# Homework 1

## References

- Lectures 1 through 4 (inclusive).

## Instructions

- Type your name and email in the "Student details" section below.
- Develop the code and generate the figures you need to solve the problems using this notebook.
- For the answers that require a mathematical proof or derivation you should type them using latex. If you have never written latex before and you find it exceedingly difficult, we will likely accept handwritten solutions.
- The total homework points are 100. Please note that the problems are not weighed equally.

In [ ]:
```
import matplotlib.pyplot as plt
%matplotlib inline
import matplotlib_inline
matplotlib_inline.backend_inline.set_matplotlib_formats('svg')
import seaborn as sns
sns.set_context("paper")
sns.set_style("ticks")

import numpy as np
import scipy
import scipy.stats as st
import urllib.request
import os

def download(
    url : str,
    local_filename : str = None
):
    """Download a file from a url.

    Arguments
    url            -- The url we want to download.
    local_filename -- The filemame to write on. If not
                      specified
    """
    if local_filename is None:
        local_filename = os.path.basename(url)
    urllib.request.urlretrieve(url, local_filename)
```

## Student details

- **First Name:** Kyle
- **Last Name:** Illenden
- **Email:** killende@purdue.edu
- **Used generative AI to complete this assignment (Yes/No):** Yes
- **Which generative AI tool did you use (if applicable)?:** ChatGPT

## Problem 1

Disclaimer: This example is a modified version of the one found in a 2013 lecture on Bayesian Scientific Computing taught by Prof. Nicholas Zabaras. I am not sure where the original problem is coming from.

We are tasked with assessing the usefulness of a tuberculosis test. The prior information I is:

> The percentage of the population infected by tuberculosis is 0.4%. We have run several experiments and determined that:

- If a tested patient has the disease, then 80% of the time the test comes out positive.
- If a tested patient does not have the disease, then 90% of the time the test comes out negative.

To facilitate your analysis, consider the following logical sentences concerning a patient:

> A: The patient is tested and the test is positive.

> B: The patient has tuberculosis.

A. Find the probability that the patient has tuberculosis (before looking at the result of the test), i.e., $p(B|I)$. This is known as the base rate or the prior probability.
**Answer:**

$p(B|I) = 0.004$

B. Find the probability that the test is positive given that the patient has tuberculosis, i.e., $p(A|B,I)$.
**Answer:**

$p(A|B,I) = 0.8$

C. Find the probability that the test is positive given that the patient does not have tuberculosis, i.e., $p(A|\neg B,I)$.
**Answer:**

$p(A|\neg B,I) = 1 - 0.9 = 0.1$

D. Find the probability that a patient that tested positive has tuberculosis, i.e., $p(B|A,I)$.
**Answer:**

$p(B|A,I) = \frac{p(A,B|I)}{p(A|I)} = \frac{p(A|B,I)p(B|I)}{p(A|I)} = \frac{p(A|B,I)p(B|I)}{p(A,B|I)+p(A\neg B|I)} = \frac{p(A|B,I)p(B|I)}{p(A|B,I)p(B|I)+p(A|\neg B,I)p(\neg B|I)} = \frac{0.8*0.004}{(0.8*0.004)+(0.1*(1-0.004))} = 0.0031128$

E. Find the probability that a patient that tested negative has tuberculosis, i.e., $p(B|\neg A,I)$. Does the test change our prior state of knowledge about about the patient? Is the test useful?
**Answer:**

$p(B|\neg A,I) = \frac{p(\neg A,B|I)}{p(\neg A|I)} = \frac{p(\neg A|B,I)p(B|I)}{p(\neg A|I)} = \frac{p(\neg A|B,I)p(B|I)}{1-p(A|I)} = \frac{(1-0.8)*0.004}{1-(0.1028)} = 0.000892$

F. What would a good test look like? Find values for

$$p(A|B,I) = p(\text{test is positive}|\text{has tuberculosis},I),$$

and

$$p(A|\neg B,I) = p(\text{test is positive}|\text{does not have tuberculosis},I),$$

so that

$$p(B|A,I) = p(\text{has tuberculosis}|\text{test is positive},I) = 0.99.$$

There are more than one solutions. How would you pick a good one? Thinking in this way can help you set goals if you work in R&D. If you have time, try to figure out whether or not there exists such an accurate test for tuberculosis
**Answer:**

From above we know that: $p(B|I) = 0.004$, $p(\neg B|I) = 1 - 0.004$, and

$$p(B|A,I) = \frac{p(A|B,I)p(B|I)}{p(A|B,I)p(B|I) + p(A|\neg B,I)p(\neg B|I)}$$

Another constraint we know is that all the probabilities must be < 1.

