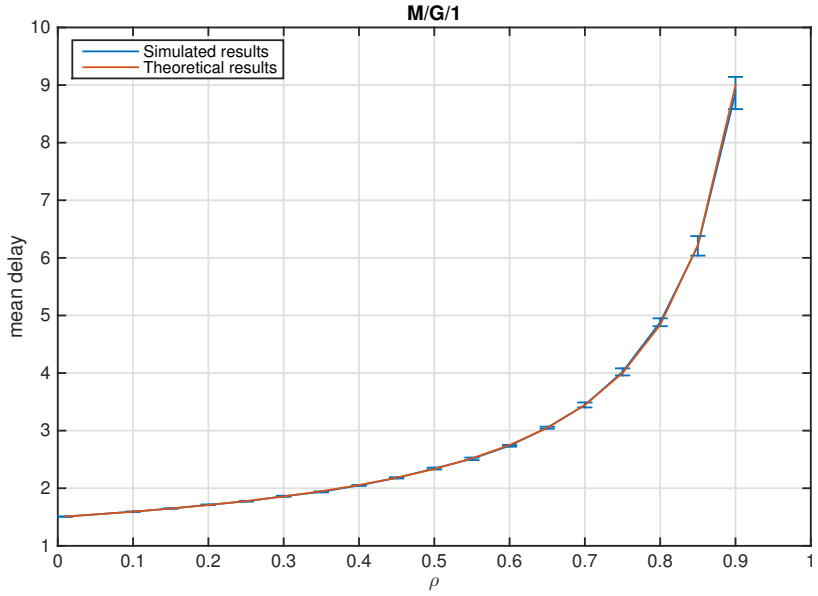


Figure 3: Delay vs. ρ for the described M/G/1 queue given $l_{slot} = 1$

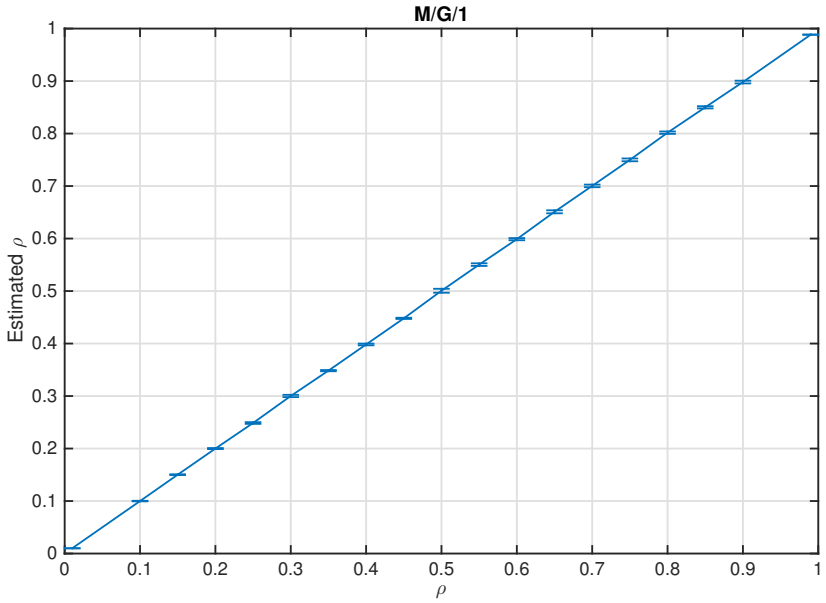


Figure 4: $\hat{\rho}$ vs. ρ for an M/G/1 queue

transitions:

$$X_{k+1} = \begin{cases} X_k - 1, & \text{wp } 0.5b \\ X_k, & \text{wp } 0.5 \\ X_k + 1, & \text{wp } 0.5(1-b) \end{cases} \tag{5}$$

which correspond to the events

- No arrivals, one departure
- No arrivals and no departures OR one arrival and one departure
- One arrival and no departures

Instead for $X_k = 0$ the transitions are

$$X_{k+1} = \begin{cases} 0, & \text{wp } 0.5 \\ 1, & \text{wp } 0.5 \end{cases} \tag{6}$$

