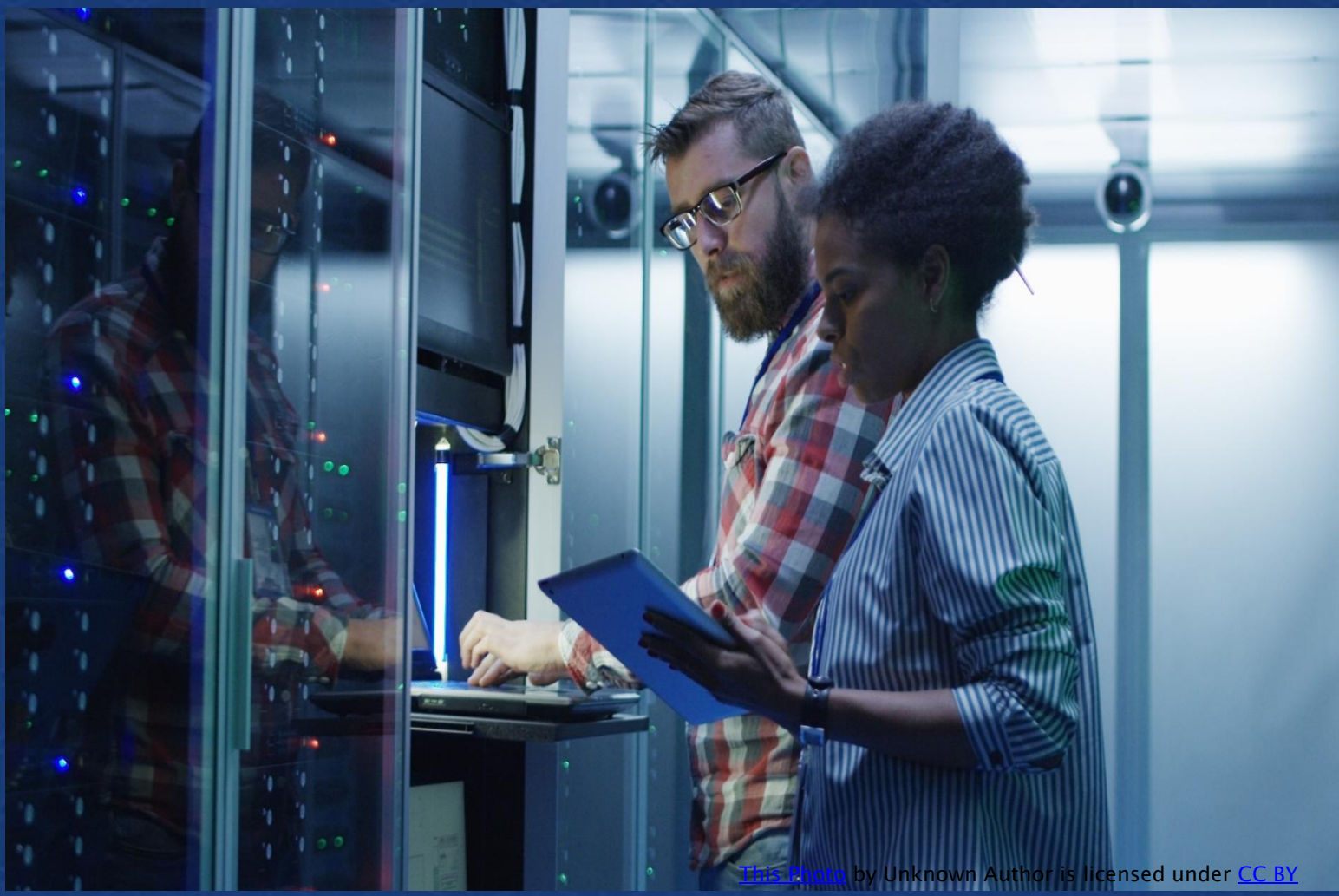
# Journey to a Digital Factory III

## Automation and the Digital Factory.



This Photo by Unknown Author is licensed under CC BY

# Introduction

Is there much difference between Automation and Digital Factories? Defining the differences, or (perhaps more importantly) recognizing how automation is a key component of a digital factory, can be helpful when thinking about moving forward incrementally and keeping a digital factory project manageable.

Automation is the piece of a digital factory focusing on speed and repeatability of process steps. A Digital Factory is a more encompassing definition covering the transformation of the process for better overall results which include automation.

While there are several types of automation as described below, this paper will focus on an automation level we call adivarent control (Tuttle, Alagarsamy, & Radl, 2023a, p. 12), which we consider a bridge between traditional automation and a full digital factory. This added layer between level 2 and 3 of the Purdue Model (ISA-95) control hierarchy provides operating flexibility while supporting the operator and best practices. This bridging layer can be challenging and is the reason so many analytical tools do not attempt to actualize the knowledge in real-time.

# Automation Overview

Automation is going to be familiar to most of the readers here, but let's just do a quick recap of the main reasons and types.

The pressure to automate a task or process can come from many directions and often results in unintended benefits and issues. The results intended may be to improve working conditions (or remove the worker from a hazardous condition), worker shortages, labor costs, limits on throughput, limits on quality, etc.

The types of automation include:

*Fixed Automation* – where a task or process follows a preset set of rules or steps to create a specific product. Normally, this term is applied to physical production processes. Some examples include automated conveyor belts to transfer sorted parts, assembly machines, and/ or automated chemical processes (batch and continuous).

