Laplace Transforms for Process Control

CONTROL Global

R. Russell Rhinehart, 2018-05-09

Preface

One can argue to not teach students to derive or invert Laplace, or z-, or frequency transforms in the senior level process control course. In my 13-year industrial career, I never used mathematical transforms; but I tuned controllers and filters, designed a ratio control system, decided what variables needed to be controlled, sized an orifice and chose its dP cell range, did trouble-shooting, chose DCS and device manufacturers, assessed patterns in SPC charts, and analyzed process attributes. Further, mostly, the daily operation of controllers (tuning and switching modes) was done by technicians, not engineers. When I had the privilege to tune a controller, I followed a heuristic procedure, and did not use mathematical transforms. There are many process control topics that are more important to career skill than the ability to use the partial fractions method to invert a Laplace transform. The process control course should not reject the practice-relevant aspects of process control to become an advanced mathematics course.

On the other hand, Laplace transfer functions and block diagram notation are efficient methods to describe dynamic behaviors, and are often the representation of the algorithms in filtering and control as presented in vendor bulletins and instruction manuals. They have become our historical legacy language of representing process dynamics. Accordingly, a control engineer, should understand what transfer functions mean; especially relating to FOPDT models, filtering, PID controller options, and feedforward methods.

Further, transfer function analysis has provided the theoretical underpinning of many control techniques and remains an important analysis and communication tool for those of us doing research in dynamic systems. So, surprisingly to most in process control, those in research or development actually think Laplace transform notation is useful.

This four-part monograph is how I presented the Laplace transform topic in the process control course to chemical engineering seniors. The first part is about deviation variables, the second is about the Laplace transform and relates transfer functions to operations, the third part is how to interpret block diagrams, and the fourth part is for those wishing to convert transfer functions into code to simulate a process or a controlled system. Hopefully, this will have utility to my readers.

Contents

Part 1 Deviation Variables

Introduction

Mathematically, a Laplace transform is represented as the integral from time zero (now) to forever into the future (infinity) of the signal weighted by the exponential term e^{-st}.

$$Y(s) = \hat{y}(s) = \int_{t'=0}^{t'=\infty} e^{-st'} y'(t') dt' \qquad (1.1)$$

Do not look at Equation (1.1) as mathematics. If you do, you'll revert to your Calculus II course perspective, and become mentally misdirected. This monograph is not about calculus. We'll understand the meaning of the equation in Part 2 of this series, but first, you must understand what the symbols represent.

Here, the symbol y represents the process variable (PV) value. However, y' is in the equation, not y. The y' represents the process variable (PV) deviation from its initial value; so, y' starts with a value of zero. In the equation, it is explicitly indicated as a function of time, $y'(t')$, also a deviation. The deviation notation is essential to simplifying the Laplace transfer functions, but it can cause a conceptual barrier; so this first part is about deviation variables.

Deviation Variables

To help explain deviation variables, consider that your car is stopped, facing east, and then it accelerates to 50 mph. Its initial speed was zero. Or was it? If one considers the rotation of the Earth, the initial speed of a stopped car may have been about 1,000 mph (near the Equator), and the car then accelerated to 1,050 mph. Claiming that the speed is 50 mph is grounded in a deviation from the Earth's surface speed. Facing west, it would have decelerated to 950 mph. Of course, the Earth is moving as it rotates, so the actual speed relative to something is different yet. However, we are comfortable considering speed as a deviation from the Earth surface, as if it were a stationary reference.

As another example, if the PV is temperature and starts with a value of 32 °F, one could equivalently consider that it is starting at 0 °C. But, is 0 °C really zero, or is that temperature 273.15 °K? Both the Fahrenheit and Centigrade scales are deviation variables. The height of a desk is about 32 inches, but this is relative to the floor, not sea level. At 980 feet above sea level, my desk at home on the first floor has a height of 982.67 feet, but downhill a bit in my school office, and up on the 4th floor, it is about 960 ft. Temperature and level of material in a tank, are deviations from a user-defined reference. But, in fitting a chair for my desk, no one would say that the height of my desk is 960 ft.

