Problem set 2: Hands on with sustainability models

This problem set will have you work through several sustainability models. To turn this in, please submit a PDF of your completed notebook and upload to the assignment on canvas. There are lots of ways to convert a completed Jupyter Notebook to PDF, including the Export feature in VS Code in the "…" options menu above this notebook. The most robust way is to use Quarto (https://quarto.org/docs/getting-started.html), which I had you install as a VS Code Extension. The easiest way to do that is open up a Command Prompt/Terminal in the folder where your notebook is and type `quarto render problem_set_2.ipynb --to pdf`. This will create a PDF in the same folder as your notebook.

Required imports

```
import os, sys
from IPython.display import display, HTML
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import dicelib  # https://github.com/mptouzel/PyDICE

import ipywidgets as widgets  # interactive display

plt.style.use(
    "https://raw.githubusercontent.com/ClimateMatchAcademy/course-content/main/cma.mplstyl
)

%matplotlib inline
sns.set_style("ticks", {"axes.grid": False})
params = {"lines.linewidth": "3"}
plt.rcParams.update(params)
```

```
display(HTML("<style>.container { width:100% !important; }</style>"))
```

```
<IPython.core.display.HTML object>
```

Helper Functions

```
def plot_future_returns(gamma, random_seed):
    fig, ax = plt.subplots(1, 2, figsize=(8, 4))
    np.random.seed(random_seed)
    undiscounted_utility_time_series = np.random.rand(time_steps)
    ax[0].plot(undiscounted_utility_time_series)

    discounted_utility_time_series = undiscounted_utility_time_series * np.power(
        gamma, np.arange(time_steps)
    )
    ax[0].plot(discounted_utility_time_series)

    cumulsum_discounted_utility_time_series = np.cumsum(discounted_utility_time_series)
    ax[1].plot(
        cumulsum_discounted_utility_time_series * (1 - gamma),
        color="C1",
        label=r"discounted on $1/(1-\gamma)=$"
        + "\n"
        + r"$"
        + str(round(1 / (1 - gamma)))
        + "$-step horizon",
    )
    cumulsum_undiscounted_utility_time_series = np.cumsum(
        undiscounted_utility_time_series
    )
    ax[1].plot(
        cumulsum_undiscounted_utility_time_series
        / cumulsum_undiscounted_utility_time_series[-1],
        label="undiscounted",
        color="C0",
    )
    ax[1].axvline(1 / (1 - gamma), ls="--", color="k")

    ax[0].set_ylabel("utility at step t")
    ax[0].set_xlim(0, time_steps)
```

```
    ax[0].set_xlabel("time steps into the future")
    ax[1].legend(frameon=False)
    ax[1].set_ylabel("future return (normalized)")
    ax[1].set_xlabel("time steps into the future")
    ax[1].set_xlim(0, time_steps)
    fig.tight_layout()
```

Question 1

Write a for loop that tests the DICE model to see how the total damages in 2100 change when there is a value of 2 versus 3 for the damage function coefficient. Output a figure for each of the two values and then report out the actual value of total damage

Reminder on some key python terms:

To iterate ver a list, you use a for loop:

```
for i in [1, 2, 3]:
    print(i)
```

```
1
2
3
```

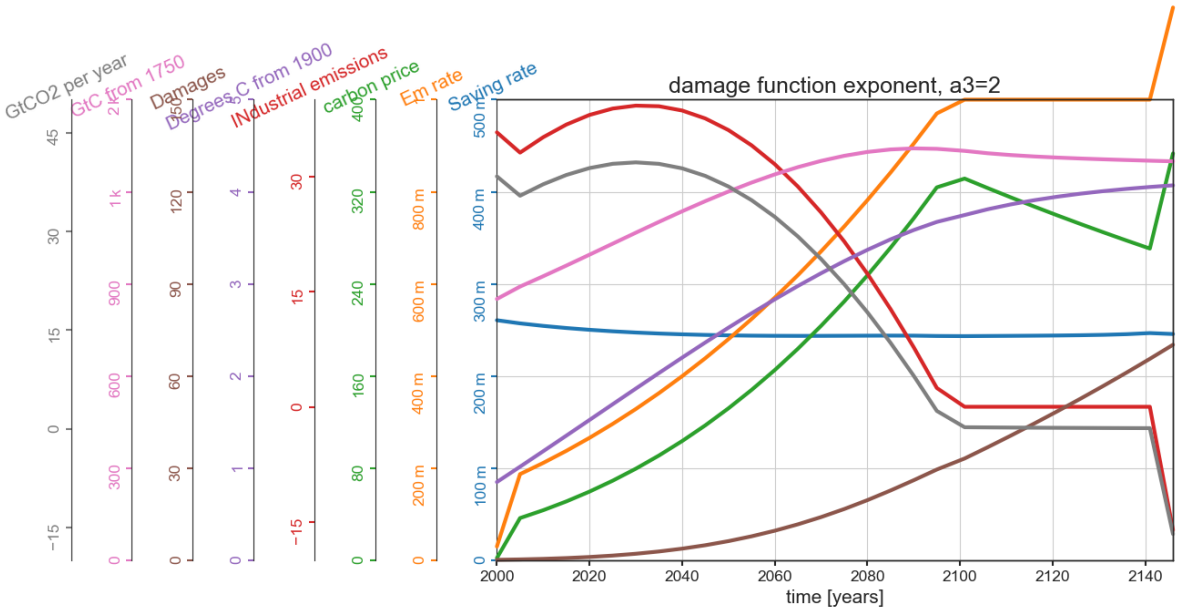
The first step to solve this will be create a new dice model object. You will then call methods from that object like this: dice.method_name()

