Realization and Simulation of a Topology Attack with FDIA Injection on a Power Grid

for the course "Systems and control methods for cyber-physical security" held by prof. Francesco Liberati

Alessandro Lamin 1762253
Guglielmo Saladini 1711622
Romolo D'Amico 1545116

Academic Year 2021-2022

Contents

Acronyms

BDA Binary Dragonfly. 2, 24

BDD bad data detection. 12, 17, 22

BFM branch flow model. 13

BIM bus injection model. 13

EA Evolutionary Algorithms. 23

EMS energy management system. 14, 17

FDIAs false data injection attacks. 2, 6, 10

GA Genetic Algorithm. 2, 23, 24, 35

HVDC High Voltage DC. 9

ISO Indipendent System Operation. 14, 17, 22

LMPs locational marginal prices. 17

MOEA/D Multi-Objective Evolutionary Algorithm Based on Decomposition. 2, 23

NAA Natural Aggregation Algorithm. 2, 21–24

NTP network topology processor. 14, 15

OPF optimal power flow. 13

SCED security-constrained economic dispatch. 17, 21, 22

TEP topology error processor. 17

WLS Weighted Least-Squares. 2, 11

Abstract

The principal aim of the project is to reproduce and to understand deeply the realization of a topology attack in the context of a power grid, using a FDIA data injection to make the attack undetectable. We will start with a brief introduction on a power system and its mathematical model. Then we will show how we performed the attack, solving a minimization problem (that we have reversed in order to maximize and extend as much as possible the attacks) using some new genetic optimization algorithms. These algorithms have shown different results, depending on which scenario or case we want to express. The final part is about the results obtained using different algorithms with different use cases.

1 Introduction

Ensuring the security of a grid is a primary concern of power system operations.

With the increasing integration of multidisciplinary technologies, modern power systems have become complex cyberphysical systems, making them vulnerable to various cyber-attacks.

In practical power systems, the network topology could be changed due to the planned transmission line maintenance and/or forced and unforeseen line outages: therefore, real-time operation of the grid is based on accurate knowledge of the network topology, which is achieved through real-time observations of breakers/switches in the network.

The tight coupling of cyber and physical infrastructures in modern power systems thus provides opportunities for cyber attackers to launch cyber topology attacks in which maliciously introduced fake grid topology information disturbs the normal operations of the grid.

Topology attacks against power systems which have been studied in the literature can be generally classified into two categories:

- *physical topology attack*, which refers to an attack in which one or more bus/line interconnections are physically destroyed by attackers;
- *cyber topology attack*, in which bus/line interconnections are not physically attacked, but the outputs of measurement devices are falsified by attackers when transmitting to the control center.

1.1 False Data Injection Attacks FDIAs

It is a very simple attack but, in this typology, it is already possible to find all the ideas that, more or less, we will find in all the other attacks.

False Data Injection means that the attack is corrupting some meausurements and this leads to very bad changes of the state estimation: the objective of the attack is to disrupt the algorithm that, in power systems, does the state estimation.

All the concern, at least in the power systems, in cyberphysical attacks originated from this one: in fact FDIA against grid state estimation was one of the first cyberphysical attacks, to power systems, to be studied in the scientific literature by Yao Liu, Peng Ning and Michael K. Reiter with two papers published, respectively, on december 2009 and june 2011: from that moment there was an explosion of interest, towards the analysis of other possible cyberphysical vulnerabilities of the power systems and other critical infrastructures, since FDIAs revealed vulnerability in the systems that were used for state estimation.

The goal of this attack is to corrupt state estimation so that then, all the other functions (that will are coming) will work with a wrong state causing, so, bad performances.

1.2 IEEE-N-bus Power System and its Power Flow Equations

We are working at the transmission level and the goal, for the operator, is to estimate the state of a transmission network: so it means very high voltage power system. In the following figure it is possible to see an example of a diagram for a transmission network.

The IEEE 24-bus reliability test system was developed by the IEEE reliability subcommittee and published in 1979 as a benchmark for testing various reliability analysis methods. The three reliability test systems are IEEE one-area, IEEE two-area, and IEEE three-area.

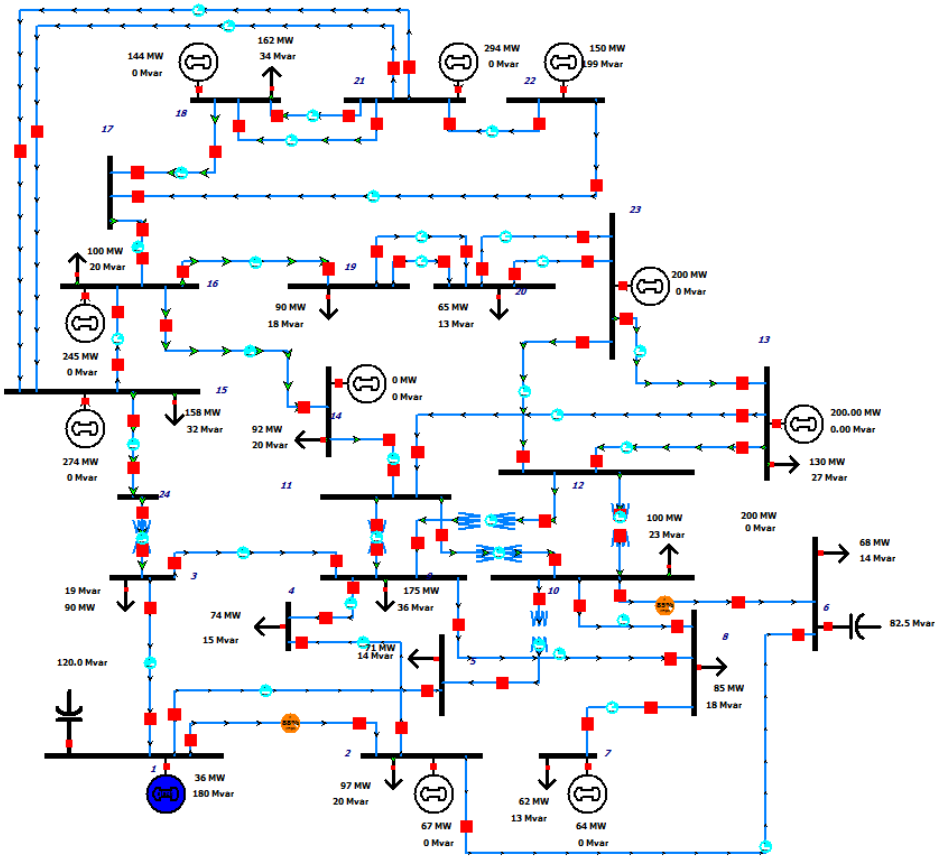


Figure 1: IEEE-24-bus Power System Diagram

The equations that govern the power flow in the networks are the so-called *power flow equations*, which are one of the main mathematical models used in power systems. They determine active and reactive power flows on the network lines:

$$P_{ij} = V_i^2 g_{ij} - V_i V_j [g_{ij} cos(\theta_{ij}) + b_{ij} sin(\theta_{ij})] \tag{1a}$$

$$Q_{ij} = -V_i^2 b_{ij} - V_i V_j [g_{ij} sin(\theta_{ij}) - b_{ij} cos(\theta_{ij})] \tag{1b}$$

Both of them have to be referred from bus i to bus j.
From these equations it is possible to go to the nodal power flow equations: imagine to have a given bus where we are consuming energy power and where we have two lines connected to this bus. The sum of the power, in that point, must be equal to zero: so it is possible to find the injected power, or the consumed power, by summing the line power flow for these two lines. That's why we have a sum in the following (nonlinear) equations:

$$P_i = V_i \sum_{j \in N_i} V_j[-g_{ij}cos(\theta_{ij}) - b_{ij}sin(\theta_{ij})] + V_i^2 \sum_{j \in N_i} g_{ij} \tag{2a}$$

$$Q_i = V_i \sum_{j \in N_i} V_j[-g_{ij}sin(\theta_{ij}) + b_{ij}cos(\theta_{ij})] - V_i^2 \sum_{j \in N_i} b_{ij} \tag{2b}$$

where:

- P_i and Q_i are, respectively, the active and reactive power injections at bus i;
- V_i is the voltage magnitude at bus i;
- g_{ij} and b_{ij} are network parameters (respectively, the susceptance and conductance of line (i, j));
- θ_i is the voltage angle at bus i;
- $\theta_{ij} = \theta_i - \theta_j$;
- N_i is the set of buses connected to bus i through a line.

1.3 Power grid

An electrical power grid is an interconnected network that delivers the generated power to the consumers: it is, sometimes, also called as an electrical power system.
These interconnected networks include two main components: buses that represent important locations of the grid (e.g. generation points, load points, substations) and transmission (or distribution) lines that connect these buses.

It is pretty straightforward, therefore, to look at power grid networks as graphs: buses and transmission lines can be represented by nodes and edges of a corresponding graph.

There are two equivalent graph models that can be used to derive the basic power flow equations:

- directed graph representation:
- undirected graph representation.

A power grid consists of:

- generating stations (power plants);
- transmission system;
- distribution system.

