Lab_07

April 7, 2025

Probability for Data Science

UC Berkeley, Spring 2025

Michael Xiao and Ani Adhikari

CC BY-NC-SA 4.0

```
[1]: # SETUP
     import warnings
     warnings.filterwarnings('ignore')
     from datascience import *
     from prob140 import *
     import numpy as np
     %matplotlib inline
     import matplotlib.pyplot as plt
     import matplotlib.patches as patches
     plt.style.use('fivethirtyeight')
     import pylab
     from scipy import stats
     import ipywidgets as widgets
     from ipywidgets import interact
```

```
[2]: def createHittingTimes(lamb, total_time):
         waiting_times = np.array([])
         while True:
             newTime = np.random.exponential(1/lamb)
             if sum(waiting_times) + newTime > total_time:
                 break
             waiting_times = np.append(waiting_times, newTime)
         hitting_times = np.cumsum(waiting_times)
         return hitting_times

     def plotHittingTimes(hitting_times, total_time, lamb):
         Plot_abstract(
             lambda i, time: plt.plot(time, 0, color='red', marker='o', ms = 5),
             lamb=lamb, t=total_time, mult=1, hitting_times=hitting_times,
       →figsize=(10,1)
```

```
        )
        plt.text(total_time, -.2, total_time, horizontalalignment='left')
        plt.xlim(-0.1, 1.1)
```

```
[3]: def Plot_abstract(plot_each, lamb, t, mult=1.5,
     hitting_times=None,figsize=(10,2)):
         plt.figure(figsize=figsize)
         if hitting_times is None:
             hitting_times = createHittingTimes(lamb, mult*t)

         plt.arrow(0, 0, mult*t, 0, width=.005*figsize[1]/2, color='black',
     head_width=0.1*min(1,t*mult/4))
         #print(min(0.1,.1*figsize[1]/4*t*mult))
         for i,time in enumerate(hitting_times):
             plot_each(i, time)
         plt.vlines(0, -1/20., 1/20., lw=2)
         plt.text(0, -.2, '0', horizontalalignment='right')
         # plt.text(1.5*t, -.2, 'Time', horizontalalignment='center')
         width = min(0.5, 0.5*t/4)
         plt.ylim(-1 * width, width)
         plt.xlim(-1, mult*t + 1)
         plt.axis('off');
         return hitting_times

     def Plot_process():
         def plot_each(i, time):
             plt.plot(time, 0, 'ro', ms = 5, color='red')
         Plot_abstract(plot_each, lamb=1, t=5)
         plt.text(1.5*2.5, -.4, 'time', horizontalalignment='center')

     def Plot_waiting_times():
         def plot_each(i, time):
             plt.plot(time, 0, 'ro', ms = 5, color='red')
             if i <= 2:
                 plt.text(time, -.2, '$T_%d$'%(i+1), horizontalalignment='center')

         Plot_abstract(plot_each, lamb=1/3, t=5)
         plt.title('$T_r = $ Time of the $r$th Arrival', fontsize=15)

     def Plot_inter_arrival_times():
         previous_time = 0
         colors = ['green', 'purple', 'orange']

         def plot_each(i, time):
             nonlocal previous_time
             plt.plot(time, 0, 'ro', ms = 5, color="red")
             if i <= 2:
```

```python
            plt.hlines(0, previous_time, time, color=colors[i], lw=2)
            plt.text(time, -.2, '$T_%d$'%(i+1), horizontalalignment='center')
            previous_time = time
    Plot_abstract(plot_each, lamb=1/3, t=5)
    plt.title('Times Between Consecutive Arrivals', fontsize=15);


def waiting_time_tail(r):
    t = 4
    def plot_each(i, time):
        plt.plot(time, 0, 'ro', ms = 5, color="red")
    hitting_times = Plot_abstract(plot_each, 0.5, 4, 1.75)
    if sum(hitting_times < t) < r:
        answer = 'Yes.'
    else:
        answer = 'No.'
    plt.vlines(t, 0., 2, color="green", linewidths=(2,))
    plt.text(t, -.2, "t", horizontalalignment='center')
    plt.title('For $r = $'+str(r)+', is $T_r > t$ ?  '+answer);


def thinHittingTimes(lamb, total_time, p_of_blue):
    allHittingTimes= createHittingTimes(lamb, total_time)
    which_blue = np.random.rand(len(allHittingTimes)) < p_of_blue
    blue_hits = allHittingTimes[np.nonzero(which_blue)]
    red_hits = allHittingTimes[np.nonzero(1-which_blue)]
    return blue_hits, red_hits


def Plot_thinned_processes(lamb, p_blue):
    blue, red = thinHittingTimes(lamb, 1, p_blue)
    def plot_each(i, time): # this is jank af
        color = 'blue' if time in blue else 'red'
        plt.plot(time, 0, 'ro', ms = 5, color=color)
    Plot_abstract(plot_each, lamb=lamb, t=1, mult=1,␣
 ↪hitting_times=list(blue)+list(red), figsize=(10,1))
    plt.title("Original Process", fontsize=10)
    plt.xlim(-0.1, 1.1 )

    Plot_abstract(plot_each, lamb=lamb, t=1, mult=1, hitting_times=blue,␣
 ↪figsize=(10,1))
    plt.title("Blue Process", fontsize=10)
    plt.xlim(-0.1, 1.1 )

    Plot_abstract(plot_each, lamb=lamb, t=1, mult=1, hitting_times=red,␣
 ↪figsize=(10,1))
    plt.title("Red Process", fontsize=10)
    plt.xlim(-0.1, 1.1 )
```

```python
[4]: def draw_rect(ax, x, y, w, h, col, opaqueness):
         ax.add_patch(patches.Rectangle((x, y),w,h,color=col, alpha=opaqueness))


     def plotDiscreteIntervals(hitting_times, total_time, intervals):
         fig1 = plt.figure(figsize=(10, 1))
         ax1 = fig1.add_subplot(111)
         plt.hlines(0, 0, total_time, lw=2)
         plt.text(0, -.2, '0', horizontalalignment='right')
         plt.text(total_time, -.2, str(total_time), horizontalalignment='left')
         plt.ylim(-.5, .5)
         plt.xlim(-0.1, total_time + 0.1 )
         plt.text(total_time/2, -.4, 'time', horizontalalignment='center')
         plt.axis('off')

         end_points = [i * total_time/intervals for i in range(0, intervals+1)]
         plt.vlines(end_points, -1/40., 1/40., lw=2) #remove this line to get rid of
     ↪tick marks
         for i in range(0, intervals):
             left_endpoint = end_points[i]
             right_endpoint = end_points[i+1]
             for time in hitting_times:
                 if time <= right_endpoint and time >= left_endpoint:
                     draw_rect(ax1, left_endpoint, 0, total_time/intervals, 1/20,
     ↪'red', 1)
                     break

     def Plot_discretized_process(lamb):
         total_time = 1
         hitting_times = createHittingTimes(lamb, total_time)
         uniform_slider = widgets.IntSlider(
             value=1,min=1,max=100,step=1, description='Subintervals')
         @interact(intervals = uniform_slider)
         def plot(intervals):
             plotHittingTimes(hitting_times, total_time, lamb)
             plotDiscreteIntervals(hitting_times, total_time, intervals)
             plt.show()
```