Therefore, we need to have a high probability for "test is positive given the person has tuberculosis" ($p(A|B,I)$) and a low probability for "test is positive given the person does not have tuberculosis" ($p(A|\neg B,I)$). Given this, I am choosing $p(A|\neg B,I) = 0.0000202832$ in order to find $p(A|B,I)$ that would allow for $p(B|A,I) = 0.99$.

NOTE: This calculation was done through the for loop below that computationally calculates the bounds. Then once the bounds were found, I divided the maximum $p(A|\neg B, I)$ bound in half. This is why the $p(A|B, I)$ becomes exactly 0.5.

$$p(B|A, I) = 0.99 = \frac{p(A|B, I)p(B|I)}{p(A|B, I)p(B|I) + p(A|\neg B, I)p(\neg B|I)} = \frac{p(A|B, I) * 0.004}{p(A|B, I)(0.004) + (0.0000202832)(1 - 0.004)}$$

$$\Rightarrow p(A|B, I)(0.004) + (0.0000202832)(1 - 0.004) = \frac{p(A|B, I) * 0.004}{0.99}$$

$$\Rightarrow p(A|B, I) = \frac{p(A|B, I)}{0.99} - \frac{(0.0000202832)(1 - 0.004)}{0.004}$$

$$\Rightarrow p(A|B, I) - \frac{p(A|B, I)}{0.99} = -\frac{(0.0000202832)(1 - 0.004)}{0.004}$$

$$\Rightarrow p(A|B, I)(1 - \frac{1}{0.99}) = -\frac{(0.0000202832)(1 - 0.004)}{0.004}$$

$$\Rightarrow p(A|B, I) = -\frac{(0.0000202832)(1 - 0.004)}{0.004} * \frac{1}{(1 - \frac{1}{0.99})}$$

$$\Rightarrow p(A|B, I) = -0.005051 * (-99)$$

$$\Rightarrow p(A|B, I) = 0.5$$

therefore, to make $p(B|A, I) = 0.99$ with $p(A|\neg B, I) = 0.0000202832$, we need $p(A|B, I) = 0.5$

In [ ]:
```
B = 0.004
nB = 1-B
BgA = 0.99
max = 1000000
min = 10000
first = False
for i in range(min,max):
    AgnB = 1/i
    AgB = -((AgnB*nB)/B)*(1/(1-(1/BgA)))
    if AgB < 1 and first == False:
        constraint_2_n = AgnB
        constraint_2 = AgB
        first = True
    if AgB > 1 and first == True or i == max:
        constraint_1_n = AgnB
        constraint_1 = AgB


print(f"p(A|-B,I) must be within ({constraint_1_n:.10f},{constraint_2_n:.10f}), which gives p(A|B,I) within ({constra

AgnB = constraint_2_n/2
AgB = -((AgnB*nB)/B)*(1/(1-(1/BgA)))

print(f"Using p(A|B,I) = {AgB:.10f} and p(A|-B,I) = {AgnB:.10f}, then p(B|A,I) = {(AgB*B)/((AgB*B)+(AgnB*nB)):.2f}")
```

```
p(A|-B,I) must be within (0.0000000000,0.0000405663), which gives p(A|B,I) within (0.0000000247,1.0000000000)
Using p(A|B,I) = 0.5000000000 and p(A|-B,I) = 0.0000202832, then p(B|A,I) = 0.99
```

## Problem 2 - Practice with discrete random variables

Consider the Categorical random variable:

$$X \sim \text{Categorical}(0.3, 0.1, 0.2, 0.4),$$

taking values in $\{0, 1, 2, 3\}$. Find the following (you may use `scipy.stats.rv_discrete` or do it by hand):

