Adding Realism to Dynamic Simulation for Control Testing

R. Russell Rhinehart
russ@r3eda.com
2024-10-13

Introduction

Use simulations for exploring control applications. There are many benefits for using simulations to understand and test control-related functions and applications, including:

1. Training operators and engineers about the process.
2. Evaluating and demonstrating the economic benefit of the next level of advanced regulatory control or of model predictive control.
3. Testing tuning algorithms.
4. Testing steady-state detection algorithms.
5. Training operators to tune controllers.
6. Testing algorithms for getting FOPDT models.
7. Testing process design options for insensitivity to disturbances (larger mixing volumes, for instance).
8. Comparing control strategies for economic benefit (quality giveaway, constraint violations, waste generation, calibration error robustness, capital cost of devices, maintenance).
9. Generating data to test Big Data algorithms for developing models.
10. Teaching concepts.

For me, understanding the problems associated with integral action that led to reset feedback as the solution, came through my observation with simulations; not through abstract mathematical derivations, not by memorizing what is written, and certainly not through trial-and-error implementations on a real process.

I encourage you to use phenomenological models in dynamic simulators that legitimately represent your process, as opposed to FOPDT, or trivial mechanistic, or empirical black-box models. Search the Internet for “chemical process simulators” to obtain lists of many commercial products that are available. Or write your own code.

Include the controller strategy you wish to evaluate in the process simulator. Be sure that it matches what you would use in the real process (series or parallel, reset feedback, sampling frequency, etc.).

I also encourage you to add environmental effects to the simulator (noise, drift, stiction, resolution, etc.) to make the simulation representative of what Nature will give you. When your process is operating, Nature does not keep the inlet humidity, fuel BTU content, ambient losses, catalyst reactivity, or tax rate constant. And Nature contrives mechanisms that add noise to measurements such as flow turbulence or imperfect mixing. When you implement control, or a

related algorithm on a process, Nature will make it a stochastic process. So, your simulation should also represent the stochastic behavior. This tutorial reveals how to add such effects.

A simulation that includes natural vagaries is termed a ***stochastic*** simulation. By contrast, most simulations are ***deterministic***. Here is the difference: A deterministic calculation returns the same value whenever or wherever you do it. For example, $3x4 = 12$ whether you did the multiplication years ago in 4th grade, or whether an astronaut does it today in the space station. A stochastic process, however, does not reproduce exact values; it produces a realization (a possible value) from a distribution of values. For example, roll a 6-sided die and the values could be 1, 2, 3, …, or 6, each with a 0.1666… probability. Rolling a die generates a stochastic outcome.

Why include stochastic effects? Here is an example. When an operator is taught to tune controllers on low-order, linear, noiseless, undisturbed example, it is easy to determine FOPDT model parameter values using the reaction-curve approach – start from steady conditions then make a step-and-hold in the controller output then wait until steady state. But what if the process did not start at a noiseless steady state, was nonlinear, and was continually perturbed by disturbances? See Figure 1 for an illustration of this open loop step test. The CV, the green trace, is continually affected by uncontrolled disturbances, and the signal is broadened by noise. The CV did not start at steady state but was drifting downward from about 90 to 80 CV units. After the step, the CV rose to about 110 then fell to about 100. How does the trainee identify when the process reached steady state, or what the change in CV is when it never settles? Should gain be determined from the initial CV value of 80 or from a projection of the downward trend to a value of about 70?

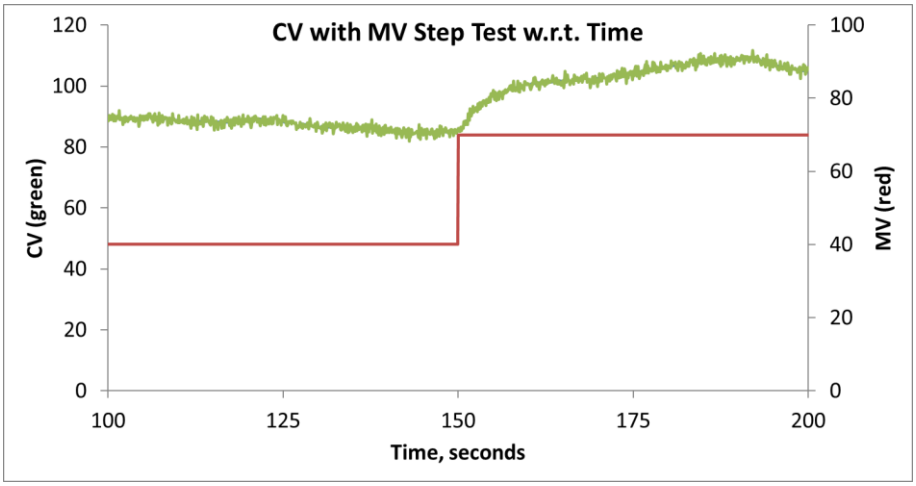

Figure 1 – A Reaction Curve Step-and-Hold test

Use stochastic simulations to train, develop, test, and evaluate control-related procedures, because they represent the reality that a procedure or algorithm must face.

Classically, to test controllers on deterministic simulations, only set point responses (with metrics about rise time, overshoot, damping ratio, etc.) have been used to evaluate controllers. Continue

to use set point changes (servo control) that mimic changes in production rates and product specs changes, but you'll need to replicate the tests many times to get the range of possible results.