# 1 Lab 7: Poisson Process (Part A Due Monday, April 7th at 5 PM, Part B Due Monday, April 14th at 5 PM)

A *point process* on the real line is a random sequence of points on the line. Often the line is thought of as an axis representing time. Then the points can be thought of as the random times of arrivals of some kind of item or occurrences of some kind of event.

In this lab you will analyze a point process called the *Poisson process*. It records the times of events that are essentially unpredictable – events that appear to occur randomly and spontaneously with no

prior warning. The process is used as a model in many areas, including queuing theory (items joining a queue or leaving it), physics (a Geiger counter recording particles), genetics (nucleotide base substitutions in a gene over time), computer vision (photons arriving at a sensor), and seismology (the times of earthquakes).

Run the cell below several times to see the main visualization that will be used throughout the lab.

```
[5]: Plot_process()
```
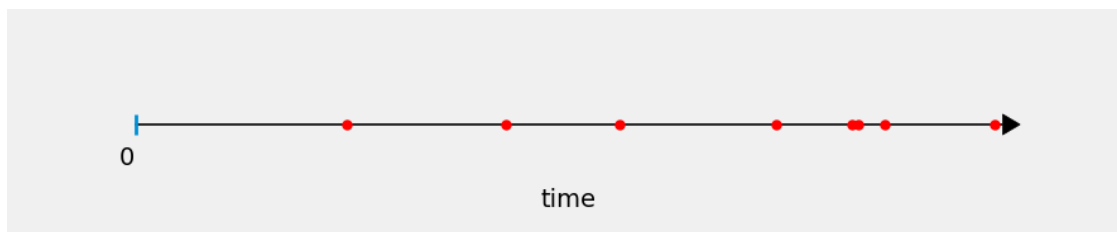

The horizontal line represents time. The arrow on the right signifies that the axis continues beyond the interval that is displayed. The time at which you start observing the process is labeled 0 and is called the time origin. Each red dot represents an arrival.

By assumption, **there are no double-hits** in the arrival process. That is, it is not possible for two arrivals to happen at exactly the same time.

Because the red dots have a diameter big enough for you to see them, two dots might have a considerable overlap if two arrivals are close enough in time. That doesn't mean that the two arrivals are at the same time – they can't be, by assumption.

