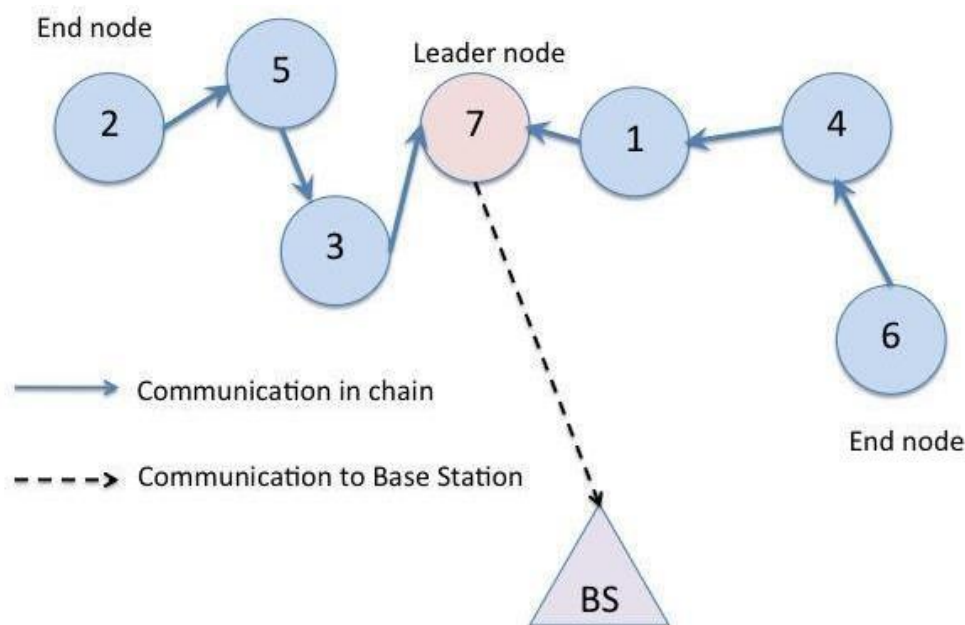# PEGASIS in WSN

Software Recommended: NetSim Standard v12.1 (32/64 bit), Visual Studio 2015/2017/2019, MATLAB (32/64 bit)

In Wireless Sensor Networks energy efficiency plays a crucial role as the sensors are generally battery powered. Hierarchical routing protocols can be used to overcome this constraint.

PEGASIS – Power Efficient Gathering in Sensor Information Systems is one such hierarchical routing protocol which follows a chain based approach and a greedy algorithm. The sensor nodes organize themselves to form a chain. If any node dies in between then the chain is reconstructed to bypass the dead node. A leader or a cluster head node is assigned and it takes care of transmitting data to the base station/ sink node. The main goal of PEGASIS is to receive and transmit data to and from the neighbor and take turns being the cluster head for transmission to the Sink Node.

PEGASIS in NetSim with MATLAB Interfacing:

PEGASIS algorithm is implemented in NetSim by Interfacing with MATLAB for the purpose of mathematical calculation. The sensor coordinates are fed as input to MATLAB and PEGASIS algorithm that is implemented in MATLAB is used to dynamically form a chain between the nodes and to elect one of them as a head node.

From MATLAB the order of devices in the chain and the head node id is retrieved to perform routing in NetSim.

All the above steps are performed during each transmission and can be defined as per the implementation. Each time a node dies, the chain will be reconstructed. Also nodes take turn to become the head node in each iteration.

The codes required for the mathematical calculations done in MATLAB are written to a **PEGASIS.m** file as shown below:

A **PEGASIS.c** file is added to the DSR project which contains the following functions:

fn_NetSim_PEGASIS_CheckDestination()

This function is used to determine whether the current device is the destination of a packet or an intermediate node.

fn_NetSim_PEGASIS_GetNextHop()

This function handles routing in the sensor network by determining the next hop device based on the chain that is formed as part of PEGASIS protocol.

fn_NetSim_PEGASIS_run()

This function makes a call to MATLAB interfacing function and passes the inputs from NetSim to MATLAB and also retrieves the computed parameters from MATLAB workspace for further calculations in NetSim.

fn_NetSim_PEGASIS_form_clusters()

This function updates the information obtained from MATLAB to identify the head node and the neighbouring nodes in the PEGASIS chain.

fn_NetSim_PEGASIS_next_closest_node()

This function is used to identify the neighbouring node and next hop for routing packets.

fn_NetSim_PEGASIS_Init()

This function initializes the parameters specific to PEGASIS algorithm and identifies the number of sensors that are alive throughout the simulation.

Static Routing:

