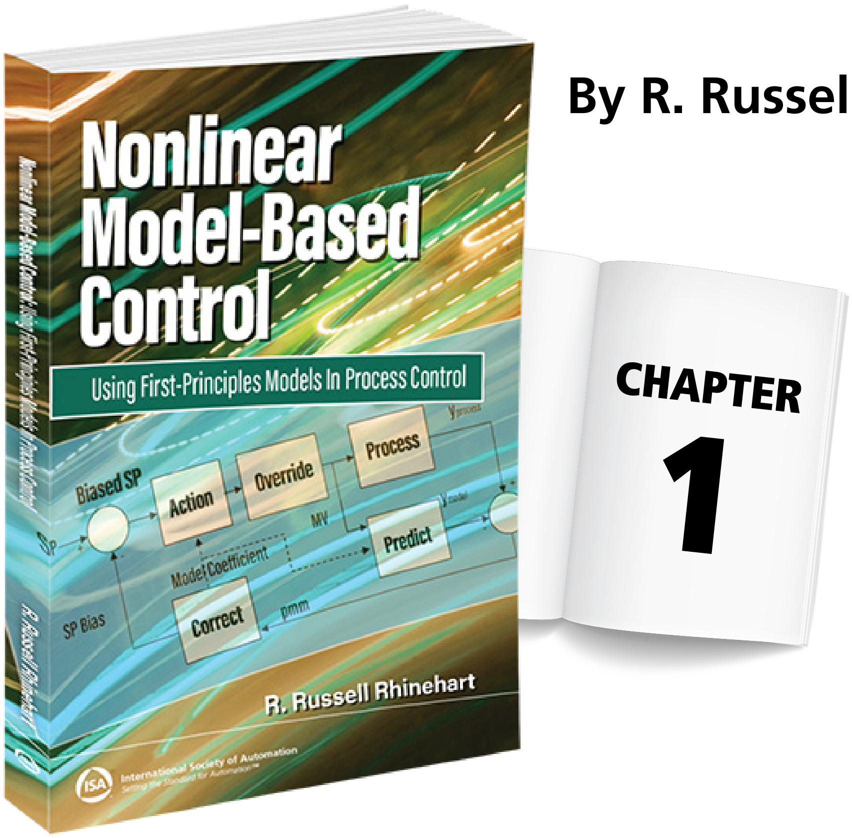
Chapter 1 of:

Nonlinear Model-Based Control

R. Russell Rhinehart

Nonlinear Model-Based Control

Using First-Principles Models in Process Control


Nonlinear Model-Based Control
Using First-Principles Models In Process Control
R. Russell Rhinehart
CHAPTER 1

By R. Russell Rhinehart

Book Table of Contents

Buy the Complete Book

Nonlinear Model-Based Control

Using First-Principles Models In Process Control

R. Russell Rhinehart

Printed in the United States of America

ISBN-13: 978-1-64331-242-2 (print)
ISBN-13: 978-1-64331-243-9 (ePub)
ISBN-13: 978-1-64331-244-6 (Kindle)

ISA
3252 South Miami Blvd, Suite 102
Durham, North Carolina, USA 27703
www.isa.org

Library of Congress Cataloging-in-Publication Data
Rhinehart, Russell.
Nonlinear model-based control: using first-principles models in process control / Russell Rhinehart.
p. cm.
Durham : International Society of Automation, 2023.
Includes bibliographical references and index.
ISBN: 9781643312422 (paperback)
LLCN: 2024932056
1. Automation and control. 2. First-principles models. 3. Generic model control. 4. Process-model-based control. 5. Predictive functional control. 6. Horizon predictive control.
https://lccn.loc.gov/2024932056

1 Introduction

1.1 Model-Based Controllers

Model-based control algorithms were developed when the digital revolution made it possible for them to do more (e.g., override, ratio, feedforward, and gain scheduling) than analog proportional-integral-derivative (PID) and associated advanced regulatory control (ARC) could do. Early model-based techniques include the Smith predictor and Dahlin's algorithm. In the 1970s, some vendors created distributed control systems (DCSs) and programmable logic controllers (PLCs), which were digital computers for control, and simultaneously, other vendors were creating affordable digital computers for off-line use. Then, many visionaries realized that the affordable digital power would support new model-based control algorithms that would have advantages over ARC techniques. After a development phase to test and refine algorithms, the 1980s saw many control approaches begin to be revealed. The advances continue today, revealing more comprehensive and powerful approaches.

These techniques, designed to solve one problem, failed to provide a comprehensive control strategy. For instance, generic model control (GMC) using steady-state (SS) models is good for nonlinear processes. It has the convenience of acting like a proportional-integral (PI) controller with output characterization and only requiring SS models, but it is not good for multivariable or constrained processes or those with ill-behaved dynamics (inverse action or dead time).

Internal model control (IMC) can handle ill-behaved dynamics but uses linear models and is not very good for multivariable or constraint handling.

Natural language processing (NLP), alternately termed *fuzzy logic* (FL) or *expert systems* (ES), and associated techniques like Takagi-Sugeno-Kang (TSK) models can also handle nonlinearities and have the convenience of using the process expert's qualitative understanding of the process response. However, these become quite complicated for multivariable processes and constraint handling.

Neural networks (NNs) provide a model-free (no predefined functionality) way to create nonlinear empirical models of the process. These could be useful in controllers, but they mask any mechanistic understanding of the model, which I believe has much value.

Predictive functional control (PFC) uses a process model to determine a step-and-hold value for the manipulated variable (MV), alternately termed the *controller output* (CO), to make the model match the model set point at a future time, termed the *coincidence point*. This has the advantage of compensating for both nonlinearity and ill-behaved dynamics. With first-principles models, it also preserves process knowledge and appropriately accounts for measurable disturbances. It is good for single-loop control and can be extended to multivariable constraint-handling control.