Power generating stations are located at feasible places - according to the availability of the fuel, the dam site or an efficient location for renewable sources. Hence, they are often located quite away from the populated areas: this is very practical since the transmission of electrical power over longer distances is a lot much economical than the relative transmission of any other fuel. Also, a hydroelectric plant must be located according to an appropriate dam site or a wind power plant may be located off-shore to harvest additional energy from the wind: thus, a long distance transmission system is needed to transmit the generated electricity to the populated areas. Moreover, a distribution system is needed to distribute the power to every consumer at appropriate voltages.

1.4 Behaviour of the power grid

In the following image it is possible to see a typical layout of an electrical power grid.

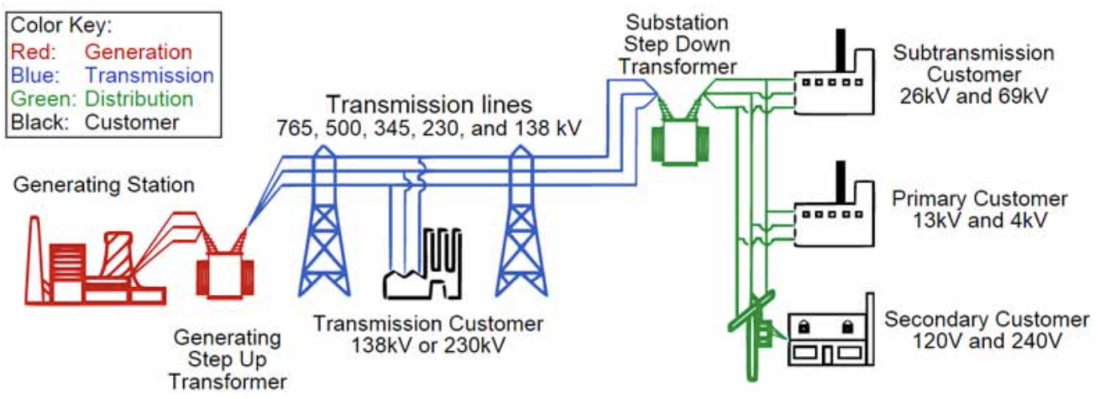


Figure 2: Typical layout of an electrical power grid

- **Power Generation:** electricity is generated in power plants which are often located far away from the populated areas. There are various types of power generating stations such as thermal, nuclear, hydro, solar, wind etc. A power plant may consist two or more 3-phase alternators which are operated in parallel and electricity is generated in power plants at voltages ranging from 11kV up to 25 kV. Generation voltage can not be much higher due to technical limitations;

- **Transmission**: for the transmission of power over longer distances, the generated voltages are stepped up to a much higher level: a step up transformer is used for this purpose, which increases the voltage level with the corresponding decrease in the current. Stepping up the voltage is necessary to increase the transmission efficiency by reducing losses in the transmission lines. Transmission voltages are generally 220kV or greater up to 765kV and transmissions lines are often seen running over tall towers at the outskirts of a city. Most commonly 3-phase AC power at very high voltage is used for the power transmission but, due to the advancements in power electronics, HVDC (High Voltage DC) has proved many advantages for longer distance transmission: so, HVDC transmission systems are being employed for very longer distance power transmission. AC power is converted into HVDC at a converter station for the transmission, and then it is converted back into AC at the other end. Also, HVDC link is the only option today for interconnecting grids with different frequencies;

- **Distribution**: power from the transmission system is then stepped down to a considerably lower voltage (say 33 to 66kV) using a step-down transformer in a primary step-down substation. The power is then carried to distribution substations or directly to very large industrial consumers: at distribution substations, the power is further stepped down (say at 11kV). Power distribution is carried out using overhead or underground distribution lines which are usually interconnected in a ring or mesh network types. Distribution transformers are used to lower the voltage further to the utilization voltage (120 volts or 230 volts) and supply several consumers using the secondary distribution lines.

1.5 Grid state estimation

Grid state estimation is one of the essential tasks to monitor and control the smart power grid: in particular it is the problem of estimating the state of the electric network/grid based on measurements coming from sensors spread across the grid. State estimation is one of the first things that we do in the control center: once the measurements reach it, first of all, we do the state estimation and then, using the estimated state, it is possible to do a lot of operations. So it is important to know that *a number of critical grid functions and operations are based on state estimation*.

In traditional AC power system, the state is expressed by voltage magnitude and voltage angles at every bus of the grid and, with this data, is possible to compute the power flow equations eq. (1a) and eq. (1b). The measurements, relative to branches active and reactive flows and power injections at each node, are used to derive the state of the grid. In our approach the measurements taken from the grid are the following: active/reactive power flows and active/reactive power generated by each generator.

The process of state estimation was performed using the tool MATPOWER, that will be presented later. It is important to notice that MATPOWER uses the AC state estimation, based on an iterative process.

The other type of state estimation is the DC one, this type is based on the linearization of the powerflow equations. As will reported later, the DC state estimation was useful, for our work, to get the attack vector a, necessary for the topology attack.

1.5.1 DC state estimation problem

Let's see now the technique (the simplest one) used before FDIAs were discovered. This state estimation algorithm is called *DC state estimation* because it works using the DC power flow equations, which are the linearization of eq. (1a) and eq. (1b): if we linearize them around an operating point we get the linearized version of DC equations. So, in the following, we are going to see the DC state estimation problem because we work on DC power flow model.

Let's define this kind of problem starting with a few of nomenclature:

- let $x = [x_1, x_2, ..., x_n]^T \in \mathbb{R}^n$ denote the vector of (n) state variables (to be estimated);
- let $z = [z_1, z_2, ..., z_m] \in \mathbb{R}^m$ denote the vector of (m) measurements;
- it must always be $m \geq n$;

The **goal of state estimation** in power systems is to estimate the state variables from the measurements; the **estimate** is called $\hat{x}$.

Let's now introduce a very importance concept, the so called **meausurement model**, which is a mathematical relation that links the state variables with the measurement variables. In our case the model is non-linear since it is related to the AC-model of the Power Grid, the relative equation is reported below:

$$z = h(\gamma, x) + e \tag{3}$$

Where γ is the current topology of the Power Grid. This means that the function h depends directly on the topology of the grid.

Since it is very complex to deal with the non-linear AC model of the grid, could be very useful a linearization the $h(\gamma, x)$, around an equilibrium point x_0, deriving the Jacobian matrix H :

$$H = \frac{\partial h(\gamma, x)}{\partial x} \tag{4}$$

Thus, the linear measurement function with the relative measurement noise is:

$$z = Hx + e \tag{5}$$

where:

- H is a (Jacobian) matrix built based on network topology and it comes from the linearization of the power flow equations. A typical case is when H is a tall matrix, so it has a number of rows m much higher than the number of state variables we need to estimate;
- e is the vector of **measurement error/noise**. It is assumed to follow a **Gaussian distribution with zero mean** and diagonal covariance matrix R;

If we want to perform estimation we need to find an equation that links the variables we want to estimate with the variables we measure: this equation is precisely, in our case, the power flow equations. So, basically, all we have to do somehow is, given z, to find a good estimate $\hat{x}$ for x.

1.5.2 Weighted Least-Squares (WLS) estimation criterion

A very popular and simple technique is the **weighted least-squares** estimation criterion.

So how do we find the estimate $\hat{x}$? We have to solve this problem, finding a value x that minimizes the following expression:

$$\hat{x} = arg \min_x \{(z - Hx)^T W(z - Hx) := \|z - Hx\|_w\} \tag{6}$$

with $W = R^{-1}$ (the diagonal elements of W are the inverse of the covariance matrix characterizing each sensor). We could write the objective function in scalar form too:

$$\sum_{i=1}^{m} (z_i - H_i x)^2 W_i \tag{7}$$

In this case we look for the value x that minimizes the above sum (the term into the parenthesis represents the *estimation error*).

It is very easy to find the solution to this problem by taking the objective function and developing the product:

$$J = (z - Hx)^T W(z - Hx) = z^T Wz - 2x^T H^T Wz + x^T H^T WHx \tag{8}$$

At this point $\hat{x}$ is easily found by means of:

$$\frac{\delta J}{\delta x} = 0 \quad \Rightarrow \quad -2H^T Wz + 2H^T WHx = 0 \tag{9}$$

and we obtain the closed form expression for the estimated measurement:

$$\hat{x} = (H^T W H)^{-1} H^T W z := Ez \tag{10}$$

with $E = (H^T W H)^{-1} H^T W$, i.e. the weighted pseudo-inverse of H.

Before using this vector we do a check, called *bad data detection (BDD)*, to make sure everything is ok with this estimate: in fact a typical problem could be the malfunctioning of one or more sensors sending, for example, completely wrong values for the measurements.

So we could have bad data in the measurement vector z (due to measurement noise, disturbances, sensor failures etc.) that will make $\hat{x}$ deviate from the real state x but it is important to underline that, this situation, is not related to the attacks: it is just the presence of bad data.

There are typically modules, called *bad data detector*, that are in charge of detecting the presence of issues in vector z: a typical technique is the **residual-based detector** in which we have to define and check the residual.