Static Routing is defined in such a way that the sensors send packets to neighbouring node in the PEGASIS chain which is closest to the head node. Once packet arrives at the head node it is forwarded to the destination or the sinknode.
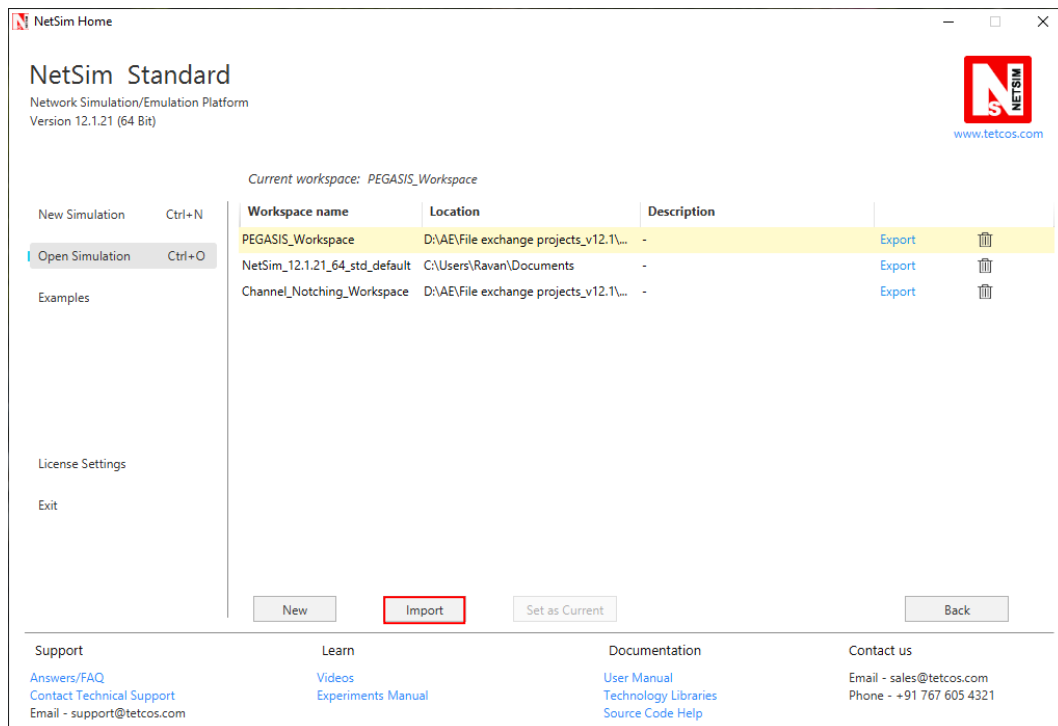
NOTE:

To run this code 64- bit version of MATLAB must be installed in your system.

Steps:

1. The downloaded project folder contains the folders Documentation, MATLAB_Code and PEGASIS_Workspace directory as shown below:
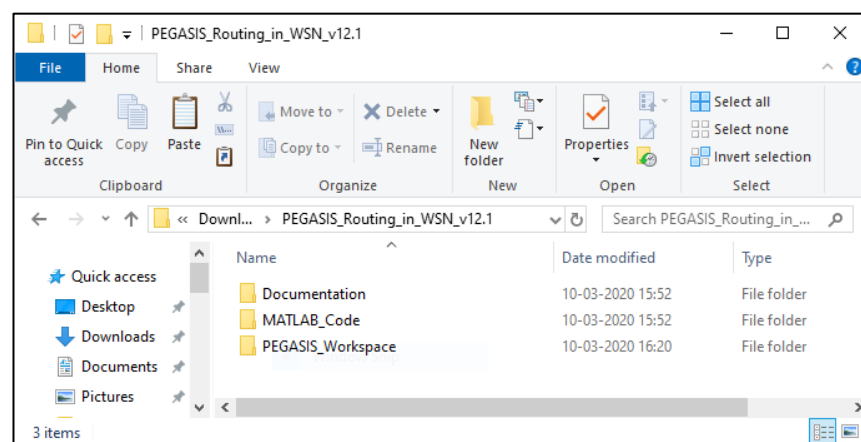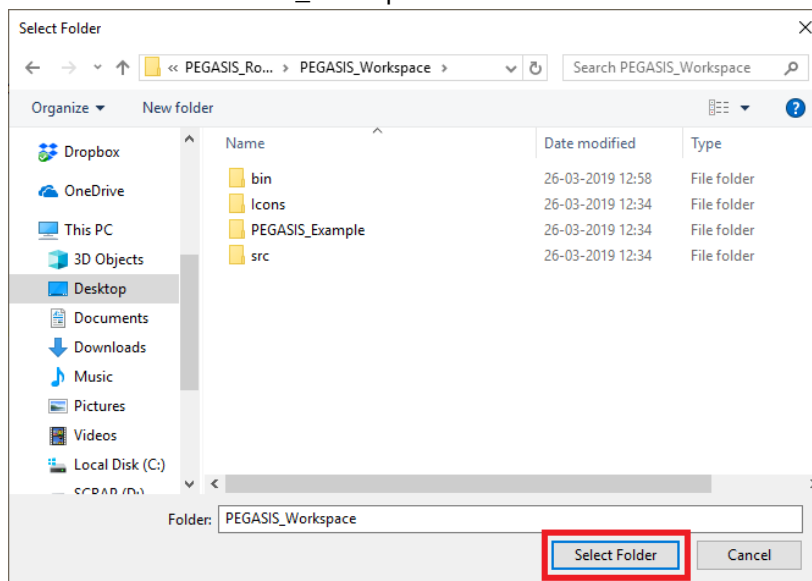

2. Import PEGASIS_Workspace by going to Open Simulation->Workspace Options->More Options in NetSim Home window. Then select Import as shown below:

3. It displays a window where users need to give the path of the workspace folder and click on OK as shown below:
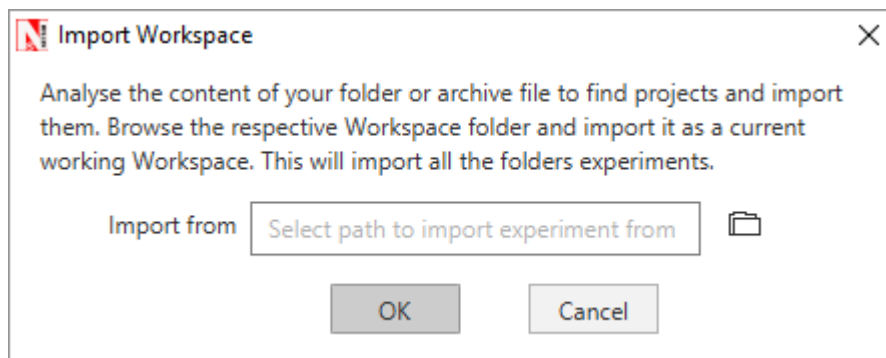