Let's now compute the steady state distribution of the states, which will be used to average the number of users in the system. Let $\boldsymbol{\pi}$ be the vector of the steady state probabilities and $\mathbf{P}$ the transition probability matrix as follows

$$\mathbf{P} = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 & & \cdots \\ 0.5b & 0.5 & 0.5(1-b) & 0 & & \cdots \\ 0 & 0.5b & 0.5 & 0.5(1-b) & 0 & \cdots \\ 0 & 0 & 0.5b & 0.5 & 0.5(1-b) & 0 \quad \cdots \\ \vdots & \vdots & & & & \vdots \end{pmatrix} \tag{7}$$

Then the steady state distribution is the solution of

$$\boldsymbol{\pi} = \boldsymbol{\pi}\mathbf{P} \tag{8}$$

which corresponds to the following system

$$\begin{cases} \pi_0 = 0.5\pi_0 + 0.5b\pi_1 \\ \pi_1 = 0.5\pi_0 + 0.5\pi_1 + 0.5b\pi_2 \\ \vdots \\ \pi_i = 0.5(1-b)\pi_{i-1} + 0.5\pi_i + 0.5b\pi_{i+1}, \quad i \geq 2 \end{cases} \tag{9}$$

which yields to

$$\pi_1 = \frac{\pi_0}{b} \tag{10}$$

$$\pi_2 = \frac{1-b}{b}\pi_1 \tag{11}$$

$$\pi_3 = \frac{1-b}{b}\pi_2 \tag{12}$$

$$\tag{13}$$

and so on. This means that

$$\pi_i = \frac{(1-b)^{i-1}}{b^i}\pi_0, \quad i \geq 1 \tag{14}$$

and that π_0 can be computed in the following way

$$\sum_{i=0}^{\infty} \pi_i = 1$$

$$\pi_0 \left(1 + \sum_{i=1}^{\infty} \frac{(1-b)^{i-1}}{b^i}\right) = 1$$

$$\pi_0 \left(1 + \frac{1}{b}\sum_{i=1}^{\infty} \left(\frac{1-b}{b}\right)^{i-1}\right) = 1$$

$$\pi_0 \left(1 + \frac{1}{b}\frac{1}{1-\frac{1-b}{b}}\right) = 1$$

and after some algebraic passages

$$\pi_0 = \frac{2b-1}{2b} \tag{15}$$

The steady state distribution can be used to compute the mean number of users in the system:

$$m_x = \sum_{i=0}^{\infty} i\pi_i$$
$$m_x = \sum_{i=0}^{\infty} i\frac{(1-b)^{i-1}}{b^i}\pi_0$$
$$m_x = \frac{\pi_0}{b}\sum_{i=0}^{\infty} i\left(\frac{1-b}{b}\right)^{i-1}$$
$$m_x = \frac{\pi_0}{b}\frac{b^2}{(2b-1)^2}$$
$$m_x = \frac{2b-1}{2b}\frac{b}{(2b-1)^2}$$
$$m_x = \frac{1}{2(2b-1)}$$

Finally, by considering the system arrival rate as $\lambda_a = b_a = 0.5$ and using Little's Law it is possible to compute the mean time in the system spent by each user as

$$m_d = \frac{m_x}{b_a} = \frac{2}{2(2b-1)} = \frac{1}{2b-1} \tag{16}$$

The results of the simulations and the theoretical curve for the delay are in Figure 5, while the estimated utilization factor is in Figure 6.