We define a **measurement residual** as the difference between the measurement vector z and the estimated measurement $\hat{z}$:

$$r = z - \hat{z} = (I - K)z \tag{11}$$

We check the norm of the residual $\|r\|$ having a threshold based approach.
When the norm is above a given threshold, $\|r\| \geq \tau$, it means we have a problem with some bad data in the measurements; instead, when the norm is below a given threshold, $\|r\| < \tau$, it means the measurements are reasonable ones and they are not affected too much by errors or bad data.

2 MatPower

MATPOWER is a package of Matlab for solving power flow and optimal power flow problems: it is intended as a simulation tool for researchers and educators that is easy to use and modify. The design of MATPOWER is provided to give the best performance possible while keeping the code simple to understand and modify.

2.1 Optimal power flow

Power flow problem serves as the key component of a much more challenging task: the optimal power flow (OPF).

OPF is an umbrella term that covers a wide range of constrained optimization problems, the most important ingredients of which are: variables that optimize an objective function, some equality constraints, including the power balance and power flow equations, and inequality constraints, including bounds on the variables.

The sets of variables and constraints, as well as the form of the objective, will vary depending on the type of OPF.

2.2 The power flow problem

There are two basic power flow models: the *bus injection model (BIM)* and the *branch flow model (BFM)* that are based on the undirected and directed graph representation of the power grid, respectively.

The power flow problem is to find a unique solution of a power system (given certain input variables) and this kind of problem, formulated as either a BIM or a BFM, define a non-linear system of equations.

There are multiple approaches to solve power flow systems but the most widely used technique is the Newton–Raphson method: the solution is sought in an iterative fashion, until a certain threshold of convergence is satisfied.

3 Topology Attack

An attack is considered successful if the cyber attacker can manipulate the database in which all measurements are consolidated in control centre: moreover, manipulating a database within the control centre is substantially easier for cyber attackers than taking over the control centre itself.

It is therefore reasonable to make the following assumptions:

- the cyber attacker has knowledge of all grid information, including the state estimation scheme, bad data detection method, topology matrix, and line parameters of the system;
- the cyber attacker is capable of falsifying any meter measurement, i.e. digital information of breakers/switches that are transmitted to the network topology processor and analog information that are transmitted to the state estimator.

3.1 Cyber-topology attack classification and detection evasion strategy

By definition, the cyber-topology attack is significantly different from the physical-topology attack since the former manipulates the network topology information at the cyber layer, while the latter actually manipulates the physical network topology.

In modern power systems, the *network topology processor (NTP)*, which is a component of *energy management system (EMS)*, is responsible for constructing the system topology from the telemetered breakers/switches' status.

By launching a successful cyber-topology attack, the network topology conceived by the *Indipendent System Operation (ISO)* would thus be inconsistent with the actual network topology.

The topology information of a network can be expressed in this way:

$$z_{digital}(i) = \begin{cases} 1 & \text{if the branch } i \text{ is active} \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

To illustrate this concept, consider a system in which the actual physical topology is depicted as shown in Fig. 3:

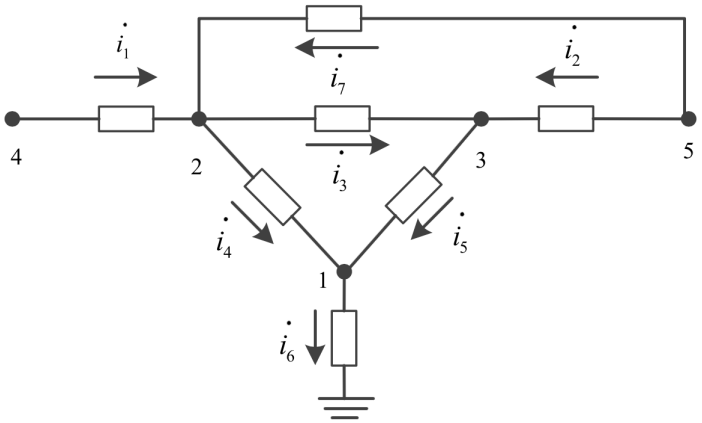


Figure 3: 5-node system topology network (under no attack)

We have to recall the mathematical concept of the Incidence Matrix which is a logical matrix that shows the relationship between two classes of objects. In the case of a Power Grid we have that the rows represents the number of the nodes and that the columns represents the number of the branches of the graph shown above. The value of each element of the Incidence Matrix could be 1,0 or −1, following the rule explained below:

$$x_{ij} = \begin{cases} 1 & \text{Outgoing branch from } j^{th} \text{ node} \\ -1 & \text{Outgoing branch from } j^{th} \text{ node} \\ 0 & \text{Otherwise} \end{cases} \tag{13}$$

In the absence of a cyber-topology attack, the NTP constructs the topology information as the following incidence matrix, based on the measured statuses of breakers/switches:

$$A = \begin{pmatrix} 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & -1 \\ 0 & -1 & -1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{14}$$

In the presence of cyber-topology attack, the incidence matrix could be altered.
For example, suppose the attacker has successfully falsified the status of the breaker on the 7th transmission line in Fig. 3 from CLOSED to OPEN: note in this case that the physical status of the breaker on the 7th transmission line is not manipulated, and only the cyber information transmitted to the control centre is falsified.

Then, as a consequence, the incidence matrix under attack is changed and shown by matrix (15):

$$\bar{A} = \begin{Bmatrix} 0 & 0 & 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & -1 & -1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{Bmatrix} \tag{15}$$

In this sense, the ISO is deceived into believing the topology of current network has become the one shown in Fig. 4:

3.2 Cyber-topology attack type classification

In an n-node network the set of all physically deployed transmission lines (denoted Ω) can, in principle, be utilized to transmit electricity: in practice, however, for a variety of reasons (e.g., line faults, line maintenance, etc.), not all physical transmission lines are necessarily employed.

We denote by Ω_C the set of physically deployed transmission lines which are currently being used and by Ω_A the set of lines not being used; clearly $\Omega = \Omega_C + \Omega_A$.

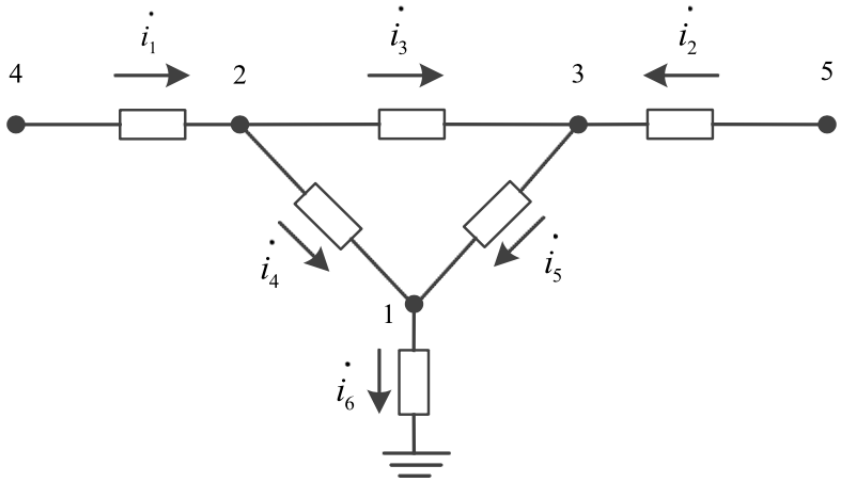


Figure 4: The control center believed 5-node system topology network (under attack)

Some transmission lines are kept connected all the time forming the backbone of the network: these lines are considered as pivotal lines of the whole network.

The connection statuses of these transmission lines are often robustly protected and difficult to falsify and it could thus be reasonable to assume that these backbone lines guarantee the observability of the power system: denoting the set of backbone transmission lines as Ω_S and remaining set of transmission lines in Ω_C as Ω_R, then there is $\Omega_C = \Omega_S + \Omega_R$.

In summary, the classification of transmission lines can be illustrated in Fig. 5.

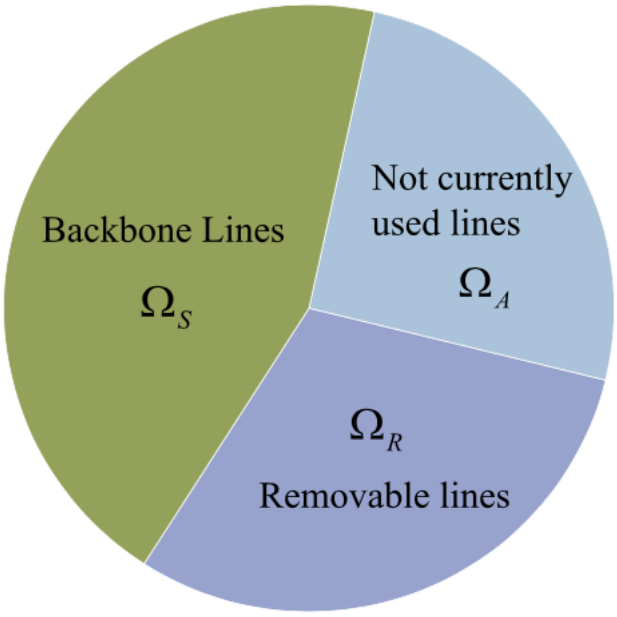


Figure 5: Transmission line sets of a network ($\Omega_C = \Omega_S \cup \Omega_R, \Omega = \Omega_C \cup \Omega_A$)

A cyber-topology attack aims to falsify the line connection status at the cyber layer, wherein the attacker has three basic forms of attack options:

1. *line-addiction attack*: some of the line connection statuses are falsified from OPEN to CLOSED in the information layer;
2. *line-removal attack*: some of the line connection statuses are falsified from CLOSED to OPEN in the information layer;
3. *line-switching attack*: some of the line connection statuses are falsified from OPEN to CLOSED, and some of the line connection statuses are simultaneously falsified from CLOSED to OPEN in the information layer.

