Cat swarm optimization and Levenberg-Marquardt for model identification and prediction identification and prediction tuberculosis disease spreading

A Nuari, A Damayanti, and A B Pratiwi

Department of Mathematics, Faculty of Science and Technology, Universitas Airlangga, Surabaya, Indonesia

E-mail: aulid@fst.unair.ac.id

Abstract. Tuberculosis (TB) is an infectious disease caused by the bacterium Mycobacterium tuberculosis and can cause death. Based on the prevalence rate, Indonesia ranks third after India and China as the countries with the highest number of TB sufferers in the world. Models identification and prediction are needed to minimize and anticipate problems that can occur. In applying the model, Cat Swarm Optimization and the Levenberg-Marquardt are used. The purpose of this study is to obtain the prediction result based on the model identification in TB disease spreading using a swarm optimization algorithm and Levenberg-Marquardt. The identification is intended to analyze the spread of TB disease based on actual data. The process begins with estimating the parameters in the model using the Cat Swarm Optimization algorithm. After getting the optimal parameters, the acquisition of the model uses Levenberg-Marquardt. Based on the implementation of the program on the spread of TB disease in East Java in the period of the first quarter of 2002 to the fourth quarter of 2017, the MSE of training and prediction process were 0.000225563 and 0.0085307, respectively. The MSE value from the prediction process shows that the Cat Swarm Optimization and Levenberg-Marquardt can be used to identify models and predict the spread of tuberculosis.

1. Introduction
Tuberculosis (TB) is an infectious disease caused by the bacterium Mycobacterium tuberculosis. This disease can be transmitted to other individuals when the bacteria that come out of an infected individual (sufferer) of TB when coughing or sneezing, is inhaled by that individual. Bacteria released by TB sufferers, generally in the form of droplets or sputum splashes. The inhaled bacteria then enter through the respiratory tract, so that the bacteria can attack the lungs and can attack other organs in the body. [2] The Cat Swarm Optimization (CSO) algorithm is one of the metaheuristic algorithms proposed by Shu-Chu Chu and Pei-Wei Tsai in 2006. The intention of this algorithm is to utilize a combination of cat behavior, namely seeking mode and tracing mode to solve optimization problems. Seeking mode describes a condition when cat is resting, looking at the surrounding conditions looking for the next position to move. Tracing mode describes condition when the cat is following carefully its target [1]. Based on the description above, this study discusses model identification and prediction of the spread of tuberculosis using Cat Swarm Optimization (CSO) and Levenberg-Marquardt.
2. Metodology

2.1 Identification System

System identification is an approach to construct mathematical representations of unknown parameters. Evaluation can be done by testing into the model approach used by entering existing data input. From the result of data input calculation in the model result can be obtained the model output. The output of this model is then compared with the actual data so that it can be evaluated by correcting certain parameters in the model used.

System identification includes several steps: planning, choosing the structure of the model to be used, parameter estimation and model validation. The system identification steps are repeated until a satisfactory system identification result is obtained that is able to approach the actual data. [4]

2.2 Model penyebaran penyakit tuberculosis

The TB spread model is divided into three compartments. These compartments include human populations that are susceptible to tuberculosis which is denoted as S, human populations infected with tuberculosis who are donated as I, and human populations that are recovered from tuberculosis denoted as R. A mathematical model of the spread of SIR type TB disease is as follows: [3]

\[
\frac{dS}{dt} = \prod - b \frac{I}{N} S - \mu S
\]
\[
\frac{dI}{dt} = b \frac{I}{N} S - (\mu + c + \mu_T) I
\]
\[
\frac{dR}{dt} = c I - \mu R
\]

where:
\[\prod\]: the number of recruitment individuals
\[b\]: transmission rates of infected populations to susceptible populations
\[c\]: natural recovery rate
\[\mu\]: natural death rate
\[\mu_T\]: death rates caused by TB

The total population is \(N = S + I + R\). Nilai \(S, I, R \geq 0\) values and parameters in the models \(\prod, b, c, \mu, \mu_T\) have positif values.