As one final PV example: You are familiar with the difference between absolute pressure and gage pressure. The steam in a 50 psi line (gage pressure, psig) actually has about a 64.7 psia (pressure absolute) value. The 64.7 value defines the steam properties. The gage pressure is a deviation from the atmospheric pressure. But, from day to day, as the barometric pressure changes, the same 50 psig, might represent 66 psia on one day and 63 psia on another. The reference for the psig deviation is not a constant. Similarly, there is no need for you to fix your reference value of the PV to a universal absolute. Regardless of the initial value at the time, you can define the scale, translate the axis, to start with a value of zero.

For transforms and transfer functions, again for convenience, to simplify them, we'll also require that $y(t = 0)$ be at an initial steady state.

Time, t', is also a deviation variable. "Now" could be considered 3:12 pm on October 7th 2019, instead it could be called 15:12 hours (pretending midnight was zero), or 2019.76712… years A.D., or we could reference the time to the big bang. For instance, if the race ended after the winner crosses the finish line at 10:37:16.492 am, we call the winning time as 43.2 seconds, because we consider that the race started at time zero. The event starts at time zero.

Example 1.1

I think that the concept of deviation variables is often missed by students when they are trying to learn the language and models of control. So, here is an example using a traditional exercise of generating a first-order plus deadtime (FOPDT) model, which is commonly used as a simple representation of process behavior for tuning feedback and feedforward controllers and decouplers. From an initial steady state, the concept is that the process is influenced by a step-and-hold influence. The influence is usually the controller output, but it might be a disturbance. Figure 1.1 shows the idealization of influence, u, and Figure 1.2 shows the ideal PV response, y. For clarity of concept, the signals are noiseless.

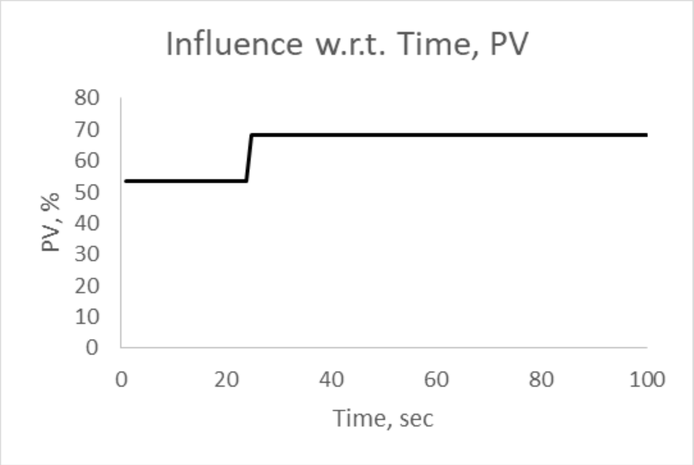


Figure 1.1 – Ideal step-and-hold influence from an initial steady state

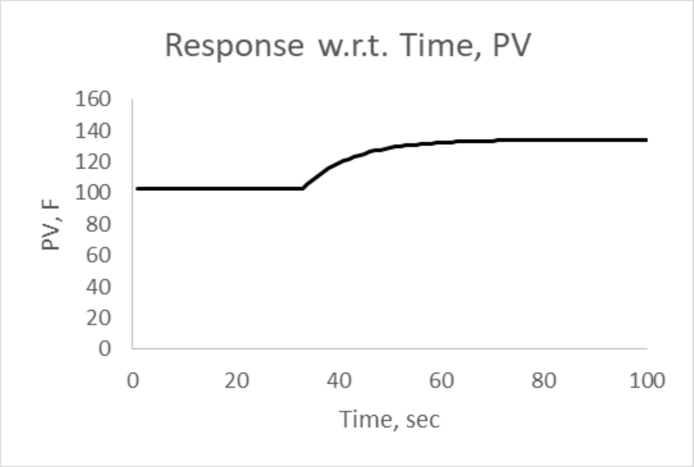


Figure 1.2 – Ideal FOPDT response from an initial steady state

Note that the time axis on both figures goes from zero to 100 seconds. However, the step-and-hold event occurred at a time of 25 seconds. It did not occur at time zero. In actuality, the zero time on the axis might have been at 2:31 pm when data collection started. Regardless of when data collection starts, we will reference the deviation time, t', from the start of the event, in this case the step change in the influence.