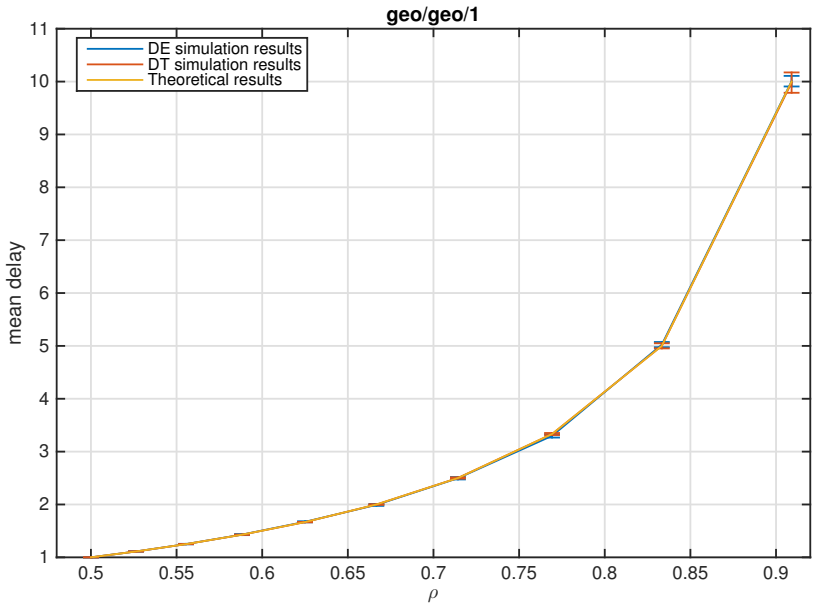


Figure 5: Delay vs. ρ for the described geo/geo/1 queue given $l_{slot} = 1$

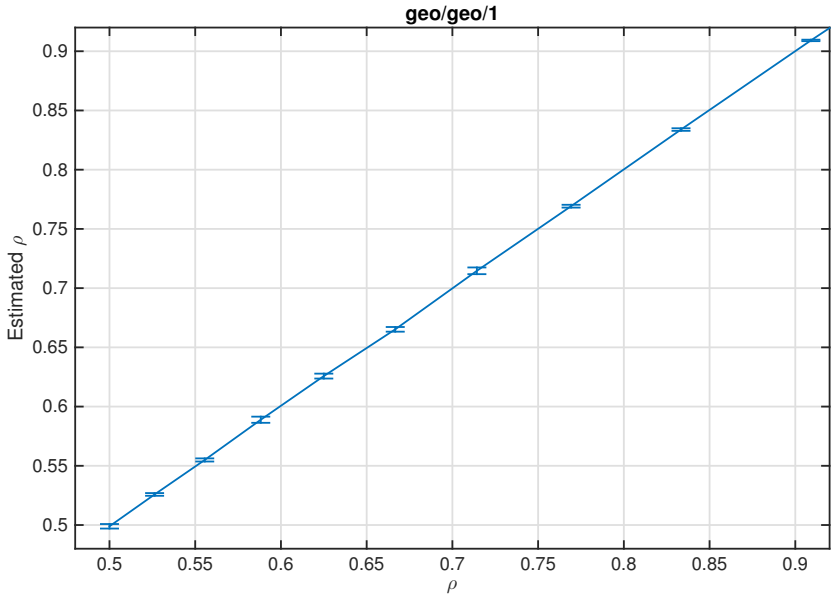


Figure 6: $\hat{\rho}$ vs. ρ for a geo/geo/1 queue

Exercise 2 - Read logs and analyze data

In this second exercise the goal was to extract a look-up table from a log file of a simulator and use it to compute some measurements on useful quantities in underwater optical communications. In particular, the log files are obtained from the simulations of the ambient light irradiance E_0 with the simulator HYDROLIGHT. From the 3 different dump files (each for a different value of $c \in [0.15, 0.4, 2.19]$ with c the attenuation coefficient) I recovered the values of E_0 and the corresponding depth z in meters using a `perl` script, which I provide in the attached code. It uses a regular expression to identify the right rows of the log and selects the columns with the desired values with a typical `perl` construct. In Figure 7 E_0 is plotted as a function of the depth z.