A. The expectation $\mathbb{E}[X]$.

**Answer:**

In [ ]:
```
ps = [0.3, 0.1, 0.2, 0.4] # probabilities
xs = np.array([1, 2, 3, 4]) # corresponding values

X = st.rv_discrete(name='X', values=(xs, ps))

print(f"E[X] = {X.expect():.2f}")
```

```
E[X] = 2.70
```

B. The variance $\mathbb{V}[X]$.

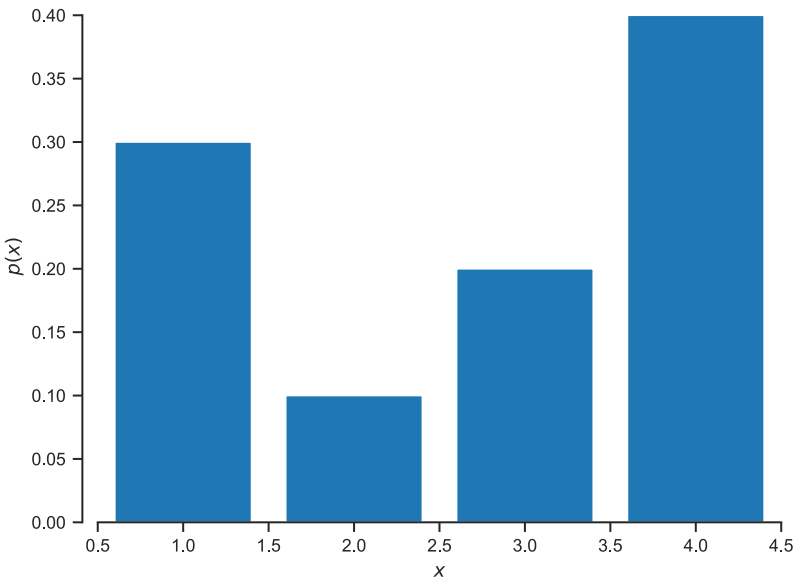
**Answer:**

In [ ]:
```
print(f"V[X] = {X.var():.2f}")
```

```
V[X] = 1.61
```

C. Plot the probability mass function of $X$.

In [ ]:
```
fig, ax = plt.subplots()
ax.bar(xs, X.pmf(xs))
ax.set_xlabel("$x$")
ax.set_ylabel("$p(x)$")
sns.despine(trim=True);
```




D. Find the probability that $X$ is in $\{0, 2\}$.

**Answer:**

In [ ]:
```
#p(X in {0,1,2}) = p(X=0)+p(X=1)+p(X=2) = n/a + 0.3 + 0.1 = 0.4
print(f"p(X=0) = {X.pmf(0):.2f}")
print(f"p(X=1) = {X.pmf(1):.2f}")
print(f"p(X=2) = {X.pmf(2):.2f}")
print(f"This can be represented by p(X=0) + p(X=1) + p(X=2) = {(X.pmf(0)+X.pmf(1)+X.pmf(2)):.2f}")
```

```
p(X=0) = 0.00
p(X=1) = 0.30
p(X=2) = 0.10
This can be represented by p(X=0) + p(X=1) + p(X=2) = 0.40
```

E. Find $\mathbb{E}[4X + 3]$.

**Answer:**

In [ ]:
```
# You can also answer with code here:
```

```
print(f"E[4X+3] = 4*E[X]+3 = {4:.0f}*{X.expect():.2f}+{3:.0f} = {4*X.expect():.2f}+{3:.0f} = {4*X.expect()+3:.2f}")
```

```
E[4X+3] = 4*E[X]+3 = 4*2.70+3 = 10.80+3 = 13.80
```

F. Find $\mathbb{V}[4X + 3]$.

**Answer:**

In [ ]:
```
print(f"V[4X+3] = (4^2)*V[X] = {(4*4):.0f}*{X.var():.2f} = {(4*4)*X.var():.2f}")
```

```
V[4X+3] = (4^2)*V[X] = 16*1.61 = 25.76
```