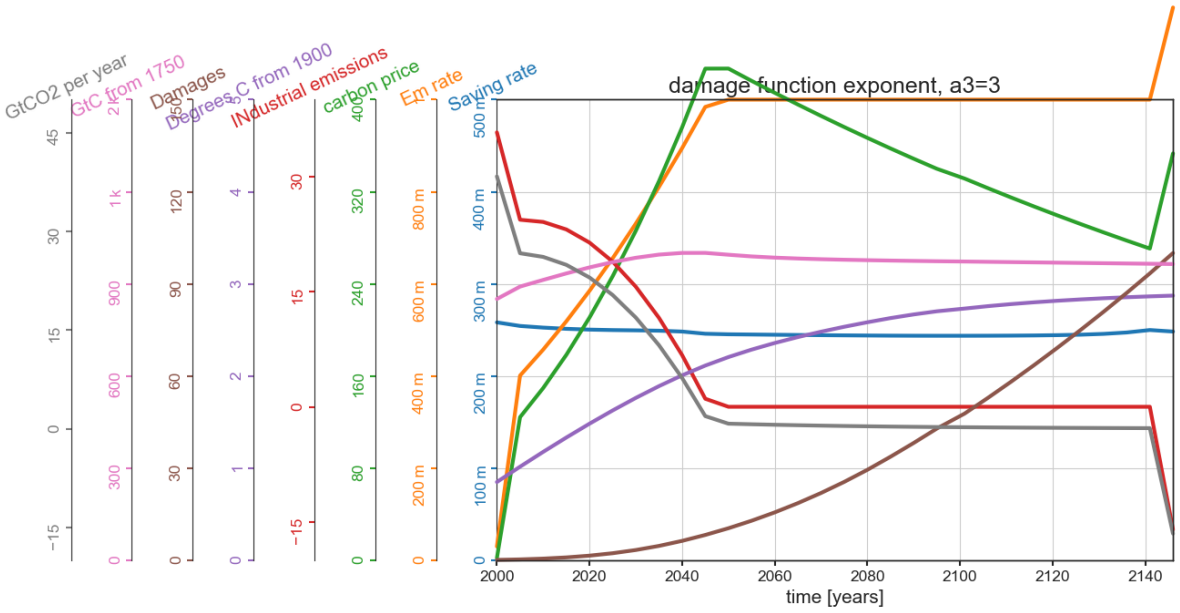
Use the init methods, but ensuring you use the correct damage function parameter. Calling the .init_parameters() method with no arguments gives you the default parameters, but you can overwrite the default by passing specific values like this: `dice.init_parameters(alpha=0.2, eta=0.3)`. You will want to change the exponent in the damage function. If you can't figure out what coefficient it is, you can learn this by poking around in the dicelib.py file. To get the total damages at the right year, you will have to inspect the model outputs and call the right year out of the list. You can access an element in a list like this: `my_value = my_list[0]`. Remember that python is 0-indexed, so the first element is at index 0.

```
# Initialize DICE model

for a3 in [2, 3]:
    dice = dicelib.DICE()
```

```
    dice.init_parameters(a3=a3)
    dice.init_variables()
    controls_start, controls_bounds = dice.get_control_bounds_and_startvalue()
    dice.optimize_controls(controls_start, controls_bounds)
    dice.roll_out(dice.optimal_controls)
    dice.plot_run("damage function exponent, a3=" + str(a3))
    inx = int((2100-dice.min_year)/5) # Calculate the index for year 2100
    total_damages_2100=dice.DAMAGES[inx]
    print(f"Total damages at year 2100 (a3={a3}): {total_damages_2100}")
```

```
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\scipy\optimize\_slsqp_py.py:437: Ru
  fx = wrapped_fun(x)


Optimization terminated successfully    (Exit mode 0)
            Current function value: -4517.318953976377
            Iterations: 94
            Function evaluations: 19029
            Gradient evaluations: 94
Total damages at year 2100 (a3=2): 33.16490083819454
Optimization terminated successfully    (Exit mode 0)
            Current function value: -4416.943986620645
            Iterations: 92
            Function evaluations: 18626
            Gradient evaluations: 92
Total damages at year 2100 (a3=3): 47.79018135805996
```





