

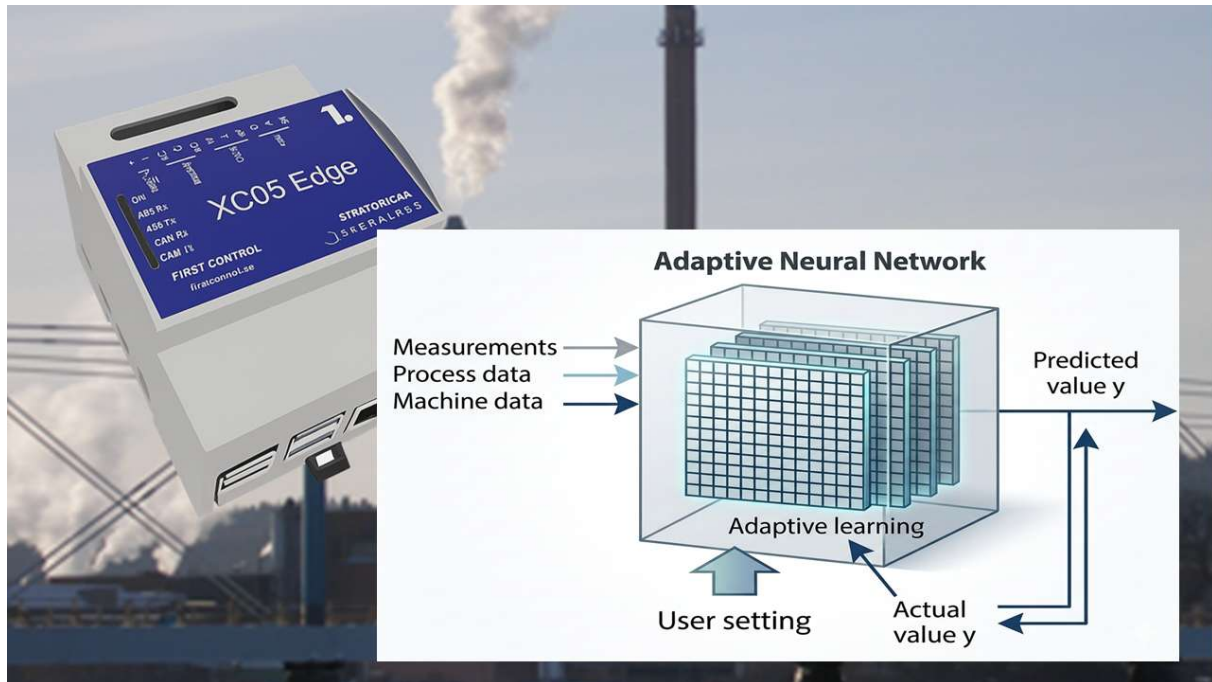
Self-Tuning AI for Industrial Automation

Practical application of Neural Networks

By

Gunnar Bengtsson, PhD, CEO First Control systems AB

Joakim Örnescans. M.Sc. Inventor of XC05Edge



Introduction

Adaptive controllers and AI have different roles in an automation system. The adaptive controllers provide the direct control: damping signal variations, keeping the process stable, and maintaining good control performance in both steady-state and transient operation, even when the process dynamics change.

AI can be used as a supervisory layer to describe nonlinear relationships, interpret and exploit large amounts of operating data, and optimize how the process should be run. At the same time, there are still very few openly reported examples where AI is used in actual operation in real industrial processes.

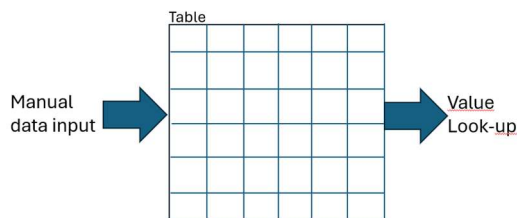
We have used AI methods in a number of installation projects to solve concrete problems in physical processes. For this purpose, we developed an adaptive neural network intended to optimize operation at major production changes or during start-up. The method can be implemented in the control system as a standard function and used by a process engineer in the same way as a controller is used today. For the adaptive

part, we have used the powerful self-tuning concept originally developed by researchers at Lund University under the leadership of Professor Karl Johan Åström.

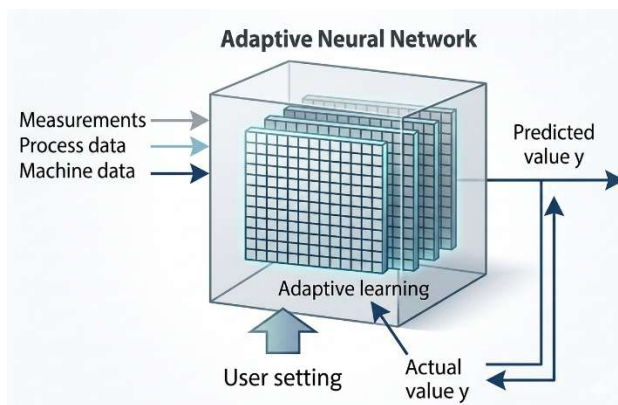
The technology has been used by First Control in real industrial installations, including rolling mills, Zn coating, and optimization of Kaplan turbines, with very good results in the applications where it was put into continuous operation.