4. Browse to the PEGASIS_Workspace folder and click on select folder as shown below:


5. After this click on OK button in the Import Workspace window.
6. While importing the workspace, if the following warning message indicating Software Version Mismatch is displayed, you can ignore it and proceed.
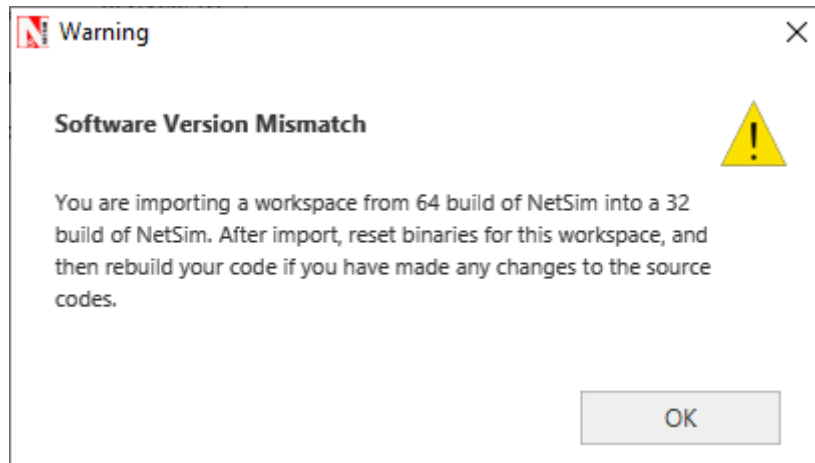
7. The Imported workspace will be set as the current workspace automatically. To see the imported workspace, click on Open Simulation->Workspace Options->More Options as shown below:
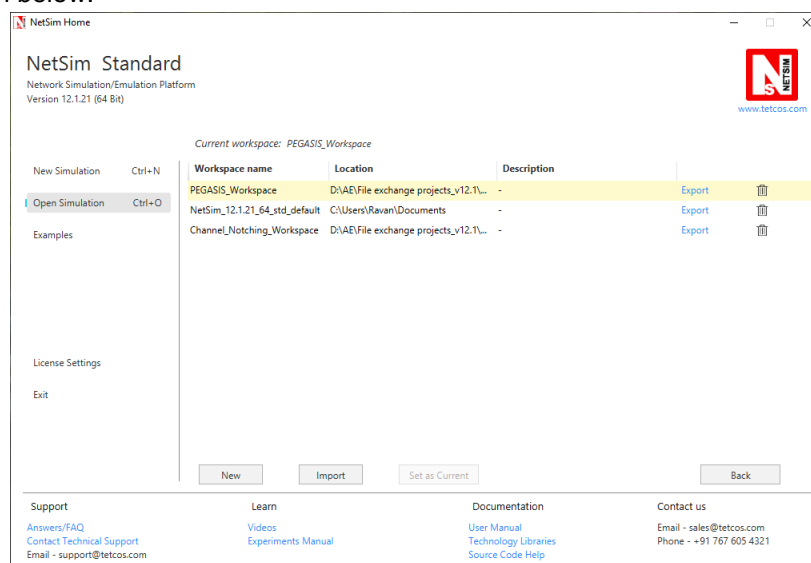
8. Create a user variable with the name of MATLAB_PATH and provide the path of the installation directory of user's respective MATLAB version.
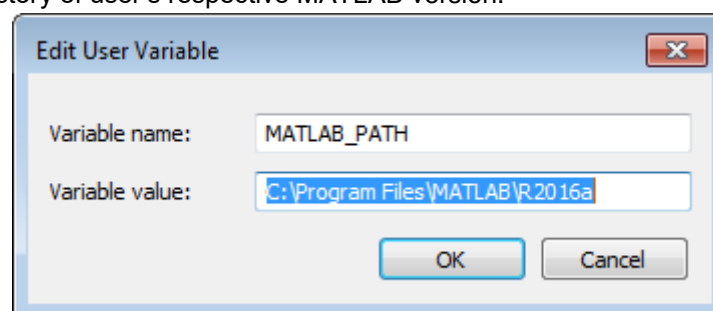

9. Make sure that the following directory is in the PATH(Environment variable)
   **<Path where MATLAB is installed>\bin\win64**
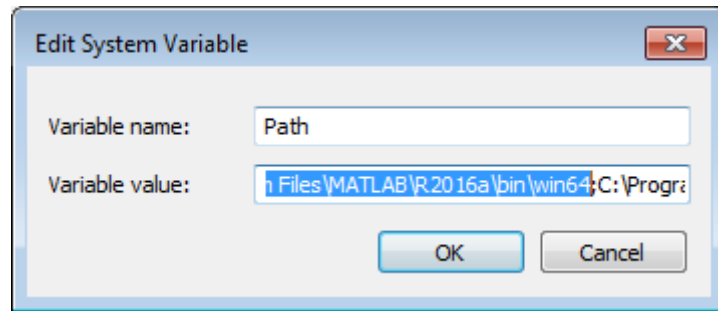
Note: If the machine has more than one MATLAB installed, the directory for the target platform must be ahead of any other MATLAB directory (for instance, when compiling a 64-bit application, the directory in the MATLAB 64-bit installation must be the first one on the PATH).

