NetSim Multi-Parameter Sweeper Python Script

Software: NetSim v13.2 (64 bit), DOT NET CORE SDK 3.1, Python 3.7.4

Project Download Link:

https://github.com/NetSim-TETCOS/Multi-Parameter_Sweeper_v13.2/archive/refs/heads/main.zip

Introduction

When users want to *sweep* one or more parameters, they change their values between simulation runs, and compare and analyse the performance metrics from each run. NetSim multi-parameter sweeper enables users to automate the sweep process.

Consider an example, where a user wishes to create and simulate a network scenario for all possible values of one or more parameters in combination and analyse a set of performance metrics across the simulation runs. This is extremely time consuming to do manually using the NetSim GUI.

The multi-parameter sweep program enables users to automate the sweep process across multiple input parameters, simulate each run, save each result, and compare specific output metrics via a spreadsheet software like MS Excel.

The sweep program runs NetSim via its CLI interface.

File Organization

The project directory consists of several binaries which are responsible for different tasks during a multi-parameter sweep:

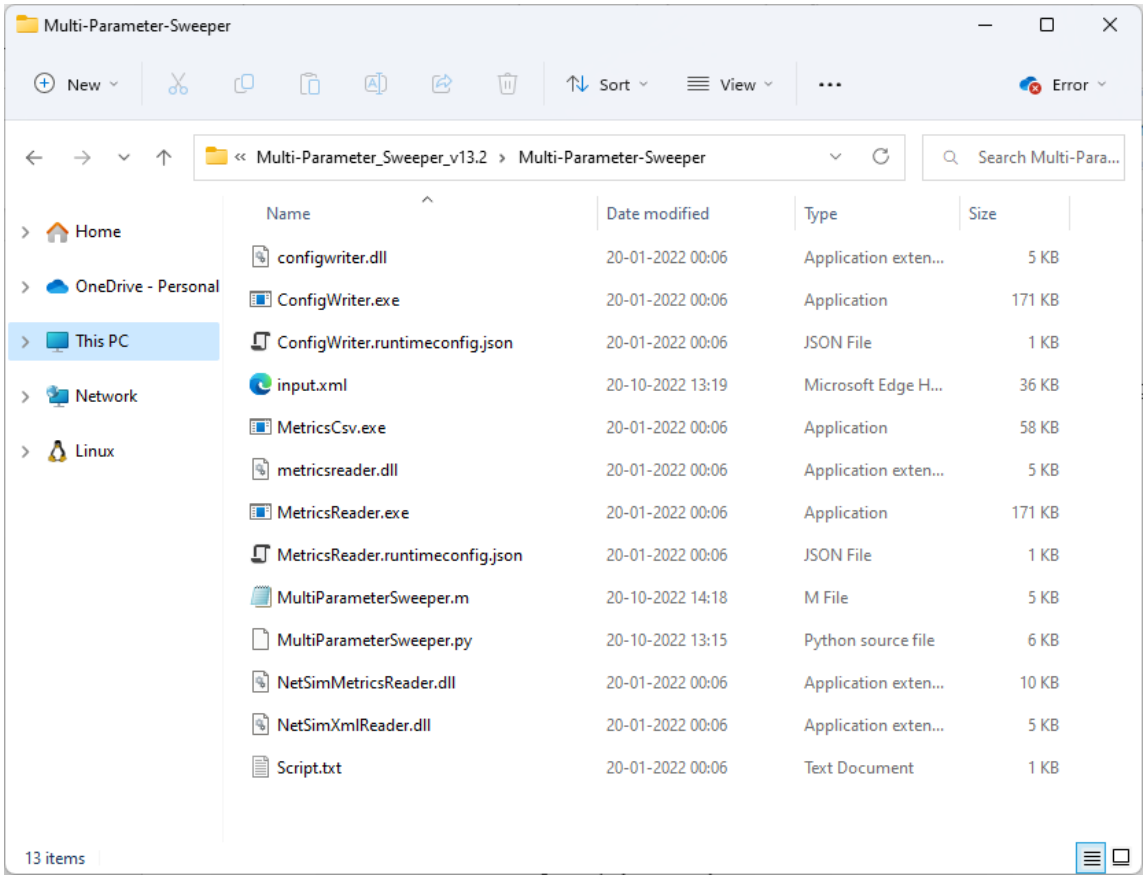


Figure 1: project directory consists of several binaries

1. **input.xml:** This file contains the base NetSim network configuration that is to be simulated. This file is created by copy pasting the Configuration.netsim file that can be obtained by saving a network configuration in NetSim and renaming it to input.xml.

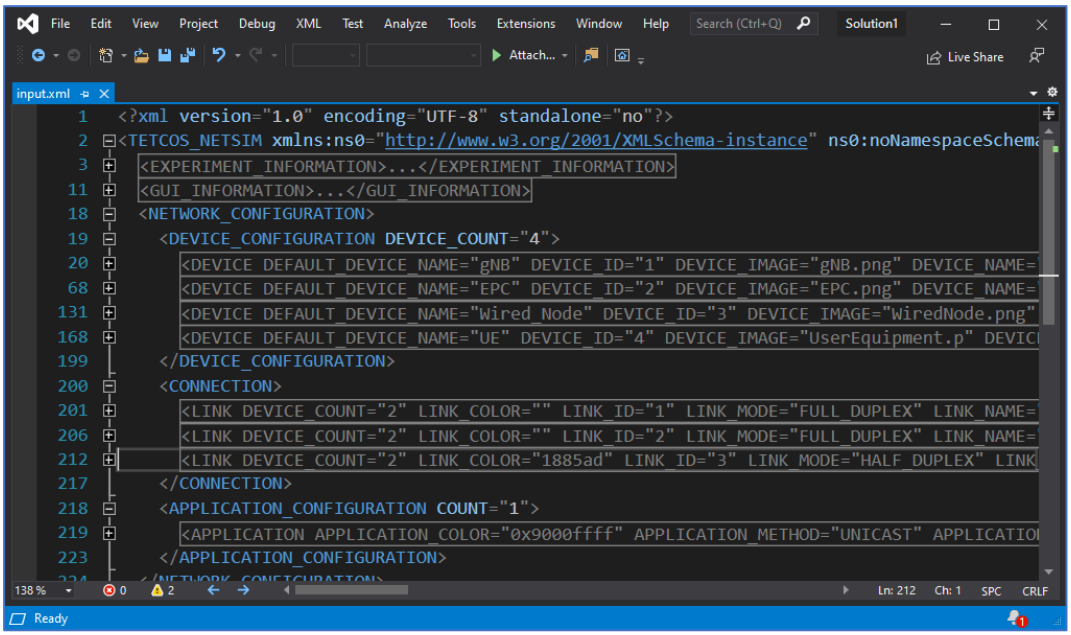


Figure 2: NetSim input Configuration.netsim file

- The values of parameters which are to be varied during each simulation run needs to be specified as {0}, {1}, {2}, etc. respectively.

- For Example, if the X and Y coordinates of a device is to be varied the values can be modified in the input.xml file as shown below:

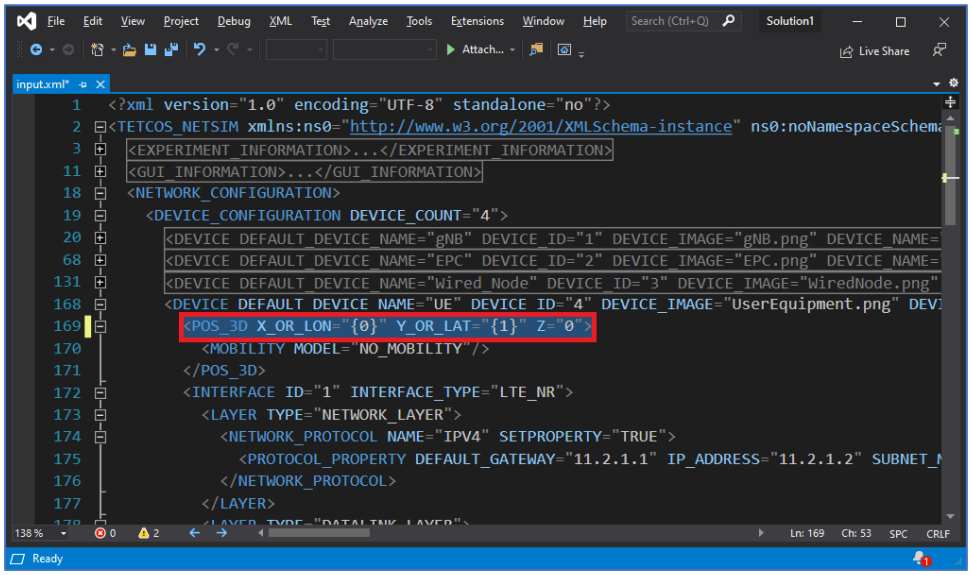


Figure 3: Modify X and Y coordinates in input input.xml file

2. **Script.txt:** This file should be updated with the parameter from the output metrics of NetSim that is to be logged at the end of each simulation run for the purpose of analysis.

At the end of every simulation, NetSim generates a Metrics.xml file which contain the performance metrics written in a specific format based on which it is loaded in the results dashboard.

Each Metric is part of a results table which can be accessed using a menu in the results dashboard. A NetSim Metrics.xml file is shown below:

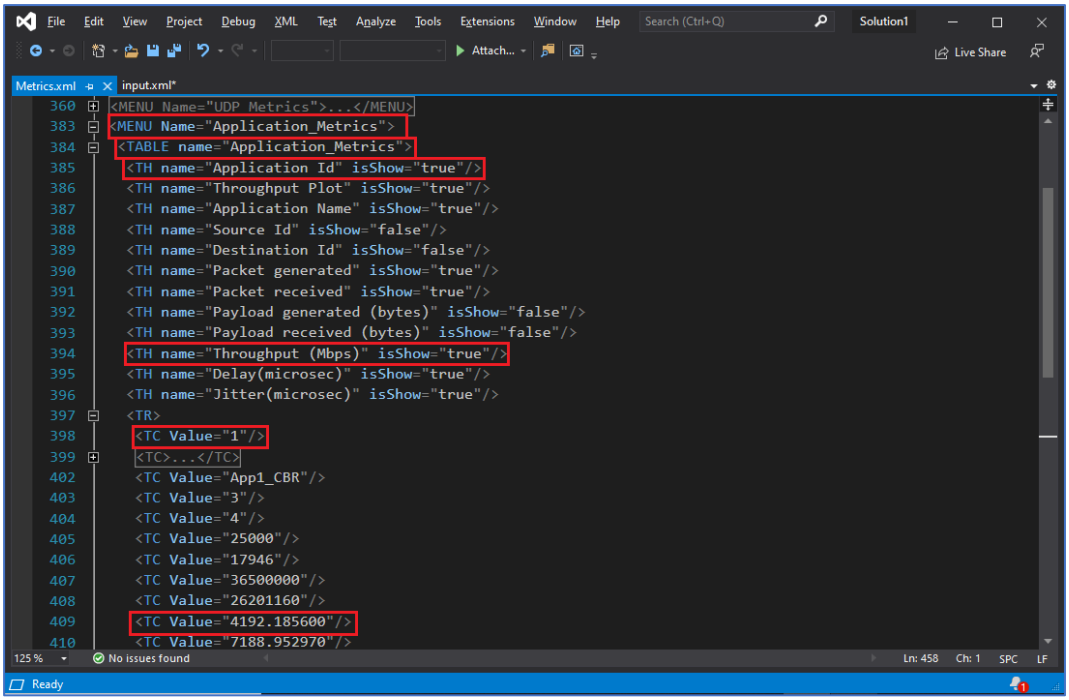


Figure 4: NetSim output Metrics.xml file

For Example, if the application throughput is to be logged for each simulation run then the script file can be updated as shown below:

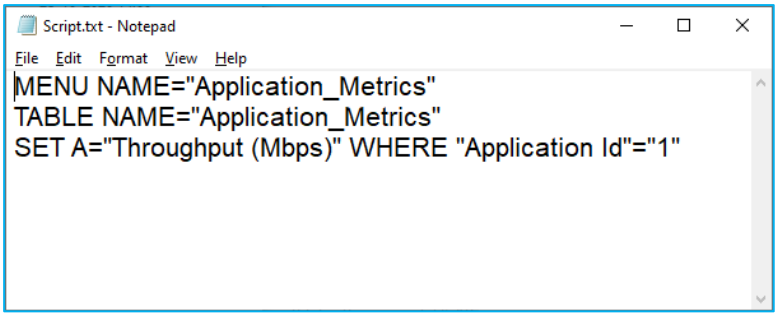


Figure 5: The application throughput is to be logged for each simulation modified in Script.txt

3. **ConfigWriter.exe**: This executable takes one or more command line arguments as input and generated Configuration.netsim file by replacing the arguments in place of the variable parameters specified in the input.xml file.

   If there are two variable parameters specified in the input.xml file ({0} and {1}) then two arguments need to be passed while calling ConfigWriter.exe.