Further, in Figure 1.1, the initial steady state value of the influence was 53.6 % and it changed to 68.4 %. In contrast, Figure 1.3 shows the deviation variable representation of the influence. In deviation variables, we will use the initial steady state value as the reference, and claim that the influence changed from zero to a value of 14.8 %' (14.8 = 68.4 - 53.6), and that it happened at a time of zero. Note that the magnitude of the change was 14.8 %, but to explicitly indicate it is relative to the initial value, I used the symbol %' for its units.

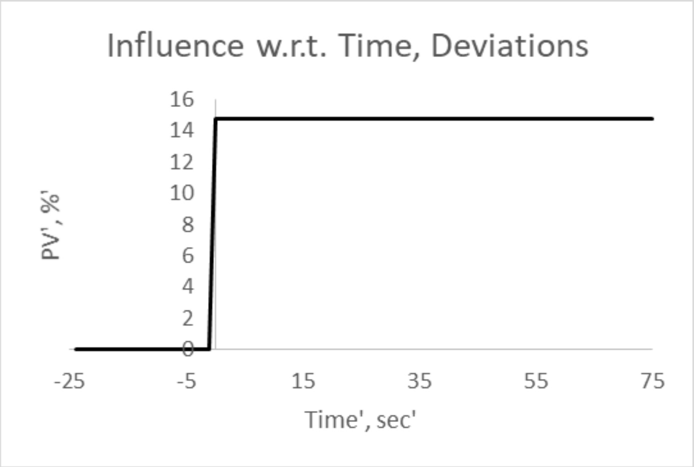


Figure 1.3 – Figure 1.1 in deviation variable notation

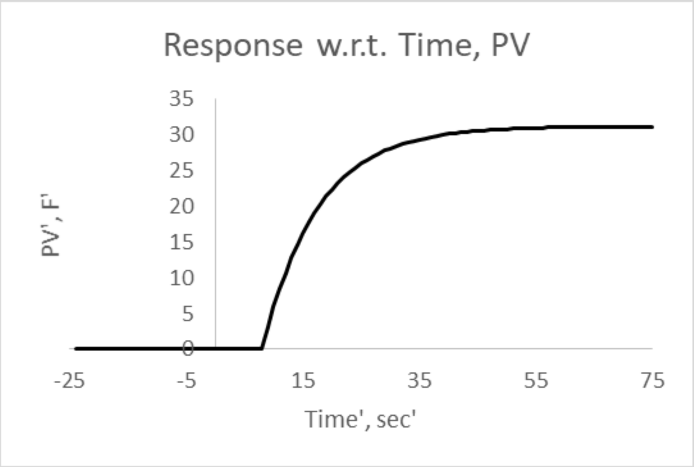


Figure 1.4 – Figure 1.2 in deviation variable notation

In Figure 1.2, the PV response rose from 102.9 °F to 134.1 °F. But, in deviation variables from its initial steady state value, it rose from zero to 31.2 °F'. The temperature is not a bit below freezing; so, rather than using °F units, to explicitly indicate the PV is a deviation, I use °F'. Figure 1.4, reveals the PV in deviation variables.

Looking at Figure 1.2, one might decide that the PV settling time is 80 seconds, which is about when it stopped rising. However, it did not take the PV 80 seconds to settle from the influence at 25 sec. Figure 1.4 more clearly represents the settling time as about 55 seconds.