However, also use long periods at steady set points (regulatory control) to determine metrics about process variability. From the extended time simulation of regulatory control, measure process variance and the frequency and magnitude of specification violations, waste generation, and on-constraint events. Then shift set points away from constraints and specifications limits to make the violations acceptable. Then evaluate the process economics associated with your control strategy (throughput, cost of material and energy, cost of upsets, etc. relative to the additional cost of devices, and control-related training and maintenance).

A deterministic simulator represents what we ideally model as the truth about Nature. In general, the exact rules and coefficient values that Nature uses to generate data are unknowable. To create a "Digital Twin" with appropriate fidelity to your process you may need to calibrate your simulator – adjust coefficient values to make it better match the process data.

Once you have calibrated a dynamic simulator, add the following stochastic features to better represent the control challenges.

Simulating Noise

Measurements are subject to noise, which are seemingly random perturbations of the true process output.

Noise on measurements could come from several sources. One is "thermal static" due to the device electronics. Another is caused by electromagnetic influences from proximity machines on the measurement transmission wires. A third is mechanical vibrations on a device. These are external to the process. However, the process might also internally generate noise-like measurement deviations. Flow turbulence causes pressure perturbations on a differential pressure measurement device leading to a noisy reported value of the flow rate. Boiling causes both pressure perturbations and thermal variations on sensors in the two-phase mixture. Incomplete mixing after an upstream injection point causes composition or temperature fluctuations on a downstream sensor. On the time scale of process measurements, these noise perturbations often appear independently distributed.

Commonly, the Gaussian (also called the Normal or bell-shaped) distribution is accepted as a representative model of noise. The Gaussian distribution of values represents the confluence of many random influences of equivalent impact. I have been very pleased with the 1958 Box-Muller (Box, G. E. P.; Muller, Mervin E. (1958), "A Note on the Generation of Random Normal Deviates", The Annals of Mathematical Statistics, 29 (2): 610–611) model of generating normally and independently distributed (NID) perturbations with mean of zero and standard deviation of σ, $NID(0, \sigma)$. Generate $NID(0, \sigma)$ values with this formula:

$$n_i = \sigma \sqrt{-2\ln\left(r_{1,i}\right)}\ \sin\left(2\pi r_{2,i}\right) \quad (1)$$

Here, the variables $r_{1,i}$ and $r_{2,i}$ represent uniformly distributed independent random numbers on the 0 to 1 range (excluding 0), typically called by a RAN or RAND function. And n_i is normally and independently distributed with a mean of zero and standard deviation of σ. The subscript i represents the sampling counter; shown to explicitly indicate that the perturbation changes with each sampling. But there is no need to include that interval counter in the simulator.

Note: If your simulator has a built-in $NID(0,\sigma)$ function, you will not need to code Equation (1). If your simulator does not have a random number generator, you could code your own. Search the Internet for "pseudorandom number generator" to find code for a function. A classic algorithm is the Linear Congruential Generator.

There are several approaches to including noise on a measurement. The most common one is to consider that the noise is additive to the true measurement. To simulate this, add the noise perturbation to the deterministic model output. Here σ in Equation (1) has the same units as the measured variable, x.

$$x_{measured,i} = x_{deterministic\ model,i} + n_i \quad (2)$$

Note: Other noise models could consider that the noise magnitude scales with the variable, which may be relevant in flow rate measurements when the intensity of process turbulence is proportional to flow rate. Or, process gains could characterize the noise. For instance, in pH processes, the measurement noise might be due to imperfect mixing, with base-rich and acid-rich fluid packets passing by the sensor. In a strong acid strong base mixture, the pH sensitivity to composition is very high in the near neutral (pH=7) region but very low in the far from neutral ranges (pH<5, or pH>9). Finally, in some processes the noise is not symmetrical. The $+$ deviations may be larger or smaller than the $-$ deviations, and you may need an alternate function to generate skewed noise. For instance, in turbulent flow induced pressure fluctuations the negative deviations are generally larger than the positive deviations. And, in a population of humans, the negative height deviations from average (which includes children) is larger than the positive height deviations from average.

You need to understand the sources of random noise on your process measurement to be able to choose a noise model.

In any case you need to choose a value for σ in Equation (1). This is easy. The 5σ range ($\mu \pm 2.5\sigma$) on a measurement includes about 99% of all values. So, if you use your experience for the range of normally noisy values that would be observed when the process is at steady state, set σ to be one fifth of that range.

$$\sigma \cong \frac{Range\ at\ SS}{5} = \frac{x_{SS\ noisy\ high} - x_{SS\ noisy\ low}}{5} \quad (3)$$

In my experience, there is no justification for the effort to determine the exact value for σ. This takes time, and the actual noise your process generates will likely not be exactly Gaussian. Balance perfection with sufficiency, and use a reasonable estimate for σ from experience.