4. **MetricsCSV.exe:** This executable is used to convert the Metrics.xml file present in the output folder into a comma separated file, MetricsPrint.csv.

5. **MetricsReader.exe**: This executable is responsible for reading the output parameter from the Metrics.xml file generated after each simulation and logging it to the results file.
   It uses the Script.txt file to determine which parameter to read from the Metrics file.

   If multiple parameters are to be read and logged, then the MetricsReader.exe can be called multiple times with Script.txt file having information about the parameter to be read each time.

6. **Supporting DLL's**: Some the supporting files such as ConfigWriter.dll, MetricsReader.dll, NetSimMetricsReader.dll, NetSimXmlReader.dll, etc. which are present in the project folder are used by other executable such as ConfigWriter.exe and MetricsReader.exe for various purposes during a multi-parameter sweep.

7. **MultiParameterSweeper**.**py** uses python programming language which is less complex and offers more flexibility as the number of input and output parameters increases.

   - Users can also write the script to run the multi-parameter sweep process in a preferred programming language as per the convenience.

   - The script can be configured to run multiple simulation iterations based on the number of parameters to be varied and the range of values of each parameter.

   - NETSIM_PATH variable can be set to the path of NetSim 64-bit binaries (bin_x64) in the install directory or workspace which is to be used to run Simulations.

   - LICENSE_ARG variable can be set to License server port and IP details in case of floating on premise licenses or the path of license file in case of node locked or cloud licenses.

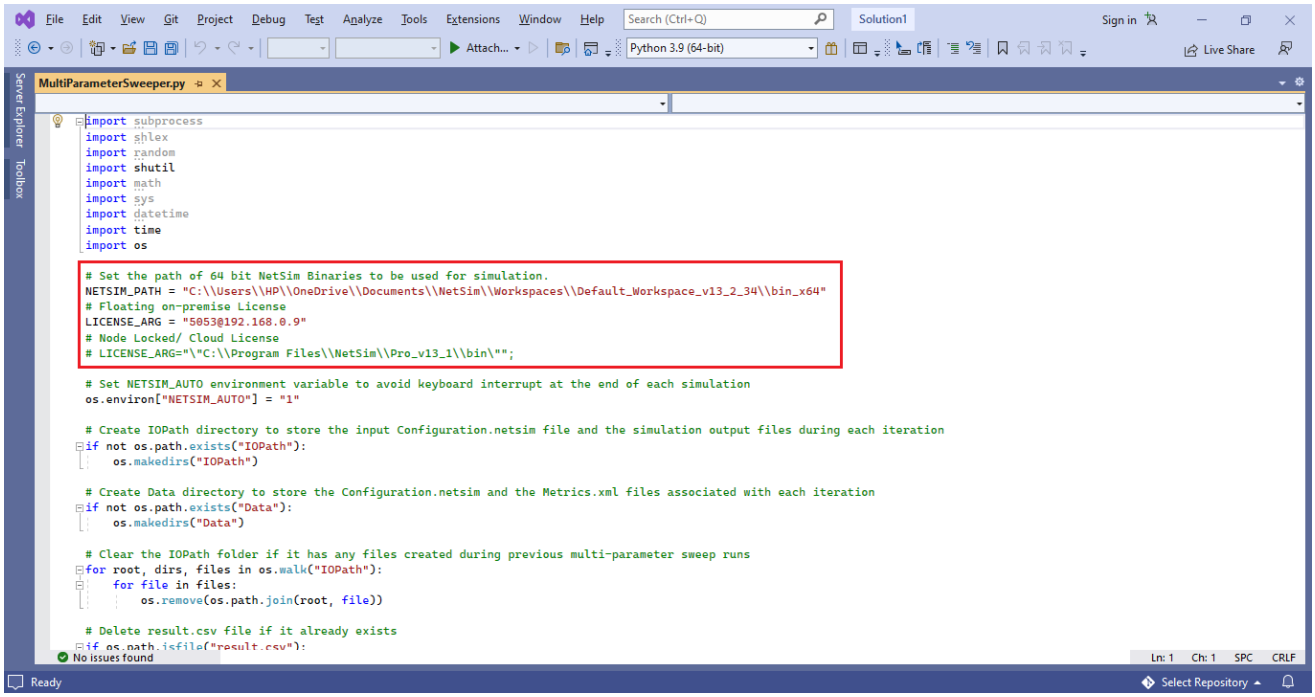


Figure 6: User need to set NetSim Path based on 64-bit

For example,

NETSIM_PATH=

"C:\\Users\\HP\\OneDrive\\Documents\\NetSim\\Workspaces\\Default_Workspace_v13_2_34\\bin_x64"

Floating on-premise license (<port no>@<server ip address>):

LICENSE_ARG = "5053@192.168.0.9"

Node Locked or Cloud licenses (path of license file):

LICENSE_ARG="\"C:\\Program Files\\NetSim\\Pro_v13_1\\bin\""

- The MetricsCSV.exe will convert the Metrics.xml file inside the output folder into a csv file.

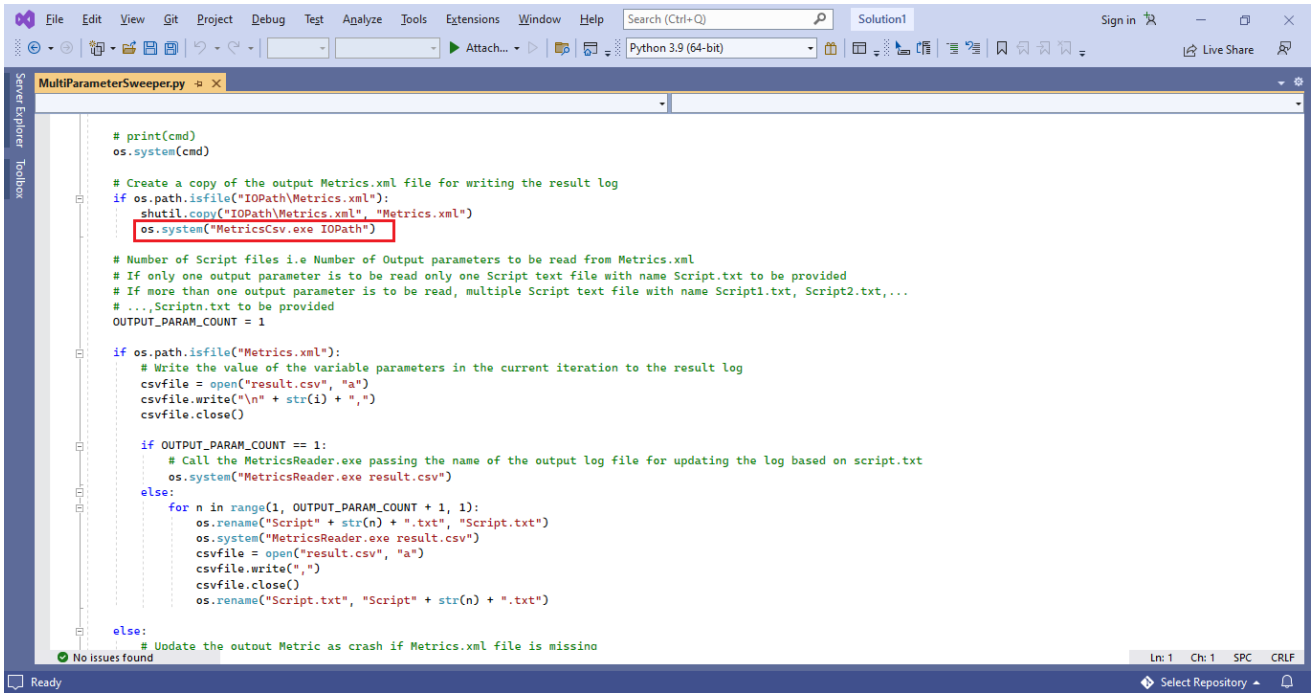


Figure 7: MetricsPrint.csv file will be created in the IOPath and then copied into the respective output folder

8. Multi-Parameter Sweeping process is started by opening command prompt in the directory of the Multi-Parameter-Sweeping project and starting the python script as shown below:

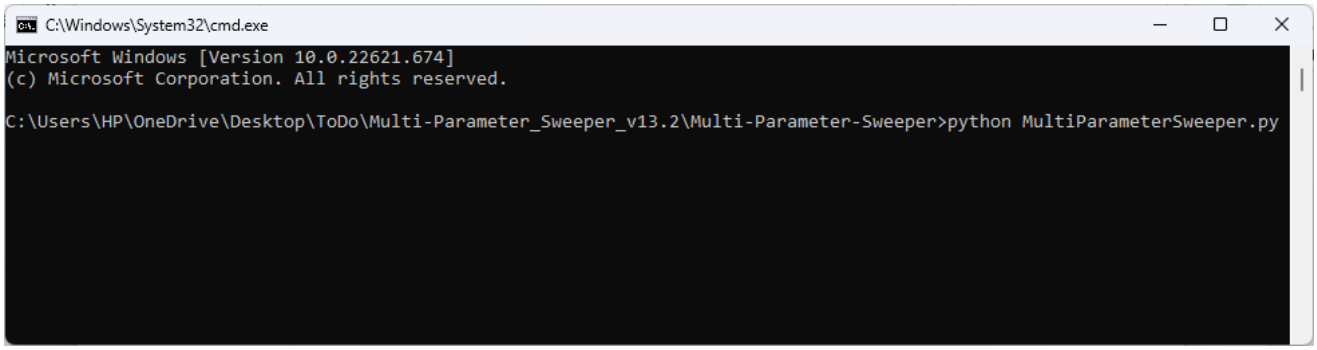


Figure 8: Running python script using cmd prompt

9. This starts the Multi-Parameter-Sweeping process which runs NetSim simulations iteratively for different values of Y parameter of UE.

10. At the end of the process the Multi-Parameter-Sweeping folder will have the following file and folders created:

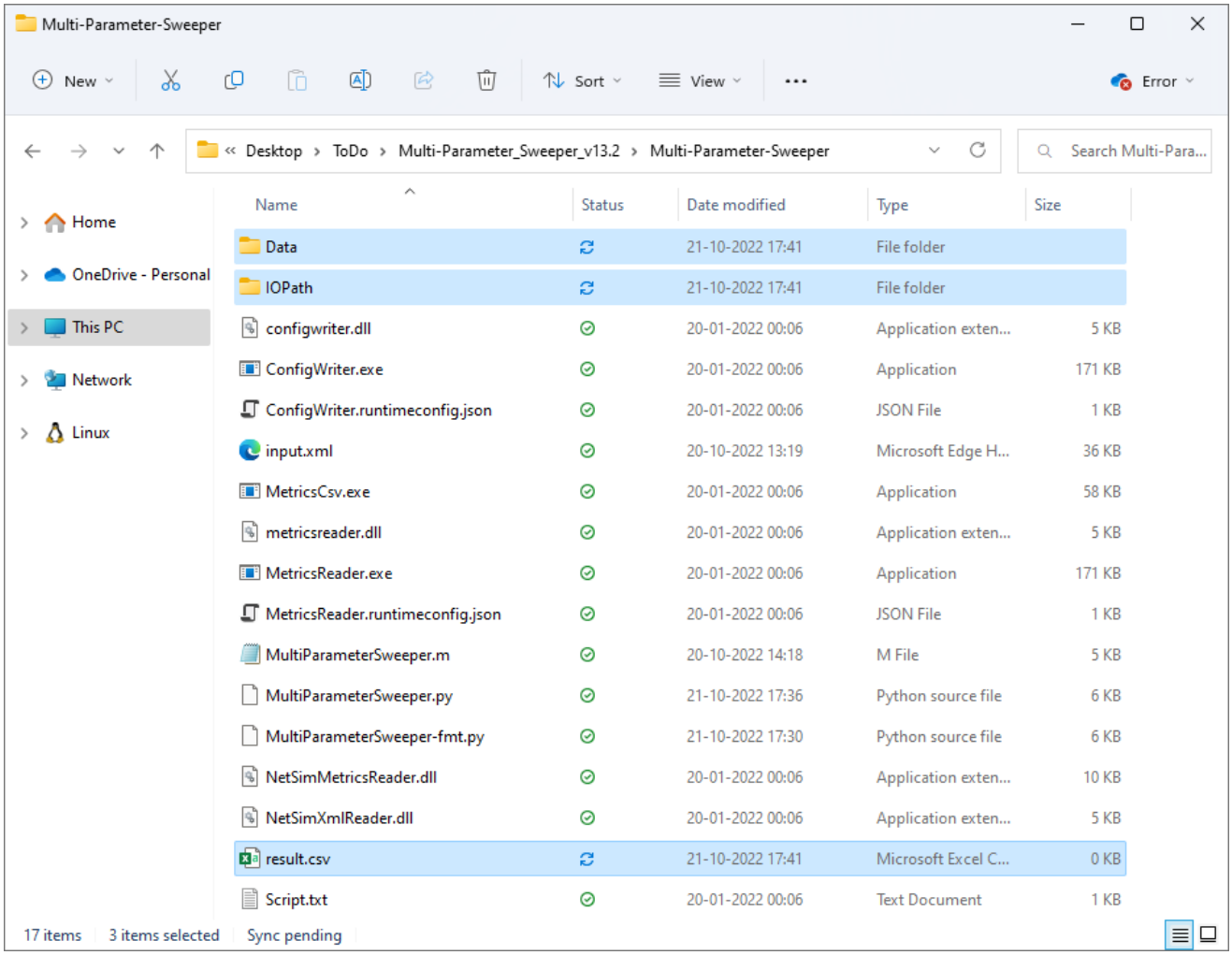


Figure 9: After Simulation Multi-Parameter-Sweeping folder contains output files like result.scv, Data etc

- **Data**: Contains multiple folders created based on date and time of simulation inside which multiple output folders corresponding to each simulation run, with its name including the value of the parameters in that iteration gets created.

