Predictive System Control Using Bat Algorithm

Nikita Mazurenko[1], prof. Ing. Milan Hofreiter, CSc.[1]

[1] *CTU in Prague, Faculty of Mechanical Engineering, Department of Instrumentation a Control Engineering, Technická 4, 166 07 Prague 6, Czech Republic*

Abstract

The main aim of this work is the design and testing of a predictive control algorithm based on the echolocation behavior of bats. The first part of the paper describes the basic principles of model predictive control. The next part of the work is devoted to the bat algorithm. It describes the working method of the optimization algorithm based on the behavior of bats, its pseudocode, and describes the implementation of this algorithm in the model predictive controller. The third part of the paper is devoted directly to simulating and testing the predictive control algorithm. The proposed and programmatically executed predictive control is simulated in a linear SISO (Single-Input-Single-Output) system using the MATLAB programming environment.

Keywords: STC; Model Predictive Control; Bat Algorithm; simulation; MATLAB

1. Model Predictive Control

Model predictive control is an advanced method for process control that has been used in industrial processes, chemical plants and oil refineries since the 1980s. The term MPC does not specify a management strategy, but define a wide range of control methods that explicitly use a process model to obtain a control signal by minimalizing a cost function. These methods can be characterized by the following common features [1]:

- Directly use of the model for predicting process outputs in future time intervals
- Calculation of the control sequence to minimizing a cost function
- Displacement of the prediction horizon at each timestep
- Reiteration of the same calculation over updated data at each timestep

MPC is a modern method of controlling, but this method is not perfect. That's why it has several drawbacks in comparison with classic methods. The most important one is that the mathematical model of process must be defined and this model has a great influence on the quality of regulation [1].

1.1. MPC Strategy

The basic principle of MPC can be described by Fig. 1.1.

1) The future outputs of a controlled system are predicted using the process model at each moment k for a specified horizon N (prediction horizon). These predicted outputs $y_p(k + i)$, for $i = 1 \ldots N$ depend on the known values of inputs and outputs up to the moment k and on the future control signals $u_p(k + i - 1)$.
2) These future control signals are calculated by optimizing a criterion to keep the process as near as possible to the reference trajectory $w(k + i)$. Where reference trajectory may be the setpoint itself or its close approximation. As a criterion for optimization is usually used a variance between the predicted trajectory of the controlled variable and its desired course.
3) Only the control signal $u_p(k)$ is transmitted to the controlled process because the value of a controlled variable $y(k + 1)$ is known and the remaining values of the control signal must be counted again. In other words, the first step is repeated.

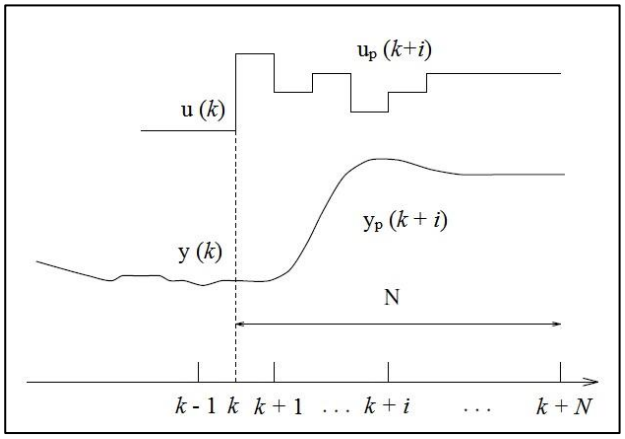

Figure 1.1. *MPC Strategy (Reprinted from* [1]*)*

To implement this strategy could be used the basic structure of the controller that is shown in Fig. 1.2.

This deterministic model is used to predict future outputs of the system. This prediction is based on the past and actual values and on proposed future values of control signal. This chosen model on the controlled process plays an important role in the control because it must be able to capture the process dynamic and be sufficient simple for implementation and understanding.

The optimizer is another essential part of the predictive control structure. It provides action signals (inputs) to the model. These inputs are calculated and optimized considering their influence on the cost function. If this cost function is quadratic its minimum can be obtained as a

* Corresponding author: Nikita.Mazurenko@fs.cvut.cz

linear function of past inputs, past outputs and future reference trajectory.