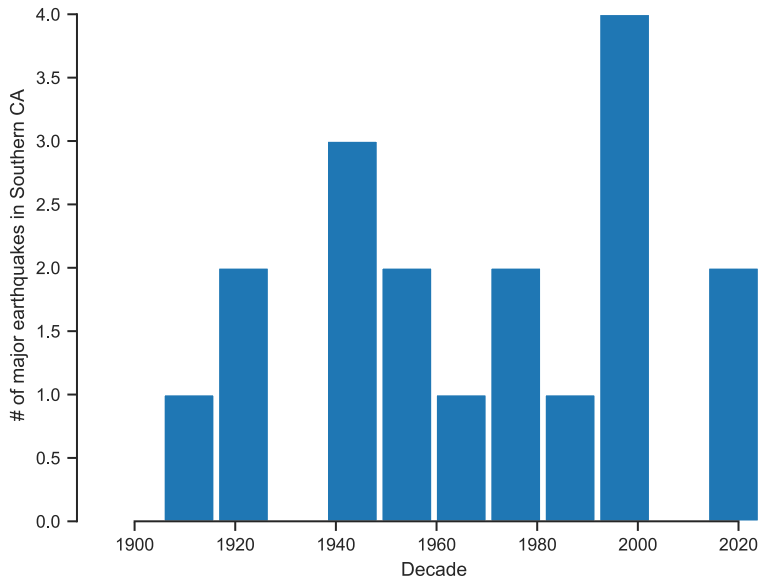
# Problem 3 - Predicting the probability of major earthquakes in Southern California

The San Andreas fault extends through California forming the boundary between the Pacific and the North American tectonic plates. It has caused some of the major earthquakes on Earth. We are going to focus on Southern California and we would like to assess the probability of a major earthquake, defined as an earthquake of magnitude 6.5 or greater, during the next ten years.

A. The first thing we are going to do is go over a database of past earthquakes that have occured in Southern California and collect the relevant data. We are going to start at 1900 because data before that time may are unreliable. Go over each decade and count the occurence of a major earthquake (i.e., count the number of organge and red colors in each decade). We have done this for you.

In [ ]:
```
eq_data = np.array([
    0, # 1900-1909
    1, # 1910-1919
    2, # 1920-1929
    0, # 1930-1939
    3, # 1940-1949
    2, # 1950-1959
    1, # 1960-1969
    2, # 1970-1979
    1, # 1980-1989
    4, # 1990-1999
    0, # 2000-2009
    2 # 2010-2019
])
fig, ax = plt.subplots(dpi=150)
ax.bar(np.linspace(1900, 2019, eq_data.shape[0]), eq_data, width=10)
ax.set_xlabel('Decade')
ax.set_ylabel('# of major earthquakes in Southern CA')
plt.legend(loc="best", frameon=False)
sns.despine(trim=True);
```

```
No artists with labels found to put in legend.  Note that artists whose label start with an underscore are ignored wh
en legend() is called with no argument.
```



B. The Poisson distribution is a discrete distribution with values $\{0, 1, 2, \ldots\}$ which is commonly used to model the number of events occuring in a certain time period. It is the right choice when these events are happening independently and the probability of any event happening over a small period of time is constant. Let's use the Poisson to model the number of earthquakes $X$ occuring in a decade. We write:

$$X \sim \text{Poisson}(r),$$

where $r$ is the *rate parameter* of Poisson. The rate is the number of events per time period. Here, $r$ is the number of earthquakes per decade. Using the data above, we can set the rate as the empirical average of the observed number of earthquakes per decade:

In [ ]:
```
r = np.mean(eq_data)
print('r = {0:1.2f} major earthquakes per decade'.format(r))
```

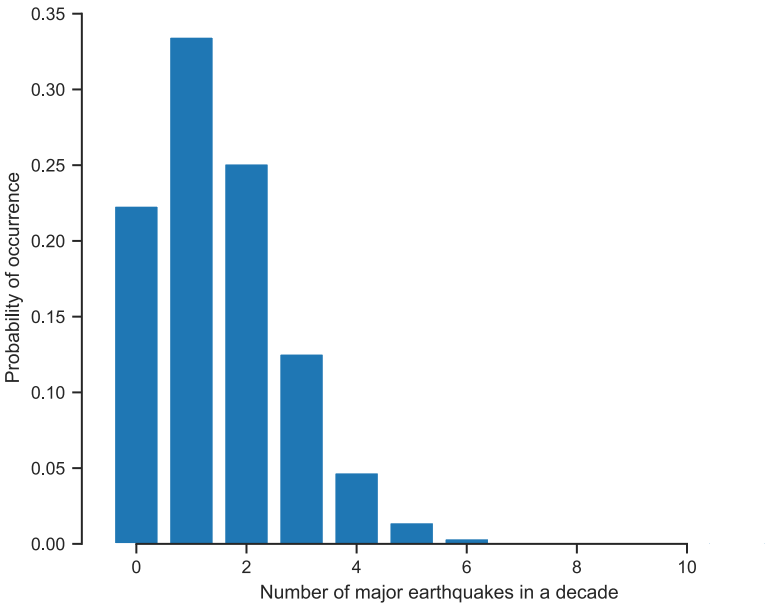
```
r = 1.50 major earthquakes per decade
```