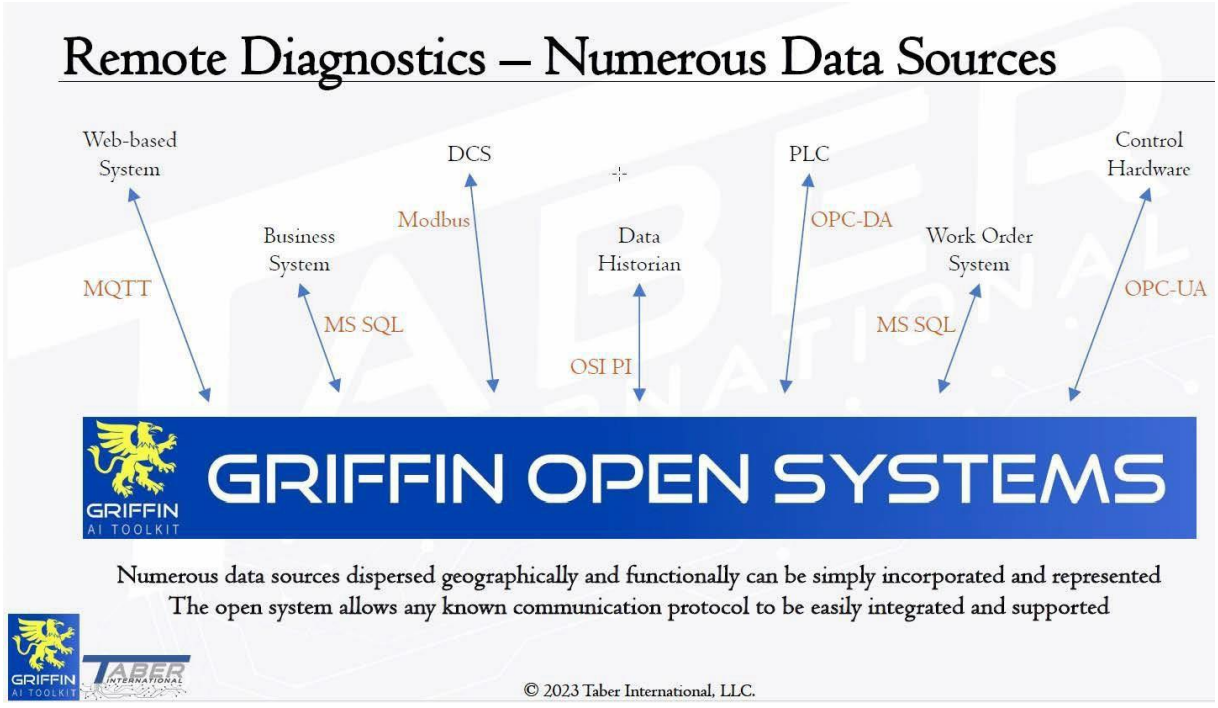


***(picture from https://encompass-inc.com/tag/fixed-automation/ website)***

*Programmable Automation* – is designed to allow for rapid changes in the production process. Examples include re-configuring equipment automatically to produce a different part, or adjusting the chemical feed rates to create a product such as paint. CNC (computerized numerical control), PLC's (programmable logic controllers) and industrial robots, are the main equipment used to achieve this type of automation.

*Flexible Automation* – Programmable automation uses flexible machines which can be set up and programmed to run different operations or sequences of steps. Flexible Automation takes this one step further and are integrated into a production system. HMI's (Human Machine Interfaces) are used to initiate changes in the production process. This further reduces the time between production runs for various products and allows a closer link between production scheduling and the actual production systems.

*Integrated Automation* – is a relatively new term reflecting an aspect of the digital factory which requires the integration of each of the above types of automation. Each automation type has advantages and disadvantages. For example, if you are producing 10 million of the same widget, you would not need the added cost for a programmable automation system. However, if the widget needs to be painted different colors, then that system may benefit from a programmable automation system. At this level of automation, systems need to talk to each other and islands of automation need to have bridges built to allow a continual transfer of information among the systems.




This paper will focus on the integrated automation aspect and working toward a digital factory implementation. The goal is a better understanding of the opportunity to be realized with a good digital factory implementation strategy and to cover strategies used to create successful solutions.

## Incremental Approach

As discussed in our earlier papers, tackling projects using incremental steps can help remove inertia that prevents projects from getting implemented in a timely fashion. But much like writing a paper, it is a good idea to have at least an outline of where you want to go with a project and at least the known steps that will need to be addressed along the way. Rewrites will likely be necessary, hence, waiting until all is figured out can be an major impediment to starting a project.

The earlier papers covered steps for agglomerating data, data conversion, data pre-processing, and overall data quality (Tuttle, Alagarsamy, & Radl, 2023b, p. 12).This was followed up by a discussion on dashboards and making the data visible to users (Tuttle, Alagarsamy, & Radl, 2023b, p. 12). Those steps are critically important and often represent the majority of time needed in a successful automation project. If these are not done correctly, especially with regard to data cleanup, then the automation part will be severely stressed to succeed. However, it is important to consider that part of the automation step is picking small steps, allowing the user to validate and verify that the data is cleaned up in a manner needed for proper application. It is highly likely situations will occur which were not foreseen and this will lead to adjustments which will provide additional improvements in how data is pre-processed. Ultimately, the best value is often achieved when you can turn the data into information used in a system acting automatically in response to process changes. Hence, it is important not implement too many steps of the automation all at once, so it is much easier to identify and correct issues.

## Operator Acceptance

An area commonly overlooked is making sure the system is setup and behaves in a manner leading to operator acceptance. There is no benefit to having a system improve efficiency by x

% only to have it continually turned off because of operator resistance. This can come from many directions, including but not limited to: training (not understanding why a system behaves differently), tradition (this is how we have always done it), control (feeling they are less in control of the process), mismatched goals (management want to optimizes x, but operator is compensated on y), job security (this is just one more step of taking away my job).