Additionally, the final values of the deviation variables are 14.8 and 31.2 as shown in Figures 1.3 and 1.4, not 68.4 and 134.1. Since transformed variables are based on deviation variables, in un-transforming notation, one must convert the deviation variable to the real variable, by adding the reference value. Again, this shift is often not included in a novice student's analysis.

A Perspective on Modeling

Often, we use FOPDT models to characterize a process response. However, real processes do not have ideal FOPDT response as implied in Example 1.1; and further, the influences are not ideal step-and-hold. When we describe a process with a FOPDT model, it is an attempt to make a FOPDT response best fit the data. The best FOPDT model to represent a real PV response is shown in Figure 1.5. The influence is the thin line that makes several steps and appears similar to a city skyline, and accordingly termed a skyline function. It does not make one step-and-hold. Most new influence steps happen before the response settles to its new value, and the jump-to value is random within a desirable range. The influence is attached to the right vertical axis. The dots represent experimental data, and the line that somewhat goes through it is the FOPDT response. Both PV response and FOPDT model are attached to the left vertical axis.

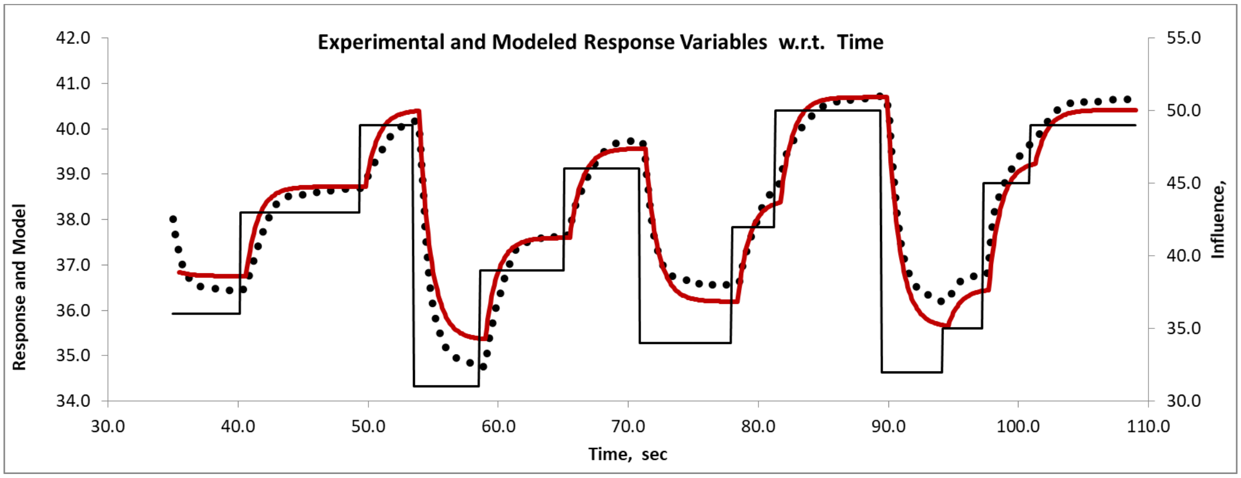


Figure 1.5 – Real FOPDT modeling

Note that the model does not exactly match the data. There is no rule requiring Nature to comply with human artifices. Just because we choose to mimic reality with a FOPDT model does not mean that it is the right model. In this case the process is nonlinear, the gain (how far it moves) in one PV range is different larger than in the other PV range, the valve has sticktion so the influence is not exactly the controller output. Lastly, there seems to be a wild disturbance progressively shifting the process values to higher values over the duration of the experiment. By contrast, the model pretends the gain is linear, the influence is what we think it should be, and there is no uncontrolled disturbance.

