Project Milestone 1

Due 17:00 on April 5

Throughout this course, you will work on a semester-long project that will allow you to engage with material from different parts of the course through a single case study. The case study is a famous set of systems dynamics models initially developed by the pioneers of systems thinking that attempt to model the global-scale dynamics of population, economic growth, and pollution. The models have received both praise and criticism in the years since they were first developed in the early 1970s. You will have the opportunity to formulate your own arguments about the models.

In this first project milestone, you will begin exploring the models and running initial simulations in Python. Then, you will leverage computational power to help you arrive at policies that drive system behavior that you have deemed sustainable.

The models

We will be using two highly related models that essentially describe the same key systems components but are different in their complexity. The `world2` model was developed by Jay Forrester and presented in his book *World Dynamics*. (This book can be a nice reference if you are interested in going further into understanding the model.) The `world3` model was developed by Dennis Meadows and a large team of collaborators. The results were presented in the book *The Limits to Growth* and the model was published in the book *Dynamics of Growth in a Finite World*. The table below summarizes the size of the two models.

Model	Stocks	Flows	Variables	Total connections
`world2`	5	8	82	121
`world3`	15	23	156	336

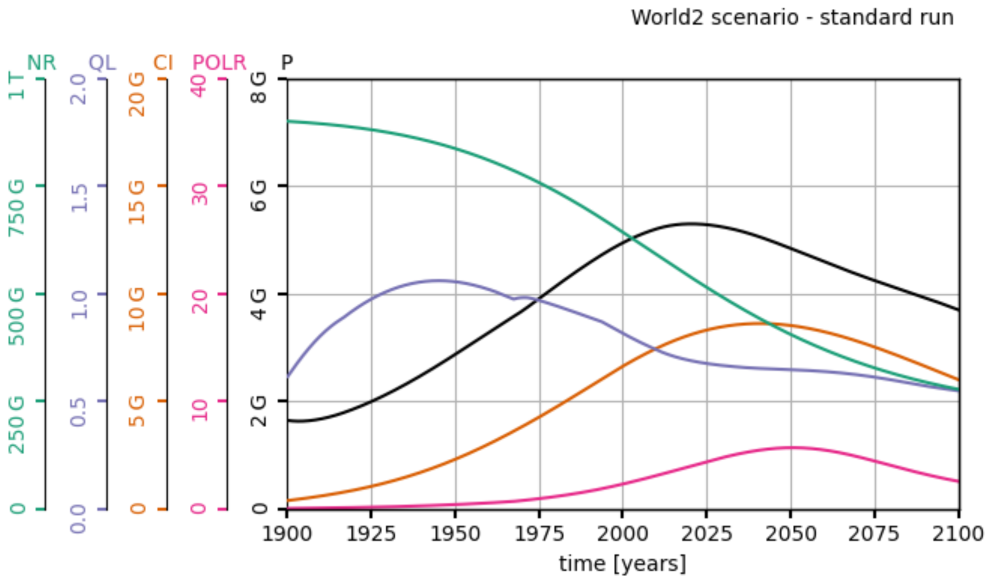
Throughout the semester, we will be working with pre-defined implementations of both models.

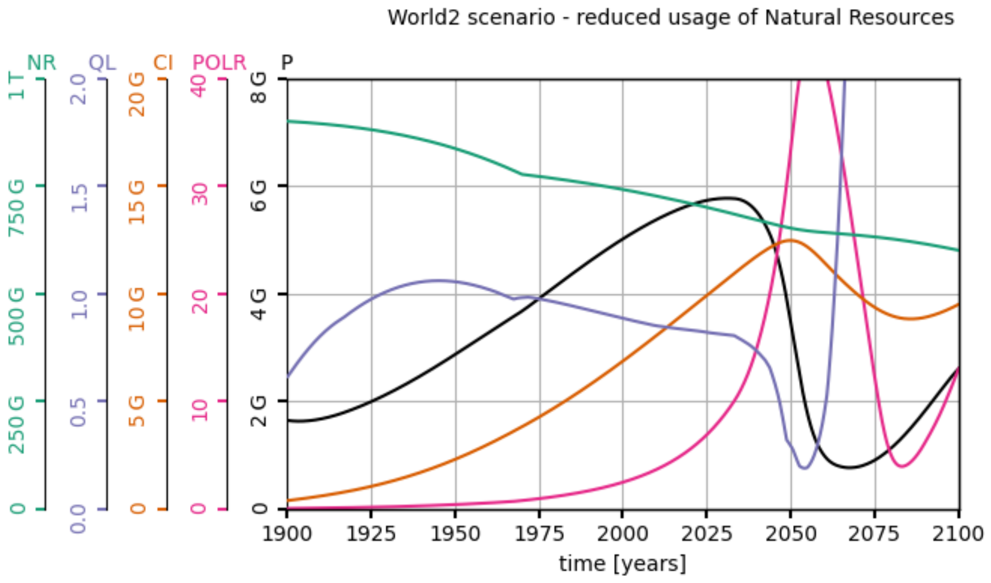
Setting up your Python environment

To get started with the project, you will need to install specific packages for working with the models. Both packages can be installed using pip. However, for maximum flexibility and the option to modify certain aspects of the model, it is better to install the folders directly. Here is the process for doing this with the pyworld2 model:

1. Go to https://github.com/cvanwynsberghe/pyworld2
2. Download the code as a zip file (you can use the green “Code” button to do this).

3. Go to https://noto.epfl.ch/
4. Create a new folder where you like and call it “pyworld2”
5. Within the pyworld2 folder you just created, upload the files contained within the “pyworld2” folder you just downloaded. Note that the “pyworld2” folder is contained within the “pyworld2-main” folder. You need to upload the files from `__init.py___` to `world2.py`. You can now create Jupyter notebooks in the outer folder and use the pyworld2 package. Test this by creating a new notebook and running the following two lines of code:
```
import pyworld2
pyworld2.hello_world2()
```
You should see the following output:





You can install the world3 model using the same steps. You should look through the source code to understand the similarities and differences in how the updated world3 model is structured. world3 has a similar structure to the world2 model. While there are more parameters in world3 (e.g., multiple mortality rates for different ages instead of one overall death rate), the same world2 diagram on Moodle provides a good proxy for the structure of world3.

Part 1: world2 model characterization

When you initialize the model, you see the standard run of the model in addition to an alternative run in which the use of natural resources is reduced by 75% in 1970. The impact of using fewer natural resources (holding all other things equal) may be surprising. One of the key reasons for the surprising behavior is the interdependencies between the population and natural resource stocks. As a starting point for this milestone, take a look at the world2 system diagram and try to identify the feedback loops that link these stocks. To identify feedback loops, it may be useful to consider the dynamics of the other stocks in each of the two scenarios as well.

In this first milestone, you should also take some time to consider varying additional parameters in the model. Parameters can be varied in different ways. The README file on Github for the models provides some details on how to vary the model parameters. Some model parameters can be varied using the `set_state_variables()` or `set_initial_state()` functions. The function `set_switch_functions()` allows you to implement changes to a select number of parameters that occur at a defined time during the model run. As described in the README file, this is done by editing the contents of a `.json` file. Note that this can be done in Python using the `json` package (already installed). Here is an example of editing a json file:

```
import json
# open the file
with open('filename.json', 'r') as file:
    # Edit the contents of the file object
    data = json.load(file)
    data[key]["key"] = 0 # access the data using the keys within the json file
    new_data = json.dumps(data)
# write the file
with open('new_filename.json', 'w') as file:
    file.write(new_data)
```

At a minimum, you should vary the following parameters: capital investment rate, birth rate, death rate, pollution production rate, but you should go deeper than this to fully explore the model dynamics. Note that you can create your own graphs by plotting individual variables as well. Variables can be accessed directly through the model instance (e.g., `[model name].[variable name]`).

Consider the usefulness and potential weaknesses of the world2 model. Identify the key feedback loops and discuss how varying different parameters affects the model runs. Remember that this is the first milestone of a semester-long project, so your understanding and opinion of the model will likely change throughout your work on the project.

Part 2: Automated policy design

Your next task is to develop a structured exploration of “policies” using an automated process that you design in Python. The goal of your automated procedure is to arrive at a policy that produces “sustainable development,” however you would like to define the term (be explicit). You may use both models (world2 and world3) in the development phase of your automated procedure, but ultimately you should define the policy for the world3 model and its parameters.

The policy you come up with involves the setting of parameters that you believe could be adjusted in a realistic manner. Beyond the values of parameters themselves, another aspect of the model is the time at which certain policies are implemented. These are described in the source codes of both world2 and world3, though they are implemented slightly differently in each version.

A key task here is to leverage the computational power of Python to explore policy scenarios in an automated fashion. One part of your strategy could be to define the most important variables and compute summary statistics of the variables for the different possible policies. This way, you can run multiple scenarios and evaluate the impacts without needing to look at the output graphs for each run separately, enabling fast comparisons of different policies.

Submission

- Groups of 3 or 4
- Max 8-page narrative
- Code for analysis (note that this is not graded at this stage, but you will need to submit your code at the end of the project)