In addition, a particular area which is important in most projects is the impact of the automation project on other systems. It is common for the tuning of one loop or sub-system to impact other control loops. In this case, the automation system must be adjusted to find a good balance between the loops and sub-systems. Our experience has found that this is also a good time to look at some root causes on these interactions, as quite often one or more loops can be improved by optimizing how they interact. This is especially true when making a process more efficient and eliminating unwanted side products or wasted energy. All wasted products/ energy must go somewhere impacting those parts of the process they touch.

## Operation Acceptance, cont.

A second common area involves nuisance alarms. Alarms are an area which should lead to greater operator acceptance, and they must be evaluated in light of your current control objectives. If an alarm was set with a 10% margin to ensure avoidance of something bad happening (e.g. an emission alarm at 90 ppm to avoid occurrences of 100 ppm) and yet the goal is to get as close to 100 ppm for best efficiency. and the control has been improved to where the system may operate at a 5% margin with no problem, then the alarm needs to be changed to reflect this new set of circumstances(i.e to alarm at 95 ppm not 90 ppm). If alarms are occurring because of changes being made, they may temporarily bump another process into alarm. The opportunity then exists to include the interaction in the logic and optimization to avoid those conditions (e.g. having a dynamic alarm limit between 90 and 95 based on the system is interacting with). The result should be a net reduction in alarms, which tends to make everyone happier or at least more content.

## Automating the Mundane Example

Automating mundane tasks is often considered basic, as the best operators seem to be able to cruise through the day with the fewest problems. When implementing, it is important to bring the operators into the process early, both to inform/confirm the goals of the system enhancement, and to seek their input on what areas would benefit them the most. The reason the best operators succeed is because they understand many of the nuances of different approaches and, therefore are able to better nudge a system and avoid non-obvious interactions. Unfortunately, DCS and PLC systems have limitations which prevent expansive decision networks to be implemented.

For example, we were approached on an Hg emission control project to lower the total PAC(Powered Activated Carbon) injection used in the control of the Hg emissions. Management had noted some shifts had only rare problems with the Hg and a concurrent issue of opacity, which was made worse with larger injections of PAC. Internally, the customer spend six months of implementing automatic strategies to bias the PAC to correct these issues and were not achieving the desired results. We were brought in to address this using the Griffin AI Toolkit®, which was already handling optimization on some related systems.

The problem seemed relatively simple. The hard goals were to keep sufficient margin on a 30 day rolling average emission rate, such that if bad coal arrived, they could stay in compliance without significant issues and not exceed a 6 minute average in opacity limit. Both of which would trigger fines for non-compliance.

There was a direct nearly-linear relationship between PAC injection rates and opacity. A non- linear relationship exists between PAC injection and Hg emissions whereby after a certain amount, adding more PAC has less impact on the Hg removal. However, factors such as coal quality and load demand affected the average Hg needing to be eliminated by the PAC. The opacity was affected by these, and several additional factors, such as equipment in service, soot cleaning operations, set points (such as excess air and windbox pressure), maintenance conditions of burners and previous mode of operation by earlier shifts. The Hg continuous emissions monitor (CEM) measures in the parts per billion (ppb) range making it difficult to accurately measure such

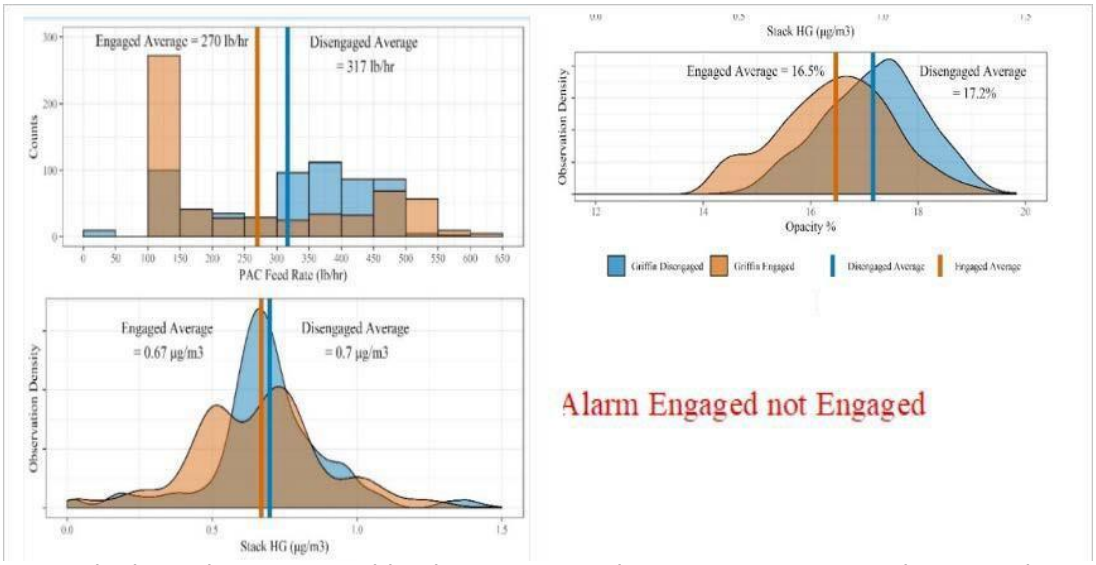
a small quantity without frequent calibration and maintenance of the CEM. This results in the PAC system running blind while the various factors impacting the Hg rate were changing.