Dynamic matrix control (DMC) and similar approaches such as identify and command (IdCom) were early multivariable, horizon-predictive, model-based controllers. They can avoid constraints, handle multivariable interactive processes, and compensate for disturbances. Many companies have developed products based on these methodologies. The present industry term for this approach is *advanced process control* (APC), while the academic term is either *model predictive control* (MPC) or *horizon predictive control* (HPC). Many vendors use their own terminology. However, the dynamic (time-dependent) models are linear. Some are FOPDT, some are state-space, but most are formulated in a vector notation, termed *finite impulse response* (FIR) models. Being linear, they do not adapt to process changes and require extensive process testing to obtain confident vector element values. They mask process knowledge.

Process-model-based control (PMBC) was developed to handle nonlinearity, preserve the engineer's first-principles knowledge, and adapt to the process as process attributes change (fouling, reactivity, ambient losses, efficiency, etc.). Tracking process parameter values is beneficial for forecasting constraint conditions and predicting when model or process maintenance will be required. It is a one-step-ahead prediction method for nonlinear processes, and although relatively simple, it does not handle ill-behaved dynamics.

PMBC embodies the "one-model" approach. Currently, steady-state models are used for supervisory set point optimization (real-time optimization [RTO]), process design, and analysis. Dynamic models (often nonlinear) are used in training, and classically linear dynamic models are used for control. When different modeling approaches are used, process engineers must understand each and adapt each as the process changes. It is desirable to use one process model for all applications, and it should be a first-principles dynamic model, which preserves process understanding that is essential for design and troubleshooting.

There are nonlinear approaches to APC (or MPC). In one approach, an NN is used to capture the process SS gain nonlinearity; then, the FIR (a vector of the response to a unit input step) model is used to approximate the linear dynamics. Like APC, this requires substantial process testing over a wide operating range. In another approach, the process expert's judgment is used to assign gain values; then, the FIR model is used to approximate the linear dynamics. This requires much less process testing because only the unknown dynamics are required. Using qualitative gains might seem to generate imperfect models, but all models are imperfect, and as long as a model is passable (perhaps it would get a grade of C in school), it is fully adequate for control.

Finally, using first-principles models in APC with PMBC adaptation has been explored industrially and is getting performance affirmation. It can handle nonlinearities, multivariable interactions, and disturbances and has constraint-handling features. It preserves process knowledge. Online adaptation keeps the model true to the process and reveals process analysis information that is useful for forecasting constraints and maintenance needs.

This book covers the best-in-class approaches (in my opinion) for using first-principles models in control—GMC with steady-state (GMC-SS) models, PMBC, PFC, and nonlinear APC (nAPC).

Why use simple PMBC?

- Unlike an empirical model such as a first-order plus dead time (FOPDT) approximation, it preserves and transmits process knowledge.
- The model only has a few adjustable coefficients to make it fit the process data, which means less process testing to develop the model.
- Once tuned, it remains tuned throughout the operating range.
- There is only one aggressiveness factor per controlled variable (CV), not several.
- When incrementally adjusted, the model parameter value tracks the process behavior to monitor process health.

- The model forecasts constraints that can be useful in higher-level optimization.
- The same model can be used in other applications, such as optimization, training, design, process analysis, constraint forecasting, and maintenance prediction.
- The variables are not deviations from an initial SS, and the units are not percentages. The variables are not transformed or expressed in abstract mathematics. They are familiar variables that engineers use with units of the process variable.
- Regardless of controller type (ARC, APC, nAPC, fuzzy, or NN), if the controller has feedforward, gain-scheduled, and decoupling features, it has the same reduced variance benefits—higher product quality, lower costs, faster throughput, reduced waste, and so on. I think nonlinear model-based control (nMBC) can be justified like ARC is, but nAPC will have a wider operating range and life.

ARC can handle many of the process difficulties within one structure. For instance, classical gain scheduling a PI controller can make it adapt to process nonlinearity. There may be three operating ranges; the control technician tunes the controller gain and integral time constant for each. PI control has two coefficient values for each of the three operating ranges and two breakpoints in the operating range that must be determined. In all, there would be eight tuning values, usually found by testing at each of the conditions. By contrast, a process-model-based controller has only one tuning coefficient that is valid for the entire operating range. For instance, with ARC, including feedforward to give advance accommodation for a PI feedback loop requires six empirically determined coefficient values for each operating range, whereas the feedforward accommodation is naturally included in a process-model-based controller. If one understands ARC, then even with the extra number of coefficients to be tuned, it might be simpler to implement and maintain than model-based control (MBC). However, I believe if the process has two or more CVs, then MBC is simpler.

1.2 Nonlinear and Nonstationary

The term *nonlinear* means not linear. If you were to plot one variable w.r.t. (with respect to) another, linear would mean the relationship is a straight line. One variable responds to the first power of the other. However, there are many nonlinear patterns. They could curve upward, curve downward, be S-shaped, and so on. Nonlinear does not say what it is, just what it is not.

In differential equations, *linear* means that if two functions are a solution, then the sum (linearly added) of the two is also a solution. This is often referred to as the *principle of superposition*. The functions do not need to be linear.

In an open-loop stable process, such as $\tau\frac{dy}{dt}+y=Ku$, where τ and K (time constant and gain) do not change in time, the response is an exponential change in time, $y(t) = Ku + (y_0 - Ku)e^{-t/\tau}$ (t is time, $\frac{dy}{dt}$ is the rate of change of the response in time, y_0 is the initial value of the response, u is the controller output, and e is the base of the natural logarithm). The graph of the response in time is not linear. However, the SS response $y_{SS} = Ku$ is a linear response to u with a gain (slope) of K. And if you look at the differential equation, the response variable y is always expressed as the first power, and so is the independent variable u. So, the response y is termed a linear response to u.

Formally and mathematically, linear can be described as the case if the second derivative of the response w.r.t. any two influence variables is zero.

$$\frac{d^2y}{dwdx}=0 \tag{1-1}$$

An example of nonlinearity in process models is the ideal squared dependence of the pressure drop due to the flow rate.