Note: In Equation (3), do not use the range of possible process values. Do use the range of the perturbations at steady state. For example, the speedometer on an automobile might have a noisy display with a σ of about 0.1 mph, which means the range is about 0.5, which means the perturbation to the true value will be about ± 0.25*, which means that at 55 mph the display fluctuates between about 54.75 and 55.25 mph. The same vehicle might have a speed range from 0 to 80 mph. Incorrectly using the speed range in Equation (3) would imply that the measurement noise has a standard deviation of (80-0)/5=16 mph, indicating that at 55 mph the display would fluctuate between 15 and 95 mph.*

Simulating Resolution

Resolution is the smallest reportable change in a signal. Your digital watch might reveal the time in seconds, and display that time persists at 11:15:26 until it jumps to 11:15:27. Time did not stop for the duration, the display just held the last reportable value, until it could report a new value. That resolution is 1 sec, or 1/60th of a minute, or 1.67% of a minute. A digital thermometer may have a resolution of 0.5 °C. As the temperature progressively changes, the display holds the prior value until the change is enough to display the new value. If the thermometer range is 25 to 45 °C and the resolution is 0.5 °C, then it can only report 41 distinct values within that range, meaning it has a resolution of 2.5% of full range. If a 10-bit processor is in the signal transmission path, then it can only hold $2^8 = 1024$ values. If the calibration range is in the middle 64% of its signal range (as 4-20 mA is a middle-ish 64% portion of the 0-25 mA total range) then only 655 values are used. This means that the signal resolution is the CV range divided by 655. The resolution, the ability to detect change, is 0.153% of the full range.

Resolution is visually detectable when the signal jumps between identical discrete values, as illustrated in Figure 2, which represents actual process temperature measurement kindly shared by Fractionation Research Inc.

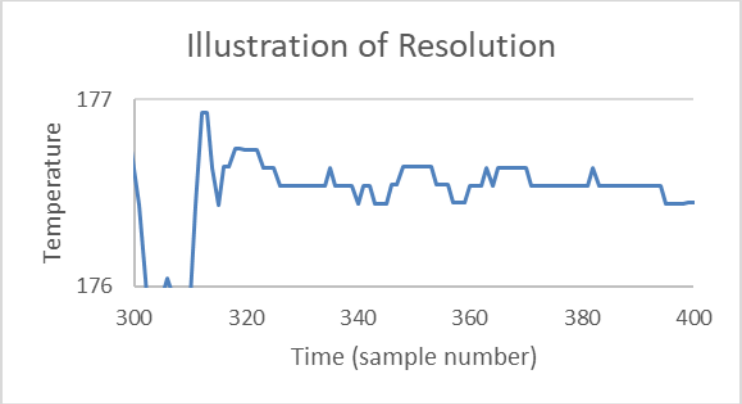

Figure 2 – Process Temperature Measurement over Time

In nonlinear processes or with nonlinear sensors (such as an orifice flow rate), discretization can be invisible at one operating range and clearly visible in another. When detectable, poor resolution can cause a controller cycle in a pattern like its response to valve stiction.

This resolution effect could be termed discretization or truncation. To model it, choose either the smallest interval for change, Δ, or the number of values, N, within a range, $R = y_{high} - y_{low}$. The relation is $\Delta = R/N$. Then, if y is the process value, and $y_{display}$ is what will be transmitted or reported, calculate $y_{display}$ from y.

$$y_{display} = y_{low} + \Delta \cdot INT\left(\frac{y - y_{low}}{\Delta}\right) \quad (4)$$

Note: Equation (4) truncates the y-value. The INT function returns the integer of the $\left(\frac{y-y_{low}}{\Delta}\right)$ value. You could round the value by using $INT\left(\frac{y-y_{low}}{\Delta} + 0.5\right)$ or a $ROUND\left(\frac{y-y_{low}}{\Delta}, 1\right)$ function.

The controller should act on the truncated $y_{display}$ value, not the simulated y value.

Simulating Bias

Noise is often termed ***random error***, which changes value independently at each sampling. However, there may be a measurement calibration bias, a permanent off-set, also called a ***systematic error***. To simulate a systematic error, simply add the error to the deterministic modeled value.

$$x_{measured,i} = x_{deterministic\ model,i} + bias \quad (5a)$$

Certainly, you can simulate both random and systematic error combined. And, as with noise, the systematic error might be scaled by (not added to) the process variable value.

Note: Bias could also be in the final control element. If A/D, i/p, or valve actuators are not perfectly calibrated, the valve position might be 34% when the controller asks the devices to make it 30%. In this case

$$x_{actual,i} = x_{desired,i} + bias \quad (5b)$$

Simulating Faults

Faults could be related to instrument failure of many types, process issues of many types, or data transmission errors of many types. These could be spurious signals, a progressive degradation,

or a step event. You might be evaluating an automation technique for robustness to faults. Be sure to simulate the fault effect in a reasonable and defensible manner.

You could schedule the event to occur at a particular simulation time or let it occur randomly, so that it does not happen at your convenience. Perhaps beginning with each simulated hour, call the random number function to generate a random number, and if the value is less than some threshold, then trigger the simulated event. The threshold value indicates the probability of the event happening. If you want to simulate the event with a probability of once in 1,000 days (24,000 hours) then use 1/24000=0.0004166... as the threshold.

If an event happens you could also simulate the repair time duration to represent a distribution of repair durations.

Simulating Ever-Changing Drifts

The simulation should include mechanisms that generate ever-changing environmental influences, input conditions, calibration drifts, and process attributes. There are five categories of such variables.

1. **Environmental influences,** e. These include wind speed, rain rate, sun irradiance, cloud cover, air temperature, which affect ambient heat losses, drying rates, etc.
2. **Input conditions, rates,** x. These include the flow rates of controlled variables and other measured inflow disturbances.
3. **Input conditions, properties,** in. These include properties of inflowing material, such as the BTU content of a fuel, raw material composition, flow rates, or relative humidity of intake air.
4. **Calibration drifts or bias,** c. These could be on any process measurement, which are related to instrument warm-up, instrument ageing, sensor fouling, linearizing functions, etc. And on any final control element due to the stages in signal transmission.
5. **Process attributes (parameter values)**, p. These represent internal conditions such as distillation tray efficiency, orifice erosion, screen plugging, heat transfer area fouling, or catalyst reactivity.