All this was managed by the best operators; however, all of these factors eventually overwhelmed the ability of the DCS engineer to keep adding logic to handle the large number of conditions. In addition, the DCS did not have a good method for a predictive emission monitor (PEM) to estimate Hg values when the PEM was out of service. Nor was it easy to program an ability to test out different strategies using predicted results for the upcoming 30 days. In the case of evaluating strategies a key component is knowledge of what daily values would soon be rolling out of a 30 day emission average. If some previously bad fuel days were about to roll out, then the system had a lot more margin to work on any opacity issues; or conversely, if good days were about to roll-out it was very important to build up more margin in the 30 day average.

With the Griffin AI Toolkit no-code system the straightforward approach was to first sit down with engineering to lay out the goals, trade-off functions, and how to handle poor or missing data situations. Within a week, this quickly led to a reduction in average PAC usage. However, many operators were still inclined to use their previous “go-to” solution of turning off the PAC systems when opacity alarms were a problem. In the phase II implementation, we spend two weeks working with the operator who had traditionally managed the system in such a way as to avoid alarms and keep the PAC system on nearly 100% of the time.

The outcome from this session was interesting in that the operator had few worries about the Hg control aspect, but was more concerned about the 6 minute opacity alarm. It turns out he had good knowledge of the many nuances that might temporarily spike opacity, versus the longer term shifts in opacity requiring a different set of actions. In addition, he added a couple of strategies to keep opacity out of alarm; which, while they necessitated looking at the big picture across a few sub-systems, were highly beneficial when they were permitted to be implemented.

The resulting system was much more pro-active in avoiding opacity spikes. The system evaluated several options based on the current conditions and selected the one which best addressed the cause of the opacity spike. In addition, this strategy automatically optimized the margins available between Hg emissions and opacity using cost curves for incremental changes. This made operator training critical as the system behavior was different than in the past. This required both training on why control rates may adjust when nothing appeared to be wrong, or why the system did not over-react if there were temporary conditions impacting one system. In addition, the results of the prediction/estimating component of the application were made available to show the expectations of the impact on emission averages of the resulting changes to the controls (through bias inputs).




First graph shows better control leading to 10% reduction in PAC usage. The second graph shows the reduction in PAC usage corresponded with lower Opacity average (which in turn correlated with a 90% reduction of alarms. The third shows how the more strategic use of PAC even lowered the average Hg emissions, while accomplishing the first 2 goals. A net win for engineering and operations.

By keeping all parties informed as the new strategy was implemented, carefully monitoring and incorporating feedback from each of the departments, the system now is considered an essential aspect of operations. This incremental step took only two weeks to put into service, the payback was achieved in under 2 months. Combining with reduction in nuisance alarms with the savings in PAC usage made this a very successful incremental project towards their overall optimization goals.

The above example uses relatively straightforward rules and holds version of the AI by capturing the knowledge of the engineering staff and operators. Combining these more complex logic streams with enhanced modeling tools in the no-code environment made the implementation practical and quick.

The next steps require more advanced machine learning techniques from the Griffin AI Toolkit. These advanced techniques allow further integration of additional control elements to optimize both a fuel additive flow rates and the control of lime to maintain another emission targets.

## Optimizing the Complex

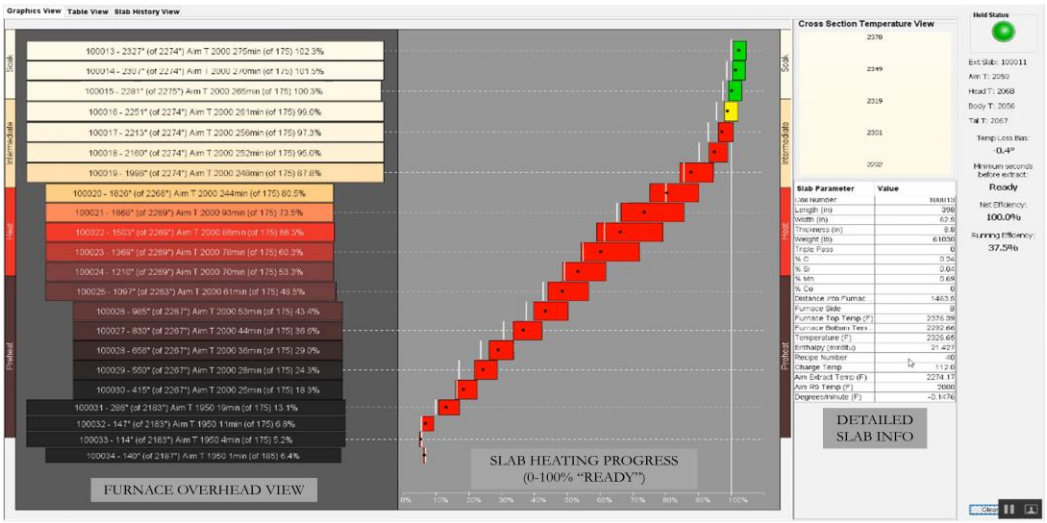
Even if the system is working perfectly for a complex process, other factors besides a good net result, will affect whether it remains in service. With more complex automation challenges, the inertia associated with 'this is how we have always done it' or 'I've not seen the process do that before, so it must be bad' are more prevalent. Training and proper information relayed to engineering and operations are key as optimization is may seem counter-intuitive.

For example, in a reheat process for a steel rolling mill. The logic seems sound that if slabs are struggling to reach the desired temperature before rolling, then cranking up the natural gas (NG) to the burners, especially near the exit, seems sound. However, when factoring in NG pressure

impact, NG flow limitations, NG non-linear valve response, bad burners and little information on a slabs heating history, this leads to rejects, scaling and a waste of NG. In this case, a digital twin modeling the heat flux in the steel allows for creating a strategic use of the NG for heating; and, more importantly a better understanding of the heat needed to reach the objectives if slabs in the reheat furnace have different heating objectives. The ideal solution may be to cut back NG flow near the slab discharge, and move that NG to earlier in the heating process, where it has more time to allow the heat to soak throughout the slab making the finishing heat more effective. As the process becomes more efficient, then other constraints from burners and the total NG maxing out are ameliorated, providing better overall control.