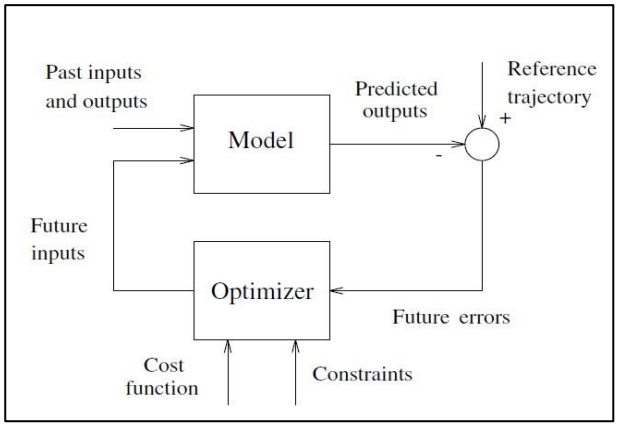

Figure 1.2. *Basic structure of MPC (Reprinted from* [1]*)*

To help understand the basic ideas that have been used in the design of the predictive control, we examine a similar strategy used by car drivers. The driver knows the reference trajectory of the car for the specified distance and knows characteristics of the vehicle. That's why the driver can decide which control action and with what intensity to use (gas pedal, brakes or steering wheel) to drive the desired trajectory. At any moment the driver makes only the first control action, which can help the driver recognize the car's response and decide which control variable (gas pedal, brakes or steering wheel) and with which value will be used in the next timepoints.

2. Control Algorithm Design

As it was said in a previous chapter, optimizer is a very important part of model predictive controller. That's why an algorithm which is implemented into optimizer plays an important role and significantly affects the control process. Various types of optimizing algorithms can be used, but the main aim of this project is to design and test Bat Algorithm for model predictive control.

2.1. Bat Algorithm

2.1.1. Introduction

Metaheuristic algorithms are now becoming powerful methods for solving optimization problems. The majority part of these algorithms has been derived from the behaviour of physical or biological systems in nature. A new metaheuristic method, which is proposed in this paper, is based on the echolocation behaviour of bats. That is why this method is named Bat Algorithm [2].

2.1.2. Behaviour of Bats

Microbats extensively use echolocation to detect prey, locate their crevices and avoid obstacles in the dark. These bats emit a very loud sound pulse and listen for the echo that reflects from surrounding objects. Microbats can correlate these pulses according to their hunting strategy. Microbats use the time delay from the emission of the pulse and detection of the echo, the loudness variations of the signals and the time difference between their two ears to build up three-dimensional map of the surrounding. Bats can detect the distance and orientation of the target, type of prey and even the speed and direction of the prey such as small insects [2].

2.1.3. Pseudo Code of the Bat Algorithm

To design bat-inspired algorithm the echolocation characteristics of microbats were idealized. This idealization can be written in the form of several rules [2]:

1) All bats use echolocation to sense distance and they also know the difference between food and barriers;
2) Bats fly randomly with velocity v_i at position x_i with a fixed frequency f_{min}, adjusting loudness A_0 and wavelength λ to search for food. They can automatically adjust the wavelength of their emitted pulses and adjust the rate of pulse emission r=[0,1], depending on the range of their target.
3) The loudness varies from a large A_0 to a minimum constant value A_{min}.

Based on this idealization, the basic steps of the Bat Algorithm can be described by the pseudo code shown in Fig. 2.1.

Cost function $f(\mathbf{x})$, $\mathbf{x} = (x_1, x_2, ...x_d)^T$
Initialize the population of bats $\mathbf{x}_i$ $(I = 1,2,....n)$ *and* $\mathbf{v}_i$
Define pulse frequency f_i *at* $\mathbf{x}_i$
Initialize pulse rates r_i *and the loudness* A_i
while *(t < Max number of iterations)*
Generate new solutions by adjusting frequency and updating velocities and locations (solutions)
  if $(rand > r_i)$
  Select a solution among the best solutions
  Generate a local solution around the selected solution
  end if
  if $(rand < A_i \ \& \ f(\mathbf{x}_i) < f(\mathbf{x}_*))$
  Accept the new solutions
  Increase r_i *and reduce* A_i
  end if
Rank the bats and find the current best $\mathbf{x}_*$
end while
Postprocess results