Stationary means that the coefficient values in the model do not change in time or w.r.t. operating conditions. *Nonstationary* means that the model coefficient values are a function of either time or some influence or response variable. If the temperature changes during a reaction and the reaction rate is temperature-dependent, then composition equations will be nonstationary. If heat transfer is dependent on flow turbulence, which depends on flow rate, then the energy equation will be nonstationary.

In general, nonlinear and nonstationary are different, but if the nonstationary aspect is due to an influence or response variable, the response is also nonlinear.

1.3 Keep It Simple and Safe

A guiding principle in industry is to keep it simple and safe (KISS). Use the simplest approach that is effective. Balance perfection with sufficiency. Nonlinear control can become gloriously complicated. Do not seek to show off your ultimate skill.

In many of my studies on which control approach is best for classic CV and MV measures, the conclusion is, "As long as the strategy accommodates the process difficulty, it is as good as any other strategy." I've explored PID, IMC, ARC, FLC (fuzzy logic control), PMBC, APC, and nAPC on diverse pilot-scale processes and many simulators. I recommend choosing the simplest approach that solves the particular process problem.

The classic CV measures are rise time and settling time for a set-point change and the integral of the squared error for the disturbance rejection mode. Classic MV measures are travel and energy. However, be aware that there are other metrics associated with desirables for control, including the engineer's time to formulate a model, the data collection effort to match a model to the process, and the multiple uses of the model. A desirable metric also includes simplicity in tuning; characteristically, the model-based controllers have only one tuning parameter per CV, while a PID has three.

The benefits of first-principles models in control are as follows:

- Process knowledge is preserved.
- The engineer is not distracted by learning other mathematical techniques.
- The one-model approach can be used for many applications (design, optimization, RTO, analysis, predictive maintenance, and training).
- Tuning is relatively simple (one parameter per CV).
- Optimization balances performance when constraints or extra degrees of freedom (DOF) happen.
- Features like ratio and feedforward are handled naturally.

1.4 What Are First-Principles Models?

First-principles models are the process engineer's models for process design, analysis, optimization, data reconciliation, and troubleshooting. They are based on elementary material and energy balances and ideal constitutive relations, and they are characteristic of college classroom teaching models for heat exchange, fluid flow, reaction engineering, separations, and other unit operations. Representing process mechanisms, they are typically nonlinear.

The models must be validated. Usually, the process engineer uses first principles to generate the calculus/algebra version of the process model, then translates that into executable code to create a process simulator (alternately termed the *surrogate model* or the *digital twin*.) However, the model might be wrong or have some missing features. To validate the model, compare it to the process data. There is likely adequate historical data, but this might require some designed trials. The process comparison could reveal which features must be included or whether the model is good enough. Once validated, it can be used in control.

1.5 Concept and Block Diagram

The diverse model-based controllers were developed independently by folks with diverse backgrounds, mathematical approaches, and jargon to solve different problems. As a result, superficially, there does not seem to be a unifying approach to the controller structures. But there is, and this book presents several techniques from that unifying structure.

There are three functions within MBC: Predict, Correct, and Action (referred to in this document as Act). The structure can be represented by the operations in Figure 1-1.

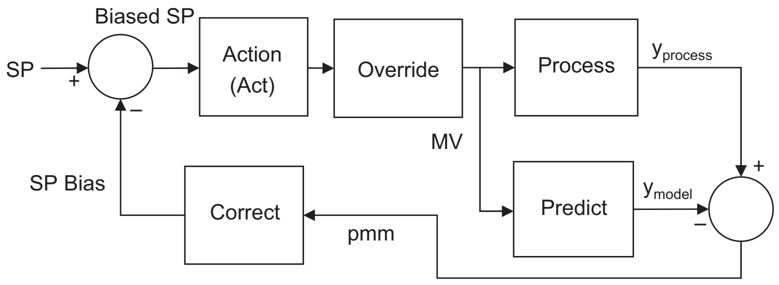


Figure 1-1. MBC representation.

1.5.1 Predict

The first model-based control (MBC) function uses the model to predict or mimic what the process output is expected to do. This function is the lower right box in Figure 1-1. It updates the past model value with past disturbances and control actions to predict the updated (current or new) modeled output. With 10 control actions within the smallest time constant or delay in the model, the control interval, Δt, is usually small enough to permit using Euler's explicit finite difference method to solve the model numerically. More advanced numerical methods are needed if the control scan time is not small relative to the model dynamics.

The acronym for the model in the Predict function is P2N, meaning past-to-now, because it implements one Δt update to reveal what the model predicts the process is doing.

1.5.2 Correct

The second function in MBC is correcting a process-model mismatch. Process-model mismatch, *pmm*, is the difference between modeled prediction and process measurement, as represented by the circle difference operation on the lower right side of Figure 1-1. The modeled value, y_m, will not match the process value, y_p, exactly for a variety of reasons, including model error or unmeasured disturbances. A simple correction approach is to bias the set point, y_{SPbias}, for the model with the process-model mismatch as follows:

$$pmm = y_p - y_m \tag{1-2}$$

$$y_{SPbias} = y_{SP} - pmm \tag{1-3}$$

The logic is illustrated in Figure 1-2. If you aim at the target bull's-eye but hit 3 cm low, aim 3 cm above the bull's-eye next time. If that shot is 0.5 cm above the bull's-eye, the pmm is the difference between where you aimed (+3 cm) and what happened (+0.5 cm). If the pmm is 2.5 cm, aim 2.5 cm above (not 0.5 cm below) the bull's-eye next time.

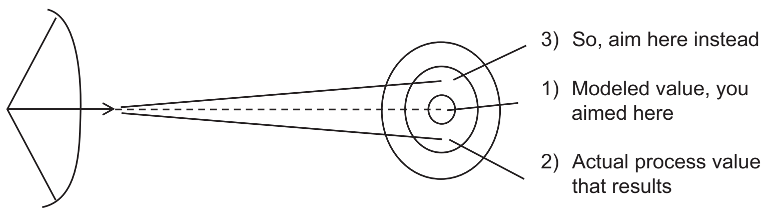

3) So, aim here instead
1) Modeled value, you aimed here
2) Actual process value that results

Figure 1-2. The concept of biasing the set point for a model using pmm.