Although many control theories place “the disturbance” as a single additive perturbation to the output of the process (Figure 3); in actuality, there are many disturbance categories, and they do not act as a single measurement bias to the simulator output. Further, despite classic modeling treatments, the “Disturbance” does not make ideal step-and-hold or first-order lag responses in time.

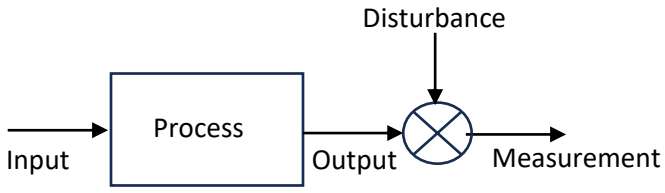

Figure 3 – Do not model disturbances like this

The disturbances, measurement noise, and calibration drift should be modeled as process influences (Figure 4).

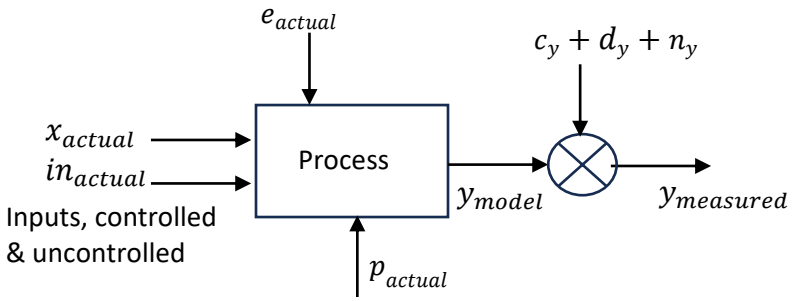

Figure 4 – Model disturbances like this

Certainly, it is easy to have the computer make step or ramp changes in an input, and some influences do switch on and off. But this is not commonly the way Nature devises it. More consistent with reality, the input pattern would be characterized by a random walk about a mean value, a stochastic pattern that does not have timing specified by the human. This section indicates how to include disturbances as autoregressive moving average (ARMA) simulator inputs.

To create a simulator, first define the deterministic equation or procedure to obtain y_{true}, the truth about nature, from $x_{desired}$, the desired value of the influence.

$$y_{true} = f(x_{desired}, p_{ideal}, e_{ideal}, c_{ideal}, in_{ideal}) \quad (6)$$

Although shown as an equation, the process simulator will likely be a complicated set of functions and subroutines. In "Equation" (6), the $x_{desired}$ represents the flow rates of the controlled and the other measured input variables. If you specify an input flow rate of 30 kg/hr, the actual might be 28 because of calibration errors on devices. If you attempt to measure the flow rate of raw material entering a reactor, the desired value might be different due to measurement error. The p_{ideal} represents the ideal values of the process or equipment attributes (catalyst reactivity, tire air pressure, heat exchanger fouling, fluid flow pressure loss coefficient, etc.) The e_{ideal} represents the ideal values of the environmental influences (temperature, irradiance, etc.). The c_{ideal} represents the ideal values of the instrument calibrations (for instance, 4-20 mA generates

3-15 psia which makes a valve stem range from 0-100% of stroke). The in_{ideal} represents the ideal values of the process inflows (temperature, raw material composition, etc.). Before using the model, first perturb each variable in each of the five categories by appropriate disturbances (drifts, d) and if sensible add noise, n. Then use the model to calculate the true response. Then finally, perturb the response with measurement drift and noise:

$$x_{actual} = x_{desired} + d_x \quad (7a)$$
$$p_{actual} = p_{ideal} + d_p \quad (7b)$$
$$e_{actual} = e_{ideal} + d_e \quad (7c)$$
$$c_{actual} = c_{ideal} + d_c \quad (7d)$$
$$in_{actual} = in_{ideal} + d_{in} \quad (7e)$$
$$y_{model} = f(x_{actual}, p_{actual}, e_{actual}, c_{actual}, in_{actual}) \quad (7f)$$
$$y_{measured} = y_{model} + c_y + d_y + n_y \quad (7g)$$

Note: Although the disturbances, d, and noise, n, are added here, it might be more appropriate to make them a multiplicative factor when you anticipate that the noise level or perturbation magnitude scales with the input value.

Note: The ideal values in Equation Set (7) do not have to be constant. You might want to consider that they are also progressing toward some value that would require maintenance.

Although these drift and noise values would change with each sampling, for clarity the subscript counter i is not included.

Usually, only measured values are considered noisy, and the only n value in the Equation Set (7) would be on n_y. Noise is not shown on the process inputs, because the input is the actual value not the measured value.

However, if your control strategy will use an input value (as does ratio or feedforward or cascade) then the control strategy cannot know the actual value. The measured value will be the actual plus calibration drift and measurement noise.

$$z_{measured} = z_{model} + d_z + n_z \quad (7h,i,j,\ldots)$$

In Equation (7h,i,j,…) the variable z represents any quantity that is measured whether it be a process variable, an environmental variable, a feed attribute, etc. Figure 5 indicates adding measurement devices that would send measured x and e influence information to a controller.