In general, the responses are not free of a collection of nonidealities (noise, disturbances, sticktion, nonlinearity, etc.); so, a single step-and-hold experiment will not represent the general behavior. Further, a single step-and-hold presumes starting from an initial steady state and holding the new value long enough to attain a new steady state. Waiting long enough pushes the process away from a desired value for an extended period. Accordingly, it is best to use a skyline influence, averaging about the nominal value and not waiting for a steady state, and fit the data over a series of shorter step-and-hold durations. See the article "FOPDT Modeling", Develop Your Potential Series in CONTROL magazine, November 2016, Vol. XXIX, No. 11, pages 46-48. I have made FOPDT software available to the public on my web site: www.r3eda.com.

Part 2 Laplace Transforms

Introduction

Perhaps the greatest value of Laplace transforms is in simplifying the mathematical analysis of a system of differential and integral equations by converting them to algebraic relations. Although that is largely irrelevant to a practitioner, Laplace transforms have provided the theoretical underpinning of many control techniques and have become the standard representation of process and controller dynamics. They are useful as a short-hand notation, and a control technologist should be able to read the language. This second part of the monograph explains the Laplace transform and relates transfer functions to common dynamic behaviors.

Mathematically, a Laplace transform is represented as the integral from time zero (now) to forever into the future (infinity) of the signal weighted by the exponential term e^{-st}. Duplicating Equation (1.1):

$$Y(s) = \hat{y}(s) = \int_{t\prime=0}^{t\prime=\infty} e^{-st\prime} y'(t')dt\,' \qquad (2.1)$$

Again, do not look at Equation (2.1) as part of your Calculus II course.

Again, it is important to understand what the symbols represent. The variable t represents time and y represents the process variable (PV). Again, however, they are indicated by t' and y' in the equation. These are deviation variables. t' is the time after the initiation of an event that causes y to change, and y' is the PV deviation from its initial steady state value.

Either the capital Y or the y-hat $\hat{y}$ symbol represent the Laplace transformed variable. As, I'll show, Y and $\hat{y}$ are not functions of time, even though the signal changes in time. Both $Y(s)$ and $\hat{y}(s)$ depend on the value of the coefficient s; so, in the equation, Y and $\hat{y}$ are explicitly shown as functions of s, not t.

The argument of the exponential must be dimensionless. Accordingly, the units of the coefficient s are the reciprocal of the units on time, t'. If you choose time in minutes, then the units on coefficient s are min^{-1}. If time is in seconds, then the units on coefficient s are sec^{-1}, or s^{-1}, an unfortunate nomenclature that can create confusion between the symbols representing the coefficient and its units.

Interpreting the Equation

I am going to provide a geometric interpretation of Equation (2.1) because it illustrates several aspects, not because it is how a transfer function is obtained. The aspects are that 1) $\hat{y}$ is a function of s, 2) not time, and 3) the units on $\hat{y}$ have no physical meaning. The term $y'(t)$ is the process signal that changes in time, as a deviation from its initial value. Figure 2.1 is an illustration of a temperature that rises in an S-shaped manner from an initial steady value of 87.3 ^{o}F. It approaches a final value of 108.6 ^{o}F, but since the graph is finite, it never quite gets there. It happens to be an ideal second-order response (with time-constants of 14 and 16 seconds) to an ideal step-and-hold influence. Notice that the graph is in deviation variable terms – both the response and time start with a value of zero, and $t' = 0$ is the time at the initiation of an event that causes y to change from its initial steady state value. So, the signal in the graph is approaching 21.3 ^{o}F' (=108.6-87.3).