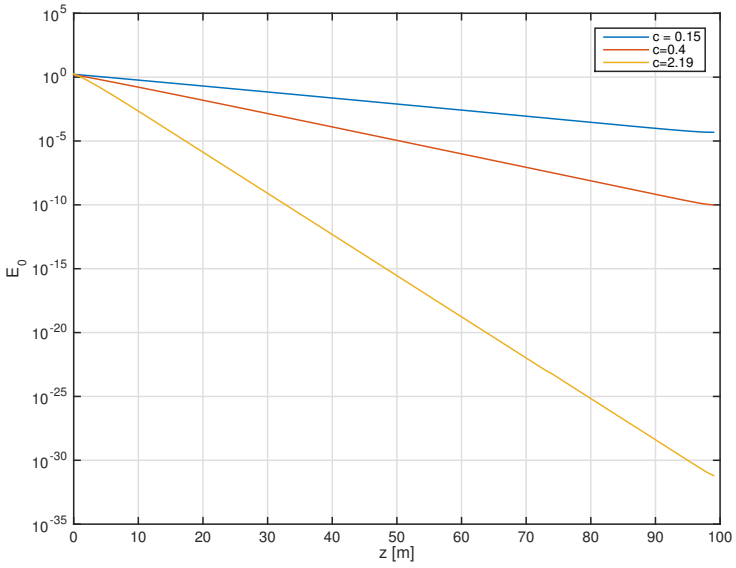


Figure 7: Irradiance E_0 vs. z

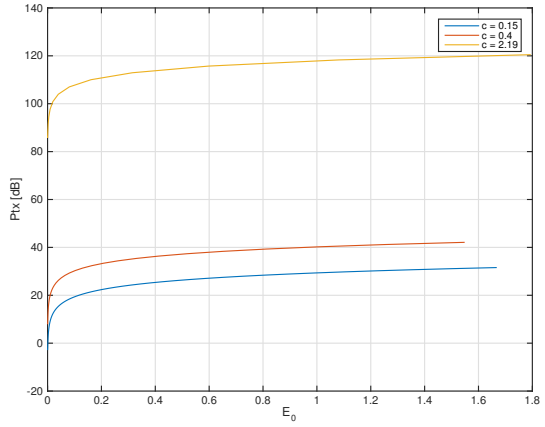
Some evaluation on the propagation are then carried out with a MATLAB script that uses the provided parameters for receiver and transmitter. The propagation model used the one proposed in [4] and assumes that there is a successful transmission if the SNR is over a certain threshold. In particular from the irradiance and the parameters S (sensitivity) and A_r (area of the receiver) of the considered modem it is possible to compute the power of ambient light noise, which is $N_A = (SE_0A_r)^2$. Let the received power P be

$$P = \frac{2P_{tx}A_r\cos(\beta)}{\pi d^2(1-\cos(\theta)) + 2A_t}e^{-cd} \tag{17}$$

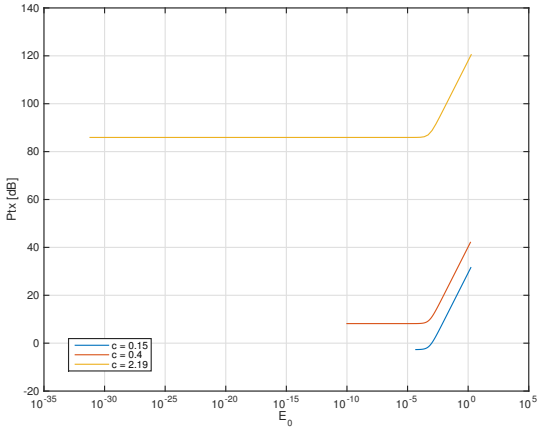
then the SNR is

$$\Gamma = \frac{(SP)^2}{2q(I_D + I_L)BW + \frac{4KTBW}{R} + N_A} \tag{18}$$

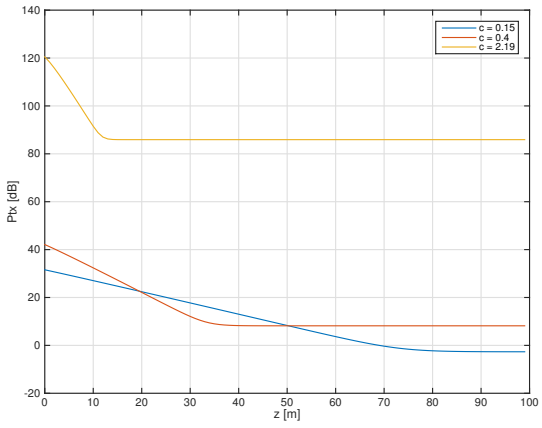
In the following, if the SNR is above the threshold of 20 dB then there is a successful transmission. In particular, by setting the threshold and a distance $d = 10$ m, by inverting (18) in order to get P and (17) in order to compute P_{tx} it is possible to know which is the minimum transmission power that can be used to reach a distance d. In Figure 8[a] there is the plot of the minimum P_{tx} against the irradiance E_0 for $d = 10$ m, while in Figure 8[c] the x axis is the depth z considered. Figure 8[b] shows instead the same as Figure 8[a] but with a logarithmic x axis, in order to visualize the behavior of the required transmitted power for very low values of E_0. It is interesting to note that for E_0 below a threshold of about 10^{-4} the required power is constant.



