

The contents of this folder are provided as supplementary materials for "Computational Implementation" by Mehmet Barlo and Nuh Aygun Dalkiran.

The programming language used for all the codes in this folder is Python 3.8. To use these codes, we recommend that you install the program called Anaconda, "a free and open-source distribution of the Python and R programming languages for scientific computing."

Below, we summarize the contents of the subfolders of this folder in an alphabetical order:

1. "Check_a_Mechanism_for_an_SCC\":

This folder contains the Python Codes that can be used to check whether a given mechanism Nash Implements a given given SCC:

This subfolder is divided into three subsubfolders according to the number of individuals.

1.1. "Check_a_Mechanism_for_an_SCC\Four_or_More_Individuals\" contains the codes for the case of four or more individuals.

1.2. "Check_a_Mechanism_for_an_SCC\Three_Individuals\" contains the codes for the case of three individuals.

1.3. "Check_a_Mechanism_for_an_SCC\Two_Individuals\" contains the codes for the case of two individuals.

Please refer to the corresponding subsubfolder for further instructions regarding how to run the specific Python Codes.

2. "Examples_in_the_Manuscript\":

This subfolder is divided into three subsubfolders according to the section the examples appear in the manuscript.

2.1 "Examples_in_the_Manuscript\Example_1_in_the_Manuscript\" presents the codes and outputs for Example 1 in the manuscript.

2.2 "Examples_in_the_Manuscript\Example_2_in_the_Manuscript\" presents the codes and outputs for Example 2 in the manuscript.

2.3 "Examples_in_the_Manuscript\Example_3_in_the_Manuscript\" presents the codes and outputs for Example 3 in the manuscript.

Please refer to the corresponding subsubfolder for further instructions regarding how to run the specific Python Codes.

3. "Given_a_Mechanism_Domain_Implementation\":

This folder contains the Python Codes that can be used to obtain the domain of preferences for a given mechanism where Nash Implementation is possible as well as the maximal domain of preferences where this mechanism Nash implements the Pareto efficient SCC:

This subfolder is divided into three subsubfolders according to the number of individuals.

3.1. "Given_a_Mechanism_Domain_Implementation\Four_or_More_Individuals\" contains the codes for the case of four or more individuals.

3.2. "Given_a_Mechanism_Domain_Implementation\Three_Individuals\" contains the codes for the case of three individuals.

3.3. "Given_a_Mechanism_Domain_Implementation\Two_Individuals\" contains the codes for the case of two individuals.

Please refer to the corresponding subsubfolder for further instructions regarding how to run the specific Python Codes.

4. "Given_an_SCC_Consistent_Collections_Necessity_Sufficiency\":

This folder contains the Python Codes that can be used to obtain the list of all consistent collections of an SCC as well as whether these consistent collections satisfy the NVP* or the EE* properties:

This subfolder is divided into two subsubfolders according to the number of individuals.

4.1.
"Given_an_SCC_Consistent_Collections_Necessity_Sufficiency\Three_or_More_Individuals\" contains the codes for the case of three or more individuals.

4.2.
"Given_an_SCC_Consistent_Collections_Necessity_Sufficiency\Two_Individuals\" contains the codes for the case of two individuals.

Please refer to the corresponding subsubfolder for further instructions regarding how to run the specific Python Codes.