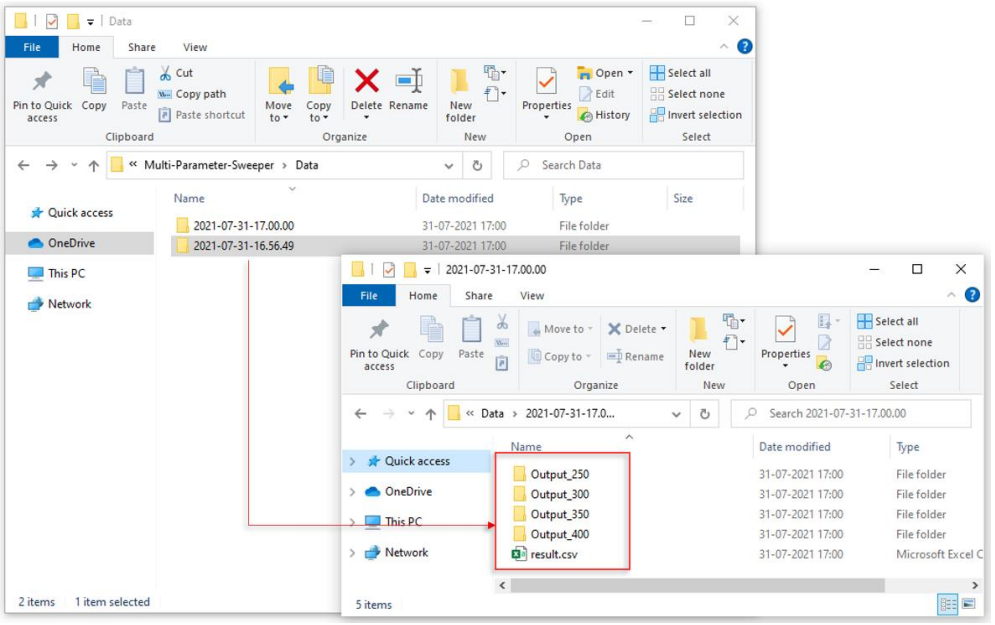


Figure 10: Based on values in the iteration, Output folder gets created in the Data directory inside a folder named as per date and time of simulation along with a copy of result.csv file.

- Each folder contains the all the output files associated with the simulation run.

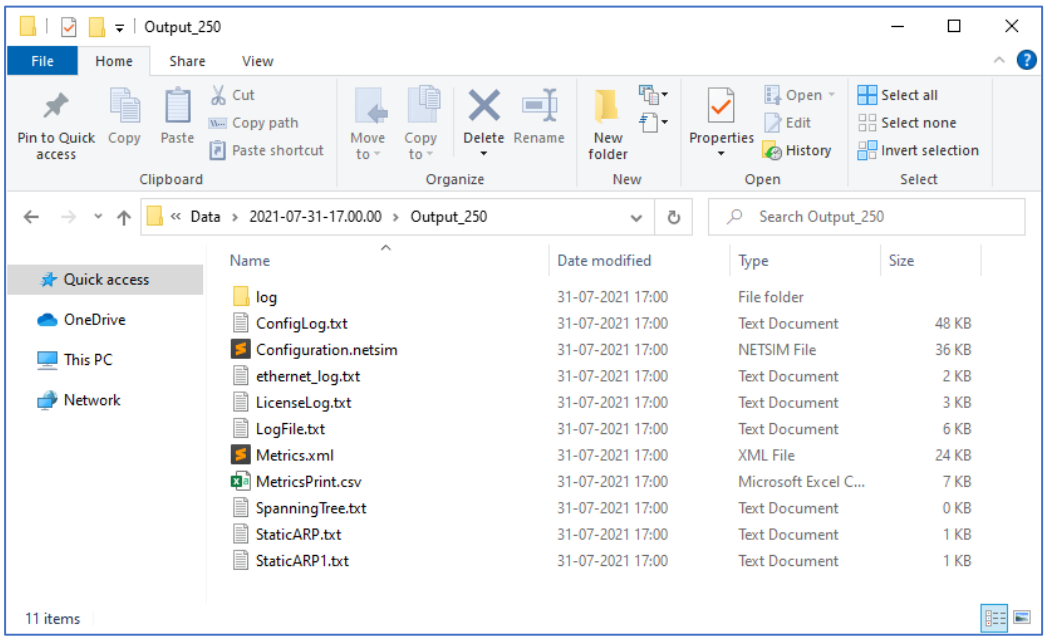


Figure 11: Each folder contains all the output files and the Metrics.xml file converted to MetricsPrint.csv file

Note: *User should keep a back-up of the data folder to avoid data loss.*

- **IO Path**: Used for storing the Configuration.netsim file and the simulation files generated during each simulation run.
- **Result.csv**: This is the output log which contains the parameter varied during each simulation run and the output parameter associated with each run. The result.csv file will also be copied into the output folder after each simulation.

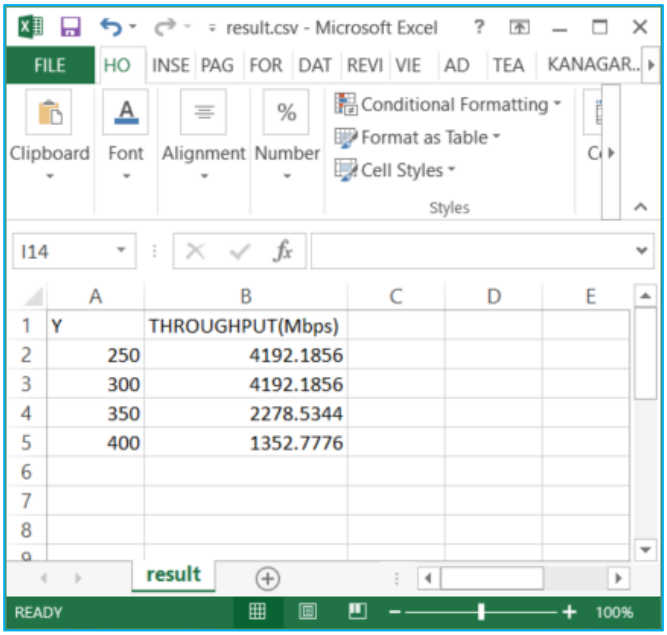


Figure 12: Iterated value and Throughput obtained listed in result.csv

Example 1: Modifying a single input parameter and logging a single output parameter

Consider the following network 5G network scenario in NetSim, comprising of a Wired Node, Router, gNB and a UE.

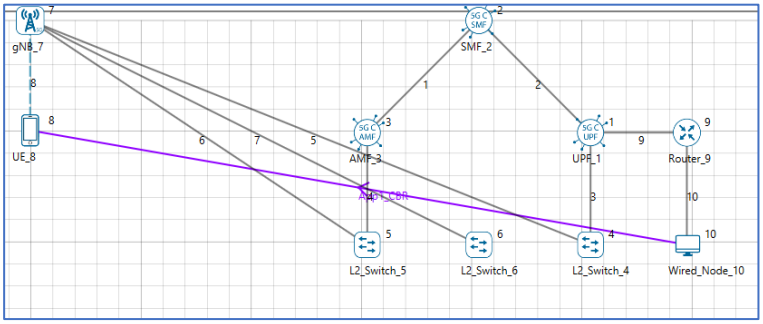


Figure 13: Network Topology

The network configuration has the initial distance between the gNB and UE as 50 meters with the gNB located at (500,0) and UE located at (500,500).

Multi-Parameter Sweeper is configured to run simulations for different distance between the gNB and UE by varying the UE Y coordinate value from 500 to 2000 in steps of 500 meters.

The network scenario is saved and the content of the Configuration.netsim file is copied to the Multi-Parameter-Sweeper directory and renamed as input.xml.

Refer to the Example 1 directory which is part of the project folder (Multi-Parameter-Sweeper_v13.1\Examples\Multi-Parameter-Sweeper-Example-1)

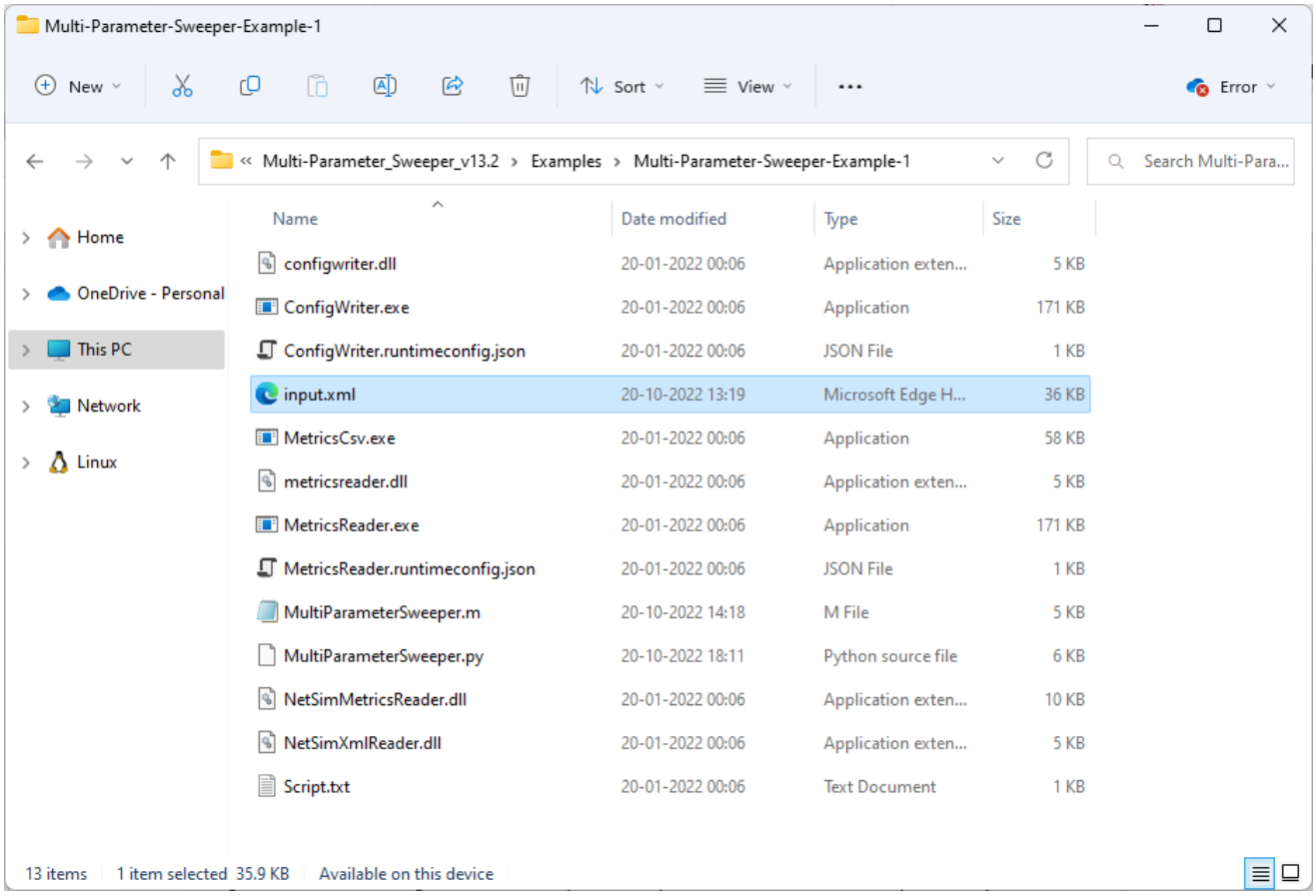


Figure 14: Renamed Configuration.netsim to input.xml and pasted in Multi-Parameter-Sweeper directory

1. The value of the Y coordinate of UE that is to be modified during each simulation run is updated (“{0}”) in the configuration file as shown below:

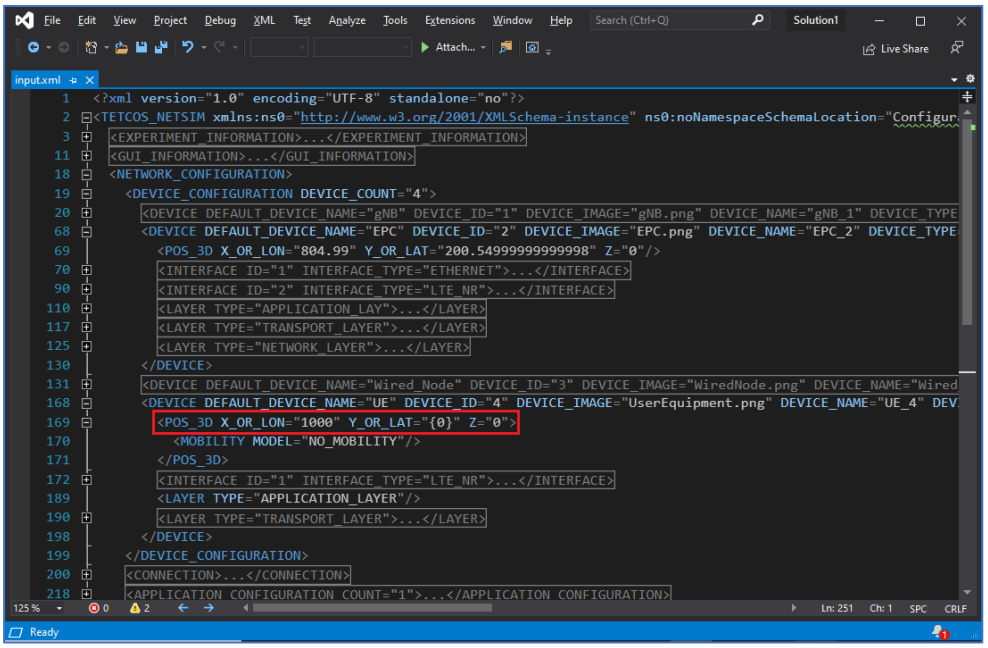


Figure 15: Modified Y coordinate of UE in input.xml

2. The Script.txt file is updated with the details of the output parameter to be read from the Metrics.xml file and added to the result csv log file. In this case the Application throughput is to be logged for each simulation run.

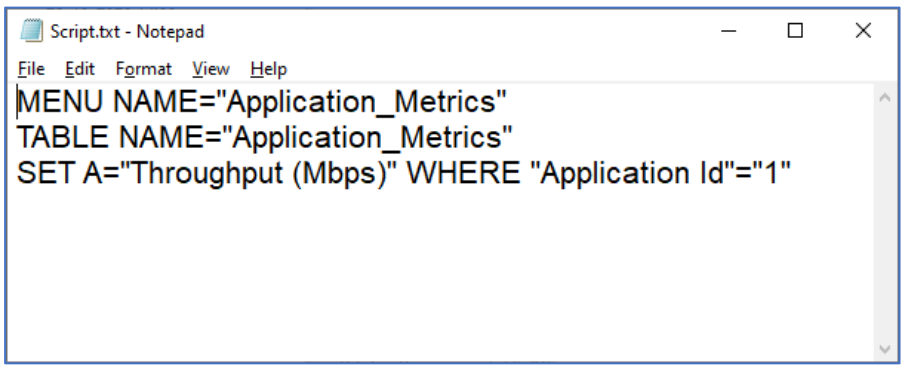


Figure 16: The application throughput is to be logged for each simulation modified in Script.txt

3. MultiParameterSweeper.py is updated to pass the Y coordinate value during each iteration to generate Configuration file run simulation and update the result csv log.

The MultiParameterSweeper.bat batch script modified for running simulations for different values of Y coordinates starting from 500 up to 2000 in steps of 500 is shown below:

- A result.csv file is created and added with headings Y and Throughput (Mbps).
- A MetricsPrint.csv is created inside every output folder.
- For loop is set to iteratively run simulations for values starting from 500 to 2000 in steps of 500.
- The value of the parameter Y in the current iteration is written to the result log file for analysis.
- The value of the parameter Y in the current iteration is passed as input to ConfigWriter executable to generate Configuration.netsim file for each simulation.
- NetSim simulation is run via CLI mode by passing the apppath, iopath and license server information
- Configuration file and Metrics file are copied and renamed appending the value of the parameter in the current iteration.

The MultiParameterSweeper.py python script modified for running simulations for different values of Y coordinates starting from 500 up to 2000 in steps of 500 is shown below:

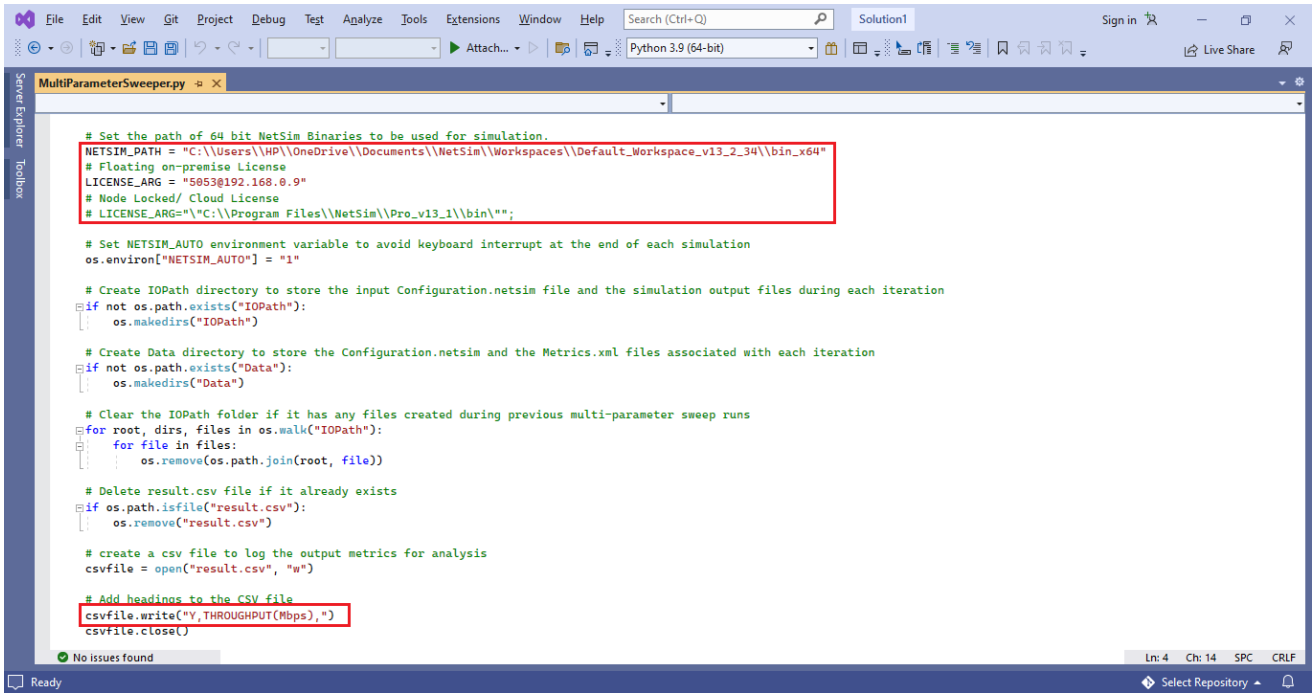


Figure 17: To Create result.csv file, added with headings Y and Throughput (Mbps) and NetSim installation Path and License information

- NETSIM_PATH variable is set to the path of NetSim 64-bit binaries in the install directory or workspace in the system.
- LICENSE_ARG variable is set to the license server details
- A result.csv file is created and added with headings Y and Throughput (Mbps).

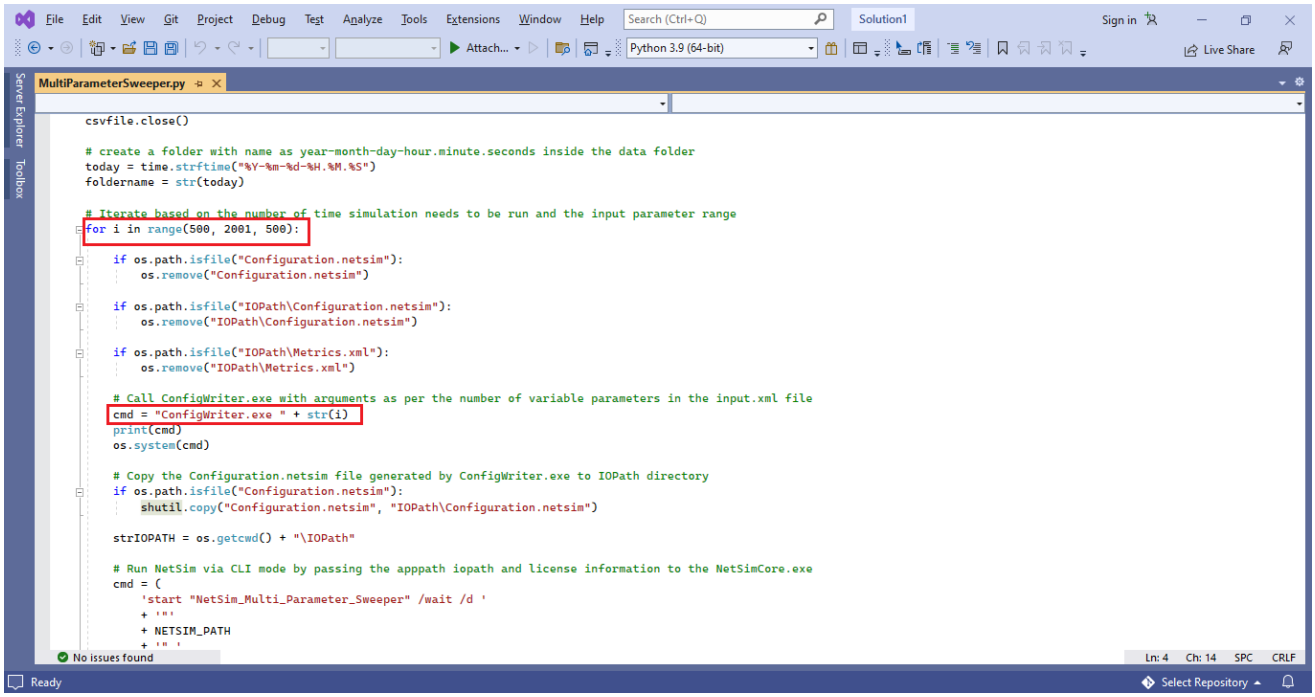


Figure 18: Varying Distance and set license server information

- For loop is set to iteratively run simulations for values starting from 500 to 2000 in steps of 500.
- The value of the parameter Y in the current iteration is passed as input to ConfigWriter executable to generate Configuration.netsim file for each simulation.

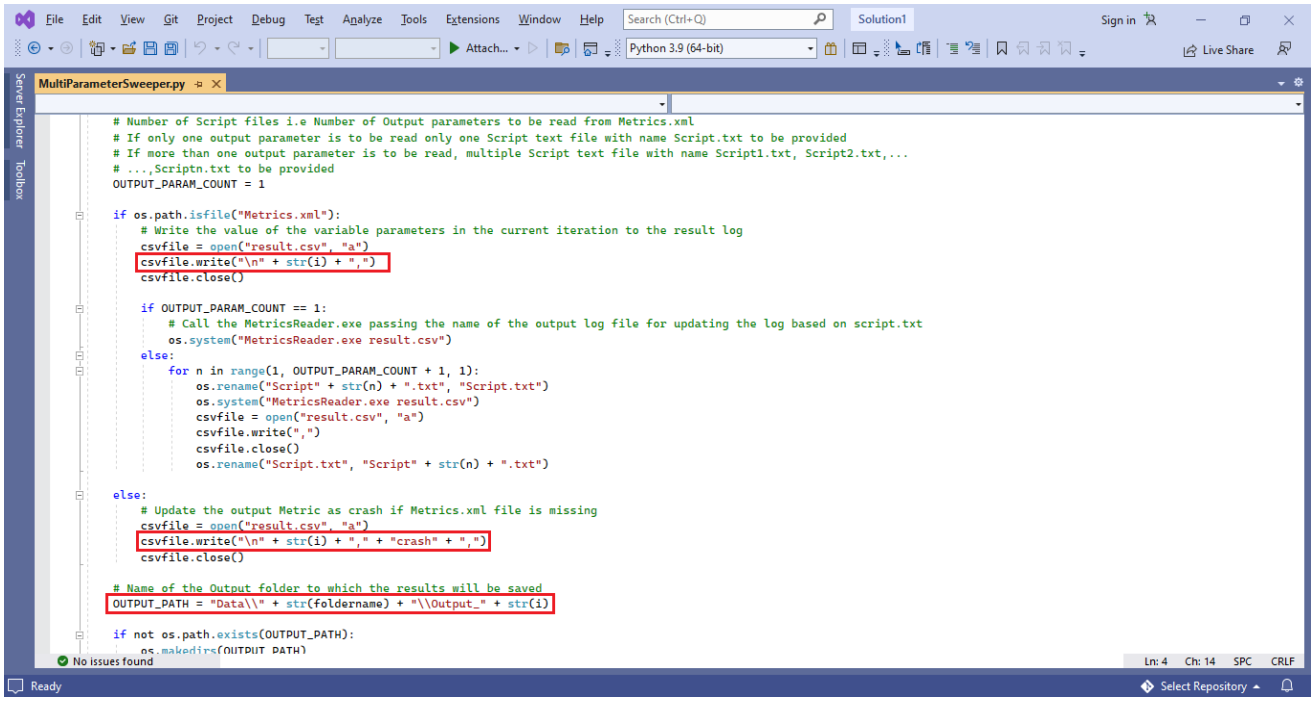


Figure 19: Modify parameters in MultiParameterSweeper.py

- The value of the parameter Y in the current iteration is written to the result log file for analysis.
- Configuration file and Metrics file are copied and renamed appending the value of the parameter in the current iteration.