The challenge here is cutting back on NG flow seems illogical when you are faced with under-heated slabs. In this particular case, we found it was essential to add dashboards showing the estimated heat content and temperatures of each slab in the furnace, and color code them with regard to readiness to be discharged and processed. A automatic delay function was implemented as an alternative to more heat, as a cold interior was a cause of rejects and even greater delays.

Previously, the operator had no feedback on interior temperatures only a moderately accurate surface temperature at the first rolling station. With a dashboard display showing current status of slabs, the training on how heating could be done differently made sense as operators now had visual feedback of how the whole process was responding. It also better highlighted when a poorly matched set of slabs with different objectives was necessitating a trade-off on the optimal heating for each, so they could both be within the specifications.




Other areas of concern, such as, burner health, air-fuel ratios and achievable NG pressure at various burners factored into a complex set of models creating a digital twin of the reheat process. This digital twin provided data on unmeasured process values by a simulation running in real-time to quickly pick up variances in the reheat process. The values are used in the both the simulation and on operator dashboards. The digital twin was composed of both hard physics models of heat flux and machine learning models and on the correlation of this data to the sensor data available.

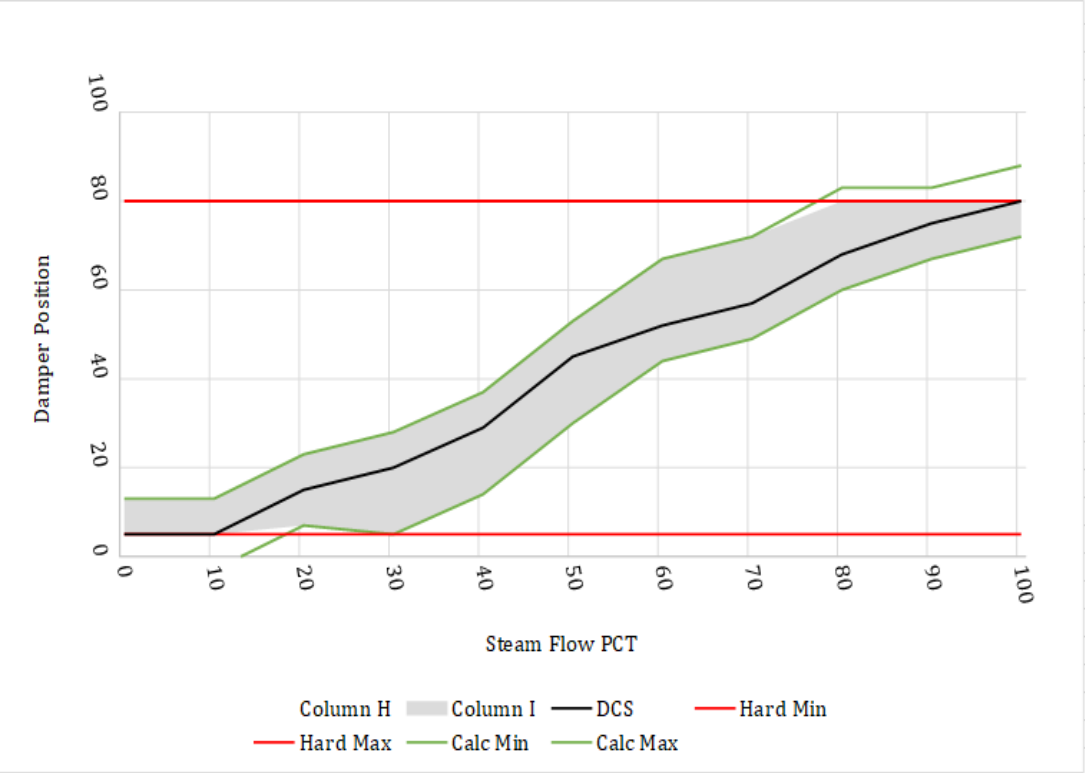
# Constraints - Modeling for Automation

As more sophisticated AI is used in automation, the challenges for a successful implementation can increase dramatically. Many of the challenges can be avoided through better use of various AI algorithms.

One common mistake is to throw all data at a single n-net with the expectation it will learn all data relationships. This rarely works beyond a small well contained system with very little data noise. Fortunately, application of good techniques make the situation both manageable and leads to improving results over time as more data is added.

The challenge in modeling, in particular with neural networks (n-nets), is if the model is complex, with hundreds of inputs and possibly a hundred or more control points to manipulate, it may need a huge amount of data patterns that may not exists to learn everything about the process.

For example, if you were modeling the combustion process on a utility scale boiler, the possible data combinations are huge. A large plant may have 8 coal mills. Each mill has air and fuel flow which are measured in the range from 0% to 100%. For starters, it makes a lot of sense to let the model know there are only 8 mills and the data range can only be 0% to 100%. Tasking the model to learn how many mills exist and what an acceptable expected range of data would be is unnecessary and creates a need to task the algorithm to not only learn what is possible but impossible. The latter while doable is impractical for most applications. When evaluating the data itself, it becomes clear that 8 mills are not run at low power and that mills only see low levels on fuel flow as they are turned off. All this knowledge if imparted ahead of time makes the modeling feasible. This is especially valuable, if you are expecting to combine the model(s) with an optimizer algorithm which is tasked with searching the whole space.