Figure 2.1. *Pseudo code of the BA (Reprinted from* [2]*)*

2.1.4. Movement of Virtual Bats

After idealizing the echolocation characteristics of bats, also the rules of bats movement must be defined. In other words, must be defined how positions $\mathbf{x_i}$ and velocities $\mathbf{v_i}$ of microbats in a *d*-dimentioal search space are updated. The new solutions and velocities at the time step *k* are given by

$$f_i = f_{min} + (f_{max} - f_{min})\beta, \qquad (2.1)$$

$$\boldsymbol{v}_i^k = \boldsymbol{v}_i^{k-1} + (\boldsymbol{x}_i^k - \boldsymbol{x}_*)f_i, \qquad (2.2)$$

$$\boldsymbol{x_i^k} = \boldsymbol{x_i^{k-1}} + \boldsymbol{v_i^k}, \quad (2.3)$$

where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution. And $\boldsymbol{x}_*$ is a current best solution (location) among all *n* bats. At each time step, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk

$$\boldsymbol{x}_{new} = \boldsymbol{x}_{old} + \varepsilon \cdot A^k, \quad (2.4)$$

Where $\varepsilon \in [-1, 1]$ is a random number, while A^k is the average loudness of all the bats at this time step.

But the loudness A_i and the rate r_i must be also updated accordingly as the iterations process. Once a bat has found its prey, the loudness usually decreases and the rate of pulse emission increases. These changes can be given by

$$A_i^{k+1} = \alpha A_i^k, \quad r_i^{k+1} = r_i^0[1 - \exp(-\gamma k)], \quad (2.5)$$

where α and γ are constants. And for any $\alpha \in (0, 1)$ and $\gamma \in (0, \infty)$ we have

$$A_i^k \to 0, \ r_i^k \to r_i^0, as\ k \to \infty \quad (2.6)$$

And if minimum constant value of loudness is $A_{min} = 0$, it means that a bat has found the prey and temporary stop emitting any sound [2].

2.2. Bat Algorithm

2.2.1. Solutions Definition

To use the BA in predictive control, it is necessary to determine exactly what the algorithm will serve for. This algorithm works as an optimizer for predictive controller and the main task of the BA algorithm for predictive control of the system is to find the best (optimal) control variables *u(k)*. These control variables route the output of the controlled dynamic system to the reference trajectory *w(k)*.

From the point of view of regulation, a more appropriate method of control is to search for the difference of the control variable $\Delta u(k)$, rather than its absolute value $u(k)$. In this case, the solution vectors $\boldsymbol{x_i}$ for the BA will be given as a sequence of differences of the control variable, and the length of these vectors will be equal to the prediction horizon *N*

$$\boldsymbol{x_i} = [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+N-1)]^T \quad (2.7)$$

To optimize and reduce the requirements for the control equipment, it was decided to use a limited number of possible differences of the control variable. The values of these differences can be written as a vector $\Delta \boldsymbol{U_0}$, that is calculated by

$$\Delta \boldsymbol{U_0} = \frac{w_U - w_L}{2} \cdot [-10, -5, -2, -1, \ 0, 1, 2, 5, 10] \quad (2.8)$$

where w_U and w_L represent the the upper and the lower limits of the proposed control area of the output variable. But the distribution of elements in the vector $\Delta \boldsymbol{U_0}$ is not uniform. This negatively affects the compilation of a new solution by randomly selection of the elements from a vector $\Delta \boldsymbol{U_0}$. Because of this, the algorithm works with indices of the $\Delta \boldsymbol{U_0}$ during the creation of new solutions. But when calculating the cost function, BA uses the vector elements corresponding to their indices.

2.2.2. Cost Function

As mentioned above, the main aim of the BA is to achieve the reference trajectory in the best (optimal) way. In other words, the algorithm must minimize the difference between the required and the real value. Therefore, it is appropriate to use a cost function in the form

$$CF_k = \sum_{n=k+1}^{k+N} (w(n) - y(n))^2, \quad (2.9)$$

3. Algorithm Simulation

3.1. Selection of the process model

A linear continuous mathematical model was chosen from [3] (in Czech). It is the SISO system and it is given by a transfer function (3.1). This continuous model was converted into discrete-time model (3.2) with sample time $T = 0.05\ s$ using the MATLAB function.