10. Open Command prompt as admin and execute the command "matlab -regserver". This will register MATLAB as a COM automation server and is required for NetSim to start MATLAB automation server during runtime.
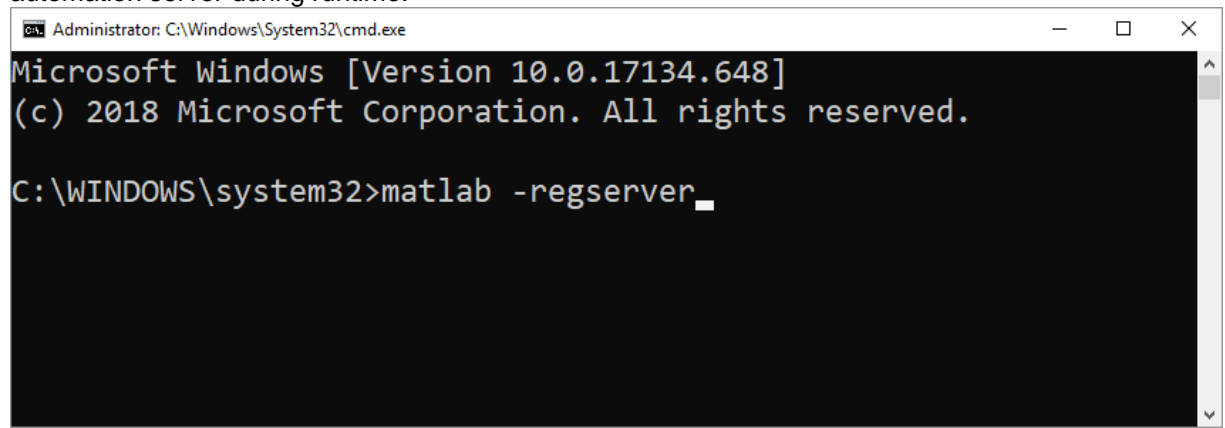

11. Place **PEGASIS.m** present in the MATLAB_Code folder inside the root directory of MATLAB. For Example: **"C:\Program Files\MATLAB\R2016a"**.
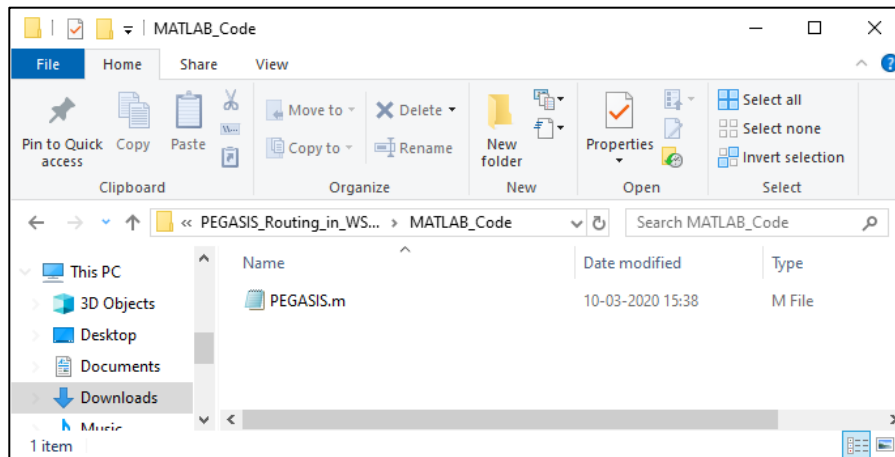

12. Open the Source codes in Visual Studio by going to Open Simulation-> Workspace Options and Clicking on Open code button as shown below:
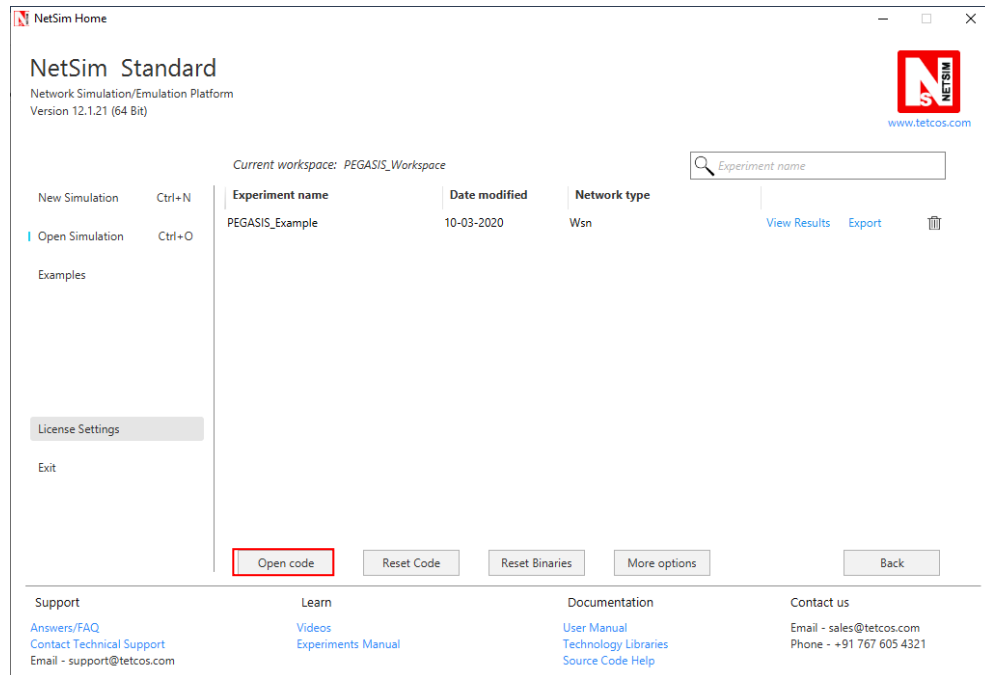
13. Under the DSR project in the solution explorer you will be able to see that **MATLAB_Interface.c** and **PEGASIS.c** files which contain source codes related to interactions with MATLAB and handling clustering in NetSim respectively.