The box labeled "Correct" in the lower left of Figure 1-1 could provide other functions. One of the alternate functions could be related to model coefficient adjustment, which has some managerial information benefits. But in this simple correction approach, it simply is a pass-through of the pmm value to bias the set point.

Notes:

1. The biased set point is the set point for the model, not the set point for the process.
2. pmm is not the actuating error. It is not the deviation from the set point (the target). pmm is the deviation between the actual hit and the model-predicted hit.

1.5.3 Act – an Elementary Choice – Simple PMBC

The third MBC function calculates the control action to determine the controller MV value. This is represented as the box labeled "Act" in Figure 1-1. This function starts with a user-defined, desired performance objective for the controller. A simple desire in the PMBC strategy is to calculate a u-value (the MV in Figure 1-1) that would push the model toward the biased set point at a rate proportional to the deviation from the biased set point. (This is an effective strategy for low-order models with inconsequential delay.)

Mathematically, this simple desire is:

$$\left.\frac{dy_m}{dt}\right|_{desired} = \frac{y_{SPbias} - y_m}{\tau_{want}} = K_c(y_{SPbias} - y_m) \tag{1-4}$$

The single tuning factor, τ_{want}, is the time constant for the desired first-order trajectory for the model to return to the biased set point. Alternately, K_c, the reciprocal of τ_{want}, can be used with a large value, creating a more aggressive controller action. This is more convenient for some, but it masks understanding the desired trajectory.

Notes:

1. The biased set point, y_{SPbias}, is for the model, and y_m is the model-predicted value.
2. $y_{SPbias} - y_m = y_{SP} - pmm - y_m = y_{SP} - y_p + y_m - y_m = y_{SP} - y_p = e$. The right-hand side of Equation 1-4 is $K_c e$, which is the normal model for proportional action. However, rather than stating it as such and masking the concept of the desired model trajectory, I will preserve the Equation 1-4 structure.
3. $\frac{y_{SPbias} - y_m}{\tau_{want}}$ is the user-desired rate of change of the modeled output, y_m. If there were no process-model mismatch, then this would also be the consequential rate of change of the process, y_p. But only rarely would a model exactly match the process behavior.

Now, to determine the action, insert the desired rate of change to the model and solve the model inverse to determine the controller output, u.

Usually, the model is used to calculate the rate of change of the modeled response, dy_m/dt, given process influences and state. However, in control action, the solution must determine the value of the MV, u, that would cause the desired rate of change. This is the inverse of the normal modeling solution procedure, often termed *solving the inverse*.

If you are lucky, the inverse will be mathematically tractable. You will be able to explicitly solve for u. However, it might not be possible to rearrange the model equation to do so, and a numerical method might be needed. Root-finding might be a first thought, and it works. However, optimization also works and has the advantage of being able to cope with constraints. This book will focus on using optimization to solve nonlinear implicit equations.

This set of Predict, Correct, and Act procedures represents the PMBC controller strategy.

Returning to Figure 1-1, each of the three functions (Predict, Correct, Act) could be specified with various options or executed with various techniques. The multitude of choices has led to a variety of controller algorithms and associated acronyms.

If the model is linear and first-order, the three-stage method described here is equivalent to an internal reset feedback version of a standard PI controller with the integral time equal to the process time constant. Note the following two points: (1) It is comforting that idealizations lead to a familiar solution, confirming the correctness of the control steps. (2) If the model is linear and first-order, use the conventional PI control technique, which will be comfortingly familiar to many of your stakeholders.

The acronym for the model in the Act function is N2F, meaning now-to-future. It is used to determine the control action to make the model behave as desired in the future.

Notes:

1. For single-input, single-output (SISO) applications, the controller output, u, could be the signal to the final control element (e.g., a valve, variable speed drive, or electrical heater). This is often termed *direct digital control* (DDC). However, in more complicated applications, it is desirable to have the model-based controller determine set points (e.g., flow rates and temperatures) for secondary loops to manipulate the final control element. The secondary loops are often regulatory PI or PID controlled, but if they have nonlinear and ill-behaved dynamics, the secondary loops could also have SISO model-based controllers.
2. The Predict and Act functions use the same mathematical model. They could use the same function to perform the calculation. However, the P2N model in the Predict function must be initialized with the prior modeled states and influenced by the actual implemented control action. By contrast, the N2F model in the Act function must be initialized with the P2N model states and influenced by the control action being contemplated by the Act function.
3. If the models are not identical in the Predict and Act functions, the states will have a steady-state offset and/or dynamic incompatibility.

1.5.4 Override

The Override function in Figure 1-1 is not necessarily part of the controller, but it could be. For instance, if the controller is tuned to be aggressive, it may calculate the controller output to exceed feasible values. It may ask for 123% or –4% in the signal to the valve. Or if the feasible flow rate range is 0 to 36 gpm, the aggressive controller might calculate 37 gpm. In such cases, the controller-desired output must be limited to what is feasible.

The operator may not want the controller to make large instantaneous changes but incrementally move the MV to a desired value. Large instantaneous changes may create thermal stresses, water hammer, and other undesirable phenomena. To prevent these, a rate-of-change (RoC) constraint can be imposed on the controller action.

Alternatively, the override might be external to the controller. It may be that an operator is controlling the valve locally or that a safety/override controller was selected to send the signal to the process. In this case, the MV value override that the controller desired is external to the controller.

Note: In any override case, the MV value used by the P2N model in the Predict function is the actual MV value, not that calculated by the Act function that may not be aware of constraints or overriding events external to the controller.

1.5.5 Infeasibilities

The controller may have square root, division, or other operations that could lead to an execution error. Usually, the override action needed to prevent the error is obvious, but it must be included in the executable code.