(a) P_{tx} to reach $d = 10$ m as a function of E_0



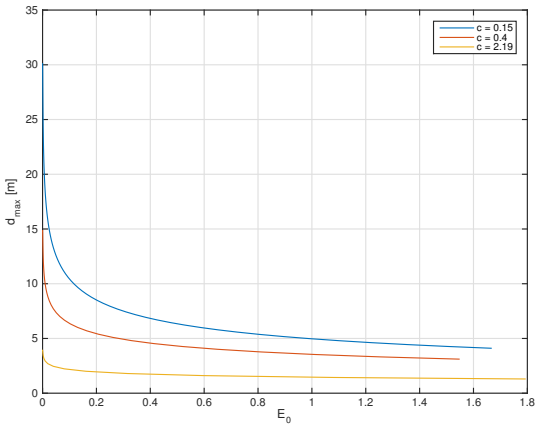
(b) P_{tx} to reach $d = 10$ m as a function of E_0, with logarithmic x axis



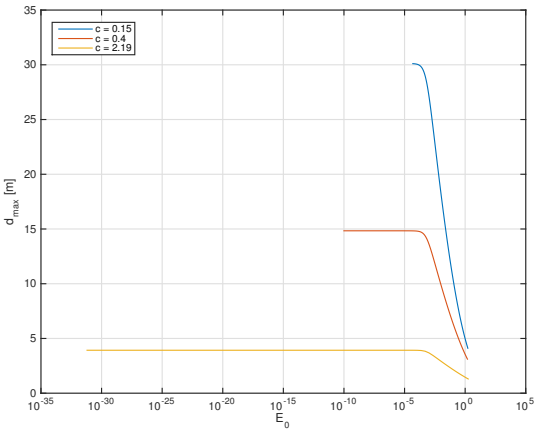
(c) P_{tx} to reach $d = 10$ m as a function of z

Figure 8: Transmitted power P_{tx} required to transmit at a distance of $d = 10$ meters

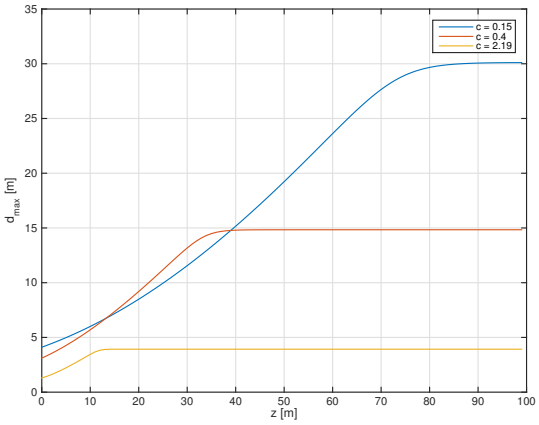
Finally, the transmit power is set to $P_{tx,max} = 100$ W and the maximum distance at which is possible to transmit is evaluated. Note that (17) is not invertible with respect to the distance d, therefore the received power P given by (17) is precomputed for $d \in [0, 50]$ m and stored in a vector `P_d`. Then, given a certain E_0, from (18) the minimum received power to guarantee an SNR of 20 dB is computed and the closest P in `P_d` is found. The corresponding value of d is chosen as the maximum distance d_{max} at which it is possible to transmit with $P_{tx,max} = 100$ W and that E_0 - or the corresponding z.
The results are in Figure 9. Figure 9[b] is the same as Figure 9[a] but with a logarithmic x axis, and it shows a behavior which is consistent with Figure 8[b].



(a) d_{max} as a function of E_0, for $P_{tx,max} = 100$



(b) d_{max} as a function of E_0 with a logarithmic x axis, for $P_{tx,max} = 100$



(c) d_{max} as a function of z, for $P_{tx,max} = 100$

Figure 9: Maximum distance d_{max} with a transmit power of $P_{tx,max} = 100$

References

[1] A.M. Law, Simulation Modeling and Analysis, 4th ed., McGraw Hill, 2006

[2] N. Benvenuto, M. Zorzi, Principles of communications networks and systems, Wiley, 2011

[3] Y. Le Boudec, Performance Evaluation of Computer and Communications Systems, EPFL, 2015

[4] D. Anguita et al., Optical wireless underwater communication for AUV: Preliminary simulation and experimental results, in Proc. IEEE/OES Oceans, Santander, Spain, Jun. 2011