It is also important to note that **the chance of an arrival at any fixed time is 0**. This is a feature of continuous time, analogous to the chance that a continuous random variable equals any fixed value is 0.

Therefore if you are counting arrivals in a time interval, you shouldn't worry about whether or not the interval includes its endpoints.

What you will learn in this lab:

- Two equivalent descriptions of the process, in terms of independent Poisson and exponential random variables
- A physical model for gamma random variables with integer shape parameters, and a formula for the cdf of the gamma (r, λ) distribution for integer r
- What happens if you randomly select arrivals in a Poisson process
- A way of thinking about statistical models for physical phenomena
- Applications of the Poisson process model, in particular to predicting the chances of earthquakes in California

Be prepared for a workout in numerous ideas you have learned in the course.

Note on Computation   Several exercises in this lab ask you to use a code cell "find the probability" or "find the value" of something. That means you should write a Python expression that

evaluates to the quantity in the question. In some exercises, there are restrictions on what you can use as part of the expression.

2 Part A of the lab starts here and is due on Monday, April 7th at 5 PM.

2.1 Identify Your Lab Partner

This is a multiple choice question. Please select **ONE** of following options that best describes how you complete **Part A** of this lab.

- I am doing Part A of this lab by myself and I don't have a partner.
- My partner for Part A of this lab is [PARTNER'S NAME] with email [berkeley.edu email address]. [SUBMITTER'S NAME] will submit to Gradescope and add the other partner to the group on Gradescope after submission.

Please copy and paste **ONE** of above statements and fill in blanks if needed. If you work with a partner, make sure only one of you submit on Gradescope and that the other member of the group is added to the submission on Gradescope. Refer to the bottom of the notebook for submission instructions.

I am doing Part A of this lab by myself

2.2 Section 1: Arrival Counts

To start off, you will examine how the Poisson distribution arises out of a simple set of assumptions about the process of arrivals.

2.2.1 Rate

The process has a numerical parameter $\lambda > 0$ that is the expected number of arrivals per unit time. The arrivals are random, of course, so there will most likely be different numbers of arrivals in successive units of time. You can think of λ as an average over time.

The higher λ is, the more arrivals you expect per unit time. So λ is called the *intensity* of the process, or the *arrival rate*, which is usually shortened to just *rate*.

In our code, we will use `lam` for λ because `lambda` has a different meaning in Python.

Notice that the rate is a constant, not a function of time. That is, the intensity of arrivals is the same at all times. Under this assumption the process is called *time-homogeneous* or just *homogeneous*.

You should only use this model in situations where the assumption of homogeneity is reasonable. You would not want to assume time homogeneity in situations where the intensity of arrivals varies over time. For example, If you are modeling people arriving at the ticket counter of a movie theater, you might expect more crowding right before movie start times than at other times.

2.2.2 Discretizing the Arrival Times

To understand the process in continuous time, it helps to create a discrete approximation.

- The first graph in the output of the cell below is a realization of the process, which we are watching on the time interval 0 to 1. We have specified $\lambda = 5$ but you can change that.
- The second graph consists of discrete approximations created by dividing the unit interval into a finite number of subintervals of equal length. A subinterval is colored red if at least one of the arrivals is in that subinterval.

At the start there is only one "sub"-interval in the second graph, and that is the entire unit interval. So the whole interval is colored red. Now move the slider (slowly!) and see what happens as you divide the unit interval more finely.

```
[6]: lam = 5
     Plot_discretized_process(lam)
```

```
interactive(children=(IntSlider(value=1, description='Subintervals', min=1),␣
 ↪Output()), _dom_classes=('widget-…
```

As the number of subintervals increases, each subinterval gets smaller and the second graph starts to look more and more like the first.

2.2.3 a) [ON PAPER] The Number of Arrivals in the Unit Interval

Formally, the discretization can be described as follows.

- Fix a non-negative integer n. At Stage n, divide the unit interval into n subintervals of length $1/n$ each.

- For $1 \leq j \leq n$, let I_j be the indicator that there is at least one arrival in subinterval j. That is, let I_j be the indicator of the event that subinterval j is colored red.

Here are the probabilistic **assumptions of the model**. The main assumption is the first one, which reflects the "unpredictable" nature of the arrivals.

- $I_1, I_2, \ldots, I_n$ are i.i.d. Bernoulli (p_n) variables.
- The sequence $p_1, p_2, \ldots$ is *consistent*, that is, $p_n \downarrow 0$ as $n \to \infty$ and $np_n \to \lambda$ as $n \to \infty$.
- Let X be the number of arrivals in the unit interval. Then $E(X) = \lambda$.

Fill in the blanks and **explain briefly**. For both (i) and (ii), the first blank should be filled with the name of a distribution and its parameters.