4. Multi-Parameter Sweeping process is started by opening command prompt in the directory of the Multi-Parameter-Sweeping project and starting the python script as shown below:

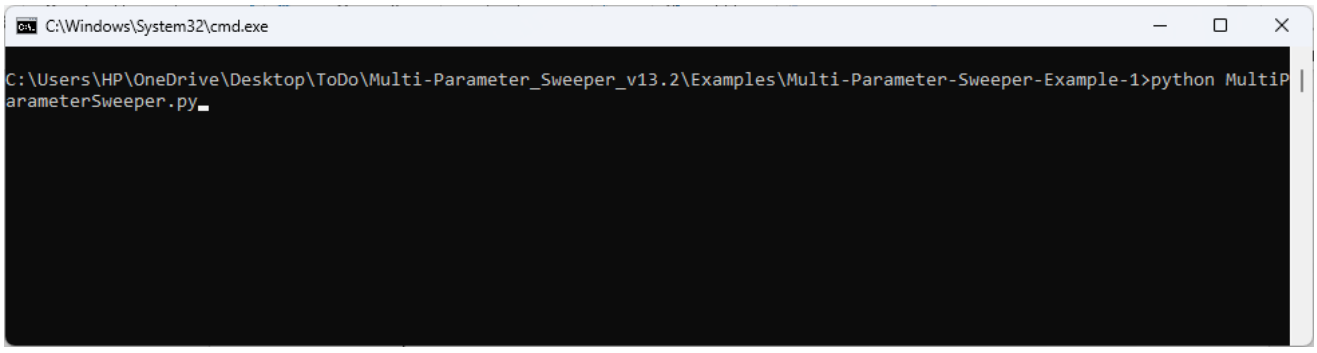


Figure 20: Running python script using cmd prompt

This starts the Multi-Parameter-Sweeping process which runs NetSim simulations iteratively for different values of Y parameter of UE.

At the end of the process the Multi-Parameter-Sweeping folder will have the following file and folders created:

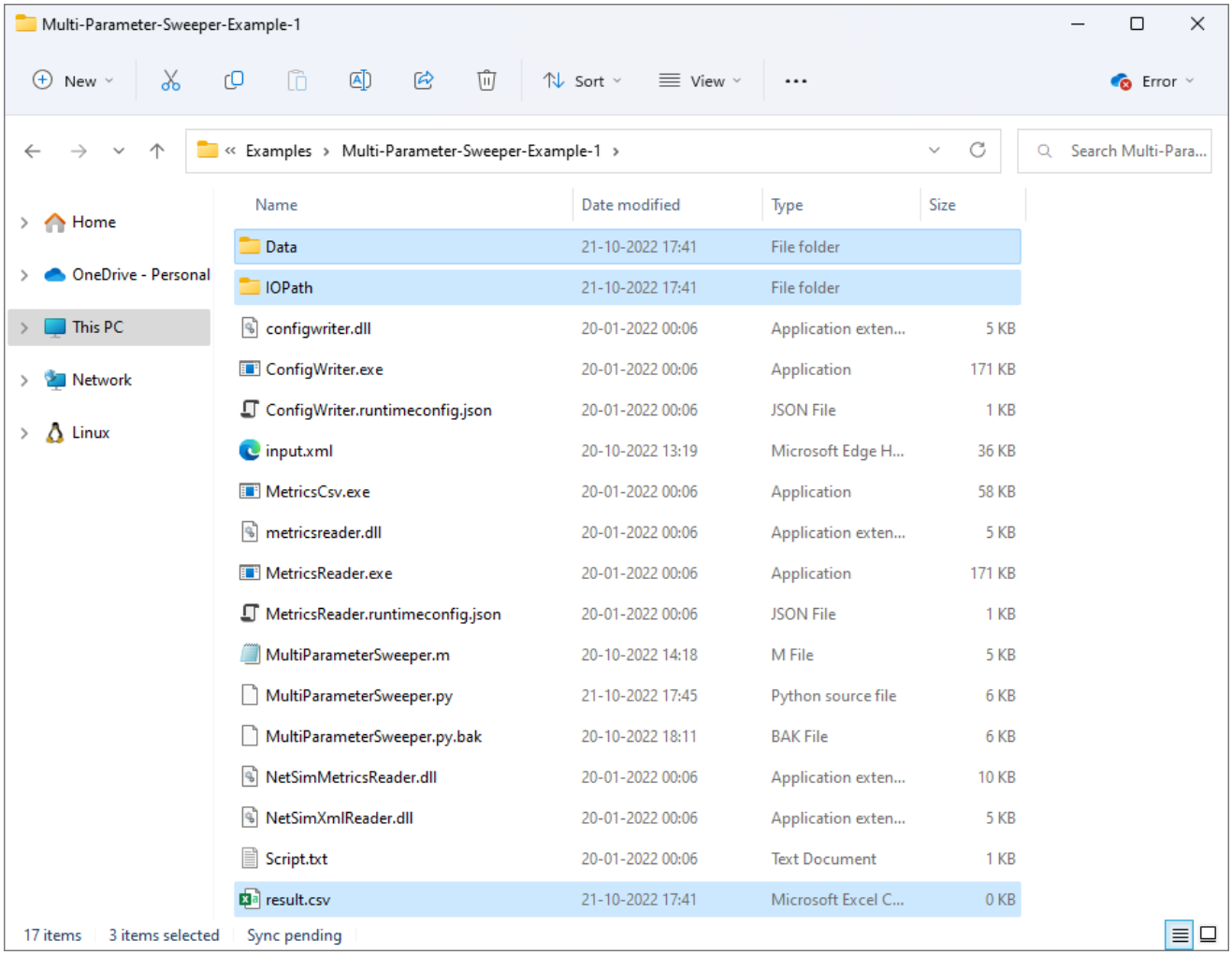


Figure 21: After Simulation Multi-Parameter-Sweeping folder contains output files like result.csv, Data etc

- **Data**: Contains multiple folders corresponding to each simulation run, with its name including the value of the parameters in that iteration. The output folders will be created inside folder with name in the format Year-Month-Day-Hours-Minutes-Seconds.

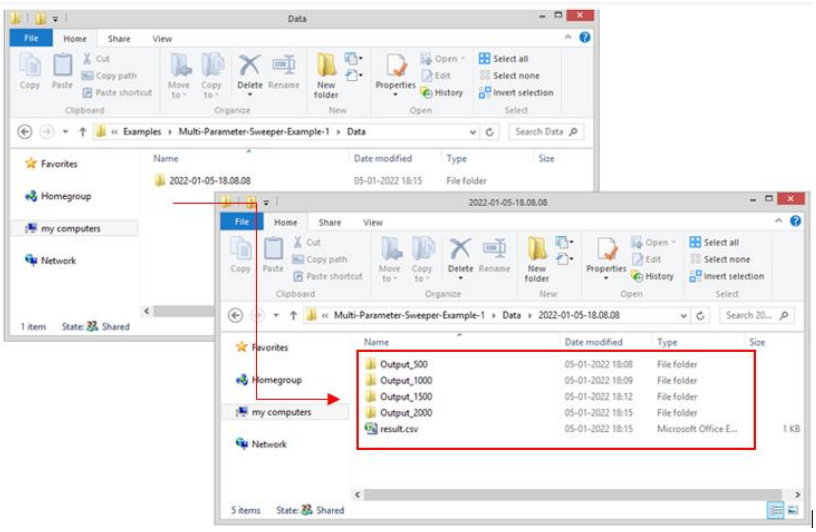


Figure 22: Based on distance Configuration.netsim files created in output folder

- Each folder contains the all the output files associated with the simulation run.

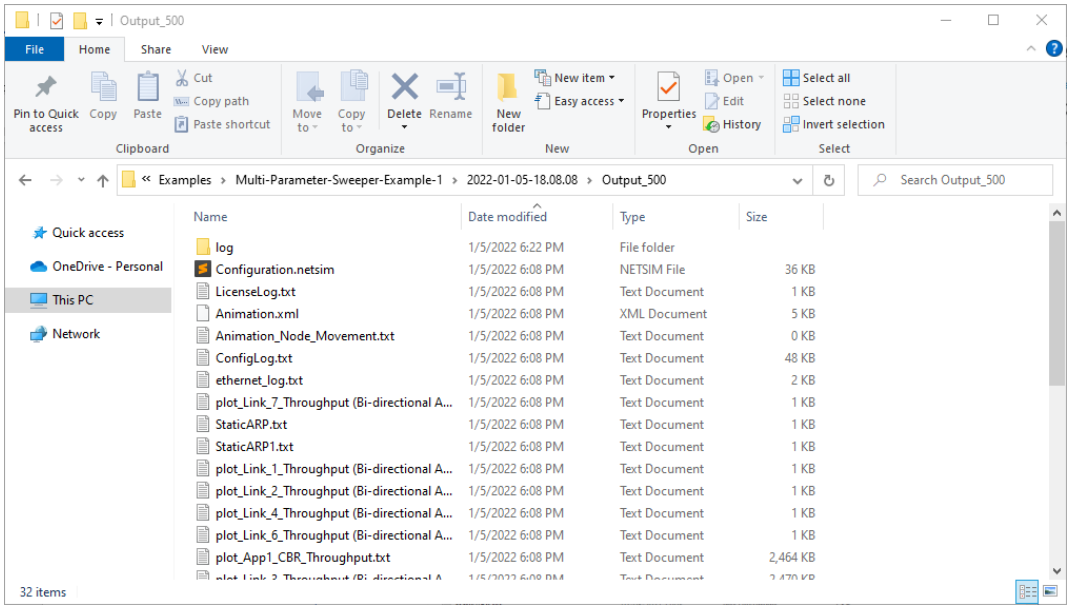


Figure 23: Each folder contains the all the output files

- **IOPath**: Used for storing the Configuration.netsim file and the simulation files generated during each simulation run.
- **Result.csv**: This is the output log which contains the parameter varied during each simulation run and the output parameter associated with each run.

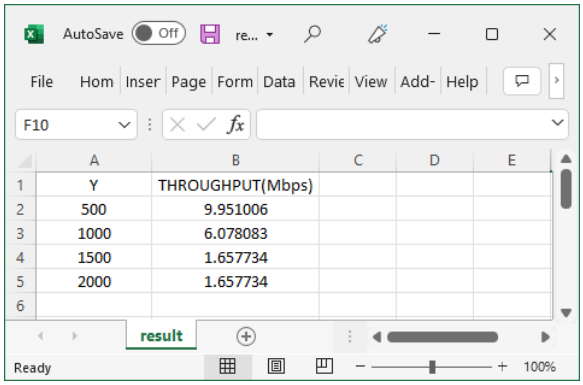


Figure 24: Distance Vs. Throughput obtained in result.csv

Varying multiple network parameters

In order to vary multiple network parameters during the multi-parameter sweep process each parameter in the input.xml file can be modified as {0},{1},{2},{3},..{n} respectively.

Logging multiple output parameters

Each output parameter that is to be logged should be part of the Script.txt file. However, the Script.txt file should contain only the details of one output parameter during the call to MetricsReader.exe.