2.3 Levenberg Marquardt

The network training algorithm using Levenberg-Marquardt is explained as follows: [5]
Step 1: Determine the initial LM factor constant (\(\lambda\))
Step 2: Determine criteria for termination of training. If the criteria are not met, then do steps 3-6.
Step 3: Calculate the parameter changes by using the following equation:
\[
\Delta X = -(J^T J + \lambda I)^{-1} J^T e
\]
where \(\Delta X\) is the change in network parameters, \(\lambda\) is the LM constant, \(e\) is a decrease in error with the parameter
Step 4: Calculate new parameters using the following equation:
\[
X_{k+1} = X_k + \Delta X_k
\]
where \(X\) is \(LW1, LW2, B1, B2\). \(LW1\) is the hidden weight, \(LW2\) is the output weight, \(B1\) is the bias hidden, and \(B2\) is the output bias.
Step 5: Update \(\lambda\) by using the following equation:
If the new objective function is greater than the previous value, then
Conversely, if the new objective function is smaller than the previous value, then

\[ \lambda_{new} = \frac{\lambda_{old}}{\beta} \]

Step 6: Return to step 2.

2.4 Cat Swarm Optimization Algorithm

The steps for the Cat Swarm Optimization Algorithm are described as follows: [1]

Step 1: Generate the cat in the CSO process.
Step 2: Randomly select a value in the maximum speed range to become the cat's speed. Then select a number of cats arbitrarily and enter in tracing mode according to MR, the rest are entered in seeking mode.
Step 3: Calculate the value of each cat's objective function by entering the position value of the cat that has been converted into a permutation representation into the objective function, and store the best cat in memory. Keep in mind that what needs to be kept is the best cat position because the best cat so far represents the best solution.
Step 4: Move the cat according to its flag, if the cat is in seeking mode, treat it according to the seeking mode process, otherwise treat it according to tracing mode. Each process has been explained before.
Step 5: Select a number of cats again and enter the tracing mode according to MR, the rest enter into seeking mode.
Step 6: Pay attention to the maximum iteration. If it is fulfilled, stop the program. Instead repeat steps 3 through 5.

2.4.1 Seeking Mode

The steps of seeking mode can be described in five stages: [1]

Step 1: Generate j imitation from the current cat position, with j = SMP. If SPC value = 1, then j = (SMP - 1), then maintain the current position as one of the candidates.
Step 2: For each clone, choose a number of dimensions according to the CDC. From each selected dimension, randomly add or subtract SRD percent from the current value and replace the previous value.
Step 3: Calculate the fitness value (FS) for all candidate points.
Step 4: If all FSs are not the same, calculate the chosen probability of each candidate point by using equation

\[ P_i = \frac{|F_i - F_{sb}|}{F_{max} - F_{min}}, \text{where } 0 < i < j \] (1)

Step 5: Calculate the cumulative chosen probability for all candidate points and do a roulette wheel to determine the new position of the cat.

2.4.2 Tracing Mode

The stages of tracing mode can be described in 3 steps as follows: [1]

Step 1: Update the speed value for each dimension (\(v_{k,d}\)) based on equation (2)

\[ v_{k,d} = v_{k,d} + r_1 \cdot c_1 \cdot (x_{best,d} - x_{k,d}) \] (2)

where \(d = 1, 2, ..., M\)
Step 2: Check that the speed is within the maximum speed range. If the new speed exceeds the range, set the value equal to the limit.
Step 3: Update the cat position based on equation (3). \(x_{best,d}\) \(d\) is the position of the cat that has the best fitness value; \(x_{k,d}\) is the position of the cat. \(c_1\) is a constant and \(r_1\) is a random number in the range [0,1].

\[ x_{k,d} = x_{k,d} + v_{k,d} \] (3)
3. Proposed Method

The steps taken to complete this research are as follows:

a. Literature study on the problem of system identification, prediction, the spread of TB disease, Cat Swarm Optimization (CSO), and the Levenberg Marquardt neural network.

b. Determine the model of TB disease referring to a journal written by Fredlina et al in 2012 where the model has 3 compartments, namely S compartment (healthy human population but susceptible to TB infection), compartment I (human population infected with TB), and R compartment (human population who have recovered from TB). The data was obtained from the East Java Provincial Health Office (2018).