(i) Fix n and let X_n be the number of red subintervals at stage n. The exact distribution of X_n is _____, and $E(X_n) = $ _____.

(ii) As $n \to \infty$, X_n converges to the number of arrivals in the unit interval. The distribution of X_n tends to the _____ distribution, which has expectation _____.

2.2.4 b) [ON PAPER] Poisson Process: First Description

One of the key assumptions of the process is that the numbers of arrivals in disjoint time intervals are independent of each other. In Part **a**, this was reflected in the assumption that at each stage n, the n subintervals are colored red independently of each other.

For $0 < a < b$, let $N_{(a,b)}$ be the number of arrivals in the time interval (a, b). Remember that the chance of an arrival at a single point (for example, at a, or at b) is 0.

(i) What is the distribution of $N_{(0,2)}$? Why? What assumptions did you make?

(ii) What is the distribution of $N_{(5,12)}$?

Poisson Process: First Description   The following description of the Poisson process will now come as no surprise.

Fix $\lambda > 0$. A *time-homogeneous Poisson process with rate* λ is a process of arrivals such that:

- The numbers of arrivals in disjoint intervals are independent.
- The number of arrivals in an interval of length t has the Poisson distribution with mean (λt).

The parameter λ has the same interpretation as before: it is the expected number of arrivals in an interval of length 1.

Remember that the term *time-homogeneous* refers to the fact that the distribution of the number of arrivals in an interval depends only on λ and the length of the interval, not on where the interval is on the time axis. For example, the number of arrivals in the interval $(1, 10)$ has the same distribution as the number of arrivals in the interval $(1001, 1010)$.

The assumption of time-homogeneity simplifies calculations, and in many situations it can be too simple as a model. Nevertheless it is often a good starting point in analyses. Sometimes it can be justified by physics, for example if the process consists of radioactive particles arriving at a Geiger counter, or photons arriving at a sensor.

2.2.5 c) Photon Noise

In a brief report written by Sam Hasinoff of Google, photon noise is described as "uncertainty associated with the measurement of light". Understanding photon noise helps increase clarity in imaging. Skim the beginning of the Theory section of the report to recognize a familiar process. You might also enjoy skimming the list of Hasinoff's publications on computer vision.

Suppose photons arrive at a sensor according to a Poisson process at a rate of 4 arrivals per unit time. What is the chance of at least 30 arrivals in each of two consecutive 10-unit time intervals?

```
[16]: lam = 4*10
      interval_each = 1 - stats.poisson.cdf(29,40)
      total_prob = interval_each * interval_each
      total_prob
```

[16]: 0.91541135465710466

2.3 Section 2: Waiting Times

The first description of the Poisson process involved the counts of arrivals in finite time intervals. Another way to look at the process is to just list the times of the arrivals.

For $r \geq 1$, let T_r be the time of the rth arrival.

The cell below shows a Poisson arrival process with T_1, T_2, and T_3 appearing as labels on the time axis. There might be fewer than three arrivals, in which case there will be fewer than three labels. Or there might be more than three arrivals, but only the first three will be labeled.

```
[8]: Plot_waiting_times()
```

$T_r =$ Time of the rth Arrival
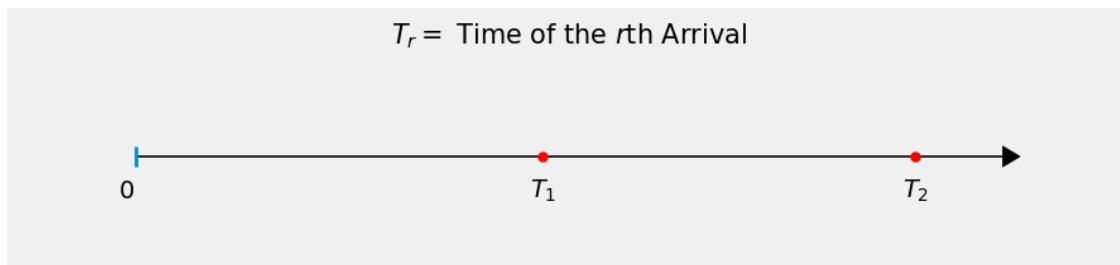
2.3.1 a) [ON PAPER] Distribution of T_1

In this exercise you will find the distribution of T_1, the waiting time till the first arrival, by identifying its survival function.

Run the cell below. In the figure, t is an arbitrary fixed time and the function is checking whether or not the event $\{T_1 > t\}$ has occurred.