To log multiple parameters, multiple script files can be used. If n output parameters are to be logged, then there can be script1.txt, script2.txt, script.txt in the sweeper folder.
For Example, there can be two Script files as shown below:

Name	Date modified	Type	Size
Data	21-10-2022 17:16	File folder	
IOPath	21-10-2022 17:17	File folder	
configwriter.dll	20-01-2022 00:06	Application exten...	5 KB
ConfigWriter.exe	20-01-2022 00:06	Application	171 KB
ConfigWriter.runtimeconfig.json	20-01-2022 00:06	JSON File	1 KB
input.xml	20-10-2022 18:40	Microsoft Edge H...	37 KB
MetricsCsv.exe	20-01-2022 00:06	Application	58 KB
metricsreader.dll	20-01-2022 00:06	Application exten...	5 KB
MetricsReader.exe	20-01-2022 00:06	Application	171 KB
MetricsReader.runtimeconfig.json	20-01-2022 00:06	JSON File	1 KB
MultiParameterSweeper.m	21-10-2022 17:16	M File	7 KB
MultiParameterSweeper.py	21-10-2022 17:45	Python source file	8 KB
NetSimMetricsReader.dll	20-01-2022 00:06	Application exten...	10 KB
NetSimXmlReader.dll	20-01-2022 00:06	Application exten...	5 KB
result.csv	21-10-2022 17:18	Microsoft Excel C...	1 KB
Script1.txt	20-01-2022 00:06	Text Document	1 KB
Script2.txt	20-01-2022 00:06	Text Document	1 KB

Figure 25: Multiple output parameters

Example 2: Modifying multiple input parameters and logging multiple output parameter

1. Consider the following network 5G network scenario in NetSim, comprising of a Wired Node, Router, gNB and a UE.

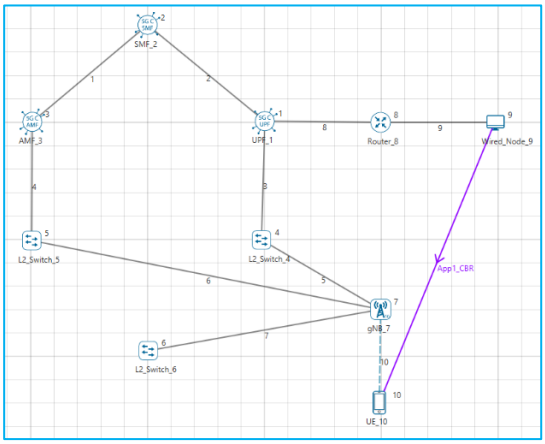


Figure 26: Network Scenario

2. Properties configured in the LTE_NR interface of the gNB is shown in the table below:

Interface(5G_RAN) Properties

Tx_Power(dBM)	40
Tx_Antenna_Count	8
Rx_Antenna_Count	4
CA_Type	Single Band
CA_Configuration	n78
CA_Count	1
Numerology	0
Channel Bandwidth (MHz)	10
PRB Count	52
MCS Table	QAM64
CQI Table	Table 1
X_Overhead	XOH0
DL UL Ratio	4:1
Outdoor Scenario	Rural Macro
LOS Mode	Standard
Wireless Link Properties	
Channel Characteristics	No_Pathloss
Wired Link Properties	
Link Speed (Mbps)	10000
BER	0
Propagation Delay (µs)	0
Application Properties	
Packet Size (Byte)	1460
Inter Arrival Time (µs)	166
Generation Rate (Mbps)	100
Transport Control	UDP
Start Time (s)	1
QoS	BE
Simulation Parameters	
Simulation Time (s)	1.1

Table 1: gNB Properties

3. Traffic is generated at a rate of 70 Mbps and upon running simulation, the throughput achieved is 59.95 Mbps.
4. We now find the max throughput for each possible bandwidth; Tx Antenna count and Rx Antenna count combination varying the generation rate based accordingly.
5. Two more parameters to be taken care include, the PRB Count and Guard Band (KHz) which vary with respect to the bandwidth.

Input Variables	Value Range
Channel Bandwidth (MHz)	10,15,20,25,30,40,50
Tx_Antenna_Count	1,2,4,8,16,32,64,128
Rx_Antenna_Count	1,2,4,8,16
PRB Count	52,79,106,133,160,216,270
Guard Band (KHz)	312.5,382.5,452.5,522.5,592.5,552.5,692.5
Reference Inter Arrival Time (Microseconds)	166
Reference Bandwidth	10
Reference DL MIMO Layer Count	2

Table 2: Input variable values

6. Inter Arrival Time for each case is calculated based on the Reference IAT Bandwidth and DL MIMO Layer Count as shown below:

$$Inter\ Arrival\ Time\ (Micro\ Seconds) = \frac{Ref\ IAT}{\left(\frac{Curr\ BW}{Ref\ BW}\right) * \left(\frac{Curr\ DL\ MIMO\ Count}{Ref\ DL\ MIMO\ Count}\right)}$$

For E.g. In case of Bandwidth of 20 MHz and DL MIMO Count of 4 inter arrival time is

$$Inter\ Arrival\ Time\ (Micro\ Seconds) = \frac{166}{\left(\frac{20}{10}\right) * \left(\frac{4}{2}\right)} = 41.5\ Mbps$$

7. The network scenario is saved and the content of the Configuration.netsim file is copied to the Multi-Parameter-Sweeper directory and renamed as input.xml.
8. Refer to the Example 2 directory which is part of the project folder (Multi-Parameter-Sweeper_v13.1\Examples\Multi-Parameter-Sweeper-Example-2).

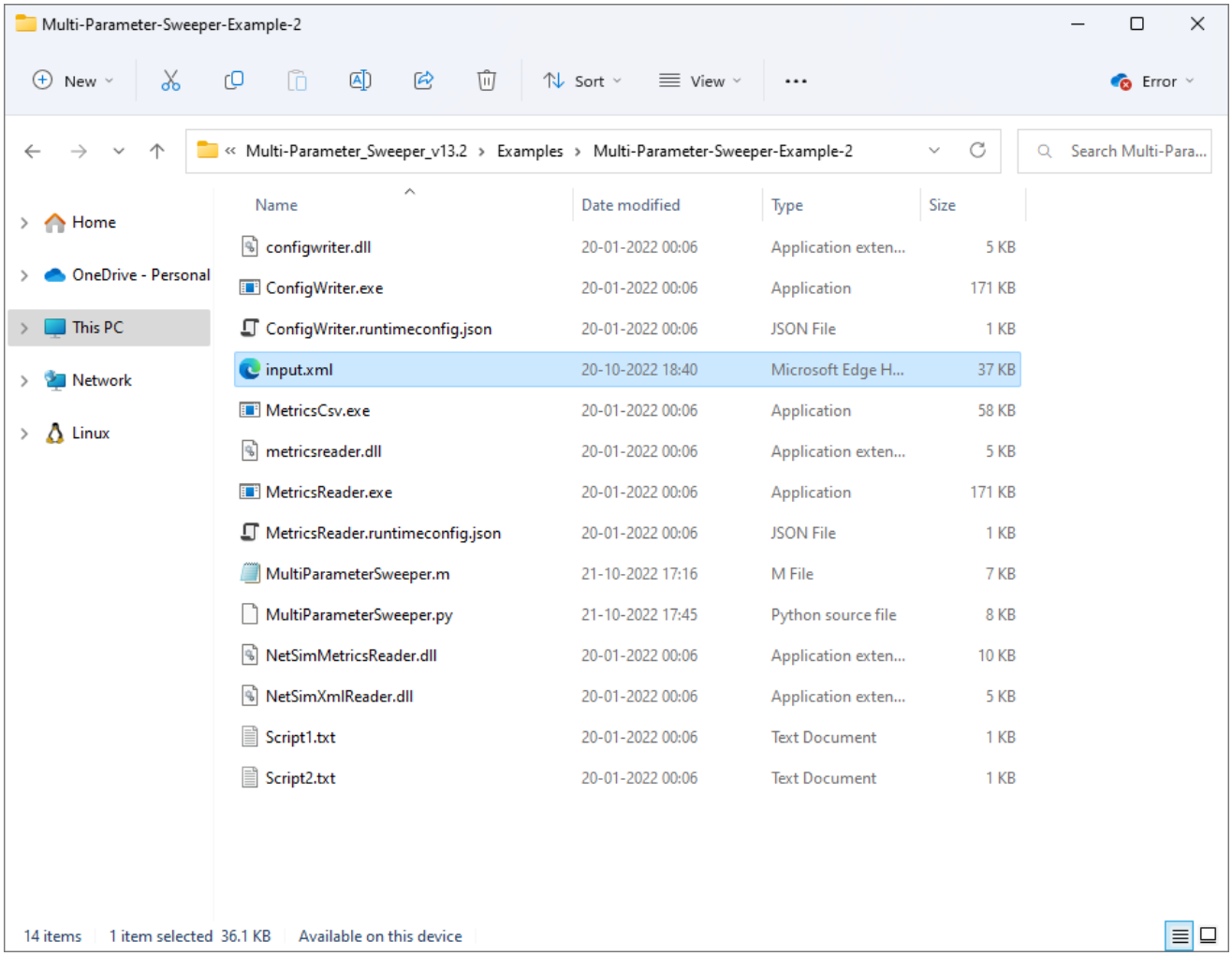


Figure 27: Renamed Configuration.netsim to input.xml and pasted in Multi-Parameter-Sweeper directory

9. In the Input.xml file the value of the input variables are modified as shown in the table below:

Input Variables	
Channel Bandwidth (MHz)	{0}
Tx Antenna Count	{1}
Rx Antenna Count	{2}
Inter Arrival Time (Microseconds)	{3}
PRB Count	{4}
Guard Band (KHz)	{5}

Table 3: Variables are modified to the input.xml file

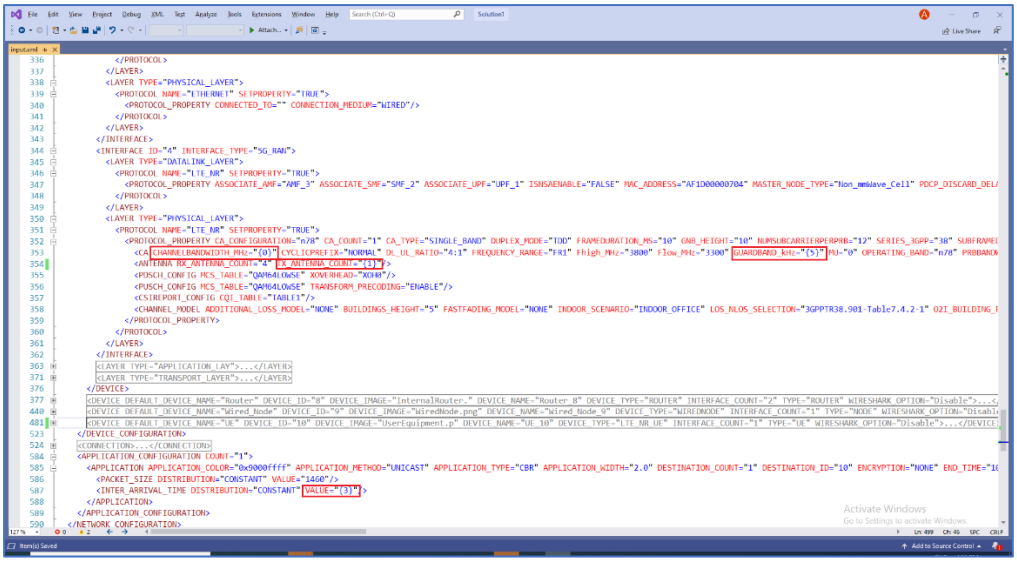


Figure 28: The above table 3 Variables are modified in input.xml file

10. The python script MultiParameterSweeper.py is modified to run simulation for all possible combinations of Bandwidth and Tx Antenna Count and Rx Antenna Count with the respective values of Guard Band, PRB Count and the IAT that is calculated.

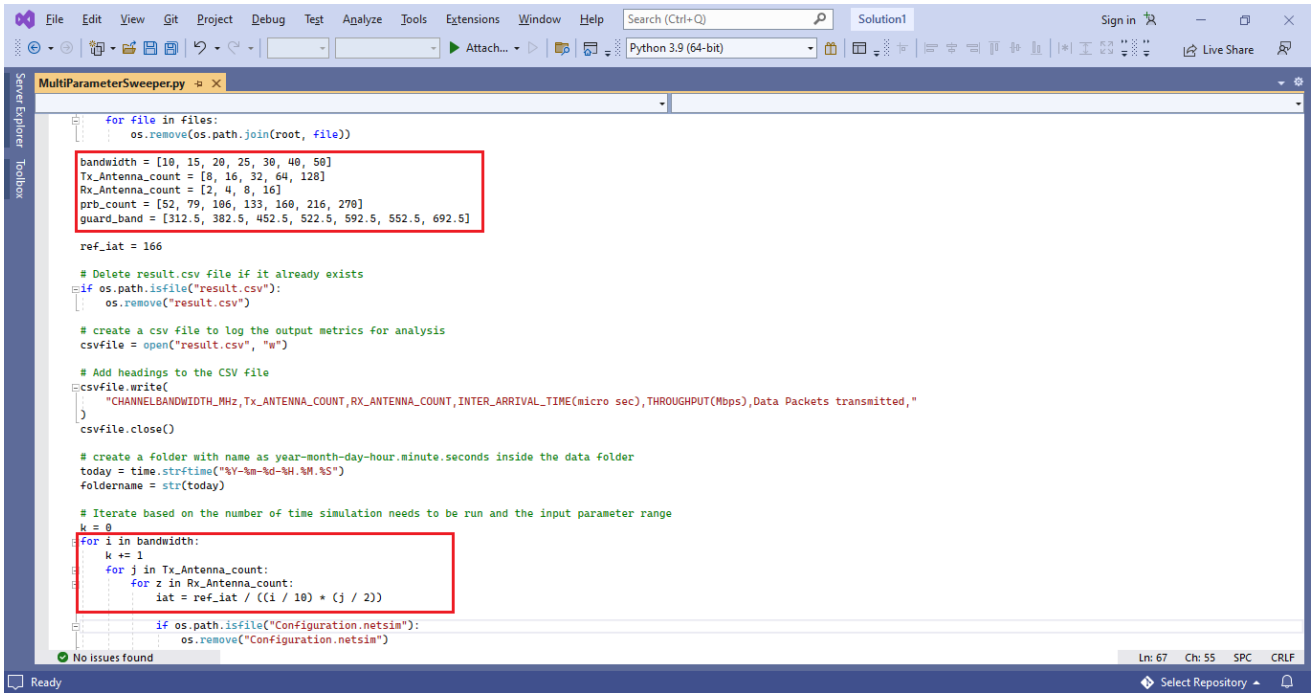


Figure 29: Modified MultiParameterSweeper.py based on input parameter

11. Multiple parameters are read from the Metrics.xml file and logged in the results.csv file along with the input parameters such as CHANNELBANDWIDTH_MHz, TX_ANTENNA_COUNT, RX_ANTENNA_COUNT, INTER_ARRIVAL_TIME (micro sec).