Method – engineering description

Engineers typically use a table to represent a nonlinear relationship or a curve in a computer. If one wants to describe a relationship $y = f(x_1, x_2)$, where x_1 and x_2 define the axes of a two-dimensional table, the table cells can then be filled with measured or estimated values.



For multidimensional relationships, this is generalized to a grid in N dimensions. Our idea was to automate this approach by replacing manual filling with an online-learning algorithm. Instead of each grid point containing only a table value, a local model is stored at each grid point and updated using a recursive identification method as new measurements arrive. Because this also includes a forgetting factor, the network will also adapt to changes in the process. The system then acts as an adaptive table: the table structure is defined by the user, but the contents are updated automatically from process data.



When a new operating point is observed, the model error - i.e., the difference between predicted and measured value - is computed. The error is then distributed to the surrounding grid points, where the models are corrected to reduce the error. The model

update uses the same algorithm as we are familiar from the self-tuning controller, supplemented with a weight for each point.

The method is intended to be generic within the control system - a module in the same way as a controller. The user specifies certain basic data:

Parameters

DIM	Network dimension
Ni	Number of points on each axis
Xi	Points along each axis
N	Network order, i.e., the number of parameters at each grid point
TIME	Forgetting factor
RESyi	Resolution of the input signals
MAX	Maximum number of grid points allowed

The network can then be controlled via the signal inputs specified by the user in the application:

- ON = Network on/off
- ADAPT = Adaptation on/off
- DUMP = Save network data
- LOAD = Load network data
- RST = Reset network

At each grid point, the identified model parameters and the state of the recursive identification algorithm are stored, so that the algorithm can take the next step when the model is to be updated.

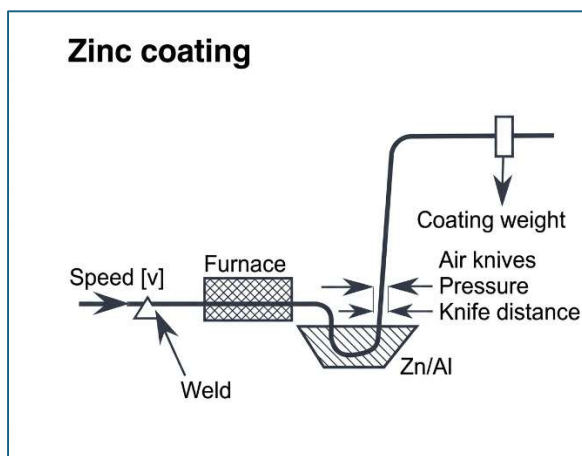
An especially important property is that the recursive identification algorithm at each grid point includes an estimate of the model uncertainty. This means that different grid points are updated to different extents: points with high uncertainty change more, while well-established points change less. In this way, a practical operating issue is handled - namely that some regions of the variable space are visited often while others are visited more rarely.

The method therefore differs both from an ordinary static table and from more conventional neural networks. Compared with a table, the model is not filled in manually but is updated continuously during operation. Compared with typical neural networks, the structure here is physically defined, because the user selects the axes and grid points. Distance and proximity in the network therefore acquire a direct process meaning that the process engineer understands.

Optimizing a Zn-coating line

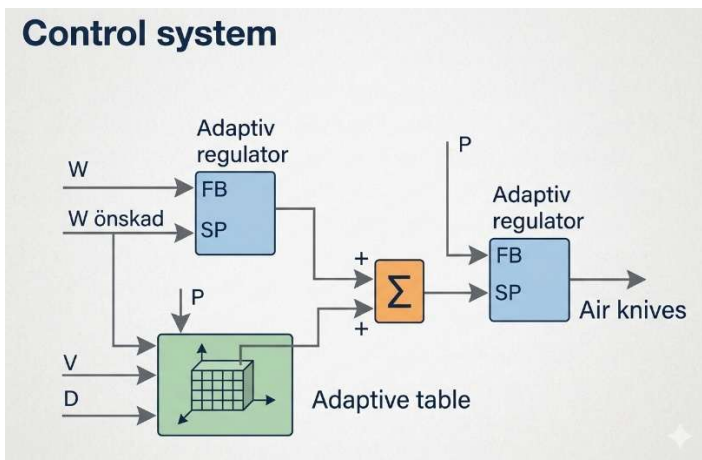
The adaptive neural network is generic and can be applied to many types of processes. The following example from a Zn-coating line illustrates how the technique can be used to reduce the amount of scrap. The principles are also applicable to similar problems in other processes.