When you see a No, think about what has to happen for the answer to be Yes. You might have to run the cell several times (ctrl-Return is helpful) to get a Yes as the answer.

```
[19]: r = 1
      waiting_time_tail(r)
```

For $r = 1$, is $T_r > t$? Yes.
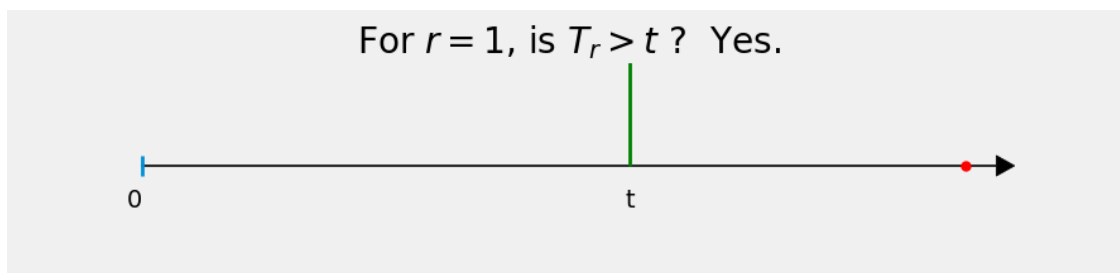
(i) Fill in the blanks based on your observations above. The first blank should be filled with math operations and numerical values, and the second with a formula involving λ and t.

For $t > 0$,

$$P(T_1 > t) \;=\; P(N_{(0,t)} \; \underline{\hspace{4cm}} \;), \quad \text{so} \quad P(T_1 > t) \;=\; \underline{\hspace{4cm}}.$$

(ii) Fill in the first blank with the appropriate distribution and its parameters, and the second with the expectation.

The distribution of T_1 is $\underline{\hspace{5cm}}$ and $E(T_1) = \underline{\hspace{2.5cm}}$.

2.3.2   b) Inter-Arrival Times

Let the *inter-arrival times* be the *gaps* or lengths of time between consecutive arrivals. By convention, the first "inter"-arrival time is simply taken to be T_1. In general, define the inter-arrival times as follows, using the letter G for "gap":

- $G_1 = T_1$
- For $i > 1$, $G_i = T_i - T_{i-1}$.

Run the cell below several times. The figure labels T_1, T_2, and T_3 as before. The length of the green segment is G_1, the length of the purple segment is G_2, and the length of the orange segment is G_3.

Remember that there might be fewer than three arrivals in the interval; there might be more, but we aren't labeling them all; and the arrivals might be so close that it's hard to read the labels.

[10]: `Plot_inter_arrival_times()`

Times Between Consecutive Arrivals
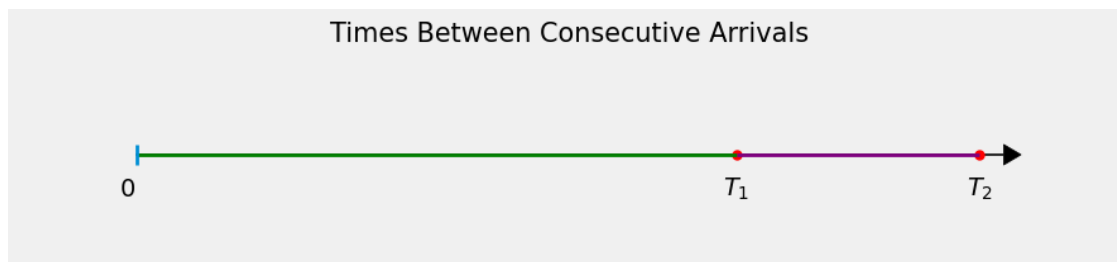
It can be shown that the process "starts over" at each T_i independently of what happened before. For example, if you stand at T_2 and look right, it's as though you were standing at the origin and observing a fresh process. This should not be very surprising. Indeed, it should remind you of many "starting over" calculations that you have done in the past.

What is the distribution of G_2? Why?

The distribution of G_2 is exponential(λ) because $G_1 = T_1$, which follows the same exponential distribution. Since the waiting times between arrivals are independent, the process resets after each arrival, allowing the distribution of the time between any two arrivals to always be the same exponential($\$ $).

For fixed n, what is the joint distribution of $G_1, G_2, \ldots, G_n$?

12

The fixed n, the joint distribution of $G_1, G_2, \ldots, G_n$ will be the product of all n of exponential(λ) because each of G_1 is independent and follows the same exponential distribution.

2.3.3 Poisson Process: Second Description

We now have a new description of the Poisson process in terms of waiting times.

Fix $\lambda > 0$. A *time homogeneous Poisson process with rate λ* is a process of arrivals such that:

- the inter-arrival times are i.i.d. exponential (λ) variables. The rate λ of the exponential distribution of gaps is the same as the arrival rate of the process.

At least informally, you have seen why the first description (in terms of Poisson counts, in **1c**) implies the second. A bit of work is needed to establish that the second description implies the first, but we'll skip that.

It is important to keep in mind is that the second description is in terms of *continuous* random variables, the inter-arrival times. The first description is in term of *discrete* random variables, the arrival counts.

