Soft Computing – project
ANFIS Application with adaptive control
2025/2026

Natália Ivanisková, xivanin00

1 Introduction

This is the implementation of the ANFIS with adaptive control for adaptive thermostat controling room temperature based on other changing factors.

1.1 Project description

Practical application of the selected algorithm: The algorithm is implemented with regard to solving a specific problem. The handling of inputs and outputs, as well as any potential interactivity of the program, is adapted accordingly.

1.2 Project tasks

Project consists of several consecutive tasks:

- Implement ANFIS controller for adaptive control of the room temperature considering other suitable factors influencing temperature in the room.
- Generate data and perform numerous test to find the best parameters setting for learning rates and number of rules regarding appropriate system characteristics such as target and output temperature difference and use of the radiator valve.
- Present behaviour of the system on the predefined scenario of temperatures and other events influencing temperature in the room.
- Create app to demonstrate adaptive behaviour of the system reacting to user controlled changes in the system.

2 ANFIS Controller

This projects implements ANFIS algorithm, that is modified for adaptive control.

2.1 ANFIS Algorithm

Adaptive-Network-Based Fuzzy Inference System (ANFIS) is a system, that combines neural networks and fuzzy logic to represent complex non-linear relationships in data inputs and outputs. Fuzzy logic is used to model human-like descriptions of data in linguistic terms (such as "low", "medium", "high"). Parameters of the membership functions and parameters of the fuzzy rules are optimized by neural networks [1]. ANFIS architecture can be divided into 5 layers:

1. **Fuzzification** - Each input is assigned a membership value, which is a conversion from sharp to fuzzy values. Some examples of the membership functions are Gaussian MF, Triangular MF, Trapezoidal MF.
2. **Rule Evaluation** - Each node corresponds to a fuzzy rule and calculates firing strength of the rule based on the fuzzy membership values. Firing strength is calculated as a product of the fuzzy membership values.
3. **Normalization** - Firing strengths are normalized to sum to one, which makes it a probability distribution.
4. **Defuzzification** - Calculation of consequent parameters weighted by the normalized firing strengths.
5. **Output calculation** - Final output is calculated as a sum of weighted outputs of all rules.

Standard offline ANFIS training consists of the optimization of the membership function parameters and consequent parameters (fuzzy rules output calculation). Training steps are forward pass to calculate the output followed by the backward pass to estimate the difference between output value and target value. The last step is the backpropagation to update membership functions parameters using gradient descent and consequent parameters update using least squared errors estimation, which is known as hybrid learning [2].

2.2 ANFIS with adaptive control

ANFIS with adaptive control differs from the trained ANFIS model by the parameters optimiyation step. For classification or regression task, model is trained on the dataset and then it is used for classification or prediction. However, in adaptive control use case, online learning is utilized. With every new input, output of the ANFIS model is compared to the predefined target value. Subsequently, difference between the output value and the target value is calculated and parameters (weights of the fuzzy rules and consequent parameters) are updated in the backward pass. Next input is processed by the model with parameters updated in the previous step.

3 Implementation

ANFIS controller was implemented for the control of the room temperature that is influenced by outside temperatures and events of the window opening. To transition the theoretical ANFIS model into a robust practical controller, several engineering enhancements were implemented.

3.1 System modeling

Thermodynamic simulation of a room was implemented to demonstrate and validate the controller. The system dynamics are modeled based on a Newton's Law of Cooling in a discretized version to model system thermodynamics:

$$T\_inside\_new = T\_inside + (heat\_gain - heat\_loss) \tag{1}$$

Where:

- Heat Gain - Calculated as valve position multiplied by a predefined coefficient representing heater efficiency (0.6).
- Heat Loss - Calculated as the temperature difference (T_inside - $T\_outside$) multiplied by predefined coefficient representing insulation loss (0.1) Insulation loss coefficient is modified in case of the open window event (0.2).

Room system simulation operates with the following parameters:

- $T\_outside$ - outdoor temperature,
- $heater\_power$ - heater valve position $(0 - 5)$,
- $window\_state$ - window state (Open/Closed).