The attacker, through his actions, tries to deceive the control center, due to the fact that the currently used transmission line set (denoted $\bar{\Omega}_C$) differing from the real actual set Ω_C.

3.3 Detection evasion strategy

In order to successfully launch the aforementioned cyber-topology attack, the attacker should adopt certain strategies to evade detection by the EMS.

In a practical network, if any inconsistency is detected (e.g., a nonzero power flow appears on a disconnected line), the control center is informed of the error and the network topology is re-estimated. Only in the absence of inconsistency the network topology is approved to be used by other modules of the EMS; therefore, if the digital data are manipulated, the corresponding analog data also needs to be manipulated in order to deceive the *topology error processor (TEP)*.

Furthermore, all analog data should be detected by *bad data detection* so as to filter false data: in this sense, when launching a cyber-topology attack, a cyber attacker needs to evade both BDD and TEP by calculating appropriate malicious injection data.

3.4 Cyber-topology attack implementation

With the cyber-topology attack, the system topology conceived for the ISO is different from the actual one. As shown in Fig. 6, arrow ① shows that the currently used transmission line set is falsified in the cyber side; arrow ② represents that the decision-making of the ISO is based on the falsified topology; arrow ③ shows the control decisions made by the ISO are applied on the actual system.

These actions could significantly cause deviations of the system.

3.4.1 SCED model

One of the most important decision-making roles of the ISO is to perform *security-constrained economic dispatch (SCED)*, which optimally dispatches generators and determines *locational marginal prices (LMPs)* of the wholesale energy market. The overall procedure of SCED is shown in Fig. 7.

The SCED model is formulated as follows:

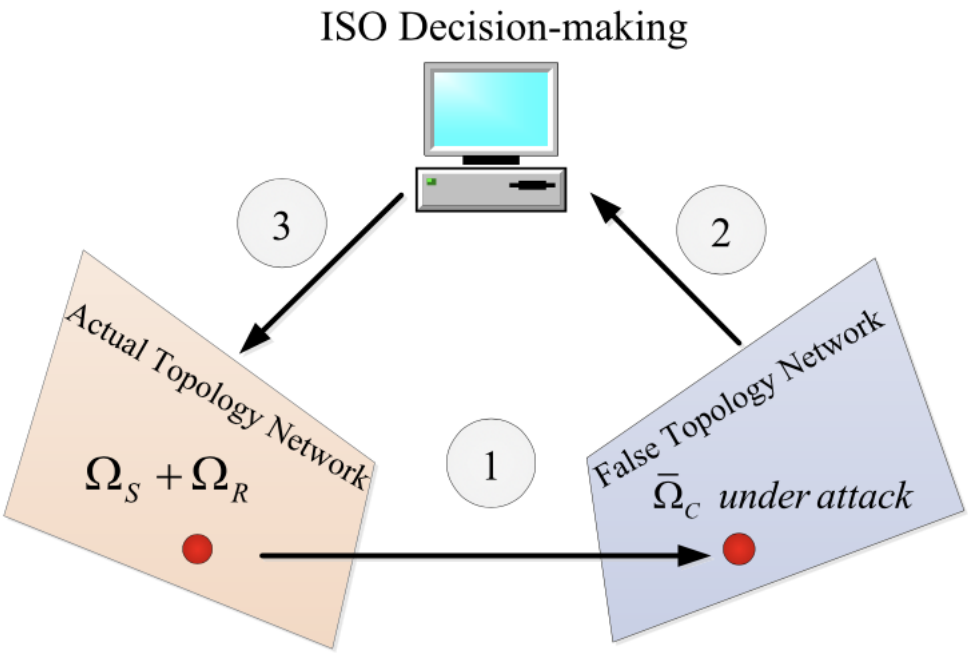


Figure 6: Cyber-topology attack flow chart

$$min \sum_{i \in K_g} c_i(Pg_i) \ \text{ s.t.} \sum_{i \in K_g} (Pg_i) = \sum_{j \in K_d} D_j \tag{16a}$$

$$P_l = -v_i v_j (g_{ij} cos\theta_{ij} + b_{ij} sin\theta_{ij}) + g_{ij} v_i^2, (i,j) \rightarrow l \tag{16b}$$

$$Q_l = v_i v_j (b_{ij} cos\theta_{ij} - g_{ij} sin\theta_{ij}) + b_{ij} v_i^2, (i,j) \rightarrow l \tag{16c}$$

$$max\left\{P_{gi}^{min}\right\} \leq P_{gi} \leq min\left\{P_{gi}^{max}\right\}, \ \ i \in K_g \tag{16d}$$

$$v_i^{min} \leq v_i \leq v_i^{max}, \ \ i \in K \tag{16e}$$

$$P_l^{min} \leq P_l \leq P_l max, \ \ l \in \Omega_C \tag{16f}$$

$$Q_l^{min} \leq_l \leq Q_l^{max}, \ \ l \in \Omega_C \tag{16g}$$

where:

- K_g is the set of generator nodes;
- K_d is the set of the load nodes;
- K is the set of all nodes;
- $c_i(\cdot)$ represents the production cost of the i-th generator;
- P_{gi} is the power output of the i-th generator, with P_{gi}^{min} and P_{gi}^{max} as the corresponding minimum and maximum limits;
- D_j is the demand of the j-th node;
- $\theta_{ij} = \theta_i - \theta_j$, with θ_i and θ_j as the voltage angle at node i and j;
- g_{ij} and b_{ij} are the conductance and the susceptance from node i to j;

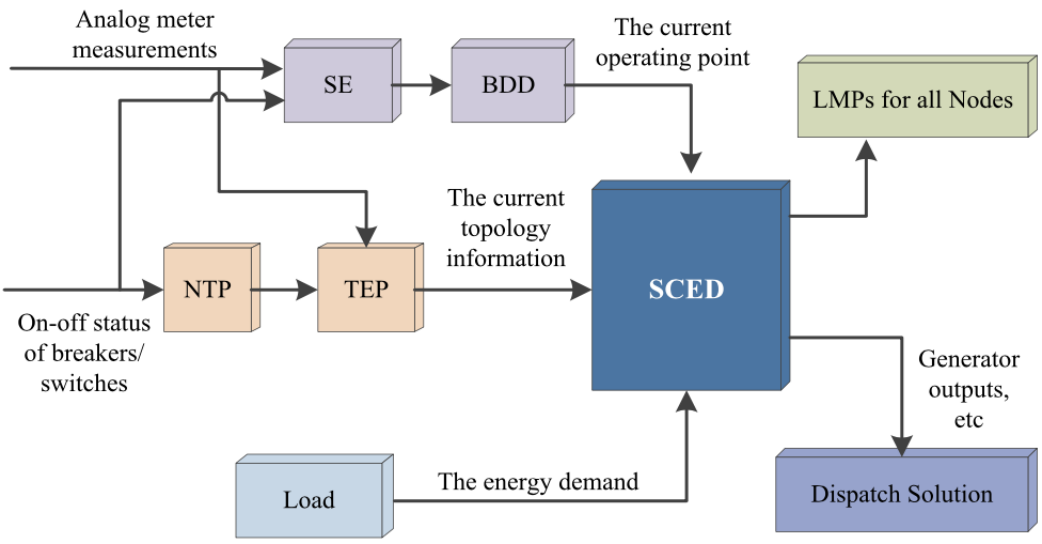


Figure 7: The flow diagram of ISO decision-making procedure

- v_i and v_j are the voltage magnitude of the i-th and j-th node, with v_i^{min} and v_i^{max} as the corresponding minimum and maximum limits;
- P_l and Q_l are the real and reactive power flow on transmission line l and P_{min}, P_{max}, Q_{min} and Q_{max} as the corresponding minimum and maximum limits;
- $(i, j) \rightarrow l$ indicates that i and j are two ends of the transmission line l.

3.4.2 Attack goals

Cyber-attackers attack the power system with various objectives, for example for causing disturbances, producing blackouts, making profits in power market, etc. In this case, our goals was the reproduction of the objective of [2]. The authors wanted to maximize the some cost function, in order to cause a physical or a economical damage. The procedure tooks advantage of the SCED process, used to find the optimal generated power of the system. In fact, if the ISO decision-making procedure, works with a different topology, whit respect the actual one, a damage to the grid could be made. So, to reproduce the paper results, we focused on two different aspect: simulating a change in the topology of the grid, using an FDIA injection to avoid to be detected, and choosing an optimal set of lines to switch on/off, in order to maximize the damage. For the latter objective, we implemented some heuristic and genetic based optimization algorithm in order to solve a complex optimization problem.

In this paper we assume that a cyber-attacker aims to introduce disturbances into the system with the following two motivations:

- causing economic losses to the system or customers;
- causing security damage on transmission lines.