2.3.4 c) [ON PAPER] Distribution of T_r for $r \geq 1$

Fix an integer $r > 0$ and let T_r be the time of the rth arival.

```
[11]: Plot_waiting_times()
```
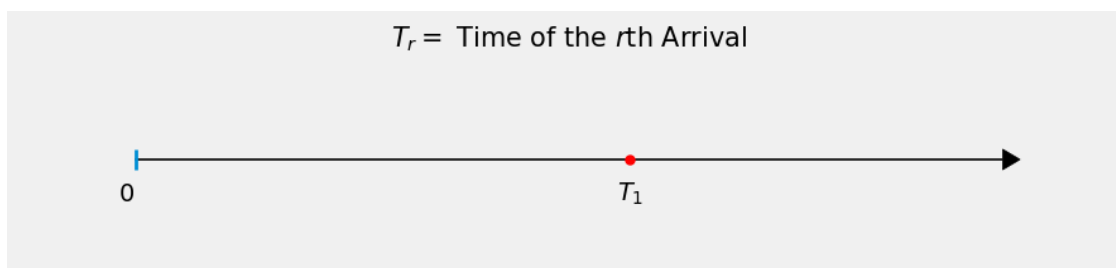

As before, let the inter-arrival times G_i be defined as $G_1 = T_1$ and $G_i = T_i - T_{i-1}$ for $i > 1$.

For integer $r > 0$, write T_r in terms of $G_1, G_2, \ldots$ and hence find

(i) $E(T_r)$

(ii) $Var(T_r)$

(iii) the distribution of T_r
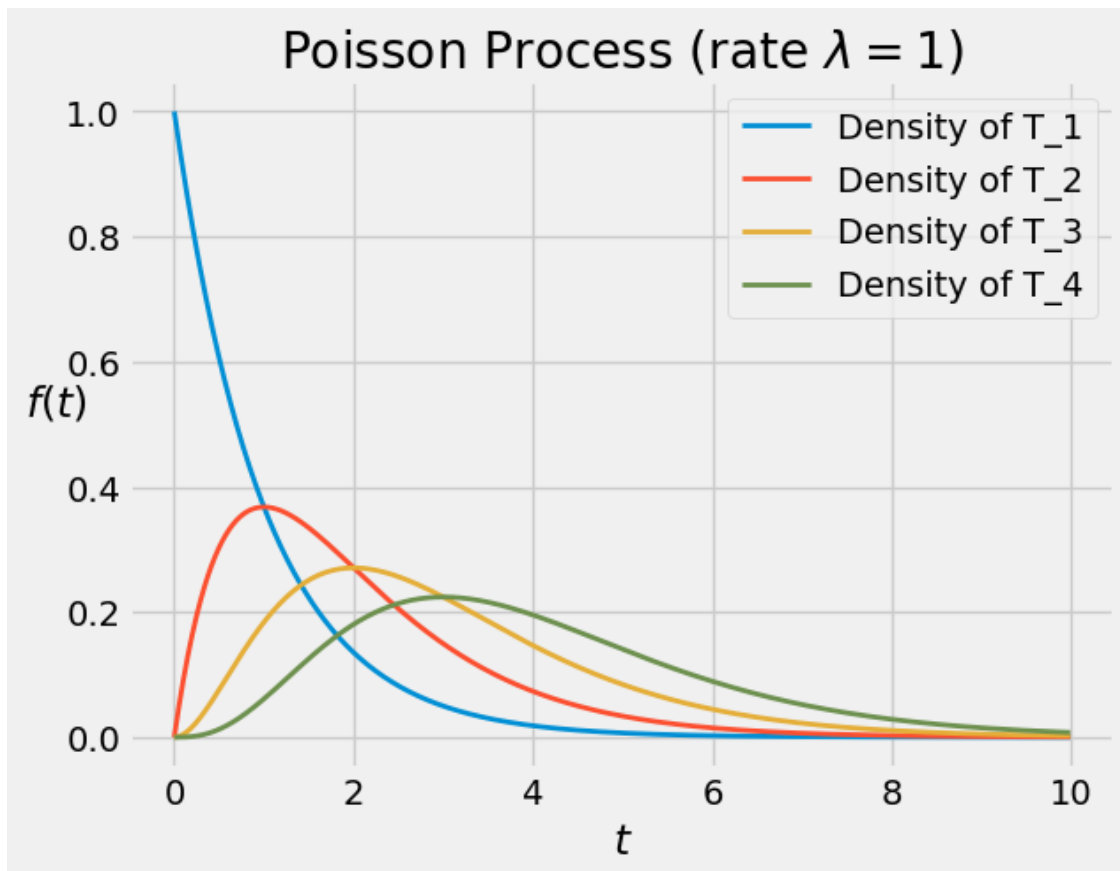
2.3.5 d) Density of T_r