*Proper use of constraints limits the data area to learn.*

# Constraints - Modeling for Automation, cont.

As outlined above, constraints become critical to good practical model building as they limit the stress on the learning algorithm(s). It does this through:

1. Eliminating the need to learn data which will never be seen. This has a few benefits. The number of patterns to learn is reduced as the training algorithm only needs to learn and map real data. The neural network structure and activation functions are allowed to map against the real nature of the data and becomes less subject to extrapolation. This can be very critical around the edges of the process data space. For example, if an air damper is only going to run in the range of 40% to 60% which is highly linear, there is no need to figure out the air flow relationship as the damper approaches 0% or 100% where non-linear data as you approach 0% or 100% can create a need for a more complex model than really needed.
2. Constraints can also be used to break the comprehensive model into multiple smaller higher fidelity models. This is very handy if major equipment being turned on or off causes a step change shift in the relationships of the data being modeled.
    a. The challenge here is how many models the software and hardware can support simultaneously, as the number of models required go up by a factorial with the number of equipment or process variable points used to break up the model.
    b. The real-time software needs to seamlessly switch between models.
3. Finally, constraints can also help eliminate bad or confounding data, not suitable for the model training.

With the above in place and good models created, a neural network (and many other types of data driven algorithms) can be used properly in a real-time automation project.

# Constraints – Optimization Background

Models and optimization are often merged together, but in reality, there are two distinct parts of an automation project. While constraints can dramatically reduce the need for data which AI based models use to learn, they can also be further utilized to improve the efficiency and, more importantly, the acceptability of results for optimization.

Optimization algorithms use models to perform 'what if' calculations. There are a variety of algorithms for searching through the modeled data space to find the 'ideal' solution. The word ideal being in quotes, because rarely is the solution the absolute best, but something close to the best, as the ability to find the best is has many limitations, including but not limited to: time to search (computing resources, algorithm efficiency), noise in the data, gaps in the data and the fidelity of the modeling system to the current real-world conditions.

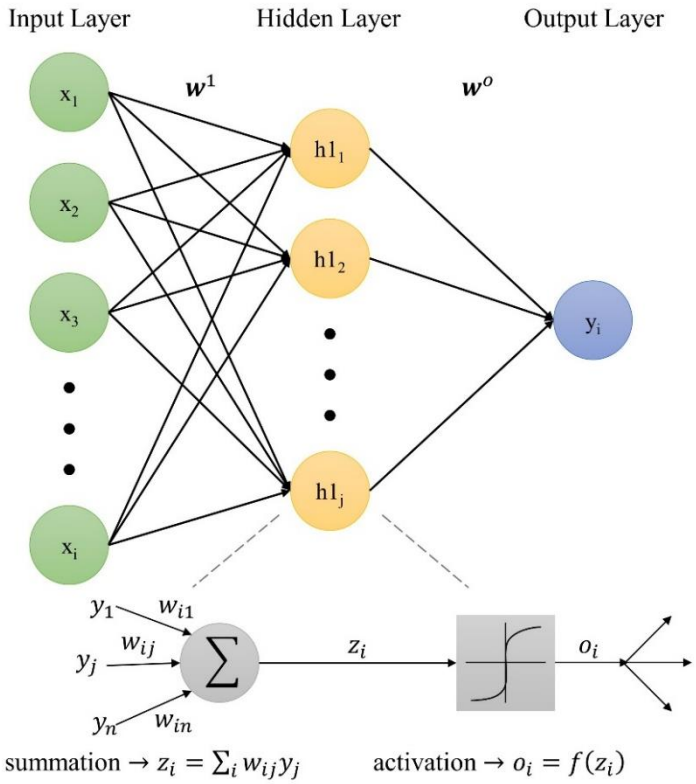
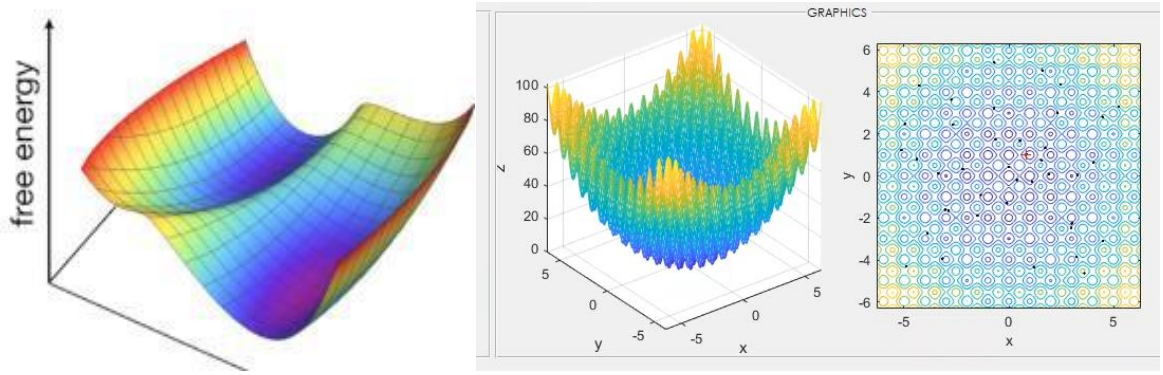



*Figure 1Neural networks process inputs through layers of neurons to produce specific outputs. Each neuron applies an activation function to scale inputs, and backpropagation adjusts neuron heights to minimize prediction errors.*

For purposes of discussion we will use the characteristics of a particle swarm optimizer (PSO). This is a good general purpose optimizer for when the data space has many interacting points and multiple objectives. This higher order data space often has a 'hilly' type of surface where there are multiple areas of good solutions broken up by areas of poor solutions. The PSO creates a number of possible solutions (particles) that are evaluated in an iterative process. The closeness (distance) to the objective is shared among the particles to help guide them toward the more successful result(s) and eventually to a near-optimal solution (ideally the optimal solution). The inclusion of random particles helps avoid local optimum solutions.