Schematic process description



A steel strip passes through a bath of molten Zn. After the bath, the excess is blown off using air knives so that the correct coating weight is obtained. The control system is designed to keep the knife pressure (P) and Zn weight (W) at their desired values. The controllers are our STREGX2 standard which is described in another article.

Control system



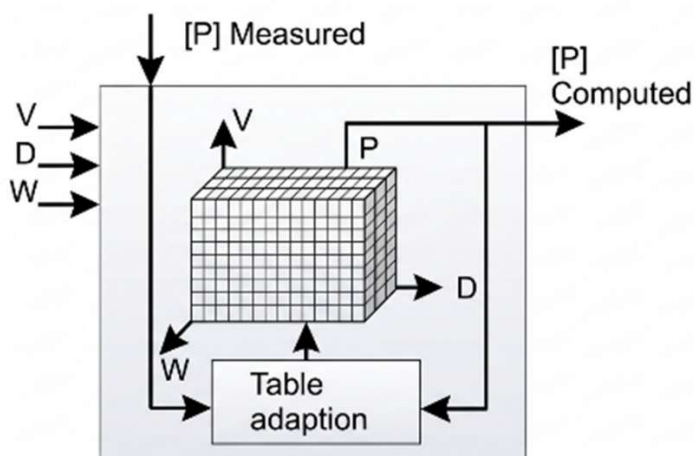
During production, changes may be made to line speed (V), knife gap (D), and desired Zn weight (W), due to planned or unplanned events. After each such change, the actual Zn

weight will change, and the local control system will, after some time, adjust the knife pressure until the desired Zn weight is reached again.

Because the weight measurement is located about 40 meters after the knives, a significant delay occurs. This means the control system needs a distance of at least 100–150 meters before the process is back in the correct state. Each changeover therefore risks turning a corresponding strip length into scrap.

The adaptive neural network

The purpose of the network is to minimize the amount of scrap during changeovers. The principle is that the network learns which knife pressure (P) is required to achieve the desired Zn weight (W) for given values of the production parameters



Once the control system has settled after a changeover, the pressure that is actually required is recorded in the network. After some time in operation, the network has learned the relationships and can, for subsequent changeovers, adjust the knife pressure directly - without first waiting for the slower feedback via the weight measurement to take effect. In this way, hundreds of meters of strip can be saved in situations that would otherwise have resulted in scrap.

The line is continuous, meaning that different strips with different desired coating weights are welded together. The weld is detected before the new strip reaches the Zn bath, and the control system can therefore track exactly when the new strip enters the process. In this case, too much Zn is acceptable, but not too little. To save material, one therefore wants to run as close as possible to the lower tolerance limit without falling below it.

This has an important consequence during changeovers. If the new strip is to have a thicker layer, the coating must be increased in advance to avoid falling below the tolerance. If the new strip is instead to have a thinner layer, the changeover can be made afterwards without the same risk. With the neural network, the changeover could be timed so that the knife pressure was adjusted at exactly the right moment.

Follow-up

The adaptive neural network ran for more than ten years without problems, until the production line was shut down. Production staff noted how accurate the changeovers became. Previously, a model provided by the supplier had been used, but it gave significantly worse results.

Of course, we could not explain the mathematical background in detail in the operating environment. However, it worked well to describe the method as a “table” that learns the relationships during operation, according to the principle described earlier in this article.

Distribution of updates within the network

In the beginning we had difficulties with how to compute the weights that govern how the update is distributed to neighbouring grid points in the network. Using optimization methods of the kind that are common today was completely out of the question. The solution had to be implemented locally in the control systems of the time, with only a fraction of the computing and memory capacity available in modern systems.

We therefore developed our own method based on the convex hull, where the weights could be computed using simple arithmetic. The method provided a unique weight distribution to $(N-1)$ surrounding grid points in an (N) -dimensional system. Because the axes had a clear physical meaning and were defined by the user, I do not see this as a limitation - rather the opposite: the structure becomes understandable and directly linked to the process.

Generalization of the method

There are several ways to generalize the method. In our installations, we primarily assumed static nonlinear relationships. This can be extended directly to also describe nonlinear dynamic relationships by using local linear dynamic models at each grid point, which are then updated in the same way as above. This has a natural physical connection, because nonlinear dynamics can often be well approximated by linear models near an operating point - which is also a central starting point for linear control theory.

A further extension is to let the weight calculation adapt to measurement quality, for example by taking account of noise level, signal-to-noise ratio, or estimated uncertainty in different signals. Since the basic idea is to develop mathematical methods to solve a practical technical problem, rather than the other way around, such a further development appears entirely reasonable.