Room system simulation considering temperature changes and window events results in calculating new indoor temperature (T_inside). ANFIS algorithm, that poses as adaptive thermostat, operates with the following:

- **Inputs**:
  - ***system_error*** - calculated as the difference of the current room temperature (after considering heat gain and heat loss factors),
  - ***T_outside*** - outside temperature (generated from a sine wave for simulation),
- **Outputs**:
  - ***heater_power*** - estimated radiator valve level to reach T_target in the room.

New radiator valve level ($heater\_power$) will be taken into consideration in the next step of the simulation and adjusted according to the situation.

3.2 Adaptive Control Strategy

However, to ensure computational efficiency for real-time iterative control, this project implements **Gradient Descent for all parameters** (both antecedent membership functions and consequent linear weights). The parameters are updated after every time step based on the instantaneous system error.

3.3 Simulation Loop

The simulation implements updates of the temperatures from sensors and updates of the radiator valve at various frequencies:

- Fast Loop (1 min): Indoor temperature, error calculation and radiator valve level are updated every minute.
- Slow loop (30 mins): The outdoor temperature "sensorüpdates every 30 minutes. The controller uses zero-order hold strategy and uses last known valid outside temperature for calculations.

3.4 Exceptional manual supervision

To simulate real world scenario, “supervisory rule” was implemented outside the neural network. If the window state is OPEN, the radiator valve is manually set to chosen level (for example 2.0). To prevent the model from learning erroneous relationships (e.g., trying to compensate for infinite heat loss), ANFIS adaptation is paused during this state and renewed once the window is detected as CLOSED.

3.5 Stability and robustness enhancements

Since simulations in the early stage of the development exhibited instability (such as valve oscillation, that were followed by large disturbances and therefore to prevent “exploding gradient” problem, two specific mechanisms were implemented to stabilize the adaptive learning:

- **Gradient Clipping:** During the backpropagation step, the error term used for weight updates is mathematically clipped to a range of $\pm 2.0°C$. This serves as prevention of the weights from extremes during transient situations (such as immediate window closing and sudden changes in the room system).
- **Insignificant deviation toleration:** Adaptive step is skipped when the absolute error is within a small threshold ($\pm 0.1°C$). During steady and stabile state of the room system, controller is prevented from overfitting to unreasinably small differences from target value.
- **Realistic Output Modification:** The output is hard-clipped to the physical limits of the radiator valve (0 to 5).

4 Testing Methodology and Optimization

To ensure the controller’s robustness and to find suitable hyperparameters ($learning\_rate$ and n_rules), numerous tests were performed and evaluated.

4.1 Monte Carlo Simulation

System was tested with Monte Carlo simulation with 100 unique runs. For each run were generated randomized environmental conditions, including varying outdoor temperatures ($0°C$ to $15°C$) and randomized window event durations.

4.2 Grid Search Optimization

To identify the optimal hyperparameters, a grid search strategy was employed. The performance was evaluated with various combinations of configurations:

- **Number of rules** (n_rules) - $[3, 5, 7]$
- **Learning rate** ($learning\_rate$) - $[0.001, 0.005, 0.010, 0.050]$

Two key metrics were recorded:

- **RMSE (Root Mean Square Error)** - to measure accuracy,

- **Valve Chattering (Variance)** - to measure system stability.

Outputs of all independent runs were aggregated and analyzed. Results are in table 1

Rules	Learning_Rate	RMSE	Chattering
3	0.0100	1.6787	0.3005
5	0.0100	1.6844	0.1856
3	0.0050	1.6925	0.1710

Tabuľka 1: Results of the grid search hyperparameters settings testing (top 3 configurations)

According to the test results, best trade-off between accuracy and stability is with the following hyperparameters values:

- $\boldsymbol{learning\_rate} = \mathbf{0.01}$
- $\boldsymbol{n\_rules} = \mathbf{3}$

Other user-modifiable parameters were set to these values:

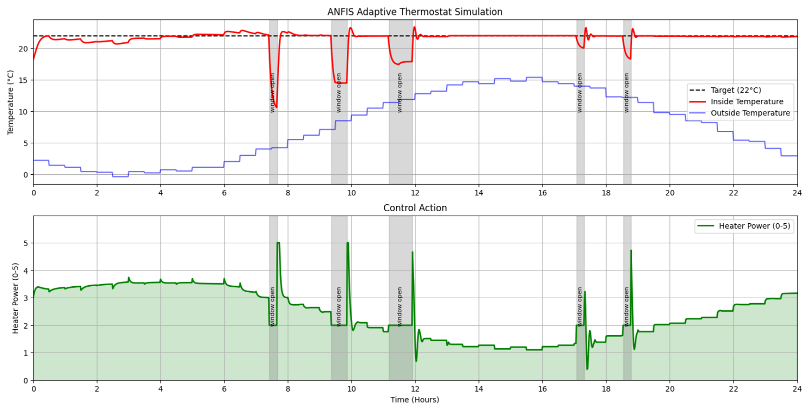
- $T\_target = 22.0$
- $T\_inside = 18.0$ (initial inside temperature)
- $initial\_heater\_power = 3.0$
- $heater\_power\_when\_window\_open = 2.0$

5 Final system application

Final version of the ANFIS system is demonstrated in static and dynamic modes.

5.1 Static Simulation

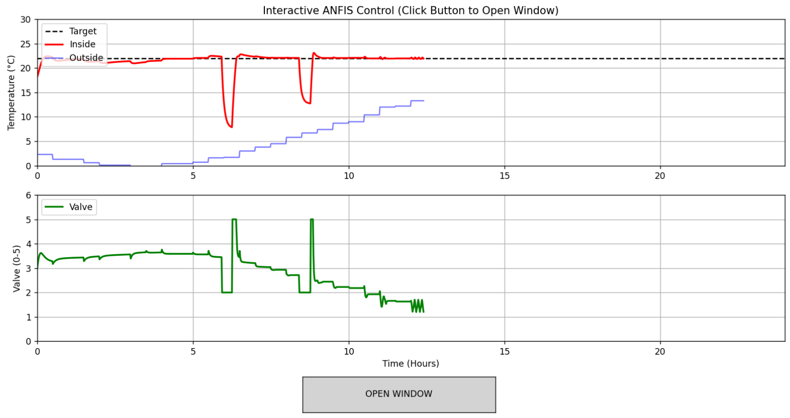
Static simulation uses pre-defined 24-hour scenario including outside temperature changes and window events. The controller demonstrated smooth valve operation during steady-state periods and rapidly increased heating power following planned window events, while keeping heater valve at the predefined level when window was set to OPEN.



Obr. 1: ANFIS controller application in static mode

5.2 Interactive Dynamic Simulation

To test the "online" adaptive capabilities in a real-time context, an interactive graphical application was developed using `matplotlib`. The application renders the system state at 10ms intervals. A GUI button allows the user to toggle the window state upon request. Changes in the system and systems adaptive response are observed in real-time simulation.



Obr. 2: ANFIS controller application in dynamic mode

6 Program usage

6.1 Installation

Dependencies are installed via following command:

```
bash install_dependencies.sh
```

6.2 Program modes usage

Final application operates in 3 modes (operated from `src` folder):

- `test` - generates data for simulation, runs test script and performs results analysis as described in 4. Results are writen on `stdout` and pictures are showed in the pop-up windows. Usage:

  ```
  python generate_data.py
  python main.py -mode test
  ```

- `static` - Static simulation with the predefined scenario from generated data. Simulation is shown in the pop-up window. Usage:

  ```
  python main.py -mode static
  ```

- `dynamic` - Interactive simulation of the system with the predefined outside temperature changes, but with the user controlled window events. Simulation is controlled in the pop-up window. Usage:

```
python main.py -mode dynamic
```

7 Conclusion

This project successfully demonstrated the application of ANFIS for adaptive control. By combining the linguistic interpretability of fuzzy logic with the online adaptability of neural networks, the controller achieved tight temperature tracking. Additional stabilitz=y and robustness mechanisms were implemented. Final app works in static mode with predefined scenario and in interactive mode with user control.

Literatúra

[1] Jang, J. S. R.
ANFIS: adaptive-network-based fuzzy inference system.
IEEE Transactions on Systems, Man, and Cybernetics, vol. 23, no. 3, pp. 665–685, May/June 1993.
Dostupné z: `https://ieeexplore.ieee.org/document/256541/`

[2] Mazraeh, A.
ANFIS (Adaptive Neuro-Fuzzy Inference System).
Medium, Feb 5, 2025.
Dostupné z: `https://medium.com/@adnan.mazraeh1993/anfis-adaptive-neuro-fuzzy-inference-system-abbba3ecb68f`