14. Based on whether you are using NetSim 32 bit or 64 bit setup you can configure Visual studio to build 32 bit or 64 bit Dll files respectively as shown below:
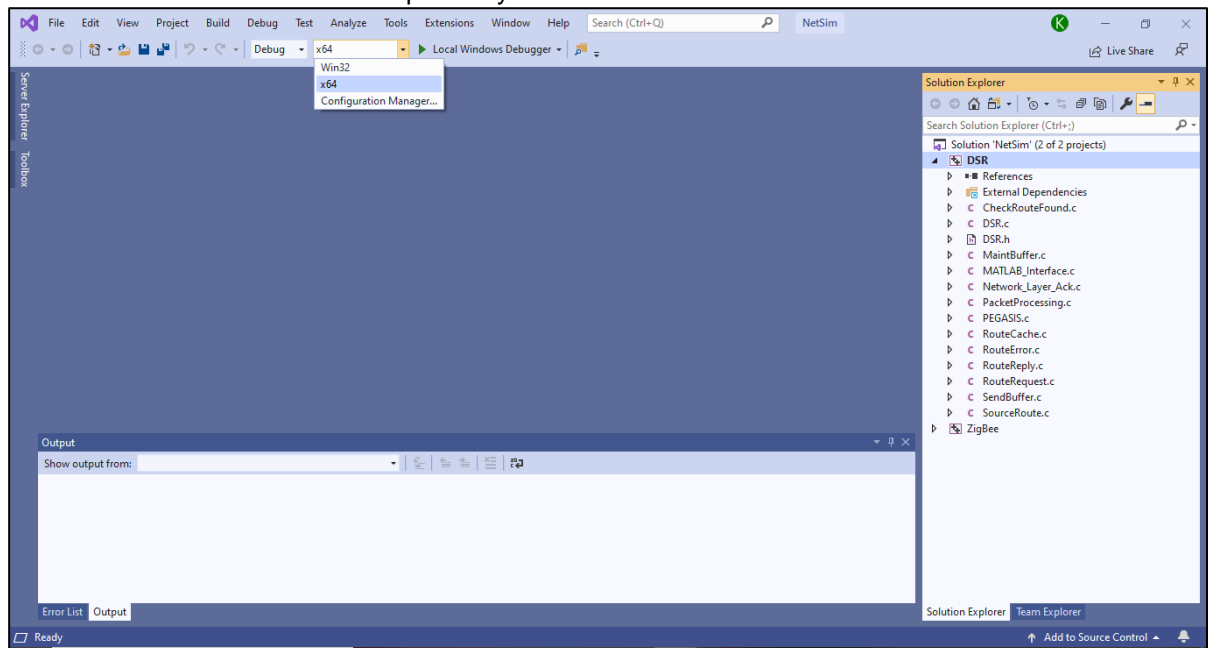
**15.** Right click on the solution explorer and select Rebuild.
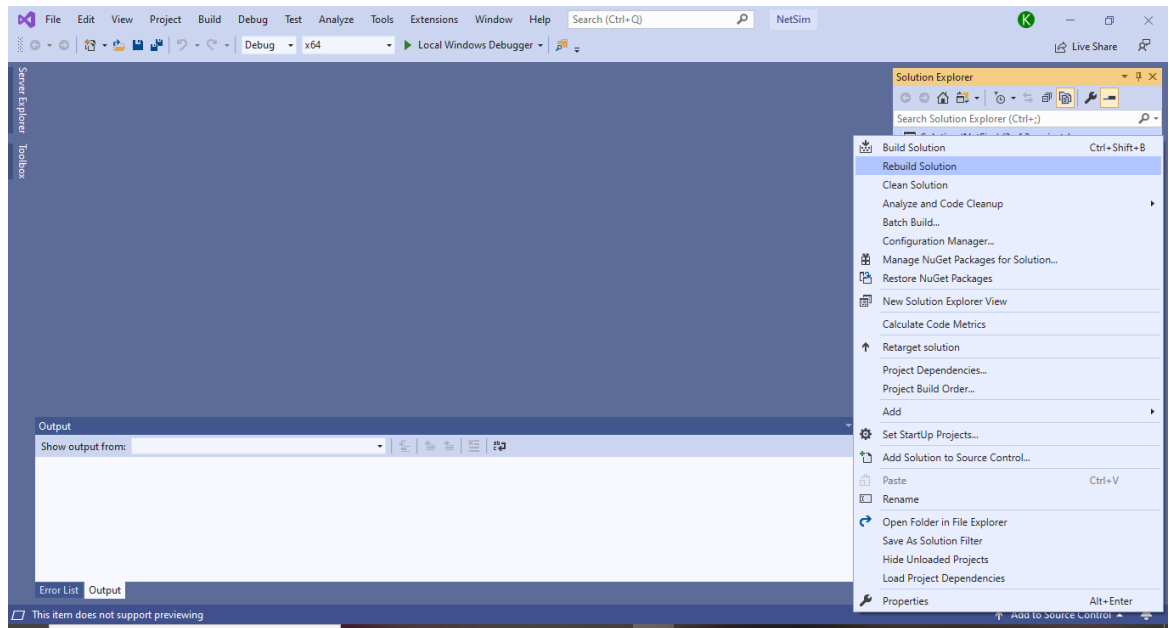
16. Upon successful build modified libDSR.dll and libzigbee.dll files gets automatically updated in the directory containing NetSim binaries.