c. Applying the CSO process to estimate parameters from the TB disease spread model, with the following steps:
   1) Input data on the spread of TB.
   2) Initialization of CSO parameters are many cats (m), seeking memeory pool (junior high), count dimension to change (cdc), seeking range dimension (srd), mixing ratio (mr), tracing constant (c), max iteration (iteration), and the number of parameters in the model are as many as five parameters.
   3) Generating the initial population of cat using real numbers at intervals [0,1] of five model parameters for each cat denoted by $x_{i}$.
   4) Generates initial velocity for each cat by randomizing real numbers [0,1].
   5) Placing Flags randomly with a number of seeking cats with flag = 1 and the remaining number of tracing means flag = 0.
   6) Calculate numerical solutions for each cat using the fourth order Runge Kutta method.
   7) Conduct initial population evaluation by calculating the objective function of each cat with the objective function being MMRE from the calculation results of numerical solutions using Runge Kutta fourth order and actual target data.
   8) Determine the value of the Self Position Consideration (SPC) the flag whose value is determined based on the evaluation value of the cat's pole.
   9) Save the paint population with the largest evaluation value as xbest.
   10) Tracing mode is the population of cat with flag = 1. Then save the best cat in this mode with the best evaluation value.
   11) Doing mode seeking is the population of paint with flag = 0. Then save the best cat in this mode with the best evaluation value.
   12) Determine the iteration solution by calculating the best cat MMRE from the smallest tracing and seeking modes.
   13) Iteration to maximum iteration.
   14) After the maximum iteration is reached, the best cat will be the solution of the parameter estimation so that it can proceed to the system identification process using the levenberg-marquardt neural network.

d. Implement the Levenberg-Marquardt process for system identification by following these steps:
   1) Initiation of a random network of weights LW1, LW2, B1, B2.
   2) Discrete models using the euler method
   3) Substitution the parameter estimation solutions obtained from the CSO process to the discrete TB disease model.
   4) Do the normalization process.
   5) The network architecture used consists of an input layer with 2 neurons, a hidden field with 2 neurons, and an output layer with 1 neuron.
   6) Conversion parameters obtained from c) are used as the weight and start of the network in the hidden layer, while the weights and biases of the output layer are generated randomly at intervals [-1,1].
   7) Calculate forward at hidden vertices and output $z_{j} = f(z_{inj})$ dan $y_{k} = f(y_{ink})$
   8) Calculate error $e$ with
The following are some variations in the parameter values used to get the smallest error value by assuming many cats = 100, cdc = 0.3, srd = 0.3, c = 2 in the process of identifying the model presented in Table 1, Table 2 and Table 3.

Table 1. Model identification results for susceptible data

| No | Iter. CSO | smp | mr | Error CSO | Stop Iterasion | MSE       |
|----|-----------|-----|----|-----------|----------------|-----------|
| 1  | 50        | 20  | 0.3| 0.2701    |                | 0.00028026|
| 2  | 50        | 20  | 0.212| 0.0052338|                | 0.00028068|
| 3  | 50        | 20  | 0.6 | 0.2615    |                | 0.00028272|
| 4  | 50        | 20  | 0.6 | 0.3007    |                | 0.00031091|
| 5  | 50        | 20  | 0.3 | 0.1725    | 100            | 0.00027998|
| 6  | 50        | 20  | 0.3 | 0.3074    |                | 0.00052268|
| 7  | 50        | 20  | 0.6 | 0.2288    |                | 0.00052317|
| 8  | 50        | 20  | 0.6 | 0.2498    |                | 0.00027998|
| No | Iter. CSO | smp | mr | Error CSO | Stop Iteration | MSE     |
|----|-----------|-----|----|-----------|----------------|---------|
| 9  | 20        | 0.3 |    | 0.2578    |                | 0.00027409 |
| 10 | 50        |     |    | 0.2659    |                | 0.0051478  |
| 11 | 20        | 0.6 |    | 0.2345    | 300            | 0.00027458 |
| 12 | 50        |     |    | 0.3110    |                | 0.00027412 |
| 13 | 20        | 0.3 |    | 0.1979    |                | 0.00027409 |
| 14 | 100       | 0.3 |    | 0.2195    |                | 0.00027414 |
| 15 | 20        | 0.6 |    | 0.2081    |                | 0.00027409 |
| 16 | 50        |     |    | 0.1974    |                | 0.00027411 |
| 17 | 20        | 0.3 |    | 0.3065    |                | 0.00027375 |
| 18 | 50        | 0.3 |    | 0.3044    |                | 0.00027369 |
| 19 | 20        | 0.6 |    | 0.3157    |                | 0.0051759  |
| 20 | 50        |     |    | 0.2987    | 500            | 0.00027371 |
| 21 | 20        | 0.3 |    | 0.2552    |                | 0.00027373 |
| 22 | 100       | 0.3 |    | 0.1727    |                | 0.00027372 |
| 23 | 20        | 0.6 |    | 0.1962    |                | 0.00027371 |
| 24 | 50        |     |    | 0.2705    |                | 0.00027377 |

