The following functions written in MATLAB comprise the algorithm for deciphering the spare capacity of a network to mitigate the effect of targeted attack. The sequence of the input and output arguments for the MATLAB functions given below is the sequence in which the variables are to be initialized. Some of the functions given below are nested within other functions that are a part of this document.

**Maincode.m**: The main code of the algorithm.

**Input Arguments** – The network to be disrupted, the number of nodes to be disrupted from the network (as a percentage of the total number of nodes in the network), the centrality measure (1-betweenness centrality; 2-degree centrality), quantum of spare capacity to be added to the network (as a percentage of the total number of edges in the original network), the degree deviation constraint (‘inf’ if no constraint is necessary).

**Output Arguments** – The original network with spare capacity, the disrupted network, the edge mapped spare capacity, the degree of the nodes in the original network in which spare capacity has been added, the normalized cost of the network with added spare capacity, an array that stores in increasing order the fraction of nodes that are successively removed from the graph based on a centrality measure (the x axis of the robustness graph), an array that stores the size of the largest connected component of the network **without** spare capacity post successive node deletions, an array that stores the size of the largest connected component of the network **with** spare capacity post successive node deletions (these two arrays are computed for the generation of the R value for the network without (Ro) and with spare capacity (R) respectively).

**Optimspare.m:** A code to iteratively add spare capacity to the disrupted network and map the nodes and edges to that of the original network (more below).

**Input Arguments** – The network for which the spare capacity has to be deciphered, the number of nodes to be disrupted from the network (as a percentage of the total number of nodes in the network), the centrality measure (1-betweenness centrality; 2-degree centrality), quantum of spare

capacity to be added to the network (as a percentage of the total number of edges in the network), the degree deviation constraint (‘inf’ if no constraint is necessary).

**Output Arguments** – The original network with added spare capacity, the original network disrupted by the centrality measure specified, the edges (spare capacity) between nodes whose node numbers have been mapped to the original network, normalized cost of the network with added spare capacity.

**Spare.m**

A function that successively adds unmapped spare capacity to a given network, given a degree deviation constraint.

**Input Arguments:** The network for which the spare capacity has to be added, the total number of edges that are to be added as spare capacity, the degree deviation constraint (‘inf’ if no constraint is necessary).

**Output Arguments:** The network with added spare capacity, the edge list of spare capacity, the normalized cost of the network with added spare capacity

**Bestedgedeglimit.m**

A function that adds unit spare capacity (one edge) to a given network

**Input Arguments:** The network for which spare capacity is to be added, the degree deviation constraint (‘inf’ if no constraint is necessary).

**Output Arguments:** A vector containing the list of all possible node pairs between the two largest connected components in the vector in an increasing order of the sum of finite path lengths of the network post the addition of unit spare capacity, the network with the added unit spare capacity, the node pair for which the spare capacity is added, the sum of finite path lengths of the network post unit spare capacity addition.

In MATLAB, when a node is disrupted, the remaining nodes downstream of that node are automatically renumbered. For example in a 10 node network, if node 2 is removed, nodes 3-10 are renumbered as nodes 2-9 respectively. Hence, one needs to map the spare capacity that is obtained from the analysis of the disrupted network to be added to the original network.

**Nodemap.m**

A function to map the nodes of the disrupted network to that of the original network

**Input Arguments:** The original (undisrupted) network, the set of nodes that that have been disrupted from the original network (using Dyn\_largestconncomp.m).

**Output Argument:** The node map for the given network and set of nodes.

**Edgemap.m**

A function to map the edges of the disrupted network to that of the original network.

**Input arguments:** The original (undisrupted) network, the set of nodes that that have been disrupted from the original network (using Dyn\_largestconncomp.m), the edge list of spare capacity.

**Output argument:** The edge map for the given network and set of edges.

**Dyn\_largestconncomp.m**

A function that dynamically disrupts nodes in the network based on descending order of a given centrality. The centrality measures are dynamically recalculated post successive node deletions.

**Input Arguments** – The network that is to be disrupted, the number of nodes to be disrupted from the network (as a percentage of the total number of nodes in the network), the centrality measure (1-betweenness centrality; 2-degree centrality)

**Output Arguments** – The node numbers that are successively disrupted from the network, the size of the largest connected component in the network post successive node disruptions, the disrupted network post the removal of the specified number of target nodes.

**Static\_largestconncomp.m**

A function that disrupts a network based on the node numbers specified.

**Input Arguments** – The network to be disrupted, the node numbers of nodes that are to be disrupted.

**Output Arguments** – The disrupted network post the deletion of all the target nodes, the size of the largest connected component post successive node deletions.