17. Run NetSim as Administrative mode.

18. Then PEGASIS_Workspace comes with a sample configuration that is already saved. To open this example, go to Open Simulation and click on the PEGASIS_Example that is present under the list of experiments as shown below:
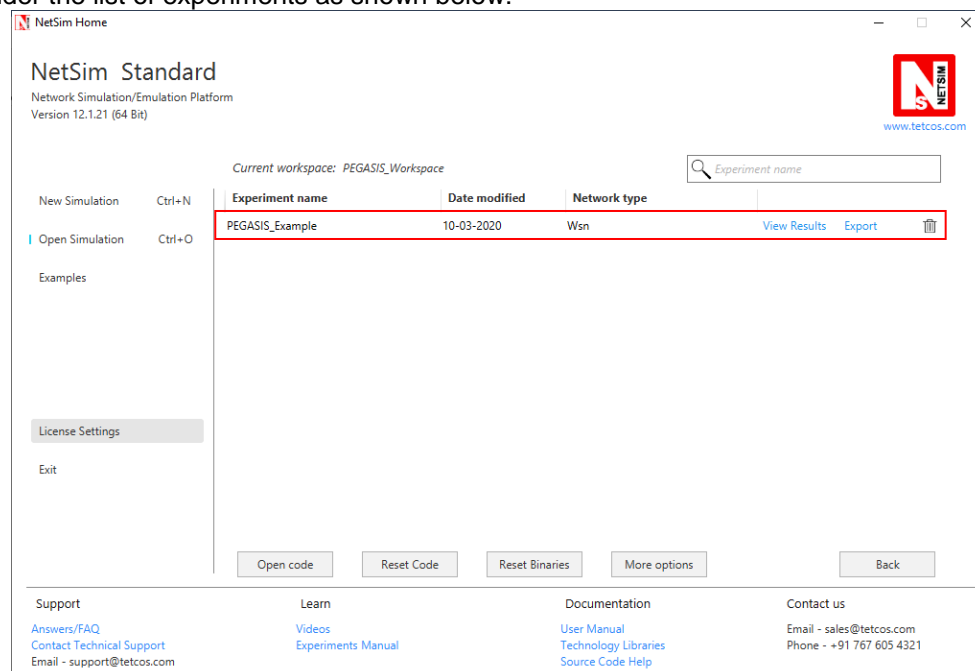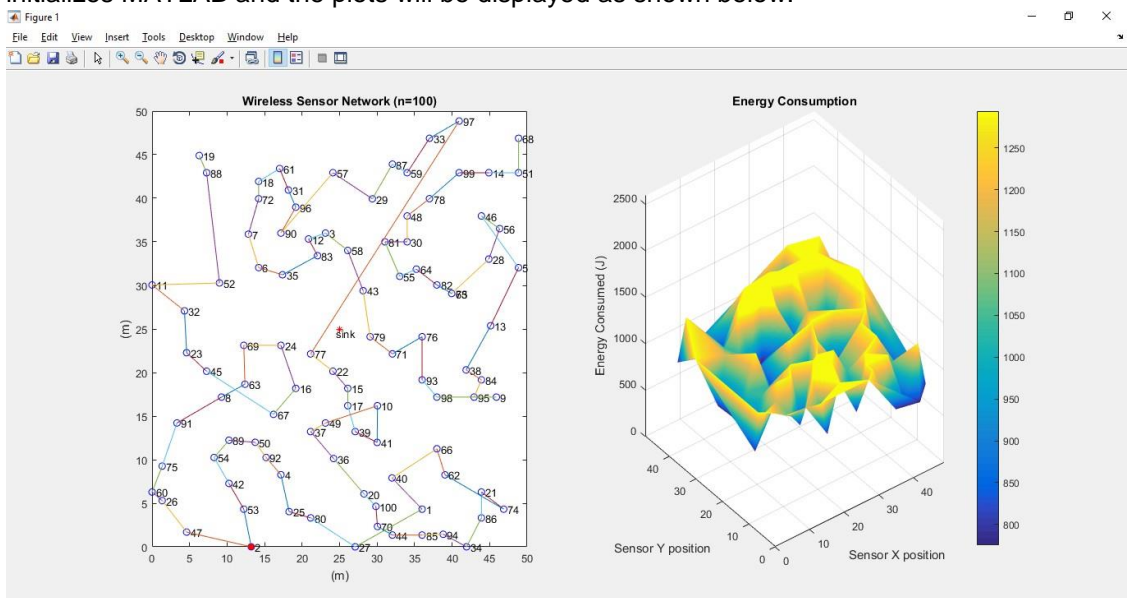