Question 2

Run the MAGICC model for the 4 main RCPs (2.6, 4.5, 7.0, 8.5). Plot the temperature of each pathway on (a single or seperate) graphs. Report out what is the expected temperature in 2100 for each RCP.

```python
from pymagicc.scenarios import rcp26
```

```python
from pymagicc.scenarios import read_scen_file
scenario_path = "C:/Users/jsnjz/miniforge3/envs/env2024a/Lib/site-packages/pymagicc/MAGICC
scenario = read_scen_file(scenario_path)
```

```
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
```

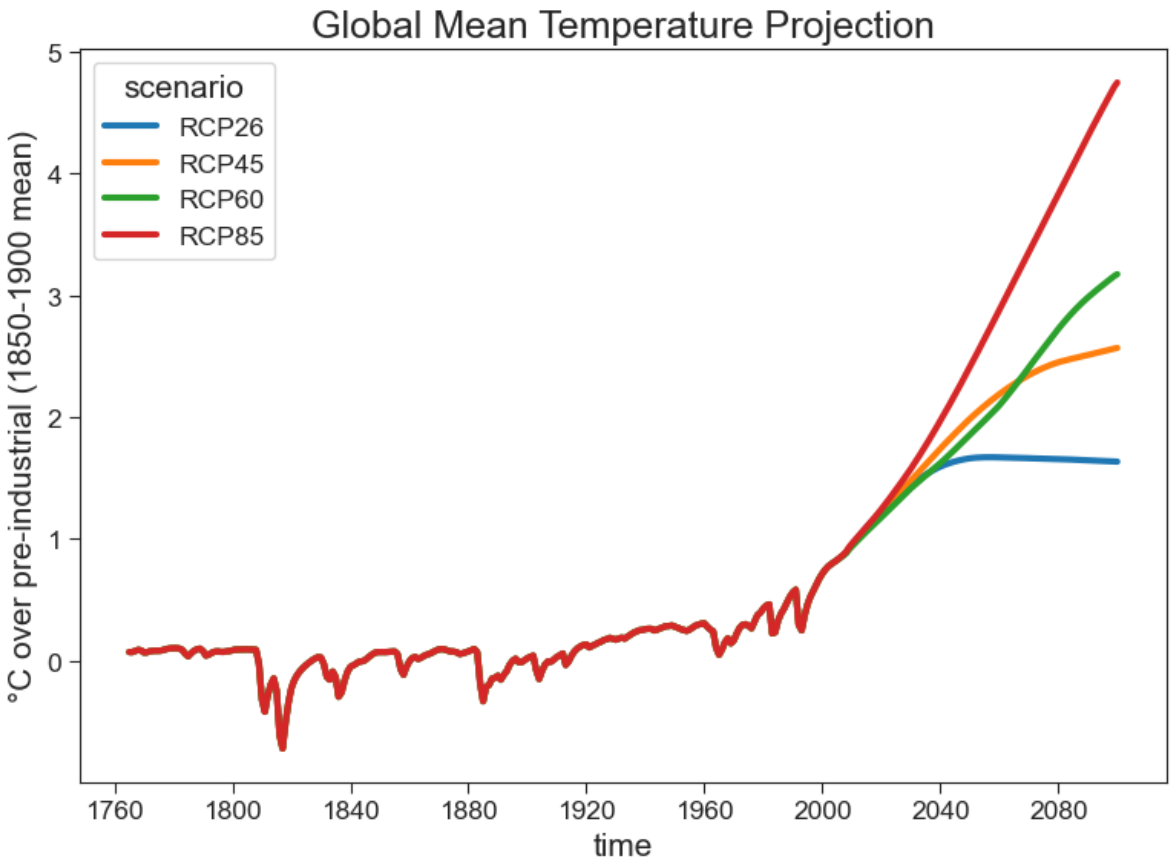
```python
import matplotlib.pyplot as plt

import pymagicc
import scmdata
from pymagicc import rcps

results = []
for scen in rcps.groupby("scenario"):
    results_scen = pymagicc.run(scen)
    results.append(results_scen)

results = scmdata.run_append(results)

temperature_rel_to_1850_1900 = (
    results
    .filter(variable="Surface Temperature", region="World")
    .relative_to_ref_period_mean(year=range(1850, 1900 + 1))
)
```

```python
temperature_rel_to_1850_1900.lineplot()
plt.title("Global Mean Temperature Projection")
plt.ylabel("°C over pre-industrial (1850-1900 mean)");
# Run `plt.show()` to display the plot when running this example
# interactively or add `%matplotlib inline` on top when in a Jupyter Notebook.
```

```
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
```

```
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
```

```
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\pymagicc\io\base.py:224: FutureWarn
  df = pd.read_csv(
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\scmdata\run.py:905: FutureWarning:
  df.columns = pd.Index(columns, name="time")
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\scmdata\run.py:905: FutureWarning:
  df.columns = pd.Index(columns, name="time")
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\scmdata\run.py:905: FutureWarning:
  df.columns = pd.Index(columns, name="time")
c:\Users\jsnjz\miniforge3\envs\env2024a\Lib\site-packages\scmdata\plotting.py:81: FutureWarni

The `ci` parameter is deprecated. Use `errorbar='sd'` for the same effect.

  ax = sns.lineplot(data=plt_df, **kwargs)
```




```
temperature_2100 = (
    temperature_rel_to_1850_1900
    .filter(year=2100)
    .values
)

# Define RCP scenario names
rcp_names = ["RCP 2.6", "RCP 4.5", "RCP 7.0", "RCP 8.5"]

# Print the predicted temperature value for the year 2100 for each RCP scenario
print("Predicted global mean temperature for 2100 (°C over pre-industrial)")
for i in range(len(temperature_2100)):
    print(rcp_names[i], "=", temperature_2100[i], "°C")
```

```
Predicted global mean temperature for 2100 (°C over pre-industrial)
RCP 2.6 = [1.63427534] °C
```

```
RCP 4.5 = [2.56807834] °C
RCP 7.0 = [3.17350514] °C
RCP 8.5 = [4.74703324] °C
```