**Table 2.** Model identification results for infected data
| No | Iter. CSO | smp | mr | Error CSO | Stop Iterasi | MSE       |
|----|-----------|-----|----|-----------|--------------|-----------|
| 1  | 20        | 0.3 | 0.2979 |            |              | 0.00024226 |
| 2  | 50        |     | 0.1913 |            |              | 0.00025153 |
| 3  | 50        | 0.6 | 0.2183 |            |              | 0.00024396 |
| 4  | 50        |     | 0.2172 |            |              | 0.00024174 |
| 5  | 20        | 0.3 | 0.2777 |            | 100          | 0.00024192 |
| 6  | 100       |     | 0.3109 |            |              | 0.00024223 |
| 7  | 20        | 0.6 | 0.2030 |            |              | 0.00024206 |
| 8  | 50        |     | 0.2993 |            |              | 0.00024262 |
| 9  | 20        | 0.6 | 0.2564 |            | 300          | 0.00023199 |
| 10 | 50        |     | 0.2894 |            |              | 0.0002181  |

Table 3. Model identification results for recovered data
After the model identification process is done with all variations of the parameter values, it is obtained that for S data the smallest error value is 0.0002737, data I the smallest error value is 0.00017217, and R data is the smallest error value is 0.00023082. The smallest error value is obtained when the parameter value for S data is max iteration = 100, junior = 50, mr = 0.6, stop iteration = 500; data I is max iteration = 100, junior high = 50, mr = 0.6, stop iteration = 500; and R data are max iteration = 50, junior high = 50, mr = 0.6, stop iteration = 500. The best weight and bias values can be seen in Table 4 and Table 5.

|   |   |   |   |
|---|---|---|---|
| 11| 20| 0.6| 0.3217|
| 12| 50| 0.3012|
| 13| 20| 0.2379|
| 14| 50| 0.2206|
| 15| 20| 0.3031|
| 16| 50| 0.2521|
| 17| 20| 0.2560|
| 18| 50| 0.3019|
| 19| 20| 0.2415|
| 20| 50| 0.3140|
| 21| 20| 0.2744|
| 22| 50| 0.2167|
| 23| 20| 0.1819|
| 24| 50| 0.2766|

Table 4. Optimal weight and bias from the input layer to the hidden layer

|   |   |   |
|---|---|---|
| 0.00023198 |
| 0.00023199 |
| 0.00023947 |
| 0.00023197 |
| 0.00023194 |
| 0.0002322 |
| 0.00023086 |
| 0.00023088 |
| 0.00023392 |
| 0.00023082 |
| 0.00023087 |
| 0.00023084 |
| 0.00023089 |
| 0.00023085 |

After the model identification process is done with all variations of the parameter values, it is obtained that for S data the smallest error value is 0.0002737, data I the smallest error value is 0.00017217, and R data is the smallest error value is 0.00023082. The smallest error value is obtained when the parameter value for S data is max iteration = 100, junior = 50, mr = 0.6, stop iteration = 500; data I is max iteration = 100, junior high = 50, mr = 0.6, stop iteration = 500; and R data are max iteration = 50, junior high = 50, mr = 0.6, stop iteration = 500. The best weight and bias values can be seen in Table 4 and Table 5.

|   |   |   |
|---|---|---|
| 0.00023198 |
| 0.00023199 |
| 0.00023947 |
| 0.00023197 |
| 0.00023194 |
| 0.0002322 |
| 0.00023086 |
| 0.00023088 |
| 0.00023392 |
| 0.00023082 |
| 0.00023087 |
| 0.00023084 |
| 0.00023089 |
| 0.00023085 |

Table 4. Optimal weight and bias from the input layer to the hidden layer

|   |   |   |
|---|---|---|
| 0.00023082 |
| 0.00023088 |
| 0.00023087 |
| 0.00023084 |
| 0.00023089 |
| 0.00023085 |

Table 5. Optimal weight and bias from the hidden layer to the output layer

|   |   |   |
|---|---|---|
| 0.00023198 |
| 0.00023199 |
| 0.00023947 |
| 0.00023197 |
| 0.00023194 |
| 0.0002322 |
| 0.00023086 |
| 0.00023088 |
| 0.00023392 |
| 0.00023082 |
| 0.00023087 |
| 0.00023084 |
| 0.00023089 |
| 0.00023085 |