19. The network scenario consists of 100 sensors deployed randomly in a 50x50m grid environment with a sink node placed in the centre.
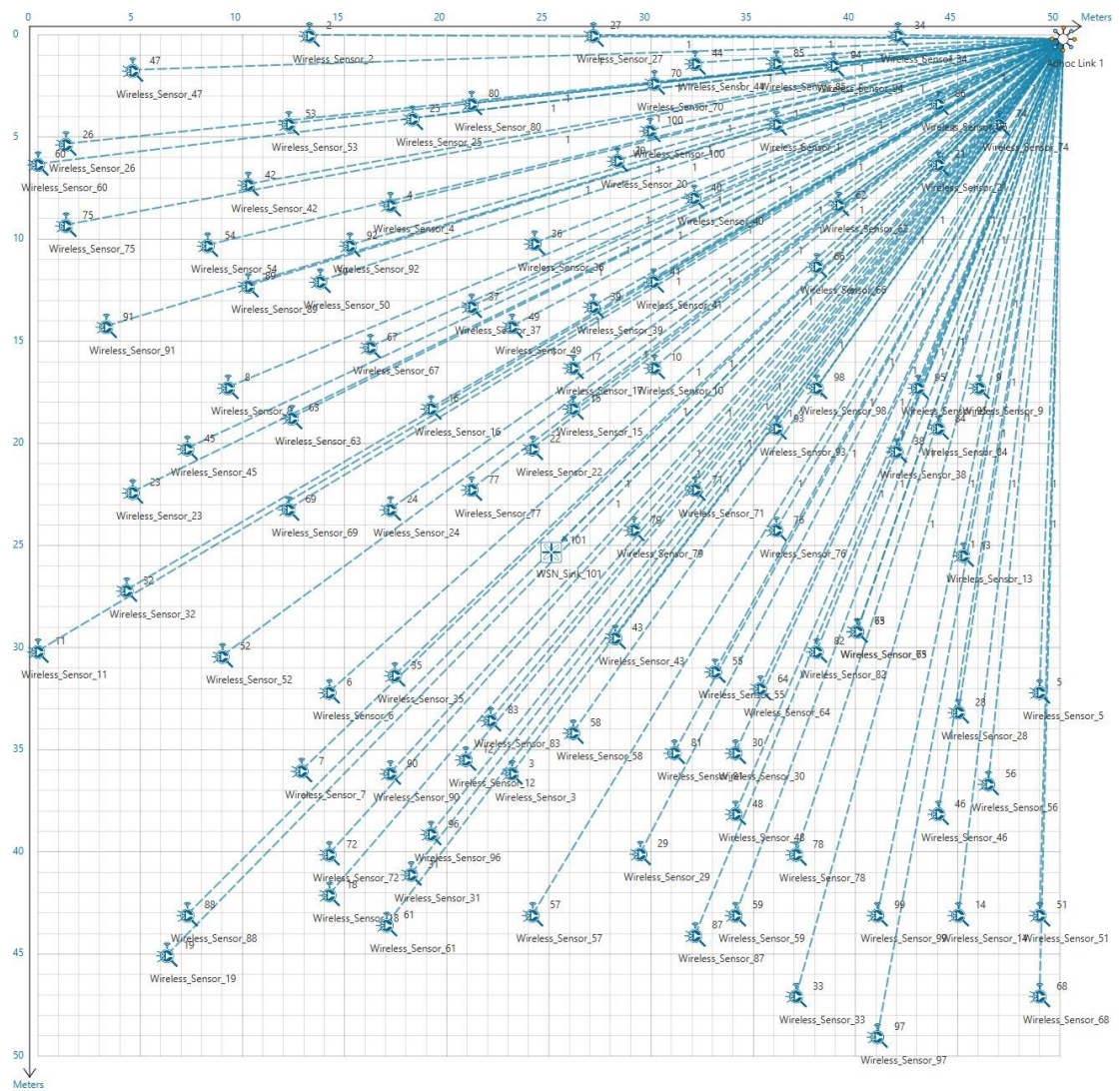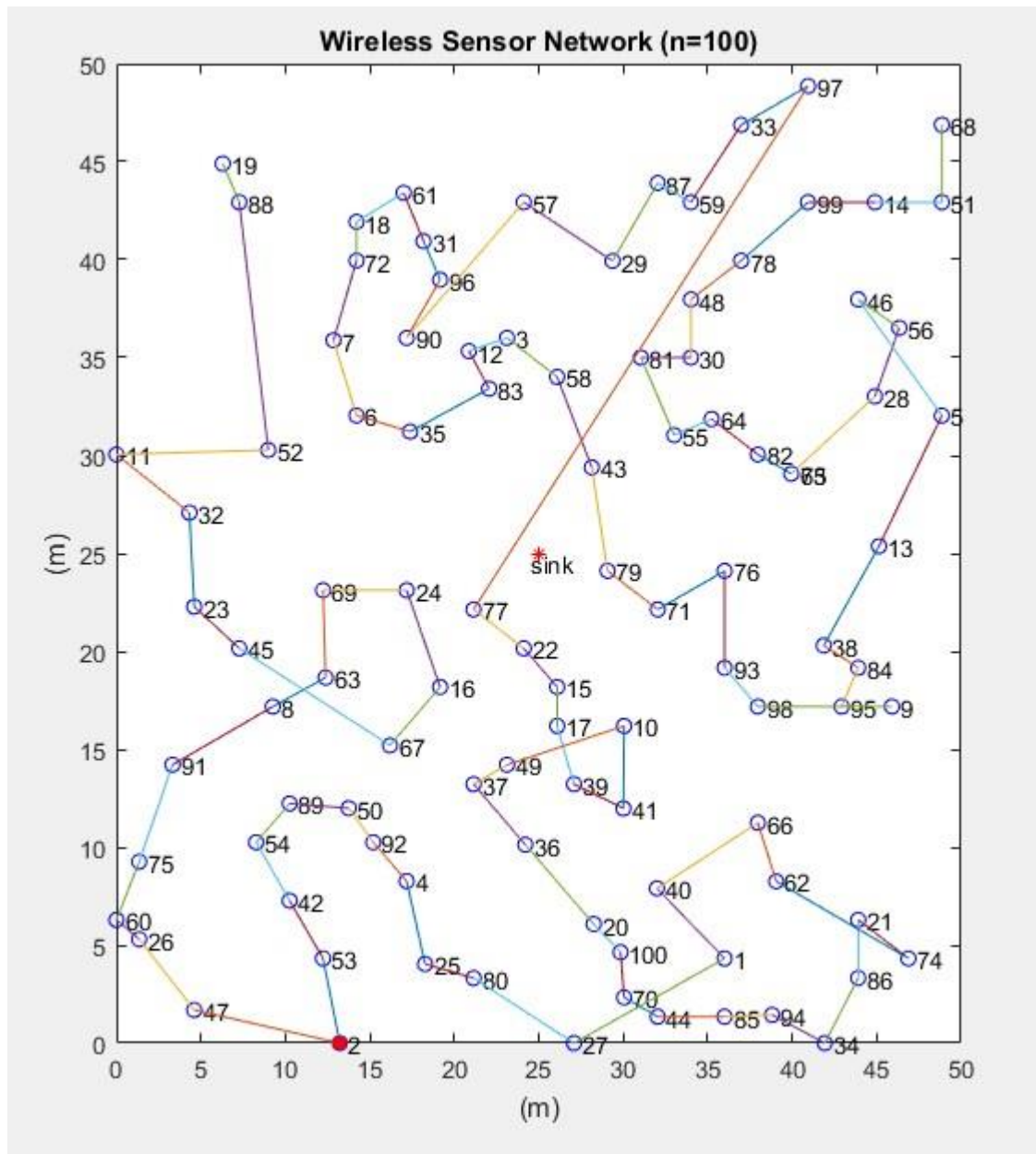
1. Run the Scenario. You will observe that as the simulation starts, NetSim automatically initializes MATLAB and the plots will be displayed as shown below:

Plot 1:

The above plot contains the nodes that are part of the network in NetSim. All sensor nodes are initially alive. N in the plot indicates the total sensor nodes that are currently alive. Sink node is highlighted in the centre and the node marked in red acts as the head node for the current transmission.

All nodes are connected to form a chain which is used to forward packets to the destination or the sink node. This chain may get reconstructed whenever one or more node runs out of energy and dies. The value of 'n' keeps getting updated during the simulation.

Plot 2:
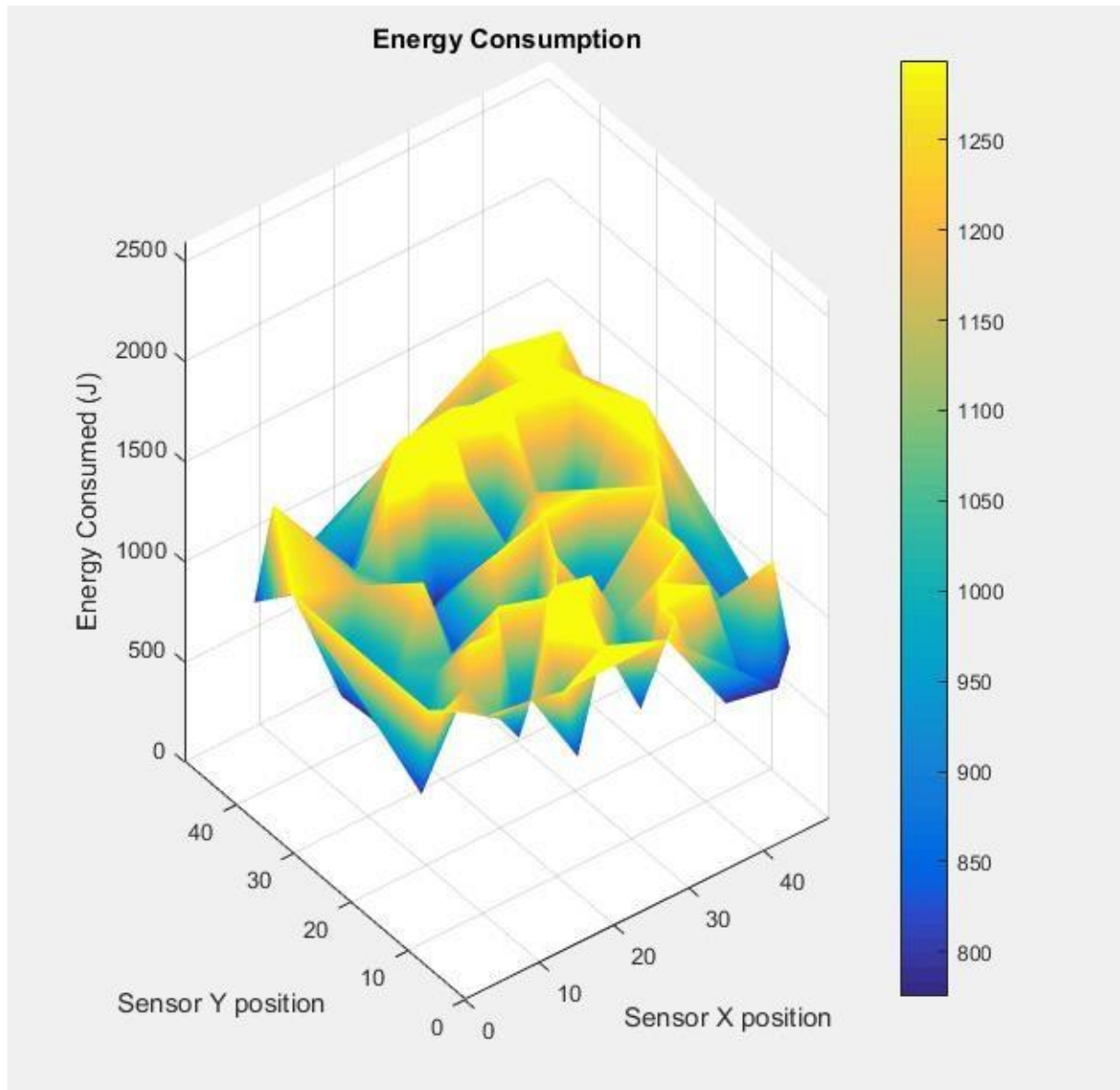
The above plot denotes the energy level in the sensor network. Sensors have different levels of energy remaining in the battery right from the simulation start. The z axis represents the power consumed while the sensors are placed on the x, y plane. As simulation progresses this plot gets updated dynamically based on energy consumed by the sensors in the network.