$$G(s) = \frac{s+230}{(s+9)(s+11)} = \frac{s+230}{s^2+20s+99} \quad (3.1)$$

$$G(z) = \frac{0.2378z + 0.1183}{z^2 - 1.215z + 0.3679} \quad (3.2)$$

The model to be used in the control system design is taken to be a state-space model. By using a state-space model, the current information required for predicting ahead is represented by the state variable at the current time [4]. A SISO system can by described by

$$\boldsymbol{x_m}(k+1) = \boldsymbol{A}\boldsymbol{x_m}(k) + \boldsymbol{B}u(k) \quad (3.3)$$

$$y(k) = \boldsymbol{C}\boldsymbol{x_m}(k) + \boldsymbol{D}u(k) \quad (3.4)$$

where $\boldsymbol{x_m}$ is the state variable vector. Using the MATLAB function, discrete-time model (3.2) was converted into a state-space model with the state variable vector $\boldsymbol{x_m}(k) = [x_{m1}(k), x_{m2}(k)]^T$ and with the system matrices

$$\boldsymbol{A} = \begin{bmatrix} 1.215 & -0.7358 \\ 0.5 & 0 \end{bmatrix}, \ \boldsymbol{B} = \begin{bmatrix} 0.5 \\ 0 \end{bmatrix}, \quad (3.5)$$

$$\boldsymbol{C} = [0.4757 \quad 0.4732], \ \boldsymbol{D} = 0 \quad (3.6)$$

3.2. Simulation of Predictive Control

In this part of the work, several simulations of predictive control of the process are described. The model for simulations was chosen and transformed into a state-space model in the previous chapter.

All simulations were performed with the same parameters of the BA. The most important parameters are: prediction horizon $N = 10$, number of bats $n = 80$, number of generations $N_{gen} = 10$. To test the control process using the designed Bat Algorithm for MPC, two reference trajectories were chosen. The first reference trajectory is a sequence of constant pulses.