Strictly speaking, **this is not how you should be calibrating models!!!** We will learn about the **right** way (which uses Bayes' rule) in the subsequent lectures. But it will do for now as the answer you would get using the **right** way is, for this problem, almost the same. Let's define a Poisson distribution using `scipy.stats.poisson` (see documentation here):

In [ ]:
```
X = st.poisson(r)
```

A. Plot the probability mass function of `X`.

In [ ]:
```
ks = range(12) # 12 was used to show the 12 decades used previously
fig, ax = plt.subplots()
ax.bar(ks, X.pmf(ks))
ax.set_xlabel('Number of major earthquakes in a decade')
ax.set_ylabel('Probability of occurrence')
sns.despine(trim=True);
```




B. What is the probability that at least one major earthquake will occur during the next decade?

**Answer:**

In [ ]:
```
print(f"This can be represented p(X>=1) = {1-X.pmf(0):.2f}") # calculate the probability of 0 and subtract that from
```

```
This can be represented p(X>=1) = 0.78
```

C. What is the probability that at least one major earthquake will occur during the next two decades? Hint: Consider two independent and identical copies of $X$, say $X_1$ and $X_2$. And consider their sum $Y = X_1 + X_2$. Read this about the sum of two independent Poisson distributions.

**Answer:**

In [ ]:
```
X_2 = st.poisson(r*2) # multiply the rate by 2 to represent calculate across the next 2 decades
print(f"This can be represented by p(X>=1) = {1-X_2.pmf(0):.2f}") # calculate the probability of 0 and subtract that
```

```
This can be represented by p(X>=1) = 0.95
```

D. What is the probability that at least one major earthquake will occur during the next five decades?
**Answer:**

In [ ]:
```
X_5 = st.poisson(r*5) # multiply the rate by 5 to represent calculate across the next 5 decades
print(f"This can be represented by p(X>=1) = {1-X_5.pmf(0):.4f}") # calculate the probability of 0 and subtract that
```

```
This can be represented by p(X>=1) = 0.9994
```

## Problem 4 - Failure of a mechanical component

Assume that you designing a gear for a mechanical system. Under normal operating conditions the gear is expected to fail at a random time. Let $T$ be a random variable capturing the time the gear fails. What should the probability density of $T$ look like?

Here are some hypothetical data to work with. Suppose that we took ten gears and we worked them until failure. The failure times (say in years) are as follows:

In [ ]:
```
time_to_fail_data = np.array(
    [
        10.5,
        7.5,
        8.1,
        8.4,
        11.2,
        9.3,
        8.9,
        12.4
    ]
)
```

Why does each gear fail at different times? There are several sources of uncertainty. The most important are:

- Manufacturing imperfections.
- Different loading conditions.

If this was a controlled fatigue experiment, then we could eliminate the second source of uncertainty by using exactly the same loading conditions.

Now, we are going to fit a probability density function to these data. Which one should we use? Well, new gears do not fail easily. So, the probability density function of $T$ should be close to zero for small $T$. As time goes by, the probability density should increase because various things start happening to the material, e.g., crack formation, fatigue, etc. Finally, the probability density must again start going to zero as time further increases because nothing lasts forever... A probability distribution that is commonly used to model this situation is the Weibull. We are going to fit some fail time data to a Weibull distribution and then you will have to answer a few questions about failing times.