Question 3

The DICE version that we ran above is the 2016 version of Nordhaus' model. Recently, Nordhaus and team released a new version, documented in Barrage and Nordhaus (2023). Read the paper and compare it to what was in the 2016 version and summarize in 2-3 paragraphs what the main differences are with emphasis on explaining the technical differences (i.e., what coefficient changed and what were its new and old values). You may want to look through the code the Lint Barrange and William Nordhaus put online, which can be found here: https://yale.app.box.com/s/whlqcr7gtzdm4nxnrfhvap2hlzebuvvm

Answer:

1. From Barrage and Nordhaus (2023), I can list the most recent update about the model can be described as follows.

(1) DICE-2023 model changed (the pure rate of social time preference) from 0.015 to 0.01 and (the constant elasticity of marginal utility of consumption from 1.45 to 1.5). The purpose is to lower the real interest rate to reflect the decline in market interest rates over recent years. The impact is that the average discount rate for 2020-2100 is revised from 4.2%/yr in DICE 2016 to 3.9%/yr in DICE-2023.

(2) For damage function, after including adjustments, damages are estimated to be around 3.12% of output at a 3 degree global warming over pre-industrial temperatures and 12.5% of output with 6 degree warming.

(3) DICE-2023 has a major revision in its treatment of GHG emissions. In earlier versions, only industrial CO2 emissions were controllable (abatable) and other GHGs and focing were taken to be exogenous. The current version includes all "abateable" emissions in the endogenous category and excludes only a small fraction of forcing as "non-abateable emissions", including land CO2, methane, chlorofluorocarbons, and other well-mixed gases.

(4) The major structural revision of the DICE-2023 model is the introduction of the DFAIR module, which represents the dynamics of the carbon cycle. Earlier versions in DICE have used linear carbon-cucle structures.

2. From DICE 2023 parameters updated in the excel on the website, I find the updates as follows.

(1) Parameters which value has been majorly changed include: damage coefficient on temperature squared, backstop price, 2050 and conservative factor (F2x).

(2) The coefficients for regressing c(1) and c(2) * IFR on alpha, to be specific, the coefficients of CCATOTALL(-1) and CCATOTALL(-1)^2.

(3) The coefficients for regressing IFR on c, temp(-1) and cacc(-1), to be specific, the coefficients of c, temp(-1) and CACC(-1).