*Examples of random particles exploring and searching space, moving toward the best solutions. Through many iterations, PSO identifies the global minimum, even among many local minima.*

# Constraints – Optimization Background, cont.

The PSO is really two strategies combined in one. Assuming the goal of the optimizer is to minimize cost, this means using a method to search the global space to ensure the optimizer avoids being stuck in a local minimum, ( i.e. to peak over the hills of the solution to see if other valleys have better minimums). The second part of the algorithm focuses on finding the minimize the cost in the valley. The PSO may be recursive to evaluate each valley minimum cost versus other previous results.

The fewer the hills and valleys to check out, the less stressed is the optimizer in seeking a good optimized solution.

# Constraints – Optimization Efficiency and Operator Acceptance

Under most circumstances, the optimizer should need to only search a portion of a models data space based on the current operating conditions/constraints, even if the model is composed of multiple models to accommodate more modes of operation or product/process mix.

Constraints can be of several source types, some of which are described next.

## Hard Costraints

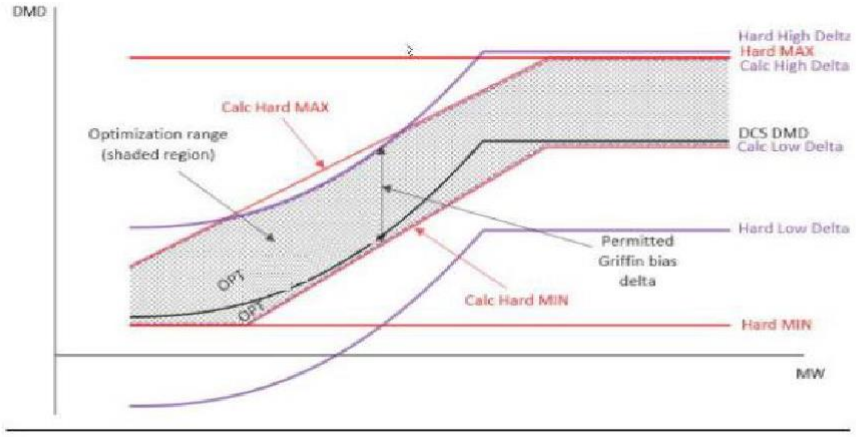
The basic concept of constraints is to have “Hard Constraints” that cannot be violated. There may be calculated hard constraints that are narrower than the absolute hard constraints which have the highest priority. Hard constraints are the process sanity check. The automation systems is designed so that it is impossible to output control or bias value to the DCS that will result in these constraints being violated.

There are two types of hard constraints: a hard process constraint on the final biased demand, and a hard constraint on the bias itself. In an automation project, the bias values would generally be the same or less than what an operator would be permitted to do manually.

Both hard constraints/limitations are iposed, and the final result will conform to whichever limit is more restrictive keeping the control output in the safe and more constrained space. Figure XYZ shows this set of restrictions.

## Soft Constraints

Soft constraints are those which are additional limitations imposed based on the current given operating conditions. These can come in several forms also, but generally are calculated bias limits on the control curve which are designed to provide guidance on where the optimizer should do its search for the optimum solution. Soft constraints may reflect the desire to keep process changes to small increments so as to avoid disturbing other sub-processes. At this point, additional algorithms may be used to calculate optimal process variable trajectories to achieve the desired setpoints.




*System of multiple constraints ensuring robust and reliable control around established DCS curve.*

Modeling can also help in the area of limiting the search space, while improving the models themselves. In the above figure, the gray area may be broken into distinct areas based on which equipment may be in-service, defining clear breakpoints in where different models would be appropriate. These local models can be treated as a merged larger global model as far as optimization is concerned. They may require a default global model where data is sparse, but the local models offer a dual benefit of lowering the search space and the creation of higher fidelity local models where the data is more prevalent, allowing better mapping of the data surfaces. In this type of set-up, it is important to automate the model learning process so models can adapt and 'fill in' as the data becomes available, avoiding the burden of manual retraining.

## Summary on constraints

It is easy to read the literature and assume the AI is a magic bullet for many problems. However, it is very important to help the AI out much as possible through clearly defining the boundaries of what really needs to learned by the models. This takes more upfront time and effort but rewards the user with a reduction in the amount of future work and makes the difference between success and failure.

The impact of this properly constraining the data region continues into the impact on the optimizer algorithms. It helps to eliminate solutions which are not possible, or poor extrapolations into areas where data will rarely exist. Narrowing the search also creates faster solutions or solutions which have had time to more thoroughly perform many more 'what if' comparisons.

**Taking the time and effort to use process expertise to more fully define the operations (data) space in which the models and optimizers work will greatly enhance both the projects likelihood of success and end-user acceptance.**

## Project time allocation

Up to 80% of the work on an automation project may be dedicated to working on system constraints and operator acceptance. When you tour a manufacturing facility you are required to stay between the yellow lines to be safe. They do not let you wander the facility and figure out which areas are safe and which are hazardous on your own. This is unsafe, and would take an extraordinary amount of time to accomplish. This does take time and effort to figure out where

to put the safety lines, and put down the safety lines and defining and putting in place a process for putting up tape in color coded fashion to further constrain where you can go based on the given conditions of the day. While time consuming the result is safe and efficient in guiding the tour. Using AI in automation needs the same type of help.