3.4.3 Measurements taken from the system

Before explaining the way in which we implemented the FDIA injection, mentioned before, it is important to describe what type of measurement of the grid we assume to be available to the ISO

decision-making procedure.

Generally, in a real grid are available the measures about the power flows between the different lines and the power injected in some nodes. Since we used MATPOWER, the method relative to the state estimation, takes as input only the generated powers (active/reactive) injected in the generation buses and the power at 'form' ends and 'to' ends of each branch of the system. So we decided to base our solution only on these measurements, proceeding then to apply the state estimation using the MATPOWER tools. Below is presented a summary of the measurement taken from the system with 46 branches and 39 nodes

- Generated active power injected (x10)
- Generated reactive power injected (x10)
- Active power flow recorded in the 'from' end of each branch (x46)
- Reactive power flow recorded in the 'from' end of each branch (x46)
- Active power flow recorded in the 'to' end of each branch (x46)
- Reactive power flow recorded in the 'to' end of each branch (x46)

3.4.4 Attack decision: (FDIA and) State Preserving Attack

As the name suggests, the attack intentionally preserves the state in order to have a sparse attack vector. Let's consider the **noiseless measurement case**:

> given $z = H\mathrm{x} \in \mathrm{Col}(H)$, the state-preserving attack vector a(z) equal to $(\bar{H} - H)$x and then z + a(z) = $\bar{H}$x $\in$ Col($\bar{H}$) causing the attack *undetectable*. The state x remains the same after the attack and, since H has full column rank, a(z) can be simply calculated as
>
> $$a(z) = (\bar{H} - H)x = (\bar{H} - H)(H^t H)^{-1} H^t z. \tag{17}$$
>
> For a(z) above to be a valid attack vector, it is necessary to be a sparse vector constrained by the meters, the data of which can be altered by the adversary.
> It is important to underline that, among all undetectable attacks, the state-preserving attack modifies the smallest numbers of meters, which is the total number of line flow and injection meters located on the target lines and the target buses.

3.4.5 Attack procedure and adopted algorithm

In order to perform the attack we proceeded in the solution of the optimization problem presented in 18. Here the function to maximize are reported in [2] The constraint relative to the SCED problem were computed using MATPOWER, and even the power flow equations reported in 18e were implemented using MATPOWER.

$$max\ G(\Omega^+, \Omega^-, a) \tag{18a}$$

$$s.t.\ \left\{P^*_{gi}, lmp^*_i, P^*_l\right\} = f^{SCED}(\Omega_C) \tag{18b}$$

$$\left\{P^x_{gi}, lmp^x_i, P^x_l\right\} = f^{SCED}(\bar{\Omega}_C) \tag{18c}$$

$$\left\{P^{\otimes}_{gi}, P^{\otimes}_l\right\} = f^{pf}(P^x_{gi}, \Omega_C) \tag{18d}$$

$$\{a\} = f^a(\Omega^+, \Omega^-) \tag{18e}$$

where

- $G(\cdot)$ represents single or multiple objective;
- $f^{SCED}(\cdot)$ represents the SCED model from Eqs. (16a)-(16g);
- eq. (18b) represents the case without attack;
- eq. (18c) represents the case with attack;
- $f^{pf}(\cdot)$ represents the power flow equations;
- eq. (18d) represents the arrow ③ in fig. 6;
- $f^{a}(\cdot)$ represent the constraints to make attack undetectable

By solving the cyber-topology attack model, the cyber attacker calculates the appropriate set of lines Ω^+ and Ω^- to manipulate and determines the injection value a to avoid being detected. The actual implementation of this problem differs from the different optimization algorithms that we implemented.

4 Adopted optimization algorithms

4.1 Natural Aggregation Algorithm (NAA)

NAA, who stands for *Natural Aggregation Algorithm*, is a new metaheuristic algorithm recently proposed by F. Luo and other colleagues to solve the single objective problem.

Recall that metaheuristic is a higher-level procedure designed to find, generate, or select a heuristic (partial search algorithm) that may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity.

NAA mimics the self-aggregation intelligence of group-living animals, balancing the exploitation and exploration steps within a search process, and uses a set of heuristic rules to perform a stochastic search in the problem space.

As a population-based evolutionary algorithm, NAA distributes the whole population across multiple sub-populations through a stochastic migration model so as to evaluate the probability of leaving or entering into the current sub-populations.

In the proposed paper, each individual is encoded as a vector with the dimension of $D^+ + D^- + D^{injection}$; the first D^+ dimensions are binary variables, representing the OPEN/CLOSED line status of Ω^+ set; the next D^- dimensions are also binary variables, representing the OPEN/CLOSED line status of Ω^- set; the last $D^{injection}$ dimensions are continuous variables, representing the injection values on sensors.

In lab experiments, NAA shows very strong global searching capability and fast convergence.

4.1.1 Implementation

The implementation of NAA is based on a cost function and a constraint handler function; the former analyzes an individual of the population returning its associated cost value. The algorithm is an heuristic, so its functioning is based on the test of a different number of individuals in order to find the one which minimizes the cost function and respect the constraint.

We implemented two different cost functions:

- the first is related to the *Case 1 in the Scenario 1* reported in [2] and its goal is to maximize the delta between the normal generation cost and the generation cost under attack;
- the second is related to the *Case 2 in the Scenario 1* reported in [2];
- the third cost function is in charge of maximizing the delta between the power flow under normal operation conditions and under the attack. This type of cost function wants to reach a very dangerous objective, since the exceeding of safety limit of power flow in the branches leads to important safety issues for the grind and to the tripping of the line.

The constraint handle function is needed to verify that the attack vector used in the current individual is able to pass the BDD test: if the current individual doesn't pass the BDD test, the function updates it computing a new a vector using the state preserving attack method.

Instead, if the individual pass the BDD test, the constraint function maintains the current individual, proceeding with the cost function computation.

4.1.2 Cost Function

The first step of the cost function is to analyze the new individual proposed by the NAA, which includes either the indexes of the line to switch on (line addiction attack) or to switch off (line removal attack) or both (line switching attack) and the values of the a vector to minimize, as requested by the paper.

The value of generated power P^*_{gi}, lines power flows P^*_l and lmp^* relative to the nominal model are already available to the function: then the modified topology is computed and the SCED problem is solved on it, using the MATPOWER $run_pf()$ function. This passage corresponds to the solution of eq. (18c).

The value found are the optimal generated power P^x_l, lmp^x_i and power flows P^x_l for the attack model. It is important to recall that these values are not optimal for the actual model; then, we put these generation outputs into the nominal model simulating the ISO decision center that, unaware of the attack, applies the optimal generation level found solving the SCED on the attack topology.

To figure out the effect of the non optimal generation power levels in the nominal model, we solved the power flow equations with the MATPOWER $run_pf()$ function, which does the operation reported in eq. (18d), returning the values of power flows $P^{\otimes}_l$ and generation $P^{\otimes}_{gi}$ reached in the real model, with the decisions made with the changed topology.

In this way it is possible to compute the cost function values reported in the following tables:

TABLE II
DEVIATIONS CAUSED BY CYBER-TOPOLOGY ATTACK

Deviation	Deviation value = value under attack - value under no attack
production cost	$\varphi^f = \sum_{i \in K_g} c_i(P_{gi}{}^{\otimes}) - \sum_{i \in K_g} c_i(P_{gi}{}^*)$
LMP	$\varphi_i^{lmp} = lmp_i{}^{\times} - lmp_i{}^*, i \in K$
Power flow	$\varphi_l{}^{pf} = P_l^{\otimes} - P_l^*, l \in \Omega_c$

TABLE III
CLASSIFICATION OF ATTACKING OBJECTIVES

Scenarios	Single-objective	Multi-objective
Scenario 1: Economy	Case 1 : max φ^f	Case 3 : Case 1 & Case 2
	Case 2 : max $\sum_{i \in K} \varphi_i^{lmp} * D_i$	
Scenario 2: Security	max $\varphi_l{}^{pf}, l \in \Omega_c$	

Also the η index is computed, in order to have a value for comparing the different attacks of the attack: it measures the delta of the difference between the generation cost without the attack and with, over the total generation cost without the attack.

4.2 Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D)

Decomposition is a basic strategy in traditional multi-objective optimization, although it has not yet been widely used in multi-objective evolutionary optimization.

MOEA/D, who stands for *Multi-Objective Evolutionary Algorithm Based on Decomposition*, decomposes a multi-objective optimization problem into a number of scalar optimization sub-problems and optimizes them simultaneously, showing low computational complexity.

Each individual solution in the population is associated with a sub-problem and each sub-problem is optimized by only using information from its neighboring sub-problems.

A neighborhood relationship among all the sub-problems is defined based on the distances of their weight vectors.

MOEA/D gives rise to a set of Pareto-optimal solutions (recall that Pareto optimality is a situation where no individual or preference criterion can be made better off without making at least one individual or preference criterion worse off): since the objectives conflict with each other, all solutions on the Pareto frontier are non-dominated and considered as optimized solutions.

4.2.1 Implementation

MOEA/D is an heuristic multiobjective optimization algorithm very similar to the NAA, except for the absence of the constraint function handle: in this case we made the assumption that the attacker can always find an attack vector a which can hide the attack.