Fill in the blank in the cell below to draw graphs of the density of T_r for $r = 1, 2, 3, 4$ in the case $\lambda = 1$. Refer to Homework 8 for relevant code.

```
[12]: t = np.arange(0, 10, 0.01)
      for r in np.arange(1, 5):
```

```
    f = stats.gamma.pdf(t,r,scale=1)
    plt.plot(t, f, lw=2, label='Density of T_'+str(r))  # plot the density of␣
 ↪T_r
plt.xlabel('$t$')
plt.ylabel('$f(t)$', rotation=0)
plt.legend()
plt.title('Poisson Process (rate $\lambda = 1$)');
```


2.3.6   e) [ON PAPER] CDF of T_r

You can write a formula for this in terms of the density you found in Part **d**, but you will not enjoy calculating the integral unless you are unusually fond of repeated integration by parts.
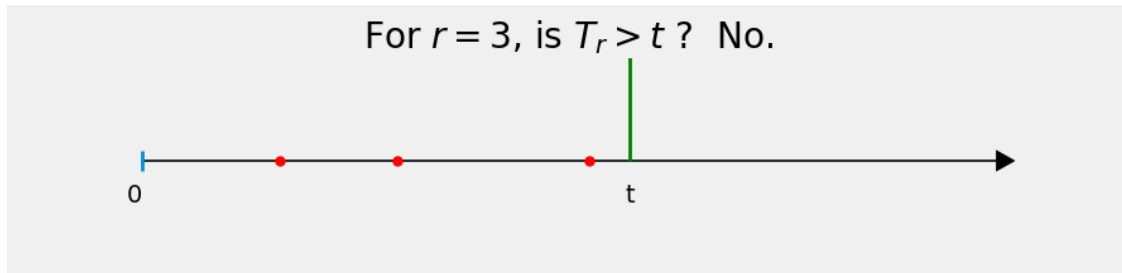
Instead, use the fact that the Poisson process has a representation in terms of counts, which are discrete variables and hence easy to work with.

Run the cell below several times. You used the same function in Part **a**.

```
[13]: r = 3
      waiting_time_tail(r)
```

For $r = 3$, is $T_r > t$? No.

Fill in the blanks below based on what you concluded after repeatedly running the cell above. Here S_{T_r} is the survival function of T_r, and F_{T_r} is the cdf of T_r.

Fix a positive integer r. For $t > 0$,

$$P(T_r > t) \;=\; P(N_{(0,t)} \underline{\hspace{2cm}}), \quad \text{so} \quad S_{T_r}(t) \;=\; \underline{\hspace{2cm}} \quad \text{and} \quad F_{T_r}(t) \;=\; \underline{\hspace{2cm}}.$$

2.3.7   f) Repair Times

Homogeneous Poisson processes are often used in reliability theory, for example to model failures of a component of a machine, assuming that the failures happen unpredictably. The component might be as simple as a lightbulb, or more complicated such as an electronic component of a cell phone.

Suppose calls for repair occur according to a Poisson process at a rate of one every two days. In your calculations below, **use only stats.poisson.pmf, stats.poisson.cdf, and arithmetic operations.** Don't use any continuous distributions.

    (i) If you start recording calls now, what is the chance that it takes more than 15 days for you to record 10 calls?

[20]:
```
probability = stats.poisson.cdf(9, 15/2)
probability
```

[20]: 0.77640761301971462

    (ii) What is the chance that the fifth call arrives less than a week after the second one?

It might help to sketch a time axis on some scratch paper and mark the second call.

[21]:
```
probability = 1 - stats.poisson.cdf(2, 7/2)
probability
```

[21]: 0.6791528011378658

15

3   Part A ends here and is due on Monday, April 7th at 5 PM.

3.1   Submission Instructions

Many assignments throughout the course will have a written portion and a code portion. Please follow the directions below to properly submit both portions.

3.1.1   Written Portion

- Scan all the pages into a PDF. You can use any scanner or a phone using applications such as CamScanner. Please **DO NOT** simply take pictures using your phone.
- Please start a new page for each question. If you have already written multiple questions on the same page, you can crop the image in CamScanner or fold your page over (the old-fashioned way). This helps expedite grading.
- It is your responsibility to check that all the work on all the scanned pages is legible.
- If you used LaTeX to do the written portions, you do not need to do any scanning; you can just download the whole notebook as a PDF via LaTeX.

3.1.2   Code Portion

- Save your notebook using `File > Save and Checkpoint`.
- Generate a PDF file using `File > Download As > PDF via LaTeX`. This might take a few seconds and will automatically download a PDF version of this notebook.
  - If you have issues, please post a follow-up on the general Lab 7A Ed thread.