Example 1-1

Create a simple model-based controller for a mixing process. Hot and cold water enters a tank with a constant volume, *V*, which is reasonably well mixed. The mixed fluid temperature, *T,* is measured, with minimal sensor lag relative to the mixing time constant, and the hot water flow rate, F_h, is manipulated to control the mixed fluid temperature, *T*. The inlet cold water flow rate, F_c, and temperature, T_c, are measured again with little sensor delay. The hot water valve also has an insignificant lag and a linear installed characteristic.

Assuming constant fluid properties, the SS temperature, *T,* model is

$$T_{model} = \frac{\mathrm{F}_h T_h + F_c T_c}{F_h + F_c} \tag{E1-1-1}$$

where subscripts *h* and *c* represent the hot and cold fluids.

And the dynamic (transient, time-dependent) model is

$$\frac{V}{F_h + F_c} \frac{dT_{model}}{dt} + T_{model} = \frac{F_h T_h + F_c T_c}{F_h + F_c} \tag{E1-1-2}$$

Using a simple finite difference numerical method, the model would be solved in the P2N Predict function as

$$T_{model,\,now} = T_{model,\,prior} + \left[\frac{\Delta t}{V}\right][F_h T_h + F_c T_c - (F_h + F_c)T_{model,\,prior}] \tag{E1-1-3}$$

The process-model mismatch and biased set point, *SP,* are

$$pmm = T_{process,\,now} - T_{model,\,now} \tag{E1-1-4}$$

$$T_{model,\,SP} = T_{SP} - pmm \tag{E1-1-5}$$

If the control objective is to have the model move toward the biased set point at a rate proportional to its deviation (the first-order CV trajectory with time constant $\tau_{desired}$)

$$\left.\frac{dT_{model}}{dt}\right|_{desired} = \frac{T_{model,SP} - T_{model,now}}{\tau_{desired}} \tag{E1-1-6}$$

Then substituting the desired rate for the modeled rate

$$\frac{V}{F_h + F_c}\frac{T_{model,SP} - T_{model,now}}{\tau_{desired}} + T_{model,now} = \frac{F_h T_h + F_c T_c}{F_h + F_c} \tag{E1-1-7}$$

And solving the inverse (for the F_h value) using the N2F model

$$F_h = \frac{\frac{V}{\tau_{desired}}(T_{model,SP} - T_{model,now}) - F_c(T_c - T_{model,now})}{(T_h - T_{model,now})} \tag{E1-1-8}$$

With a linear installed valve characteristic, the valve stem position x, fraction of full travel, linearly affects the flow rate

$$F_h = xF_{h,max} \tag{E1-1-9}$$

And because the controller output, u, is a percentage of full scale, the action is

$$u = 100\frac{\frac{V}{\tau_{desired}}(T_{model,P} - T_{model,now}) - F_c(T_c - T_{model,now})}{F_{h,max}(T_h - T_{model,now})} \tag{E1-1-10}$$

Here it is assumed that V is constant. An interaction with the level creates a multidimensional control problem. This simplifying assumption is made to illustrate calculations.

Nominally, use Equation E1-1-3 to get the P2N model value and use Equations E1-1-4 and E1-1-5 to determine the model set point. Use Equation E1-1-10 to determine the controller output to the hot water valve.

There is a divide by V in Equation E1-1-3. If $V = 0$, then the modeled T is the SS T from Equation E1-1-1. Use an IF-THEN conditional to select the right equation.

If $\tau_{desired} = 0$, then there is a divide by zero in Equation E1-1-10. This user error would mean a desire to immediately, instantly have the model value become the modeled set point. Use an IF-THEN conditional to switch from Equation E1-1-10 to the inverse using Equation E1-1-1, which becomes

$$u = 100\frac{F_c(T_{model,SP} - T_c)}{F_{h,max}(T_h - T_{model,SP})} \tag{E1-1-11}$$

If $T_h = T_{model,now}$ there is a divide by zero in Equation E1-1-10. If this is the case, no control action is possible. Increasing or decreasing F_h cannot change the modeled temperature. Perhaps the cold water flow is zero, or the hot and cold temperatures are the same. Might as well stop wasting hot water. Use a conditional to set $u = 0$.

The controller output must be limited to within 0% and 100%. If u exceeds the limit, use a conditional to reset u to that limit.

1.6 Multiple Model Use – A Caution

The process model is used in two places in the controller: in the Predict (the P2N calculation) and Act (the N2F calculation) functions.

The Predict function updates the past modeled value with the actual implemented MV to predict what the CV is doing now. This is a one-time-step update using one-step past values. I use the name past-to-now, or P2N. Initialize the one-time-step update with the prior P2N modeled value.

The Act function uses a model to forecast what might happen in the future, associated with a trial solution for the MV value. In the example, the model inverse could be explicitly solved. But an iterative search may be needed in some models: Guess at an MV and see what the modeled CV does. Re-guess at an MV value until the future modeled CV matches the desired rate of change. I call this the now-to-future (N2F) model. Initialize the N2F prediction with the current P2N modeled value. After each MV guess, the N2F model must be re-initialized with the current P2N value.

The P2N and N2F models must be identical. If the model is complicated, the functions can be the same subroutine or function. But they must be initialized as appropriate.

However, there are three process models in control simulation and testing. A process model is also used as a surrogate for the process response. This is the upper right "Process" box in Figure 1-1. This simulation model should be different from the controller models. We never seem to be able to describe Nature exactly, which seems to be more complicated and less ideal than our concepts. Process responses are affected by internal noise (turbulence and incomplete missing) and unmeasurable disturbances (environmental effects). It would be cheating to demonstrate a controller using the same model in the simulator as in the controller. Ensure the surrogate process model is a few steps less ideal than the controller model.

The process model in the controller is not the true process model. Although it is intended to be a model of the process and often is referred to as a *process model*, it is not.

1.7 Model Sufficiency/Perfection

The process model in the controller must adequately represent the process, but it does not need to be an exact (or true or perfect) representation. We use FOPDT models effectively in control. Those first-order plus deadtime idealizations mimic a high-order response with a reasonable (not perfect) fit. Further, the empirical determination of the gain, dead time, and time constant for the "process model" are all somewhat in error, subject to the experimental vagaries. FIR models seem more perfect than the simpler FOPDT models, and they are a better fit for matching experimental data, but they too are linear, and the coefficient values also represent experimental noise and uncontrolled disturbances. Both of these models are adequate for control.