The Weibull has parameters and we are going to fit them to the available data. The method we are going to use is called the *maximum likelihood method*. We haven't really talked about this, and it is not important to know what it is to do this homework problem. We will learn about maximum likelihood in later lectures. Here is how we fit the parameters using `scipy.stats`:

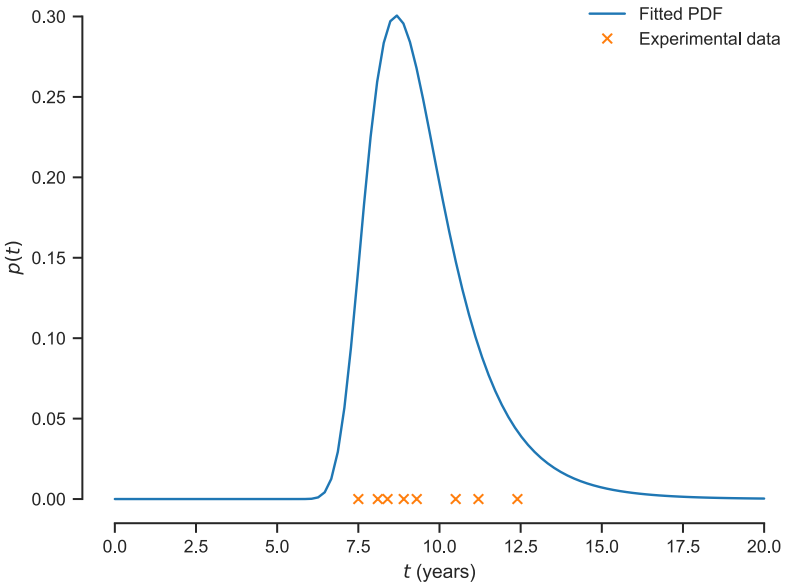
In [ ]:
```
fitted_params = st.exponweib.fit(time_to_fail_data, loc=0)
T = st.exponweib(*fitted_params)
print(f"Fitted parameters: {fitted_params}")
```

```
Fitted parameters: (448.066965711728, 0.7099665338918923, 3.4218808260575804, 0.41627831297126994)
```

Let's plot the fitted Weibul PDF and the data we used:

In [ ]:
```
fig, ax = plt.subplots()
ts = np.linspace(0.0, 20.0, 100)
ax.plot(
    ts,
    T.pdf(ts),
    label="Fitted PDF"
)
ax.plot(
    time_to_fail_data,
    np.zeros_like(time_to_fail_data),
```

```
    "x",
    label="Experimental data"
)
ax.set_xlabel(r"$t$ (years)")
ax.set_ylabel(r"$p(t)$")
plt.legend(loc="best", frameon=False)
sns.despine(trim=True);
```




Now you have to answer a series of questions about the random variable $T$ that we just fitted.

A. Find the mean fail time and its variance. Hint: Do not integrate anything by hand. Just use the functionality of `scipy.stats`.

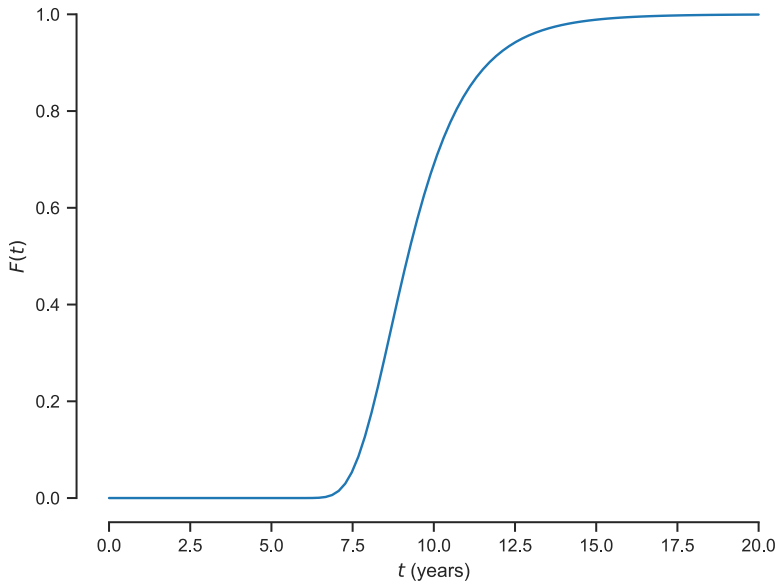
In [ ]:
```
t_mean = T.expect()
t_var = T.var()
print(f"The mean can be represented by E[T] = {t_mean:.2f}")
print(f"The variance can be represented by V[T] = {t_var:.2f}")
```

```
The mean can be represented by E[T] = 9.53
The variance can be represented by V[T] = 2.88
```