The reason why is that the MOEA/D support only binary optimization variables, so is not possible to minimize the a vector as well as in the NAA.

Another important constraint is to prevent the attack system to be unobservable: in order to do this, the algorithm returns a cost, $+\infty$, if the attack topology is not unobservable.

The remaining part of the cost function computation works exactly like the NAA, so this was already discussed in section (4.1.2): the only difference is that, in this case, are returned two cost functions, since this is a multi-objective optimization.

4.3 Genetic Algorithm (GA)

4.3.1 Introduction and working method

Genetic Algorithm (GA) is a metaheuristic method which takes inspiration directly from the process of the natural selection and it belongs to the class of the Evolutionary Algorithms (EA). In the case of optimization problems at each iteration a population of candidate solutions, that in our situation we called "individuals", is proposed.

Normally a simulation starts from a population of randomly generated individuals and, during an iterative process, firstly there is an evaluation of the fitness, which is the value of the objective function in the optimization problem solved, of every individual, then more fit individuals are stochastically selected from the current population and the properties (or the genome) of each individual is modified to create a new generation. The new candidate solutions will be used in the next iteration of the algorithm.

The algorithm will be stopped in a predefined number of iteration, that it is possible to set previously, or when the fitness reaches a satisfactory value. Initially the algorithm needs:

- a genetic representation of the solution domain;

- a fitness function to evaluate the solution domain.

The implementation of the cost function and in general of the functions structure used in “utils.m” is comparable and equivalent to the one used for the DragonFly Algorithm and for the NAA.

The fitness function is implemented in three different ways, depending on which scenario and the case we want to simulate:

- Scenario 1 (Economy) Case 1;

```
fitness = sum(lmp_cross-lmp_star);
```

- Scenario 1 (Economy) Case 2;

```
fitness = sum(totcost(gencost, Pg_circled_times)) - sum(
   totcost(gencost,Pg_star)););
```

- Scenario 2 (Security);

```
fitness = norm(PFrom_circled_times - PFrom_star);
```

4.3.2 Principal limitations

The Genetic Algorithm have a lot of disadvantages and limitations that could have an important impact on their use for optimization problems, while researchers are trying to find new solutions and implementations to make this algorithms suitable for this types of problems:

- normally and practically finding the optimal solution to complex high-dimensional and multimodal problems requires very expensive fitness evaluations. The evaluation of a single function could lasts a several amount of time and so the deal is aimed by the use of approximated fitness functions;
- Genetic Algorithm do not scale well with complexity. That is, where the number of elements which are exposed to mutation is large there is often an exponential increase in search space size;
- the ”better” solution is only in comparison to other solutions. As a result, the stop criterion is not clear in every problem;
- in many problems, Genetic Algorithm have a tendency to converge towards local optimal than to global optimal of the problem. This means that it does not ”know how” to sacrifice short-term fitness to gain longer-term fitness;
- genomes converges on solutions than could be no longer valid in case we use dynamical sets;
- Genetic Algorithm cannot effectively solve decision problems in which the only fitness measure is a single right/wrong measure, as there is no way to converge on the solution (no hill to climb).

4.4 Binary Dragonfly (BDA)

4.4.1 Introduction

Dragonfly algorithm has shown its ability to optimize different real-world problems and it has three different variants: we have focused the work on the Binary Dragonfly. The Dragonfly algorithm is a part of the SI-based (Swarm-Intelligence) algorithms which are considered to be low cost, fast, and robust for complex real-world problems.

DA is mimicking the swarming behaviours of a dragonfly and the swarming could be divided in two types: dynamic (migration) or static(hunting). The direction of the dragonflies is given and oriented by five weights:

- separation weight (s)
- alignment weight (a)
- cohesion weight (c)
- food factor (f)
- enemy factor (e)
- inertia weight (w)

Tuning the swarming weights (s, a, c, f, e, and) adaptively during the optimization process is another way to balance exploration (high alignment and low cohesion) and exploitation (low alignment and high cohesion).

$$S_i = \sum_{j \in N} X - X_j \tag{19a}$$

$$A_i = \frac{\sum_{j \in N} V_j}{N} \tag{19b}$$

$$C_i = \frac{\sum_{j \in N} X_j}{N} - X \tag{19c}$$

$$F_i = X^+ - X \tag{19d}$$

$$E_i = X^- + X \tag{19e}$$

For the fourth and the fifth equation X^+ represents the position of the source food and X^- represents the position of the enemy while V_j represents the velocity of the j^{th} neighbouring dragonfly. In order to improve randomness, stochastic behaviour, and exploration of artificial dragonfly individuals, dragonflies use a random walk (Lèvy Flight) to move around the search space.
For position updating in the search space, artificial dragonflies use two vectors: step vector $_t$ and position vector X. The first is defined as :

$$\Delta X_{t+1} = (s * S_i + a * A_i + c * C_i + f * F_i + e * E_i) + w * \Delta X_t \tag{20}$$

When the step vector calculation is finished, the we calculate the position vectors :

$$\Delta X_{t+1} = X_{t+1} + \Delta X_{t+1} \tag{21}$$

The best and the worst solutions found at each iteration so far become the food source and enemy, respectively. This makes convergence and divergence towards the promising area and outwards nonpromising area of the search space, respectively. In the case of the binary dragonfly

algorithm the position vector could be equal to 0 or 1 so normally is used the following equation to calculate the changing probability of the position of all artificial dragonflies:

$$T(\Delta X) = |\frac{\Delta X}{\sqrt{(\Delta X)^2 + 1}}| \tag{22}$$

In order to update the position was created the search agent's position in binary search spaces:

$$\begin{cases} \Delta X_{t+1} = X_t & r < T(\Delta Xt + 1); \\ \Delta X_{t+1} = X_t & r >= T(\Delta Xt + 1); \end{cases} \tag{23}$$

4.4.2 Pseudo-Code

The cost function and the fitness functions for the different Scenarios are equal to one shown in the section of the GA. We can look deeply to the pseudo-code:

Algorithm 1 BDA

Initialize the dragonflies population X_i $(i = 1, 2, \ldots, n)$
Initialize ΔX_i $(i = 1, 2, \ldots, n)$
while iteration=1, 2, ... satisfying end condition/max iteration **do**
    Calculate the objective values of all (evaluating each dragonfly
    Update Food Source and Enemy
    Update the main coefficients (w, s, a, c, f, and e)
    Calculate S, A, C, F, and E using equations (19a)–(19e)
    Update step vectors using equation (20)
    Calculate the probabilities using equation (22)
    Update position vectors using equation (23)
end while

5 Simulation Results

For our considerations we refer to the following sets:

- Ω_R, set of removable lines:

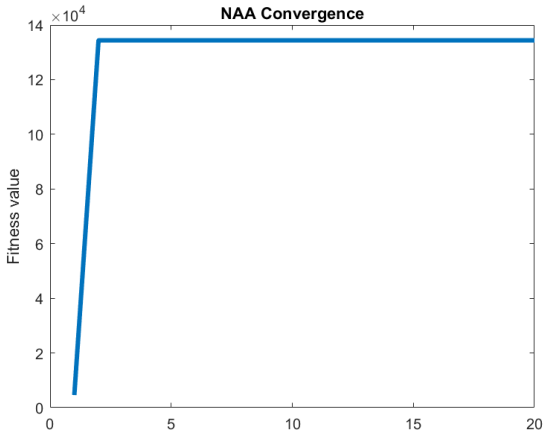
  6 - 7;
  10 - 13;
  5 - 6;
  13- 14;
  2 - 3;
  4 - 5;
  3 - 4.

- Ω_A, set of additionable lines:

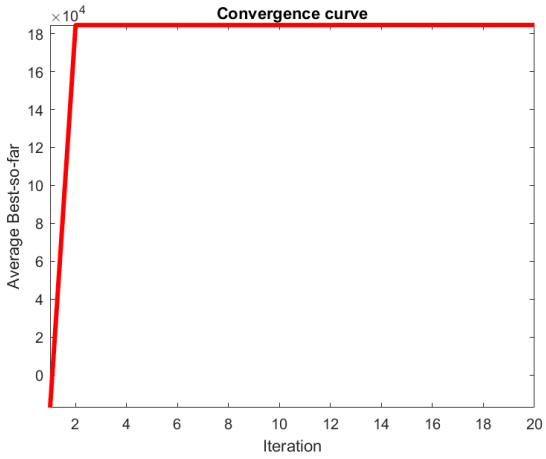
  10 - 11;
  16 - 21.