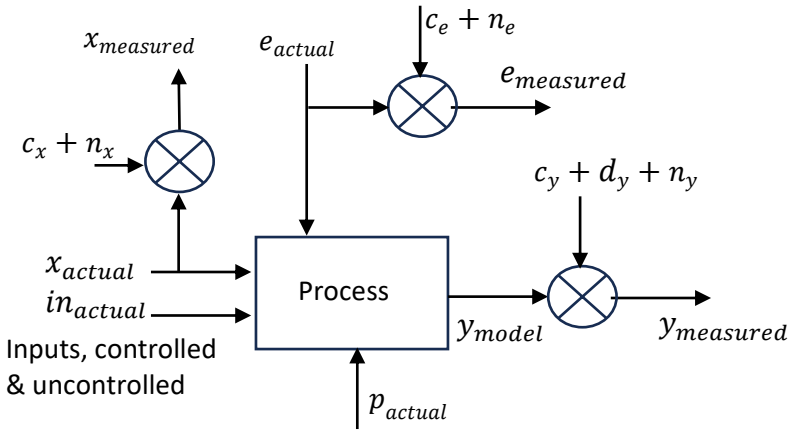

Figure 5 – Model measured disturbances like this

Looking at measurement sequence or time, there are two basic concepts of variation. One is independent variation, in which sample-to-sample variation is uncorrelated. This is often termed noise. With noise, what causes one perturbation from ideal does not persist, and the next perturbation is the result of independent causes.

The other mechanism for variation reflects a cause that has persistence, in which sequential perturbations are correlated. The ever-changing disturbances, labeled as d in Equation Set (7), represent autocorrelated, time-dependent trends, or drifts, in an influence or model coefficient. The symbol n will represent uncorrelated, independent, noise as calculated by Equation (1).

Many influences on experiments have persistence. For example, if a cloud covers the sun, the shadow persists for a while. If the blocked sunlight would influence a temperature disturbance, producing a negative d-value at one sampling (in the shadow of the cloud), then it should be followed by a negative d-value at the subsequent sampling (because the shadow persists). Other persisting deviations could be in intake air humidity, raw material batch composition, etc.

For dynamic, time-dependent simulators, a convenient and realistic method to simulate the way disturbances (systematic error in measurements, as well as uncontrolled influences on the process) change in time is to consider that they are driven by random events that have a first-order persistence.

For those interested in the theory, a drifting influence on the j-th variable with a first-order persistence driven by $NID(0, \sigma_d)$ noise is modeled as:

$$\tau \frac{d}{dt} d_j(t) + d_j(t) = n(t) = \sigma_d \sqrt{-2\ln(r_{1,i})}\ \sin(2\pi r_{2,i}) \qquad (8)$$

Again, the subscript i represents the sample number, the time interval. The σ_d indicates the amplitude of the noise driver.

Expanding this differential with a forward finite difference, and rearranging, leads to the autoregressive moving average (ARMA) model for $d_j(t)$.

$$d_{j,i} = (1-\lambda)d_{j,i-1} + \lambda\sigma_d \sqrt{-2\ln(r_{1,i})}\ \sin(2\pi r_{2,i}) \tag{9}$$

Where

$$\lambda = 1 - \mathrm{e}^{-\Delta t/\tau} \tag{10}$$

and Δt is the simulation time step.

If $\Delta t \ll \tau$, which is usually the case in simulation, then the simplified approximation $\lambda = \Delta t/\tau$ can be used.

For those not interested in the theory, all you need is Equation (9).

Since the first-order persistence tempers the response to the random influences, the driver, σ_d, needs to be greater than the desired variation on the disturbance.

In creating a simulation with a stochastic influence, the user would choose a time-constant for the persistence that is reasonable for the effects considered, and a σ_d-value that would make the disturbance have a reasonable variability. At each sampling, Equation (9) would provide the perturbation value for the variables x, p, e, c, or in that are used in Equation Set (7). In its first use, the values for $d_{j=1}$ in Equation (9) should be initialized with zero.

How should one choose values for λ and σ_d? First consider the time-constant, τ, in Equation (8). It represents the time-constant of the persistence of a particular influence. Roughly, $\tau \approx 1/3$ of the lifetime of a persisting event, (because the solution to the first-order differential equation indicates that after 3 time-constants, d_j has finished 95% of its change toward the final value. So, if you consider that the shadow of a cloud persists for 6 minutes, then the time-constant value is about $(1/3)\cdot 6 = 2$ minutes. Or you could use $\tau \approx \frac{1}{4}$ of the persistence period representing 98% of the time to change. Once you choose a τ-value that matches your experience with Nature and decided for a time interval for the numerical simulation, Δt, calculate λ from Equation (10).

To determine the value for σ_d, propagate variance in Equation (9). (You don't have to do it!) The result is Equation (11). Use your choice of the variability on variable z, σ_z (and λ from Equation (10), which is dependent on your choices for Δt and τ) to determine the value for σ_d. The subscript z represents any of the variables x, p, e, c, or in.

$$\sigma_d = \sigma_z \sqrt{\frac{2-\lambda}{\lambda}} \tag{11}$$

To choose a value for σ_z, the resulting variability on the z-variable, consider a range of fluctuation of the disturbance. You should have a feel for what is reasonable to expect for the situation that you are simulating. For instance, if it is barometric pressure, the normal local range of low to high might be 29 to 31 inches of mercury. If outside temperature in the summer, it might be from 70 to 95 ºF. If catalyst activity coefficient, it might be from 0.50 to 0.85. The disturbance value is expected to wander within those extremes. Using the range, R, as

$$R = HIGH - LOW \tag{12}$$

And the standard deviation, σ_z, as approximately one-fifth of the range, then

$$\sigma_d = \frac{R}{5} \sqrt{\frac{2-\lambda}{\lambda}} \tag{13}$$

Although I presented a lot of equations, the bottom line is very simple.