Alternatively, one could use rigorous models in the controller to attempt to explain every process nuance perfectly. However, first-principles models have an appropriate balance of sufficiency with perfection and are fully adequate for control. Contrasting modeling perfection, consider that humans can catch a fly ball or steer a car through a curve without rigorous models, with just intuitive continual correction. Like humans in the previous example, PI controllers, which are based on linear, stationary, simplistic (FOPDT) inexact models, are also adequate for control. Here is a qualitative explanation as to why this is: If there are 30 or so control intervals during a process transient, then even if the model is only 70% correct, leaving a 30% error in its first action, at the next control action the remaining error is 70% corrected, leaving a $0.3 * 0.3 = 0.09$ error. By the time a few control actions have been taken, the error is smaller than the discrimination interval for the digital representation of numerical values, visually appearing to be zero.

1.8 Nonstandard Code

The equations and code for any process-model-based controller, including overrides to trap execution errors, represent the model; this makes the code unique to the process and how it was modeled. Although the controller structure and stages within are the same from one application to another, the code is unique to the application. This would be a disadvantage.

1.9 Characteristics of Process Control Applications

Characteristic difficulties of processes, process control, and the implementation environment within the process industries shape the methods that are practicable. These characteristics differ from other control applications (robotics, flight, etc.). Here is a list of characteristics of control applications in the chemical process industries. See Chapter 5 for more details.

- Nonlinear
  - Gain, or sensitivity, changes with the MV (or CO) or the CV (or PO).
  - Nonstationary: Time constants and parameter values change in time or change with respect to operating conditions.
- Ill-behaved dynamics
  - Large dead time relative to time constants
  - High-order
  - Integrating
  - Open-loop unstable
  - Inverse-acting
- Slow dynamics
  - This is usually a benefit, contrasting the speed of response of electronic, mechanical, and aeronautical applications, permitting less expensive devices.
- Multivariable (Interactions)
  - DOF $= 0$ – "Square," meaning the same number of MVs and CVs
  - DOF < 0 – more CVs than available MVs
  - DOF > 0 – more MVs than CVs
  - DOF changes with constraints
  - DOF changes with operator action (manual mode [MAN], override, maintenance)
- Constraints
  - Soft or hard due to environmental, health, or safety
  - Soft or hard due to product specification or equipment life
  - Soft or hard due to equipment limitations (e.g., a valve cannot open more than 100%)

  - Now or into the future (as current MV or disturbance changes impact the future values of a CV or auxiliary variable)
  - Either the rate of change or value or both
- Overrides
  - Safety or constraints
  - Process maintenance taking over an MV in a MIMO system
  - Operator placing some MVs in MAN in a MIMO system
- Disturbances
  - External uncontrolled "environmental" factors. These usually have some persistence but wander about a nominal value. They could, however, progressively change due to fouling, aging, deactivation, or other conditions.
  - Measured or unmeasured influences.
  - Equipment switching, such as pressure-swing absorption and ion-exchange regeneration.
- Noise
  - Seemingly random independent perturbations on the measurement. These are often attributed to the measurement system, not a real process phenomenon. They are often Gaussian distributed.
- Instrument, control system, and process costs
  - To measure everything desired
  - To keep processes linear
  - To meet economic profitability in process design
- Faults
  - Measurement errors due to sensor failure or control implementation due to final control element problems (stiction, steam utility pressure, instrument air pressure, etc.).
  - Measurement errors due to calibration drifts.
  - Process attributes such as bypass, blockage, cavitation, and collapsed internals.

  - Control system capacity such as high traffic-missed transmission or delays, central processing unit (CPU) overload, and alarm priority overload.
  - Spikes in a signal due to electronic interference.
  - Calibration issues such as discrimination errors or a process variable (PV) beyond the calibrated range.
  - Inaccurate input data. Data validation or reconciliation may be required to solve faults. Garbage in, garbage out (GIGO) is an old mantra. If the input data is erroneous, the control action will also be. Perfection is not required, but the data must be free of major faults.
- Models
  - The cost to generate and maintain them is not conventional.
- Resolution
  - Digital discretization
  - Truncation
  - Stiction
  - Dead band, play in the linkage
  - On-off action
- Calibration
  - Measurement devices are presumed linear and calibrated as such.
  - Transducers (current to pneumatic to digital), actuators, sensors, and so on must be calibrated. They are nominal, not precise.
- Capability of operators/managers
  - Operators/technicians/maintenance personnel could have associate degrees, but many have less education. Accordingly, human-machine interface (HMI) understandability and overview, the complexity of the control algorithm, and training are all issues that must be accommodated.
  - Similarly, the engineer will have a BS degree and perhaps some equipment and basic control training courses, so the controller must be understandable and easily adapted by the process control engineer.

- Operation
  - Throughput, inventory level, product grade, equipment reconfiguration, and process attributes (fouling, catalyst activity) are continually changing, effectively changing process gain, time constant, and dead time.
  - Control is frequently switched between MAN and AUTO (automatic) modes, needing initializations for bumpless transfer.
- Slow dynamics – Settling times from minutes to hours
  - The control system cost must be low, but fault-free reliability to handle aberrations must be high.
  - Control action must be gentle for the stability of the operation. Some control strategies switch to new constraints to "save a penny," causing an upset with a much larger impact.
- Batch or continuous
  - Continuous processes, ideally, run continuously under the same conditions. The CV is continuously measured and displayed. Occasionally a set point changes, but mostly control keeps disturbances from causing the CV to deviate from the set point.
  - A batch process, ideally, follows a recipe that guides it through stages. During the batch, the material is moved toward its desired final value. At the end of the batch, the CV can be measured. That batch is complete, and the CV cannot be adjusted. However, controls such as temperature, mixing rate, pH, or oxygenation may be continuously implemented during the batch. Typically, these represent nonlinear control opportunities because the process attributes change during the batch.
  - However, the continuous process might have a large measurement delay, such as with a gamma-ray or chromatograph composition detector. Then, as with a batch, it is too late to change that material once a deviation is reported. Control is needed to adjust the process operation, but continual feedback may be responding to noise and tampering with the process. Similarly, the end-of-batch results could provide feedback information to adjust the recipe. Both could be considered end-of-batch control applications.