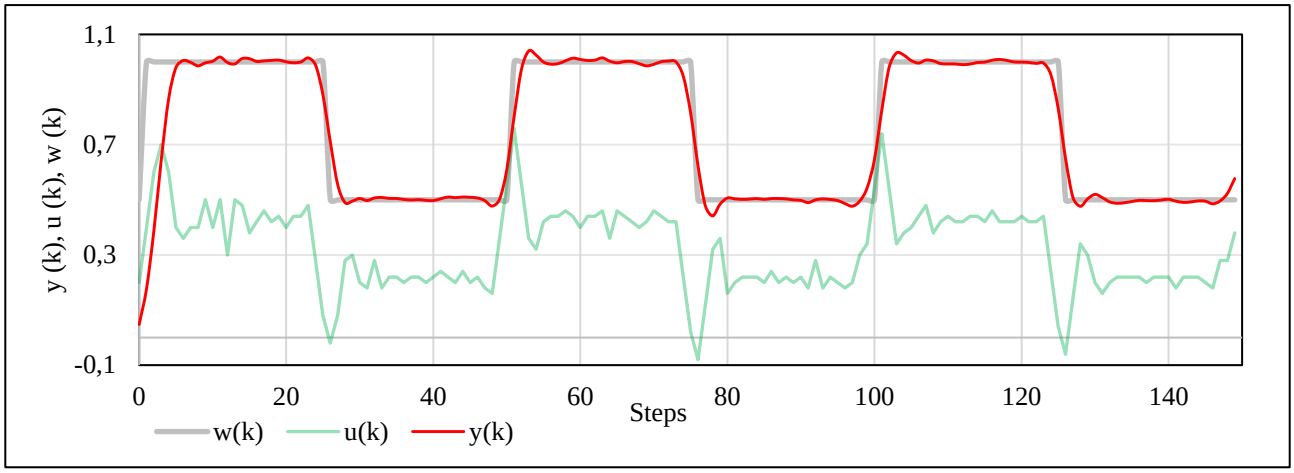

Figure 3.1. *Control process with w(k) as constant pulses*

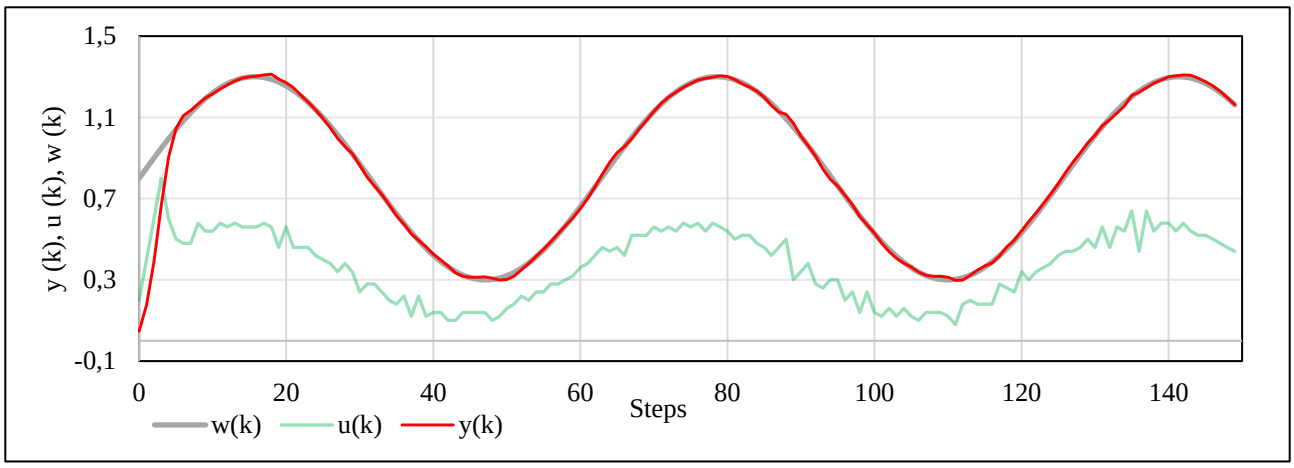

Figure 3.2. *Control process with w(k) as sine-graph*

The control process of the system is shown in Fig. 3.1. The second reference trajectory is the sine-graph and the process is shown in Fig. 3.2.

A very important criterion for estimating the algorithm of predictive control is the time for performing calculations in each timestep. This criterion allows not only to evaluate the algorithm, but also to determine the processes in which it can be used. Measured average cycle time during the simulations was *0,35* seconds.

4. Conclusion

The aim of this project was the study the Bat Algorithm, design the algorithm for Model Predictive Control based on the BA and simulation of the algorithm with a linear Single-Input-Single-Output system.

According to the results in the previous section I can conclude, that the Bat Algorithm for MPC proved to be excellent in all test performed. During the tests it was proved that the designed algorithm quickly searches for an optimal solution and optimizes this solution in every timestep. This property of the algorithm is well displayed in cases where the reference trajectory is not constant. Due to prediction control of the system, the BA starts to change the control variable before the reference trajectory changes.

Up to this point, within the project, the work and testing of the algorithm was simulated only. But based on the results of simulations, we can assume that the designed algorithm will work well with real processes. The use of BA in real laboratory work is one the next stage of the project. Another stage of the project is the simulation with non-liner systems, because the behavior of the algorithm with more complex, non-linear systems can be very interesting.

Acknowledgement

I would like to thank my supervisor prof. Ing. Milan Hofreiter, CSc for his guidance during the working with this project.

This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/210/OHK2/3T/12

Symbols

A	loudness (dB)
A_o	original value of loudness (dB)
A_{min}	minimal value of loudness (dB)
CF	cost function (-)
f	pulse frequency (Hz)
$G(s)$	continuous transfer function (-)
$G(z)$	discrete transfer function (-)
n	number of bats (-)
N	prediction horizon (-)
N_{gen}	number of generations (-)
r	rate of pulse emission (-)
u	input of the system (-)
u_p	predicted input of the system (-)
w	reference value (-)
y_p	predicted output of the system (-)
y	output of the system (-)
λ	wavelength (m)

References

[1] CAMACHO E.F., BORDONS C., "*Model Predictive Control. 2nd ed.*," 2007. [Online]. Available: http://een.iust.ac.ir/profs/Shamaghdari/MPC/Resources. [Accessed 12 4 2017].

[2] YANG Xin-She, *Nature-inspired metaheuristic algorithms. 2nd ed.*, Frome: Luniver Press, 2010.

[3] T. BAROT, *Prediktivní řízení procesů s rychlou dynamikou*, Zlín: Univerzita Tomáše Bati ve Zlíně. Fakulta alpikované informatiky. Automatické řízení a informatika, 2016.

[4] L. WANG, *Model Predictive Control System Design and Implementation Using MATLAB*, London: Springer-Verlag London Limited, 2009.