## Safety

Safety is of paramount importance when evaluating any changes to control systems. Utilizing an adivarent control system strategy where a computer assists the operators and the primary control system is often the best setup. This keeps all safety systems intact, and requires relatively simple changes to the DCS or PLC to accept biases to the control curves. If operations had manual bias control in place before, this configuration permits having either manual bias control or the optimizer bias control. This leaves the constraints in the DCS or PLC in place, while permitting the adivarent control system to impart further constraints based on the given conditions existing in the process at that time.

When taking the time and effort to factor in constraints for the model, optimizer, operator acceptance and improved margin on safety, the modification to the standard control curves will be impacted through hard values (process limits), soft (bias limitation), and dynamic (based on current conditions) constraints, providing multiple layers of process value checks.

## Ongoing 'Tweaks' – Incremental improvements

When moving from Dashboards to closed loop bias control of the process or production activity, it is important to spend time imparting knowledge into the logic and constraints of the system being developed. It is extremely rare when addressing complex optimization projects to think of everything ahead of time. Hence, it is very important to have the proper set of software tools to continually update and improve the system. This can take the form of tweaks (minor rule additions) or adding incremental sets of logic or AI components to further the intended gains in the process.

A well designed system will, in particular, free up more time for an engineer and process expert than is spent on these incremental improvements areas. We have found that while our customers are receiving a good rate of return on the automation projects, the greatest satisfaction comes from the change in their work environment. Namely, they enjoy making continual improvements to the system versus struggling to stay “above water”.

## Software Platforms – quick summary -

It is very important to have a system of software tools which are enable continuous improvement and which do not lock a company into a long term large expense.

While we deal with these characteristics in our other papers and blogs, the key items to keep in mind are:

1. Open Systems is different than open source. Open source is has its benefits if you are programmer, while an open system gives you control for easy modification of the existing system through higher level interfaces, including no-code and low code systems. With an

open system here is an ability to use internal or potentially multiple outside suppliers. In summary, avoiding vendor lock-in can be huge cost and time savings.

2. No-code, low code, and training. Each user has preference and or personnel constraints on how much they can do and learn. Finding tools which can be used by more than one person or easily learned offer longer term value to keeping the project successful. No-code programming is a good start, especially if it is open and allows low code or internal proprietary code to be included.

3. Hardware requirements. Not all software has the same efficiency when running on standard computing hardware. A software vendor which provides AI components executing in 20% of the time of another will significantly lower the scalability cost.

4. Architecture – Deciding on where the computing occurs is a key decision. In addition to cost, it is important to be very clear about who owns the process data when it is sent to the cloud (Griffin Open Systems, 2023).

## Summary and Conclusion

There are many, many tools for data analytics, manufacturing data analytics, modeling, and optimization. These provide insight as to how to improve a process or manufacturing activity. However, the vast majority rely on users to implement changes on an ongoing basis. This negates a huge portion of the potential benefit by missing the opportunity to improve the process 24x7.

With automation it is generally better to get started with something small. This generally will pay for itself while allowing the all the parties concerned to become comfortable and more knowledge about how the automation will impact actual operations.

## Next Step: Finding a Vendor

Once you have identified a problem set to address, who the target audience will be, and the purpose of the installation, it will be time to pick a vendor offering a software platform to meet your needs.

It will be important to keep in mind your requirements, both initially and for your blue sky goals. The worst case scenario is when you are trying to integrate many different software packages which may be 'best in field' but are not designed to broadly handle your need for data links, data aggregation, dashboards, automation, digital factory models, data analytic tools and AI tools for optimization. It may be best to focus on low code and no code tools permitting rapid prototyping and iterative development. This ease for adapting the system is crucial as the digital factory is a learning process and the industry, rapidly evolving with new techniques and ideas from implementation.

The industry has a lot of glitz and marketing dollars. Do research on vendors offering manufacturing dashboards and digital factory software platforms, have experience in your industry, offer the specific features and functionality you require, and have a proven track record of success.

## Next Step: Finding a Vendor, cont.

Keep in mind the long-term costs, especially the more nefarious ones, such as forced software upgrades, vendors relying on change orders for revenue, or 'buggy' software requiring extensive attention to end-user support. While your organization may not plan on making changes internally, having the ability to do so or have alternate outside support if the main vendor is slow or unresponsive should be a consideration.

Keeping these ideas in mind will help the user select a vendor which can lead to quick positive results, helping build momentum for further expansion into automation and more process improvements as the organization moves toward a digital factory.

## Prologue

Griffin Open Systems, LLC is a software platform provider for dashboard tools and AI Tools built for use in manufacturing and process industries. Griffin's open design for the software tools helps businesses build digital factories for improved efficiency, environmental compliance and leveraging existing valuable personnel.

Please visit our website for an easy to install dashboard and more informational whitepapers, blogs, and videos.

### Get Started with Adivarent Control Today

For future white papers and notices when new videos for the digital factory become available, please register your name, company, and email on the contact us page of the Griffin Open Systems website. You may also request the free version of the Griffin Dashboard Toolkit.

**email:** info@griffinopensystems.com | **visit:** GriffinOpenSystems.com

# Citations

1. Tuttle, J., Alagarsamy, S., & Radl, B. (2023a). The Journey to a Digital Factory Part I: From Levers & Spreadsheets to Automation & Digitization. Retrieved from https://www.griffinopensystems.com/whitepapers/the-journey-to-a-digital-factory-part-i/.

2. Tuttle, J., Alagarsamy, S., & Radl, B. (2023b). *The Journey to a Digital Factory Part II: Data, Data, Data Everywhere*. Retrieved from (https://www.griffinopensystems.com/whitepapers/the-journey-to-a-digital-factory-part-ii/).

3. Griffin Open Systems. (2023). Glossary of Commonly Used Terms. Retrieved from https://www.griffinopensystems.com/glossary-of-commonly-used-terms/