1.10 Control Solutions

Often, nonlinearity is the primary problem for SISO chemical process control. One solution is to design a process for control success—design the process for linear responses or large inventories to reduce interaction and temper upsets. However, design is an act of balancing multiple objectives; other desirable issues, such as capital cost, flexibility, resource use, energy integration, and sustainability, would have to be sacrificed to accommodate process control. Practicable techniques for nonlinear control can ease design constraints and enable operating more competitive processes.

With nonlinear processes, gain scheduling is a conventional, control-oriented solution to accommodate nonlinearity. In gain scheduling, the PID controller coefficients are changed to reflect the current operating region. The engineer can use process knowledge to create tuning values (or empirically tune the controller at the different operating regions), which are either placed in a lookup table or expressed in equations. The controller first looks up (or calculates) PI (or maybe PID) coefficient values; then it uses the conventional PI(D) algorithm for control.

However, if the engineer's process knowledge is expressed as either a dynamic or SS nonlinear model, it is about as easy to use a PMBC to implement gain scheduling. Several PMBC approaches have succeeded in industrial applications, and there are several commercial products for relatively simple, nonlinear, process-model-based controllers. Engineers have implemented versions in-house, and control integrators and service providers have been implementing model-based controllers for decades.

SISO or MISO (multiple-input, single-output) PMBC control has several advantages over either PI(D) or gain-scheduled PI(D). PMBC has a single tuning parameter, nonlinear compensation throughout the entire operating range, process knowledge preservation, and continuous process monitoring, which offers improved process health, predictive maintenance, and constraint recognition. For multiple-input, multiple-output (MIMO) processes, PMBC can decouple nonlinear interaction, balance deviations from set points when MV constraints are hit, and determine economic optimum MV values when there are extra DOF.

In contrast to the "large model," multi-step-ahead APC techniques characterized by HPC or MPC, the one-step-ahead controllers presented here can solve many problems and can be implemented by a process engineer.

1.11 Control Frequency

The control frequency is the number of control actions per time interval and is often measured in hertz (Hz), the number of control events per second, also called *scan rate*. Control frequency is also referred to as the *control interval* or alternate names such as *scan time* or *period*, which would be the reciprocal of Hz.

The control interval should be small relative to the process response time but not excessively small to create a computational burden. A rule of thumb has been that there should be 30 or more control actions within the open-loop settling time. Faster execution is better for precise control, but the precision improvement is a diminishing returns effect and barely noticeable. If the open-loop settling time is dead time plus three time constants (an FOPDT model concept), then the control interval should be about $\Delta t_{control} \approx (\theta + 3\tau)/30$. An alternate rule is $\Delta t_{control} \approx \theta/5$. Round this to the nearest convenient interval in your software.

Also, consider the numerical solution of the models using the heuristic rule that the simulation time interval must be small relative to any time constants or delays in the model. The smaller the simulation time interval, the truer the numerical solution to the ideal. However, very small modeling time intervals do not substantially improve the numerical model fidelity to the continuum calculus concept. Sufficiency trumps perfection when balancing functionality with computational speed. The simulation time interval should be less than one-tenth of either the modeled delay or the smallest time constant, $\Delta t_{simulation} < min(\theta_1, \theta_2, \theta_3, \ldots \tau_1, \tau_2, \tau_3)$.

It is convenient to have the control interval match the simulation interval, but if the two are disparate, make the control interval, $\Delta t_{control}$, an integer number, *N*, times the model solution time interval, $\Delta t_{simulation}$, and run the simulator for *N* iterations within each control interval.

1.12 Desired Model-Based Controller Attributes

Desirable attributes of a controller can be partitioned into various categories of being operator-friendly and having technical benefits.

1.12.1 Convenience for the Operator/Engineer

- It is simple, easy, and fast to set up.
- It is simple to tune, debug, understand, and agree with its action. If the operator is uncomfortable, it goes to MAN. This is particularly important when the controller is avoiding future constraints, compensating for interactions, and having variables with delays or that act much more slowly than others. The controller may need to reveal to the humans why it is making its choices.
- Balancing constraint deviations is intuitive.
- An operator can place elements within a MIMO controller in MAN or remove measured variables with a fault, and the rest works.

- Understanding the algorithms and internal functions is simple, and they can be easily maintained.
- There is one model version, not two, and certainly not different forms.
- The one-model approach is the same as that used in the supervisory optimizer that determines the set points for the MPC.

1.12.2 Technical Performance

- Can be tuned to provide a wide range of actions from aggressive to temperate.
- Handles hard constraints on the MV value and rate of change.
- Handles soft constraints on CV and AuxV (auxiliary variable).
- Includes economic factors to determine the best economic action when DOF > 0.
- Does not wind up when on a constraint.
- Reflects the nonlinear character of the processes so that behavior is similar in all operating regions.
- Supports bumpless transfer from MAN to AUTO.
- Supports a zero SS offset.
- Are robust to noise and various disturbance patterns.
- Avoids execution errors. Perhaps it is inherently robust (linear with no divides, square roots, or logarithms that could create an execution error), or there are appropriate overrides or error-trapping logic to prevent those or others such as subscript out of range (if the delay becomes longer than an array has dimension).
- Does not get misdirected by root-finding that is diverted in the wrong direction or a local optima or surface aberration in optimization.
- Is guaranteed to return a good solution within a specified execution time. It must complete all calculations within the sampling interval, which must be at least 1/30 of the process settling time.
- Adapts the model to make it useful for other supervisory process management tasks—forecasting when regeneration is needed (catalyst activity, heat transfer fouling), assessing energy or material consumption per product unit for process analysis, forecasting which operational conditions are possible (maximum conversion, maximum production), or revealing faults or undesired situations (flooding, screen blockage).