As a summary: To generate stochastic inputs for dynamic simulators, choose a time-constant for persistence of events and a range of the disturbance variable. Use Equation (10) to calculate λ, then Equation (13) to calculate σ_d. Then, at each simulation time interval, use Equation (9) to determine the perturbation to a variable.

Figure 6 is an illustration of one 400-min realization of a stochastic variable calculated as above. The time-constant is 40 min, the nominal input value is 20, and the range is 3 units. In this one realization of the drifting influence, notice that the variable averages about its nominal value of 20 and that the difference between the high and low values is nearly the specified 3 units. From a period between 150 and 275 min. the value is below the average, indicating a persistence of 275-150=125 min., about three times the specified 40 min. time-constant. Some persistence values are shorter, some longer.

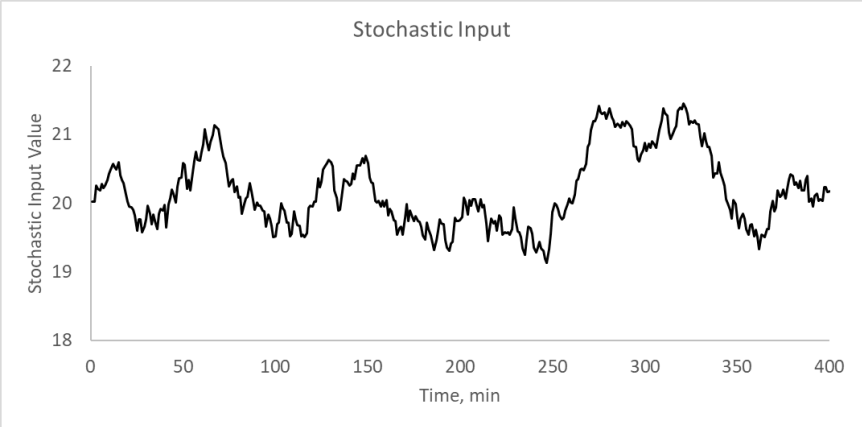

Figure 6 – Illustration of generating a stochastic influence for a process

Note: This approach modeled a drifting influence as a first-order response to a Gaussian distributed driver. One certainly could model drifts as higher-order or driven by alternate forcing function distributions. However, my view is that this adequately mimics how I consider randomly driven drifts in natural disturbances to respond, and it provides a relatively simple method to both execute and determine reasonable parameter values.

Differentiating Noise and Drift

As sampling rate increases, what appears to be noise could appear as an autocorrelated disturbance. Consider incomplete mixing in a pipeline where titrant is injected into a wild flow, or hot and cold water are mixed, or a concentrate is being diluted. Packets of rich and lean fluid pass through the analyzer sensor. Consider an optical analyzer that has inconsequential lag. If rich and lean packets are randomly sequenced and pass by at a rate of ten per second and the analyzer reports a value once every second, then it "sees" every tenth random packet. Then what affected the last measurement has completely disappeared from view, and the new measurement is completely independent of the prior measurement. The measurement sequence will be independently distributed (random) and appear as noise.

However, if the analyzer reports values on a milli-second rate, then sequential measurements will be influenced by the same packet, which has a tenth of a second residence time in the analyzer view and persists in the view for 100 samplings. In this case the measured data series will be autocorrelated, with a persistence of a tenth of a second. Because the analyzer will be observing several packets simultaneously, passing at different speeds, with irregular shapes, and various compositions, the signal will appear as an autocorrelated drift driven by Gaussian noise (many random events).

Also, if the measurement device has substantial lag, such as a thermocouple in a thermowell, or a filter on an orifice measurement device, then what should appear as independent noise will be smeared by the measurement lag, and the measurement appear as an autocorrelated drift.

Also, if the phenomenon is truly an autocorrelated drift, but infrequently sampled, the measurement could appear as random noise. Look at Figure 6, and consider that the sampling is at 50-minute intervals. The data will appear to be uncorrelated noise.

The choice of sampling rate or measurement device could make noise look like drift or drift look like noise. This tutorial has noise and drift modeled as separate phenomena. The reality may be that they are idealistic views of a single mechanism.

The simulation developer should have the process experience that permits a legitimate choice as to whether uncorrelated noise or autocorrelated drift or both should be considered.

Simulating Stiction

Stiction is a slip-stick phenomenon common to pneumatically activated flow control valves. The valve stem must slide through the valve body to open and close the inner valve. This hole in the body could permit process material to leak out, so there is packing material between the stem and body. If the packing is too loose, process material can slip out. But if it is too tight, the packing can grab and hold the valve stem. Normally, when the controller wants to open the valve a bit, the air pressure on the actuator changes a bit, and the force imbalance between the actuator and the spring permits the stem to slide to a position that makes the forces balance. But if the packing grabs the stem, the stem will not move until the force imbalance overcomes the static friction. Then the valve stem starts moving, and since sliding friction is less than static friction, the stem jumps to a new position, stopping at random position between the current location and the ideal target where the static friction again overcomes the actuator-to-spring force imbalance.

Note: I described stiction as the result of tight packing. But the cause of stiction might be corrosion or paint or a slight bend on a part of the valve stem.