3.1.3   Submitting

- Combine the PDFs from the written and code portions into one PDF. Here is a useful tool for doing so.
- Submit the assignment to Lab 7A on Gradescope.
- **Make sure to assign each page of your pdf to the correct question.**
- **It is your responsibility to verify that all of your work shows up in your final PDF submission.**

If you are having difficulties scanning, uploading, or submitting your work, please read the Ed Thread on this topic and post a follow-up on the general Lab 7A Ed thread.

1ai. $I_1, I_2, \ldots, I_n \sim$ iid Bernoulli(p_n)

    ↳ indicator values that can either
         represent an arrival (1) or no
         arrival (0) in the interval

    $X_n \sim$ # red subintervals at stage n
        $\therefore X_n = I_1 + I_2 + \ldots + I_n$

    $X_n \sim$ Binomial (n, p_n)

    $E(X_n) = n p_n$

1aii.   we know … $X_n \sim$ Binomial (n, p_n)
       - iid
         - small p_n
         - large n $(n \to \infty)$
            $\therefore$ definition of Poisson Distribution!

    $\therefore X_n$ tends to the Poisson(λ) distribution,
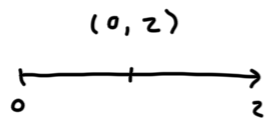   which has expectation λ

1bi.   iid $\to$ # arrivals in disjoint time
         interval don't affect each
         other + each length has
         Poisson(λ) arrival

           $(0,1)$
          ⊢————————→
        0 1

    $N_{(0,1)} =$ Poisson (λ)
       ↳ X_n for unit time $(0,1) \to$ Poisson(λ)

$$E[X_n] = \lambda$$

$$(0, 2)$$

$$\vdots \; N_{(0,2)} = \text{Poisson}\,(\lambda + \lambda)$$
$$= \text{Poisson}\,(2\lambda)$$

1bii. $$N_{(5,12)} = \text{Poisson}\,(\lambda\,(12 - 5))$$
$$= \text{Poisson}\,(7\lambda)$$

2ai. $T_1 \sim$ waiting time till the first arrival

$$\therefore P(T_1 > t) = P(\text{no arrivals in } (0, t))$$
$$= P(N_{(0,t)} = 0)$$

$$N_{(0,t)} = \text{Poisson } (\lambda t)$$
$$= e^{-\lambda t} \cdot \frac{(\lambda t)^0}{0!}$$
$$= e^{-\lambda t}$$

$$\therefore P(T_1 > t) = e^{-\lambda t}$$

2aii. $P(T_1 > t) = e^{-\lambda t}$
$$P(T_1 \le t) = 1 - P(T_1 > t)$$
$$= 1 - e^{-\lambda t}$$

$$f_{T_1}(t) = \lambda e^{-\lambda t}$$
$$\therefore T_1 \sim \text{Exponential} (\lambda)$$
$$E[T_1] = \frac{1}{\lambda}$$

2ci. $T_r = G_1 + G_2 + \dots + G_r$

$G_i \sim \text{Exponential } (\lambda)$

$$E[T_r] = E[G_1 + G_2 + \dots + G_r]$$
$$= E[G_1] + E[G_2] + \dots + E[G_r]$$
$$= r \cdot \frac{1}{\lambda}$$
$$= \frac{r}{\lambda}$$

2cii. $\text{Var}(T_r) = \text{Var}(G_1 + G_2 + \ldots + G_r)$

$$= \sum_{i=1}^{r} \text{Var}(G_i)$$

$G_1, G_2, \ldots G_r$ are independent

$$= r \cdot \text{Var}(G_i)$$

$$= r \cdot \frac{1}{\lambda^2}$$

$$= \frac{r}{\lambda^2}$$

2ciii. $T_r = G_1 + G_2 + \ldots + G_r$

$G_i \sim \text{Exponential}(\lambda)$

$\sim \text{Gamma}(1, \lambda)$

$\therefore T_r \sim \text{Gamma}(1 \cdot r, \lambda)$

$\sim \text{Gamma}(r, \lambda)$

2e. $P(T_r > t) = P\left(\begin{array}{c}\text{fewer than } r \text{ arrivals}\\ \text{in interval } (0, t)\end{array}\right)$

$$= P(N_{(0,t)} < r)$$
$$= P(N_{(0,t)} \le r - 1)$$

$N_{(0,t)} \sim \text{Poisson}(\lambda t)$

$\therefore S_{T_r}(t) = P(T_r > t)$

$$= \sum_{i=0}^{r-1} e^{-\lambda t} \cdot \frac{(\lambda t)^i}{i!} \qquad i = 0, 1, \ldots, r$$

$\therefore F_{T_r}(t) = P(T_r \le t)$

$$= 1 - P(T_r > t)$$

$$= 1 - \sum_{i=0}^{r-1} e^{-\lambda t} \frac{(\lambda t)^i}{i!} \qquad i = 0, 1, \ldots, r$$