Table 5. Optimal weight and bias from the hidden layer to the output layer

|   |   |   |
|---|---|---|
| 0.00023082 |
| 0.00023088 |
| 0.00023087 |
| 0.00023084 |
| 0.00023089 |
| 0.00023085 |

After the model identification process is done with all variations of the parameter values, it is obtained that for S data the smallest error value is 0.0002737, data I the smallest error value is 0.00017217, and R data is the smallest error value is 0.00023082. The smallest error value is obtained when the parameter value for S data is max iteration = 100, junior = 50, mr = 0.6, stop iteration = 500; data I is max iteration = 100, junior high = 50, mr = 0.6, stop iteration = 500; and R data are max iteration = 50, junior high = 50, mr = 0.6, stop iteration = 500. The best weight and bias values can be seen in Table 4 and Table 5.
The best weight and bias values obtained from the training process using Levenberg Marquardt are used for the validation process. Model validation is used to obtain error values from the calculation output results in training data. Model validation process is done by forward propagation process in neural networks where the weights and biases used are weights and biases in Table 4 and Table 5. After obtaining an output value in the forward propagation process, an error value is obtained. The output results of the model validation process can be seen in Table 6.

| $t$ | Data $(S)$ | Hasil Identifikasi $(S)$ | Data $(I)$ | Hasil Identifikasi $(I)$ | Data $(R)$ | Hasil Identifikasi $(R)$ |
|-----|------------|--------------------------|------------|--------------------------|------------|--------------------------|
| 1   | 35695589   | 357036 01                | 4567       | 4648                     | 2681       | 2636                     |
| 2   | 35695620   | 357031 86                | 5207       | 5170                     | 2310       | 2400                     |
| 43  | 37938683   | 379369 42                | 1009 5     | 10107                    | 9688       | 9697                     |
| 44  | 37939887   | 379367 47                | 9666       | 9681                     | 8913       | 8919                     |

Meanwhile, the model validation process graph $S$ (Susceptible), $I$ (Infected), $R$ (Recovered) data, can be seen in Figure 1, Figure 2, and Figure 3.

![Grafik Kesalahan JST vs Target](image)

**Figure 1.** Model validation for data $S$
Furthermore, the prediction process will be carried out with new data where the weights and biases used are the weights and biases from the results of the training data. Furthermore, MSE values obtained from the prediction results of each population test data S, I, and R in order of 0.011445, 0.0087631, and 0.005384. The graph of the results of the process of prediction of S, I, and R consecutive can be seen in Figure 4, Figure 5, and Figure 6.
Based on the results of the identification in Figure 1, Figure 2, and Figure 3, it is obtained a relatively small error value. So the network model is ready to be used to make the prediction process. For the results of the prediction process in Figure 4, Figure 5, and Figure 6, MSE error values are obtained which are relatively small values so that the artificial neural network that is formed is good enough to do the prediction process.

5. Conclusion

Based on the implementation of the program and simulation for the best parameter values carried out on the spread of Tuberculosis data for the population of S, I, and R in East Java, the average MSE results obtained in the training process amounted to 0.000225563. As for the population prediction process S, I, and R which are done using test data obtained an average MSE of 0.0085307. Therefore, models of artificial neural networks can be used for the prediction process.

References

[1] Chu S C dan Tsai P W 2006 Computational intelligence based on behaviour of cat International Journal of Innovative Computing, Information, and Control 3(1)
[2] Departemen Kesehatan RI 2005 Pharmaceutical care untuk penyakit tuberculosis (Jakarta: Direktorat Bina Farmasi Komunikasi dan Klinik)
[3] Fredlina K Q, Oka T B, dan Dwipa I M E 2012 Model SIR (Susceptible, Infectious, Recovered) untuk penyebaran penyakit tuberkulosis, e-Jurnal Matematika 1 52-58
[4] Rubio J J and Yu W 2007 Nonlinear system identification with recurrent neural network and dead-zone Kalman filter algorithm Neurocomputing 70(13-15) 2460-2466

[5] Van Gorp J 2000 Nonlinear Identification with Neural Networks and Fuzzy Logic, Dienst Algemene Elektriciteit and Instrumentatie (ELEC) Pleinlaan 2 B-1050 (Brussel, Belgium)