A convenient and realistic way to simulate stiction is to separately model the target stem position (that desired by the controller), x_{target}, and the actual stem position, x_{actual}. If the difference between the target and actual is less than a threshold difference, then the actual position does not change. However, if the difference exceeds the threshold, Δ, then the actual jumps to a random spot near to the target.

$$IF\ \left(\left|x_{target} - x_{actual}\right| < \Delta\ \right) THEN(\ x_{actual} := x_{actual}) \tag{14}$$

$$IF\ (|x_t - x_a| \geq \Delta\ )\ THEN(\ x_a := x_a + (1 + r)(x_t - x_a)/2) \tag{15}$$

In Equation (15) the target and actual subscripts are replaced with "t" and "a". The r represents a uniformly distributed random number in the 0 to 1 interval. The $(1 + r)/2$ functionality in this model permits the valve stem to jump at least halfway to the target prior to stopping, randomly. The ≔ notation indicates a computer code assignment statement, in which the "prior" and "new" subscripts are unnecessary.

The user would specify the threshold, Δ. And certainly, the user could make the Δ-value dependent on the stem position (modeling the local impact of defects) or the jump-to range larger or smaller.

Ideally in a SISO application, stiction is evidenced by a square wave (alternating steps) in the CV and a corresponding sawtooth wave (alternating ramps) in the MV. The pattern might not be exactly regular, and it may be shifted by a delay or tempered by a filter. The evidence may be more complicated in a multi-variable process and with continually drifting disturbance effects.

Simulating Lost Motion

Lost motion happens when there is play in mechanical linkages. This could be on a rotary or butterfly valve in the process industry, or in positioning devices in mechanical or robotic devices. With play or slop in linkage, the final element follows the target position always behind by a small value. Then, when the target position reverses, the final element does not change until the linkage slop is removed and the final element can begin to move. Often this is termed deadband, but lost motion is more appropriate because the term deadband normally refers to the band around a set point used to trigger on-off control action.

A convenient way to simulate this is to separately model the target final element position (that desired by the controller), x_{target}, and the actual position, x_{actual}. If the difference between the target and actual is less than a threshold difference, then the actual position does not change. However, if the difference exceeds the threshold, Δ, then the actual follows the target with a difference of Δ.

$$IF\ \left(\left|x_{target} - x_{actual}\right| < \Delta\ \right) THEN(\ x_{actual} := x_{actual}) \tag{16}$$

$$IF\ (x_t - x_a \geq \Delta\ )\ THEN(\ x_a := x_t - \Delta) \tag{17}$$

$$IF\ (x_t - x_a \leq \Delta\ )\ THEN(\ x_a := x_t + \Delta) \tag{18}$$

In Equations (17) and (18) the target and actual subscripts are replaced with "t" and "a". The user would specify the threshold, Δ. And certainly, the user could make the Δ-value dependent on the final element position (modeling the local impact of rotary arm angle).

A View of a Control Loop

Figure 7 presents a way to include noise and disturbances in the control loop. From left to right, the actuating error is the deviation between the set point and the measured value. The controller acts on the actuating error to calculate a desired value to go to the process. But several devices (D/A, i/p, valve actuator) all must be perfectly calibrated for that exact message to go to the process. c_u represents the calibration error on field devices, and u_{actual} would additionally be affected by truncation, stiction, and lost motion (not explicitly shown). The process responds to u_{actual}, but also to all the other inputs that affect the process as illustrated in Figure 4. Finally, the output of the process simulator cannot be known exactly, because it is measured. The measurement has calibration error, possible drift, noise, and it would be digitally discretized to generate the $y_{measured}$, the "lie" that is returned to the controller.

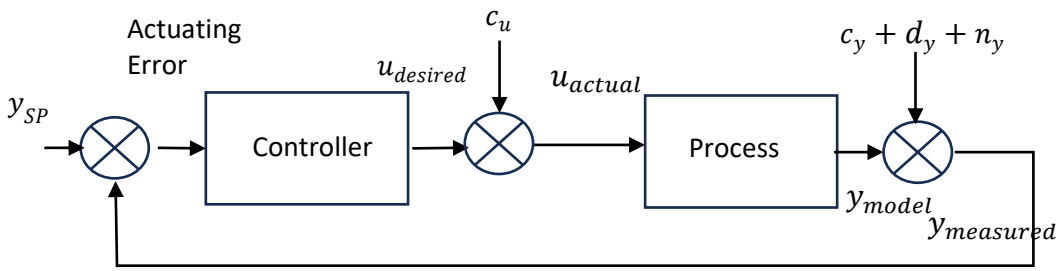

Figure 7 – A view of a control loop

Evaluating Process Variance

Under control, a deterministic simulator will asymptotically approach a steady state. At some point the deviation from an ideal steady state value will be so small (less than a pixel, or less than a discretization) that the process will have seemed to hit steady state.

However, a stochastic process will have the signal confounded by noise and continually perturbed from the ideal steady state by ever changing disturbance, environmental, and parameter values.

As Figure 8 illustrates the controlled stochastic process never settles to a steady state but fluctuates about it. The set point (blue) makes a step-and-hold at a time of 200 s. The fuzzy green band is the process response, the controlled variable (CV). It wiggles with about a 15-sec. persistence representing disturbances, and the fuzzy nature with some spikes represents the measurement noise. The manipulated variable (MV) (controller output) is the red curve that has persisting deviations, which are also of about 15 sec. countering the disturbance effects on the CV. Note that in the 250-300 and in the 350-400 sec. periods the MV had a value of about 40%, but in the 325-350 sec. period it averaged about 60%. The disturbance to the process was considerable, requiring strong control action.