Output Parameters
Throughput (Mbps)

Data Packets transmitted

Table 4: User need to modify these two parameters in Script files

12. Two script text files namely Script1.txt and Script2.txt are created with information to read each of the parameters from the Metrics.xml file. The variable OUTPUT_PARAM_COUNT is set to 2 as per the number of Script files.

Script1.txt

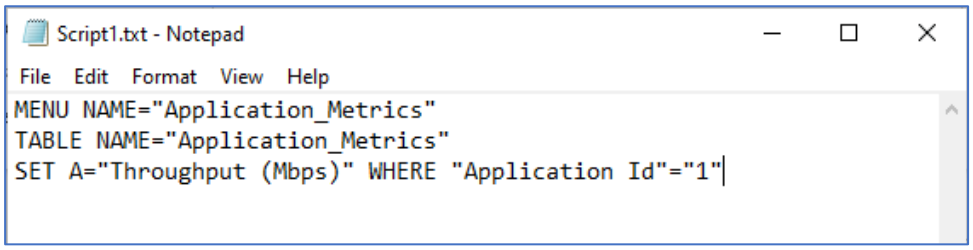


Figure 30: The application throughput is to be logged for each simulation modified in Script1.txt

Scritp2.txt

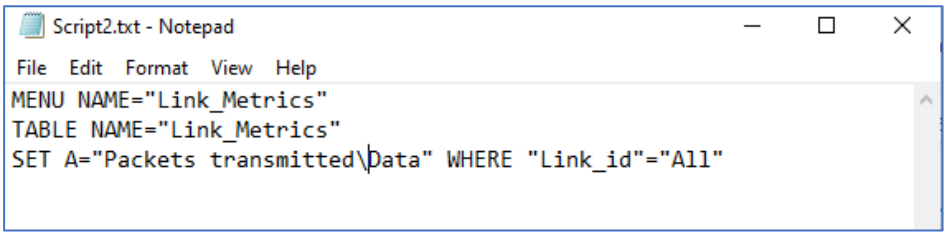


Figure 31: The Data Packets transmitted is to be logged for each simulation modified in Script2.txt

13. In the python script MultiParameterSweeper.py, MetricsReader is called to log each parameter specified in the script text files separating the entries with a comma (“,”). If simulation crashes, without generating the output Metrics.xml, then “crash” message is written to the log for each output parameter. The input parameters that were varied during each simulation run are also logged in the results.csv file.

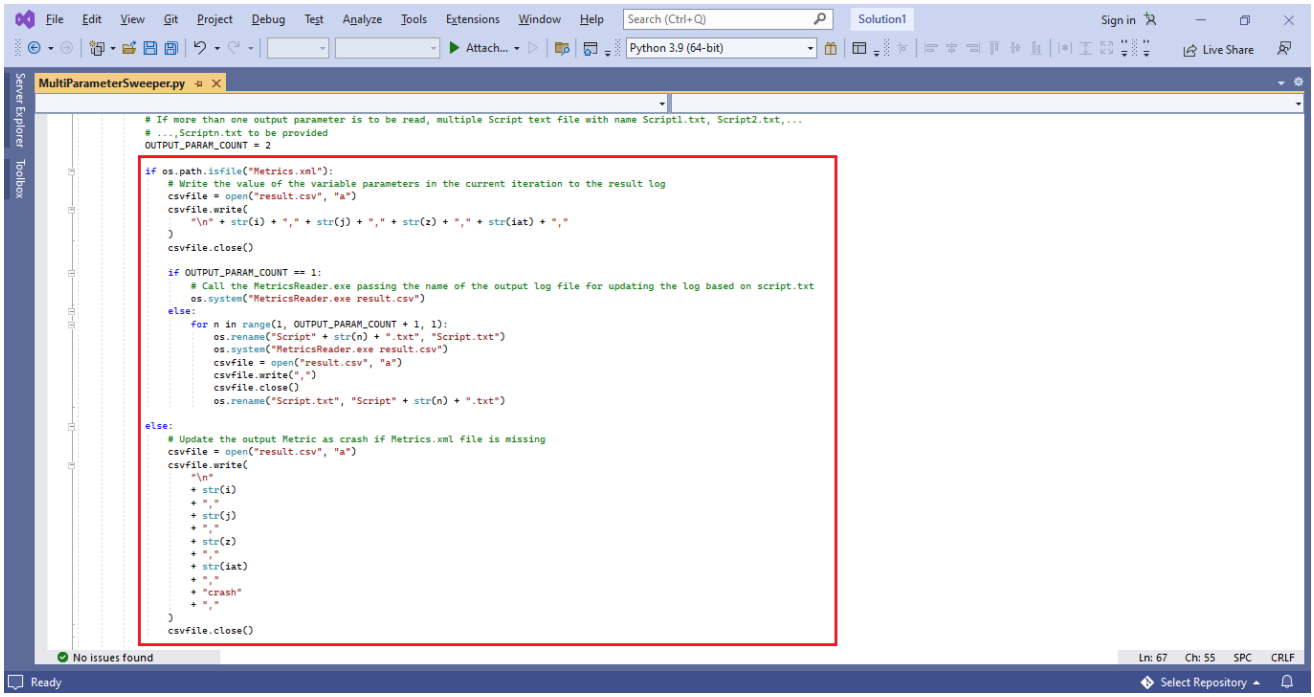


Figure 32: Modify python script MultiParameterSweeper.py file

14. The simulation Configuration file and all the output files associated with each simulation run is saved to folders with name including the bandwidth and DL MIMO count values that were used during each simulation run.

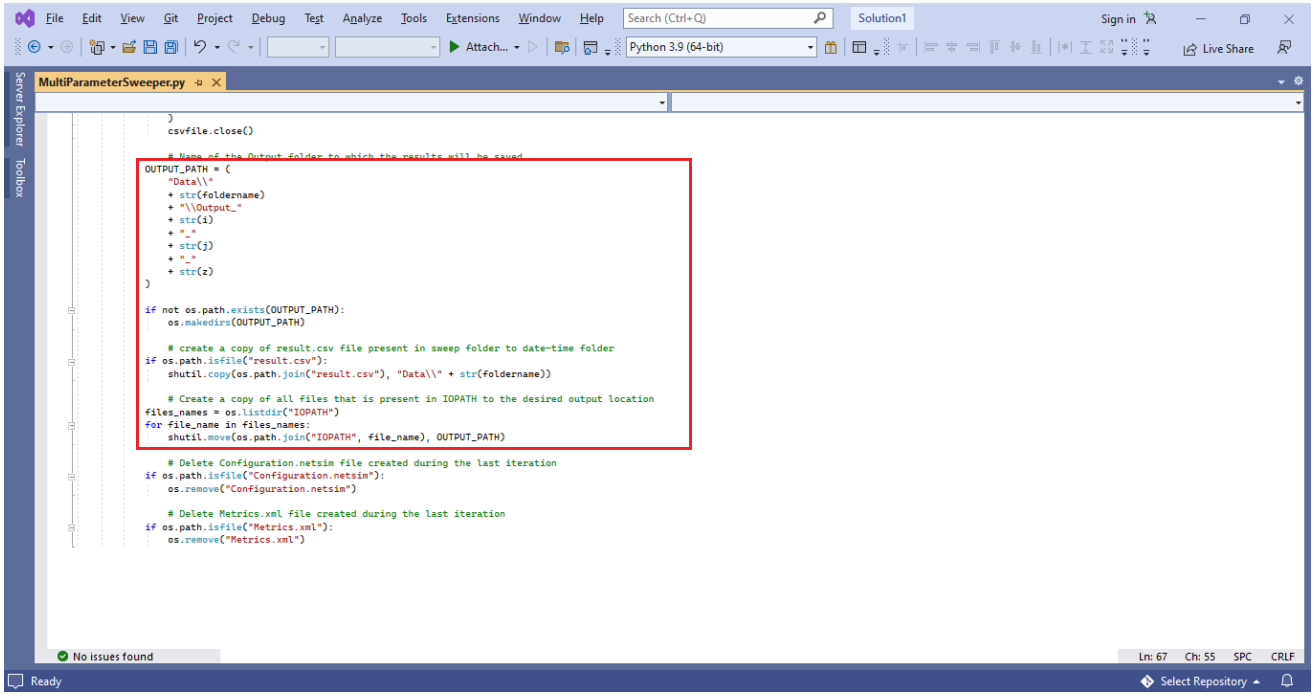


Figure 33: Modify python script MultiParameterSweeper.py file

15. Multi-Parameter Sweeping process is started by opening command prompt in the directory of the Multi-Parameter-Sweeping project and starting the python script as shown below:

Python Script:

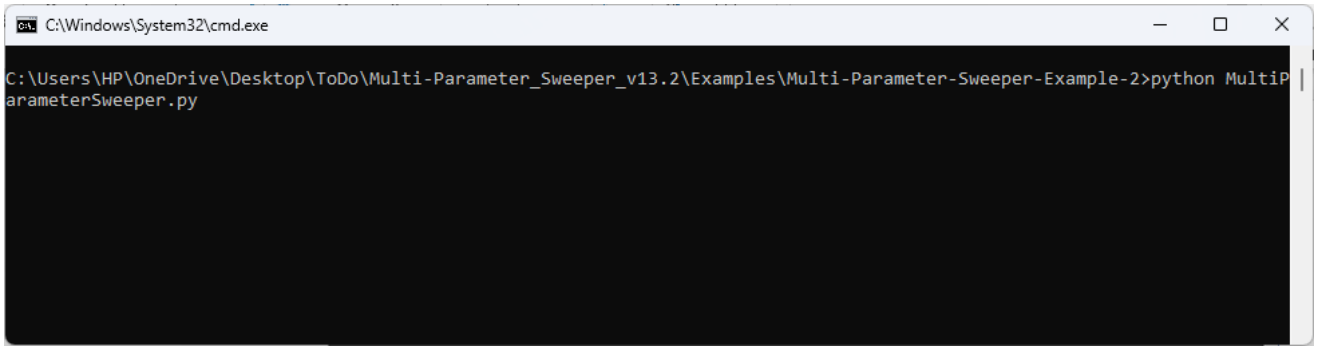


Figure 34: Running Python Script using cmd Prompt

This starts the Multi-Parameter-Sweeping process which runs NetSim simulations iteratively for different combinations of input parameters.