5.1 Scenario 1 - Case 1

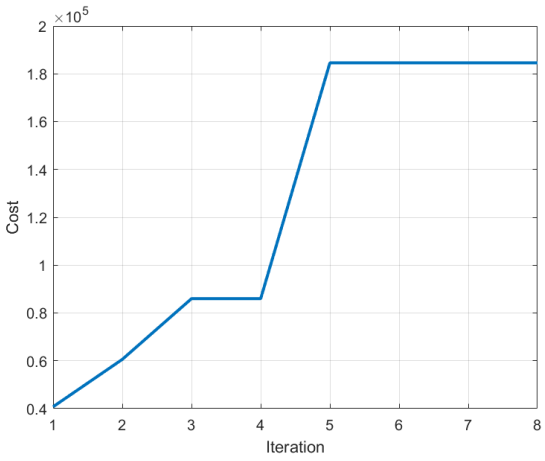
SCENARIO 1 - CASE 1				
Algorithm Situation	**Best Cost Function**	η	**Removed Lines**	**Added Lines**
NAA	1.3438e+05	6.72%	[6-7, 13-14, 2-3, 3-4]	[16-21]
BDA	1.8455e+05	9.23%	[6-7, 13-14, 2-3, 3-4]	[10-11]
GA	1.8458e+05	9.23%	[6-7, 13-14, 2-3, 3-4]	[10-11]
MOEA/D Scenario 1	[0.189175; 1.259]e+05	6.3%	[13-14, 2-3]	[10-11]



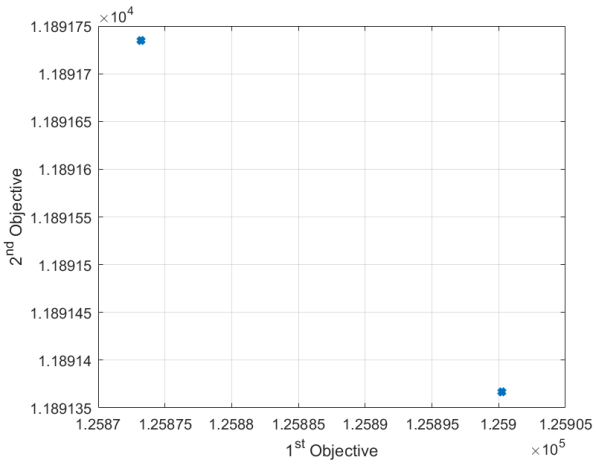
(a) NAA Scenario 1 Case 1



(b) BDA Scenario 1 Case 1



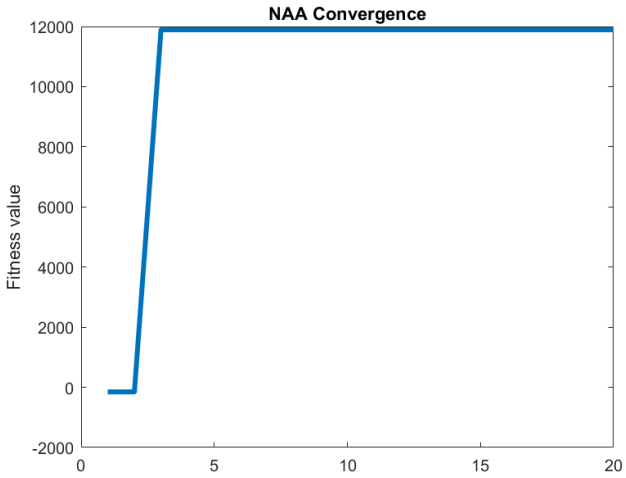
(a) GA Scenario 1 Case 1



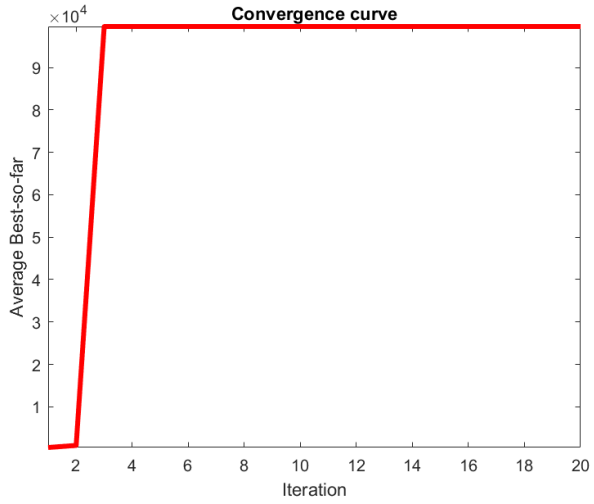
(b) MOEA/D Scenario 1

5.2 Scenario 1 - Case 2

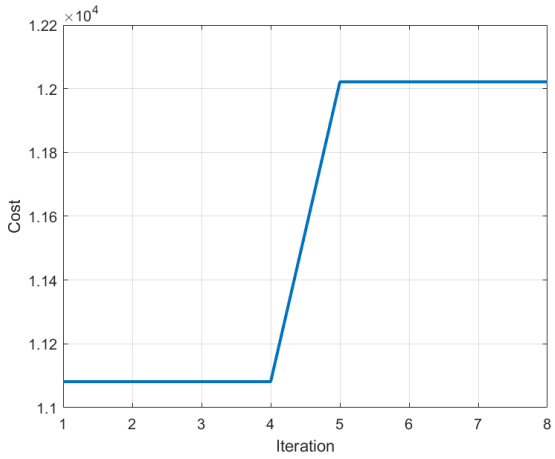
SCENARIO 1 - CASE 2				
Algorithm Situation	**Best Cost Function**	η	**Removed Lines**	**Added Lines**
NAA	1,1891e+04	0.5953%	[10-13, 13-14, 3-4]	[10-11]
BDA	9.9674e+04	9.2379%	[6-7, 10-13, 13-14, 2-3, 3-4]	[10-11]
GA	1.2021e+04	-3.6259%	[3-4, 5-6]	[16-21]
MOEA/D Scenario 1	[0.189175; 1.259]e+05	6.3%	[13-14, 2-3]	[16-21]



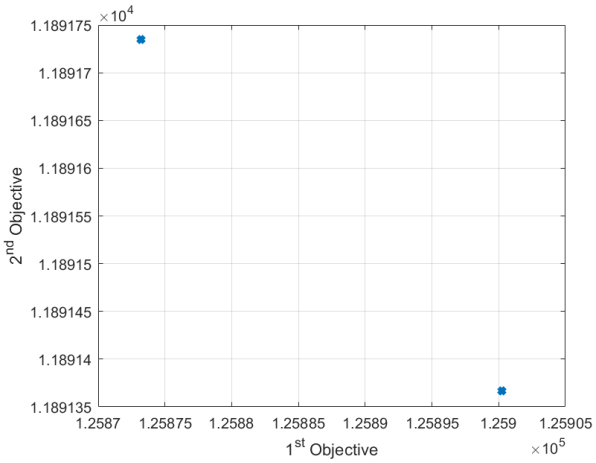
(a) NAA Scenario 1 Case 2



(b) BDA Scenario 1 Case 2



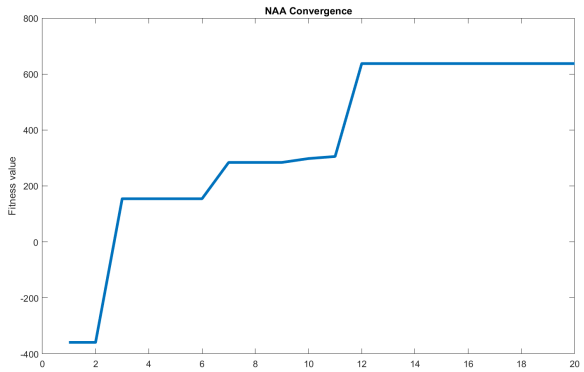
(a) GA Scenario 1 Case 2



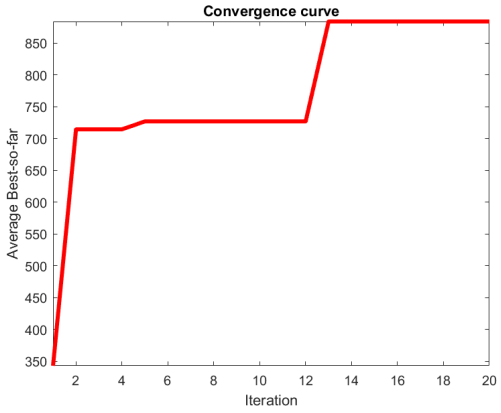
(b) MOEA/D Scenario 1

5.3 Scenario 2

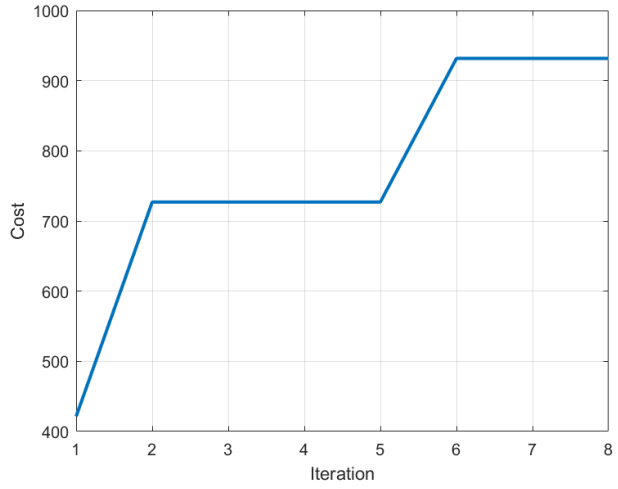
SCENARIO 2				
Algorithm Situation	**Best Cost Function**	η	**Removed Lines**	**Added Lines**
NAA	637.2822	-5.58%	[3-4]	[10-11, 16-21]
BDA	884.0429	-1.1702%	[2-3, 4-5, 5-6]	[10-11, 16-21]
GA	931.659	3.1867%	[2-3, 4-5]	[10-11, 16-21]
MOEA/D	[0.0475; -1.1892]e+04	6.3%	[10-13, 13-14, 2-3]	[10-11]