Note: The impact of the disturbances is not cyclical (not a sine function, not periodic, not a uniform amplitude). And neither is the MV response. Frequency analysis is inappropriate to analyze this stochastic behavior.

The dashed line represents a constraint or a specification limit, which cannot be exceeded. Observe that the controller is strong enough to prevent a constraint violation. If the controller were designed to be less aggressive, the disturbance affecting the process at 350 sec. would have caused the CV to violate the constraint. Additionally, if the set point were closer to the constraint, then the constraint would have been violated at a time of about 350 sec.

The term "quality giveaway" refers to the distance away from the constraint that the set point must be to prevent a violation. Whether the impact is increased quality or a safety margin, it generally means higher energy use and a more costly product. The manufacturer does not like to operate on average conditions that give away quality, prevent occasional specification violations, or require a safety margin. So, a key metric for goodness-of-control is the standard deviation of the controlled process response during steady conditions when near to the constraint, which defines the offset between specification and set point.

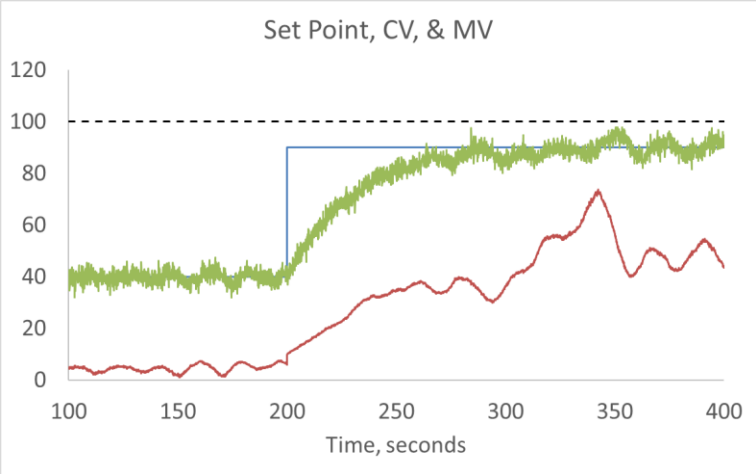

Figure 8 – Illustration of a controlled stochastic process.

One of the classic metrics for goodness of control is the integral of the squared error, ISE, which would become infinite because the stochastic process never settles. In actuality, ISE would be numerically estimated from time $t1$ to $t2$ with Δt representing the sampling interval. Equation (19) uses the rectangle rule of integration to estimate the integral. But, the ISE value would be proportional to the collection time interval $(t2 - t1)$.

$$ISE = \int_0^\infty e^2\, dt \cong \sum_{t1}^{t2} e^2\, \Delta t = \Delta t \sum_{t1}^{t2} e^2 \tag{19}$$

If the process is at steady conditions, averaging at the set point so that $y_{SP} = \bar{y}$, normalizing ISE by the collection time results in the process variance.

$$nISE = \frac{\Delta t}{(t2-t1)} \sum_{t1}^{t2} e^2 = \frac{1}{N} \sum_N e^2 = \frac{1}{N} \sum_N (y_{SP} - y)^2 = \frac{1}{N} \sum_N (\bar{y} - y)^2 = \sigma^2 \tag{20}$$

From which the standard deviation can be calculated.

$$\sigma = \sqrt{nISE} = \sqrt{\frac{1}{N} \sum_N e^2} \tag{21}$$

If one wants to place the set point far enough from the constraint so that there is only a 0.0001 chance of a violation, then Gaussian statistics indicates that the set point should be about 3.5σ

from the constraint. σ is the key value in determining quality giveaway, the set point offset from the constraint.

Equation (21) is predicated at the process being at a steady state. So, one should not use data from the transient of about 200 to 300 sec. to calculate the σ. Start collecting $nISE$ data when the process settles (about at 300 sec.) then collect data for a long enough time to be sure that all sorts of confluences of disturbances are reflected in the value of σ. The two choices, when to start collecting data and when to stop, are subjective; but longer collection period will lower the importance of either choice.

What to Do and Not Do

Don't learn by memorizing these messages or studying the math behind the derivations. Learn by implementing the techniques in a simulator and exploring their impact. Implement one thing at a time exploring how user choices (τ, Δ, σ, etc.) affect the signals.

When you are using simulations to guide technique development, for training or teaching, for demonstration, or for validation or marketing of a new algorithm, do not use deterministic simulators. For credibility and legitimate understanding, add realism to deterministic simulators to better match the vagaries that Nature will impose on your process.

- - - - - - - -

This tutorial is an enhanced version of articles in CONTROL magazine, Vol. 37, No. 9, September, 2024, pp 33-34, and Vol. 37, No. 10, October, 2024, pp xxxxx. It is also excerpted from Chapter 3 of Rhinehart, Nonlinear Model-Based Control using First-Principles Models in Process Control, International Society of Automation, Durham, North Carolina, 2024, ISBN 13:978-1-64331-242-2.

Russ Rhinehart started his career in the process industry. After 13 years and rising to engineering supervision, he transferred to a 31-year academic career, serving as the ChE Head at Oklahoma State University for 13 years. Russ is a fellow of AIChE and ISA. Now "retired", he enjoys coaching professionals through books, articles, short courses, and postings on his web site www.r3eda.com. He can be reached at russ@r3eda.com.