At the end of the process the Multi-Parameter-Sweeping folder will have the following file and folders created:

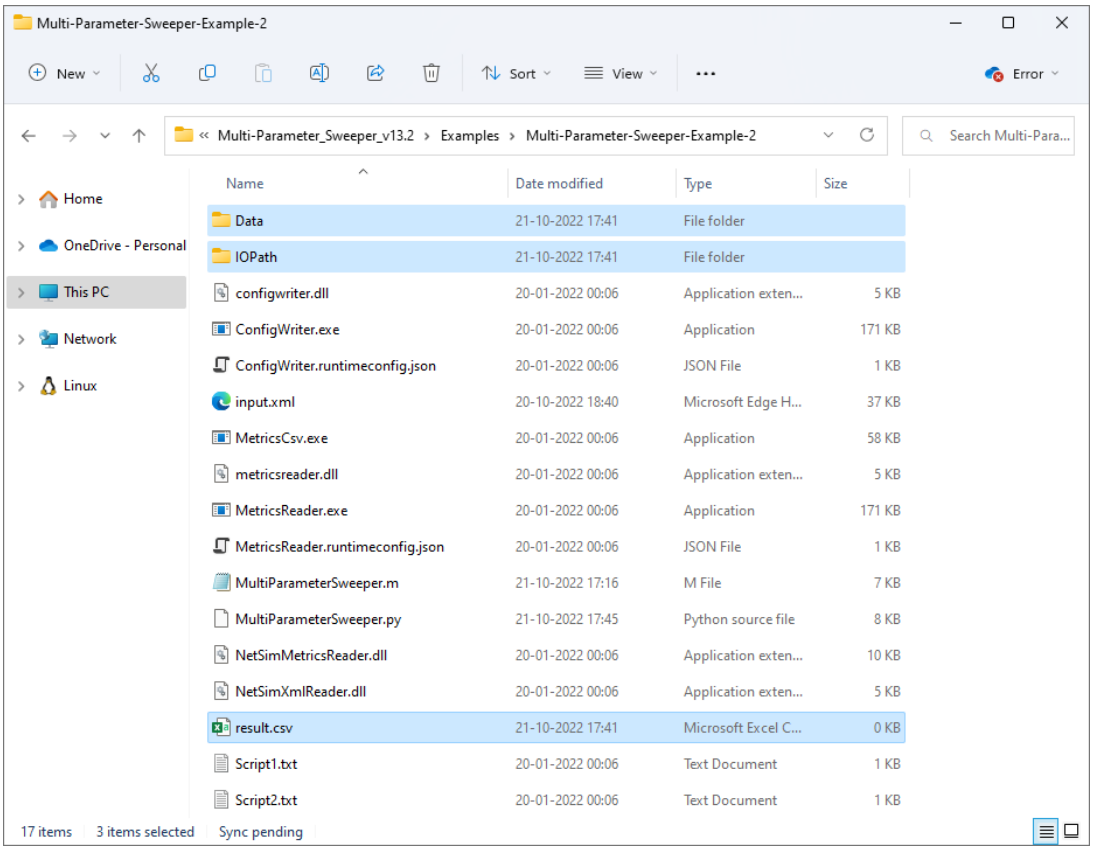


Figure 35: After Simulation Multi-Parameter-Sweeping folder contains output files like result.csv, Data etc

16. **Data**: The Data directory contains multiple output folders with the output files associated with each simulation run.

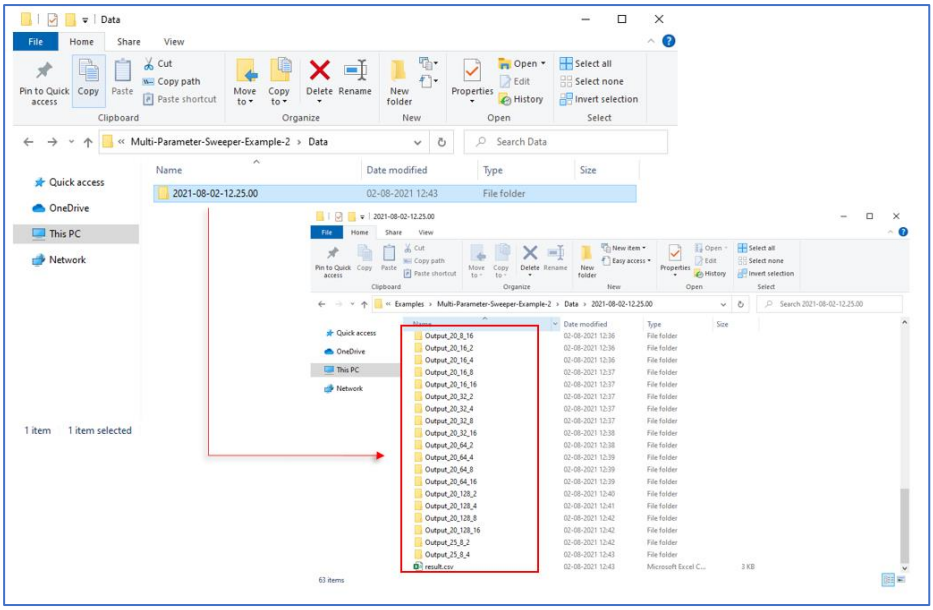
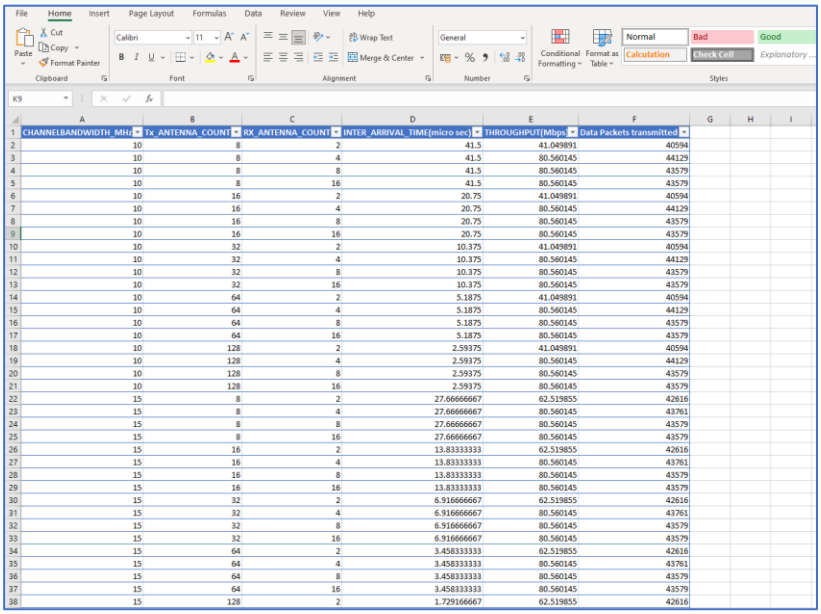


Figure 36: Based on input parameter Configuration.netsim and other files created in output folder

17. **IOPath**: Used for storing the Configuration.netsim file and the simulation files generated during each simulation run.
18. **Result.csv**: This is the output log which contains the parameter varied during each simulation run and the output parameter associated with each run.

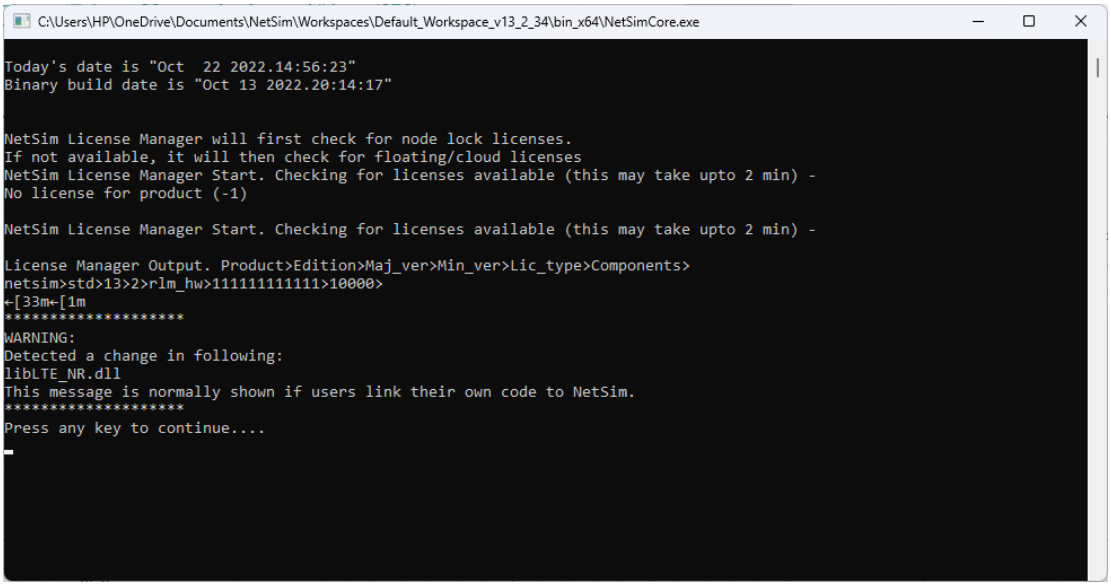


CHANNELBANDWIDTH_MHz	Tx_ANTENNA_COUNT	RX_ANTENNA_COUNT	INTER_ARRIVAL_TIME(micro sec)	THROUGHPUT(Mbps)	Data Packets transmitted
10	8	2	41.5	41.049891	40594
10	8	4	41.5	80.560145	44129
10	8	8	41.5	80.560145	43579
10	8	16	41.5	80.560145	43579
10	16	2	20.75	41.049891	40594
10	16	4	20.75	80.560145	44129
10	16	8	20.75	80.560145	43579
10	16	16	20.75	80.560145	43579
10	32	2	10.375	41.049891	40594
10	32	4	10.375	80.560145	44129
10	32	8	10.375	80.560145	43579
10	32	16	10.375	80.560145	43579
10	64	2	5.1875	41.049891	40594
10	64	4	5.1875	80.560145	44129
10	64	8	5.1875	80.560145	43579
10	64	16	5.1875	80.560145	43579
10	128	2	2.59375	41.049891	40594
10	128	4	2.59375	80.560145	44129
10	128	8	2.59375	80.560145	43579
10	128	16	2.59375	80.560145	43579
15	8	2	27.66666667	62.519855	42616
15	8	4	27.66666667	80.560145	43761
15	8	8	27.66666667	80.560145	43579
15	8	16	27.66666667	80.560145	43579
15	16	2	13.83333333	62.519855	42616
15	16	4	13.83333333	80.560145	43761
15	16	8	13.83333333	80.560145	43579
15	16	16	13.83333333	80.560145	43579
15	32	2	6.916666667	62.519855	42616
15	32	4	6.916666667	80.560145	43761
15	32	8	6.916666667	80.560145	43579
15	32	16	6.916666667	80.560145	43579
15	64	2	3.458333333	62.519855	42616
15	64	4	3.458333333	80.560145	43761
15	64	8	3.458333333	80.560145	43579
15	64	16	3.458333333	80.560145	43579
15	128	2	1.729166667	62.519855	42616

Figure 37: Based on script result stored in result.csv file

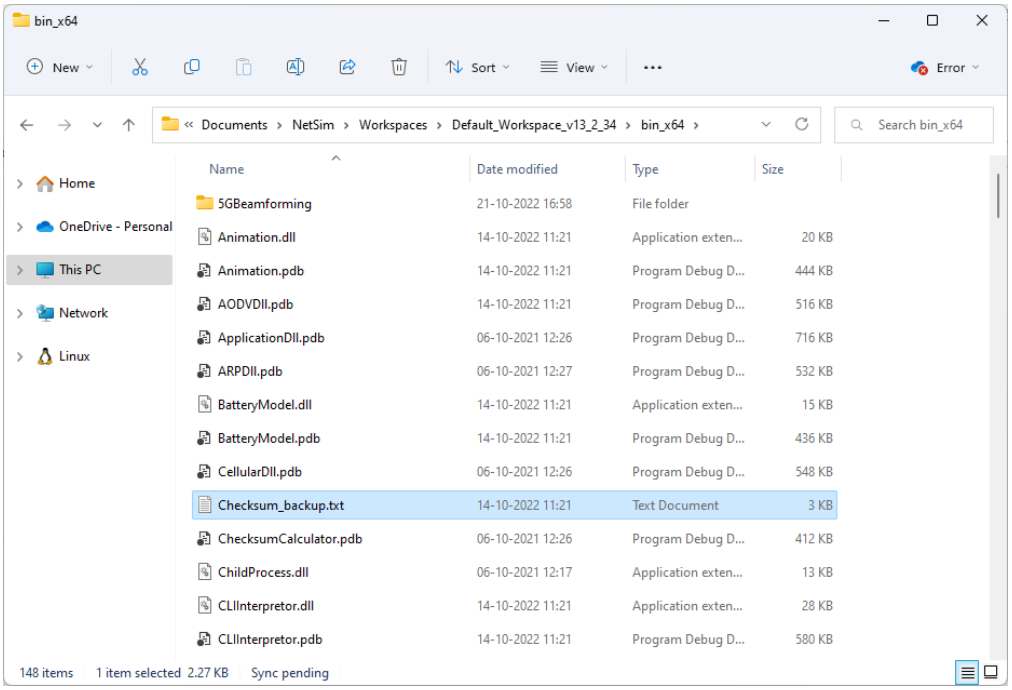
Advanced use cases:

1. Running the sweeper with binaries of a modified workspace.
   When the source codes of a workspace are modified, a warning message will be displayed in the simulation console waiting for user interrupt, as shown below:



To suppress this message and user interrupt, so that the sweeper runs without requiring any intervention, follow the steps given below:

- Open the current workspace location by going to Your Work-> Workspaces-> Open Workspace Location.
- Go the bin_x64 directory and rename the Checksum.txt file to say, Checksum_backup.txt as shown below:



Now upon running any further simulations, warning messages will not be printed in the simulation console and no user intervention will be required.

2. Sweeping Configuration files that has other associated files

There can be cases where a Configuration file requires other supporting files such as:

- Mobility – Mobility text file.
- Static route text files

- ACL input
- SUMO configuration files in case of VANET, etc

In such cases, the associated files can also be placed in the sweeper folder which contains the input.xml file and code can be slightly modified to copy all associated files when copying the Configuration file to the IOPath.