B. Plot the cumulative distribution function $F(t) = P(T \leq t)$ of $T$.

In [ ]:
```
# Your code here

fig, ax = plt.subplots()
ax.plot(ts, T.cdf(ts))
ax.set_xlabel("$t$ (years)")
ax.set_ylabel("$F(t)$")
sns.despine(trim=True);
```



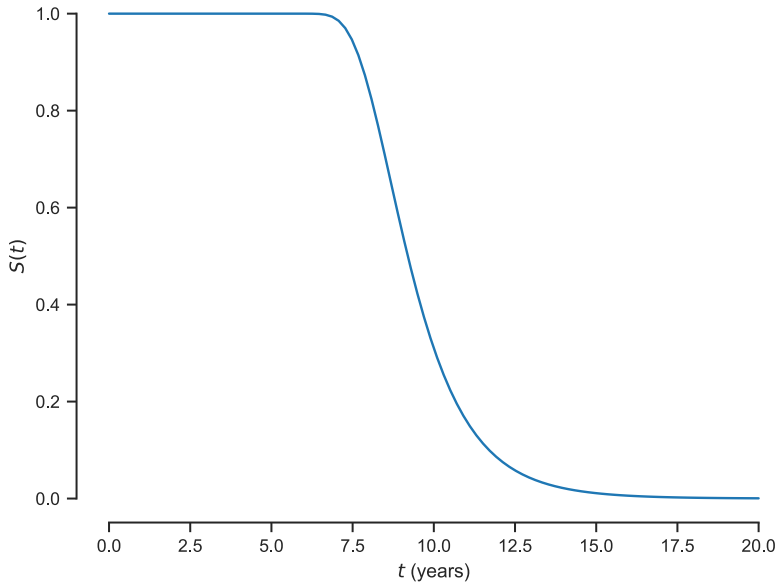

C. Plot the probability that gear survives for more than $t$ as a function of $t$. That is, plot the function:

$$S(t) = p(T > t).$$

Hint: First connect $S(t)$ to the cumulative distribution function $F(t)$ of $T$.

In [ ]:
```
F = T.cdf(ts) # p(T<=t)
S = 1-T.cdf(ts) # 1-p(T<=t) = p(T>t)

fig, ax = plt.subplots()
ax.plot(ts, S)
ax.set_xlabel("$t$ (years)")
ax.set_ylabel("$S(t)$")
sns.despine(trim=True);
```




D. Find the probability that the gear lasts anywhere between 8 and 10 years.

In [ ]:
```
a = 8 # first year (< b)
b = 10 # second year (> a)
prob_T_in_ab = T.cdf(b) - T.cdf(a) # calculating the difference between the two CDF values to get the probability bet
print(f"p({a:.2f} <= Z <= {b:.2f}) = {prob_T_in_ab:.2f}")
```

```
p(8.00 <= Z <= 10.00) = 0.53
```

E. Find the time $t^*$ such that the probability that the gear fails before $t^*$ is $0.01$.

In [ ]:
```
# F(t*) = p(T<=t*) = 0.01

# Executing inverse of the cdf computationally
for t_star in np.linspace(0.0, 20.0, 100000):
    if T.cdf(t_star) >= 0.01:
        #print(T.cdf(t_star))
        print(f"p(T<=t*) = 0.01, where t* is {t_star:.2f} using a hard-coded for loop")
        break

# Executing inverse of the cdf through a built-in scipy.stats function
t_star= T.ppf(0.01)
print(f"p(T<=t*) = 0.01, where t* is {t_star:.2f} using scipy.stats ppf function")

print(f"Given both methods finding the same results, t* = {t_star:.2f} years")
```

```
p(T<=t*) = 0.01, where t* is 6.98 using a hard-coded for loop
p(T<=t*) = 0.01, where t* is 6.98 using scipy.stats ppf function
Given both methods finding the same results, t* = 6.98 years
```