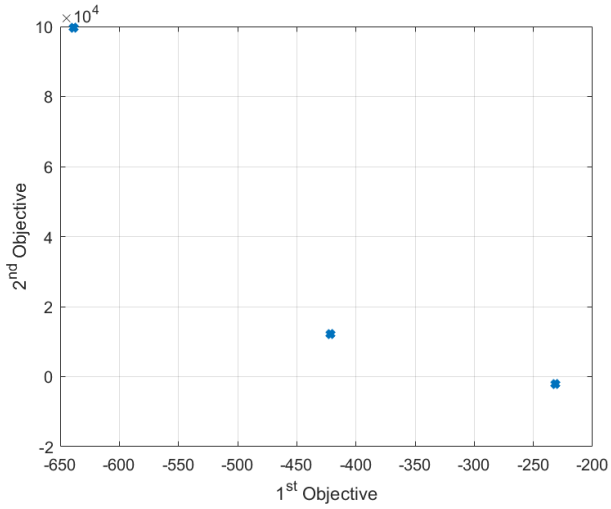
(a) NAA Scenario 2



(b) BDA Scenario 2



(a) GA Scenario 2



(b) MOEA/D Scenario 2

6 Conclusion

With all the simulation that we performed, we were able to attack the power grid maintaining the attack undetected. We tested different algorithm and use cases which turned out in different impacted lines, type of damage and values of the optimal cost function.

Firstly, we worked on the lines reported in 5, they are divided in Removable Lines Ω_R and Additionable Lines Ω_A. We didn't take the same sets of the original paper, sice using too many lines we found some issues in the convergence of the algorithms. Therefore, we restricted the lines to the ones, that in the results of the original paper results as impacted most frequently.

The results of our simulations on the different use cases are reported in the tables above.They report the performance index and optimal solution associated with each run. The eta value is the percentage of the ratio between the delta relative to the nominal total generation cost and the total generation cost under attack and the nominal total generation cost. The eta value is used as an index to evaluate objectively the real impact of the attack.

The results are divided in the three different scenarios. The first is Scenario 1 Case 1, in table 5.1. This case deals with causing an economical damage to the grid, meaning that the eta is a very good indicator for the impact of the attack. The BDA and GA are the most impacting algorithm. Their eta and best cost function values are very similar, since the use the same cost function implementation. This is even the reason why the optimal choice about lines to switch off and on are the same. The other algorithms reaches a value of eta which smaller that the one found with BDA and GA. Despite this, the BDA is able to converge faster that the GA. Even the NAA, although return an best cost function value less that the GA and BDA, converges faster than the GA. The MOEA/D is used for the multi objective optimization, this is the reason why the output of the algorithm is a Pareto front, composed by two different optimal solutions. In this case the MOEA/D, despite it tries to maximize the deviation between the power grid operational cost with and without the attack and the lmp values with and without the attack, is not able to find an eta value higher than the ones of BDA and GA.

The second group of results is relative to Scenario 1, Case 2, table 5.2. Even in this case the BDA is able to find the highest value of eta. In contrast with the Case 1 of Scenario 1, the GA is not able to reproduce the performances of the GA, showing even a different optimal solution (i.e. lines to switch on and off). The NAA, even in this case shows worst performances than the other algorithms. It is important to notice that, even in this case the GA converges slowly than the other algorithms.

The third group is relative to the scenario 2, table 5.3. This scenario deals with the maximization of the physical damage to the grid. In this case the eta value has the same meaning of before. So to show which is the algorithms that creates the higher damage it is necessary to read the 'Best Cost Function' value. The GA, has the best score. The NAA has the worst score and the worst convergence velocity, but it is associated with the best eta value. In this case the MOEA/D was used to maximize both economical and physical damage to the grid, since can implement two different cost functions to optimize.

These results show that, in the average of all cases the GA and BDA performs well than NAA and MOEA/D. A possible reason is that GA and BDA are highly specialized binary algorithm, instead of the NAA which works even with real values and MOAD/D which has more than one cost functions. The NAA was used in our case even to find the best value of the attack vector for the FDIA injection, since can handle real optimization variables. Therefore the worst result

can be associated to a more complex optimization problem to solve. It is important to recall that the MOEA/D is binary algorithm but can handle two different cost functions, resulting in a more complex problem spece were the algorithm is in charge of finding one or more optimal solutions.

7 Code

7.1 Initializing Sets

```
%% The index is composed by |omega_minus||omega_plus||a vector|
%% The indices corresponds to |omega_r||omega_a|
omega_minus_index = omega_r_set(find(ind(1:omega_r_set_size) == 1));
omega_plus_index = omega_a_set(find(ind(omega_r_set_size+1 :
    omega_r_set_size + omega_a_set_size) == 1));

%% We change the status of the transmission lines

z_digital(omega_plus_index) = 1;
z_digital(omega_minus_index) = 0;

Pg_star = userObj.Pg_star;
lmp_star = userObj.lmp_star;
PFrom_star = userObj.PFrom_star;
```

7.2 Solving SCED model under attack

```
%% Creating the new model with the new topology
attack_model = utils().change_topology_model(nominal_model, z_digital);

%% Solving the SCED model under attack
display('RUN OPF ON ATTACK TOPOLOGY');
opf_attack_model_res = runopf(attack_model);
if (opf_attack_model_res.success == 0)
        fitness = +Inf;
        return;
end

%% Calculating the Power Generated and the other parameters "crossed" (
    in the system attacked")
Pg_cross = opf_attack_model_res.gen(:,PG);
lmp_cross = opf_attack_model_res.bus(:,LAM_P);
PFrom_cross = opf_attack_model_res.branch(:,PF);

%% We substitute the Power Generated in the nominal model
nominal_model2 = nominal_model;
nominal_model2.gen(:,PG) = Pg_cross;

display('RUN POWER FLOW EQUATION WITH NOMINAL TOPOLOGY');
[MVAbase, result.bus, result.gen, result.branch, success, et] = runpf(
    nominal_model2);

```

```
%% Showing results
display(result);
PFrom_circled_times = result.branch(:,PF);
Pg_circled_times = result.gen(:,PG);
display('Old_PG')
display(Pg_cross);
display(PFrom_circled_times);
display('New_PG');
display(Pg_circled_times);
```

7.3 Fitness Function

```
%% Set fitness function the fitness function i relative to Scenario 1 -
    Case1_Case2
gencost = nominal_model.gencost;
% case 1 and case 2 (two different fitness functions)
% fitness = sum(totcost(gencost, Pg_circled_times)) - sum(totcost(
   gencost,Pg_star));
fitness = sum(lmp_cross-lmp_star);
eta = (fitness / sum(totcost(gencost,Pg_star) ) ) * 100;
display(sum(totcost(gencost, Pg_circled_times)));
display(sum(totcost(gencost,Pg_star)));
display("ETA VALUE");
display(eta);
display(fitness);

%% Since the NAA makes the min, to have the max is necessary to change
   the sign of the fitness function
fitness = -fitness;
```

7.4 Creation of attack vector *"a"* and state estimation

```
%% This function creates the attack vector a
function [a, success] = make_attack_vector(model, omega_r_set,
   omega_a_set, omega_plus, omega_minus, z_digital, z_analog, sigmas,
   a_limit);
```

```
%% State Estimation
function [H, baseMVA, bus, gen, branch, success, et, z, z_est,
   error_sqrsum] = custom_SE(model, z_analog,sigmas);
```

References

[1] *https://matpower.org/doc/*

[2] Gaoqi Liang, Steven R. Weller, Junhua Zhao, Fengji Luo and Zhao Yang Dong: *A Framework for Cyber-Topology Attacks: Line-Switching and New Attack Scenarios*

[3] Jinsub Kim and Lang Tong: *On Topology Attack of a Smart Grid: Undetectable Attacks and Countermeasures*

[4] Chnoor M. Rahman and Tarik A. Rashid: *Dragonfly Algorithm and Its Applications in Applied Science Survey*

[5] Dana Bani-Hani: *Genetic Algorithm (GA): A Simple and Intuitive Guide*

[6] S. Mirjalili: *Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems, Neural Computing and Applications*

[7] Mostapha Kalami Heris: *Binary and Real-Coded Genetic Algorithms in MATLAB*

[8] Mostapha Kalami Heris: *MOEA/D in MATLAB*

Figures

List of the figures in the report:

- Figure (1) taken from slides "False Data Injection Attacks (FDIAs) Against State Estimation in Power Systems" of "Systems and Control Methods for Cyber-physical Security" course held by Prof. Francesco Liberati;
- Figure (2) taken from "https://www.electricaleasy.com/2016/01/electrical-power-grid-structure-working.html";
- Figure (3) taken from [2];
- Figure (4) taken from [2];
- Figure (5) taken from [2];
- Figure (6) taken from [2];
- Figure (7) taken from [2];
- Figures (**??**) & (**??**) taken from [2];