- Accounts for multivariable interactions.
- Handles ill-behaved dynamic processes (dead time, inverse, integrating, open-loop unstable).
- Can use empirical models.
- Has a sufficient number of future MV trial solutions to shape the modeled CV to the desired reference trajectory.
- Has a minimal number of measurements to increase reliability and minimize the probability of a data fault.
- Is supervisory and transmits set points to lower-level SISO PI(D) controllers so the lower-level controllers keep the process running smoothly if the model-based controller fails or is taken offline.
- Is devised to handle external safety overrides. This places a constraint on the MV values that the optimizer can choose. The overridden value is also needed in the P2N model update.
- Identify mathematical infeasibility, alert the operator, and revert to regulatory control.

1.13 Modeling Approaches

This book focuses on using first-principles models in control. However, many empirical modeling approaches exist, such as NNs, fuzzy FL, and TSK models. These are useful when the process is too complicated to develop phenomenological models. They are developed from empirical testing, similar to the experimental approaches used to devise FOPDT and FIR models for APC. Such models can be successfully implemented within the general MBC structure shown in Figure 1-1 and several MBC approaches described in subsequent chapters.

Examples include the following:

- Dutta and Rhinehart explored the use of NNs in a simple MBC of a lab-scale distillation process in their paper "Experimental Comparison of a Novel, Simple, Neural Network Controller and Linear Model-Based Controllers" presented at the 1995 American Control Conference in Seattle, Washington.[1]

1. P. Dutta and R. R. Rhinehart, "Experimental Comparison of a Novel, Simple, Neural Network Controller and Linear Model Based Controllers," in *Proceedings of the 1995 American Control Conference*, paper TA10-3 (Seattle, WA, 1995).

- M. Parekh, M. Desai, H. Li, and R. R. Rhinehart demonstrated FL (a human heuristic model) as a control approach to lab-scale, in-line pH control, which was later demonstrated on a commercial-scale process in their article "In-Line Control of Nonlinear pH Neutralization Based on Fuzzy Logic" published in June 1994.[2]
- Ou and Rhinehart used multiple NNs to forecast future CV values in horizon predictive control for a lab-scale distillation.[3,4]

1.14 Takeaways

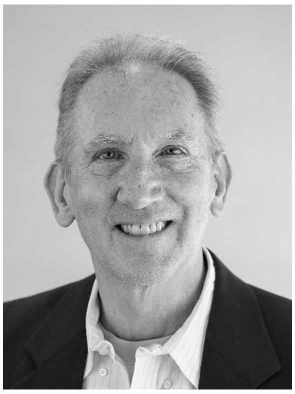
- There are many sorts of model-based controllers. The ones addressed in this book were selected because they (1) embody key approaches and (2) are most practicable for industrial application.
- When you judge a control strategy, do not consider just the CV and MV performance metrics. Consider all the issues summarized in Section 1.12.
- Although it takes engineering analysis to create the equations and code for a model-based controller, the many advantages include those summarized in Section 1.3.

Exercise

1.1 Take a relatively simple process unit that you can model and develop a simple model-based controller, as shown in Example 1-1.

2. M. Parekh, M. Desai, H. Li, and R. R. Rhinehart, "In-Line Control of Nonlinear pH Neutralization Based on Fuzzy Logic," *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, part A, vol. 17, no. 2 (June 1994): 192–201.
3. J. Ou and R. R. Rhinehart, "Grouped Neural Network Model Predictive Control," *Control Engineering Practice* 11, no. 7 (2003): 723–732.
4. J. Ou, and R. R. Rhinehart, "Grouped Neural Network Modeling for Model Predictive Control," *ISA Transactions* 41, no. 2 (April 2002): 195–202.

About the Author



Dr. R. Russell Rhinehart, professor emeritus in the School of Chemical Engineering at Oklahoma State University, has experience in both industry (13 years) and academia (31 years), and was head of the school for 13 years. Russ is a past president of the American Automatic Control Council, was editor-in-chief of *ISA Transactions* from 1998 to 2012, and Director of the ISA Automatic Control Systems Division (now Control and Robotics). He is a Fellow of both ISA and AIChE and a Process Automation Hall of Fame inductee. He received the 2009 ISA Distinguished Service Award and the 2013 Fray International Sustainability Award.

Inspired by his industrial experience, his mission has been to bridge the gap between industry and academia. Russ was the codirector of two industrial consortia (one at Texas Tech and a second at Oklahoma State) and built pilot-scale laboratories for dual use in undergraduate education and graduate research. He left industry in 1982 with a vision to use engineers’ process models in control and pursued many aspects of doing so in his academic research career. This book is his collection of practicable methods. His goal is for it to be a useful guide to others seeking to use nonlinear models in control.

His 1968 BS in chemical engineering and subsequent MS in nuclear engineering are both from the University of Maryland. His 1985 PhD in chemical engineering is from North Carolina State University.

Russ is the author of *Nonlinear Regression Modeling for Engineering Applications* and *Engineering Optimization*, published by John Wiley & Sons, and he is coauthor of the CRC Press textbook *Applied Engineering Statistics*. He also authored six handbook chapters on modeling, uncertainty, process control, and optimization and has presented numerous tutorials and workshops. He maintains a website (*www.r3eda.com*) to provide open access to software (including simulators to support this text) and technique monographs.

Russ taught modeling, optimization, laboratory, and process control courses and has developed short courses for industrial participants offered through ISA or directly to companies related to statistical process control, instrument and control systems, modeling, model-based control, optimization, and uncertainty analysis.

In "retirement," he offers consulting services related to engineering analysis, provides monthly "Develop Your Potential" articles for *CONTROL* magazine, and serves on several ISA, American Automatic Control Council (AACC), and International Federation of Automatic Control (IFAC) committees.

Contents