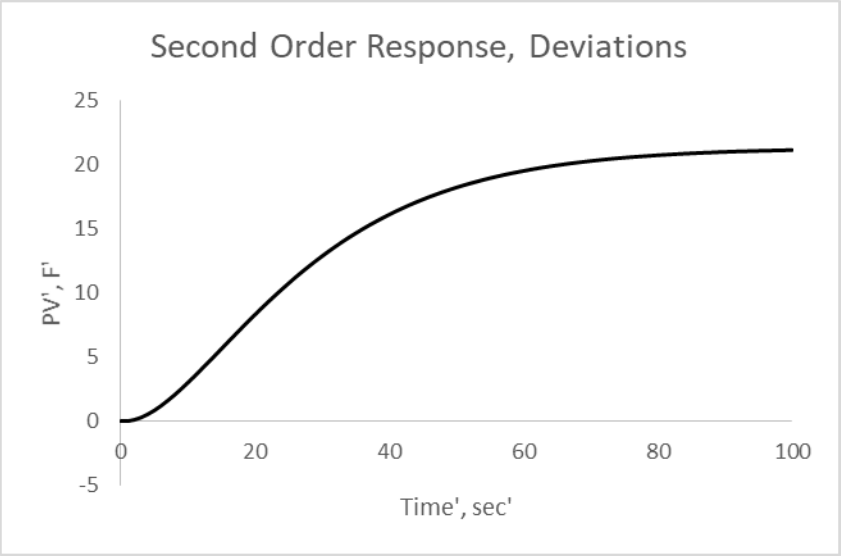


Figure 2.1 – A Second-order response in deviation variables

The term $e^{-st'}$ is a simple exponential decay. When $t' = 0$ its value is $e^{-s0} = e^{-0} = 1$ regardless of the choice of the value of coefficient s. And, when $t' = \infty$ its value is $e^{-s\infty} = e^{-\infty} = 0$ regardless of the choice of the value of coefficient s. Figure 2.2 indicates the exponential decay for several choices of the coefficient s. Notice that regardless of the s-value, each curve starts at 1.0 and tails toward 0.0.

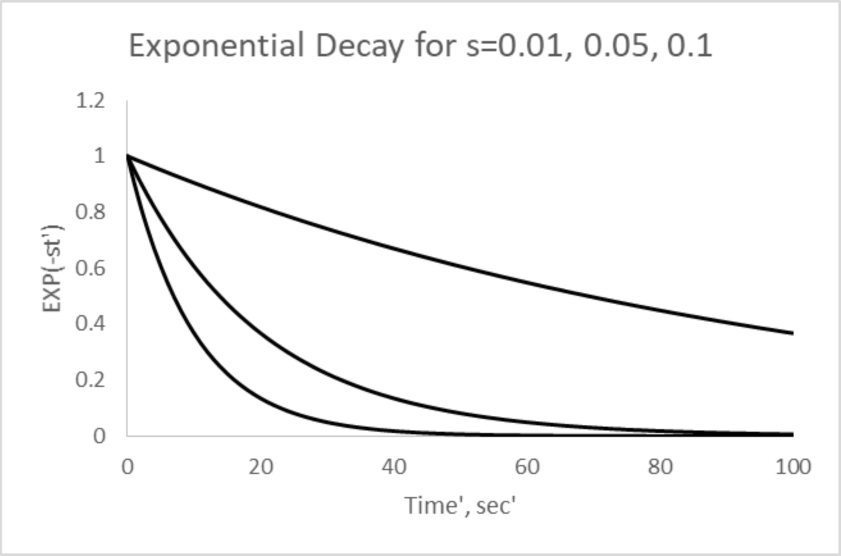


Figure 2.2 – Illustration of an exponential decay for several s-values

The argument of the integral in Equation (2.1) is a product of terms, $e^{-st'}y'(t')$. This is easily computed. Choose a value for the coefficient s, say the middle curve of $s = 0.05$ s^{-1}. At a time of 40 seconds the value of $e^{-st'}$ is 0.135 (dimensionless) and the value of $y'(t')$ from Figure 2.1 is 16.1 F' so the product is 2.18 F'. Figure 2.3, indicates how the product $e^{-st'}y'(t')$ changes with time, when the value of s is 0.05

s^{-1}. The integral argument value, the product value, is the solid line, which is attached to the left vertical axis. The exponential term is dimensionless, so the product retains the units of the process variable.

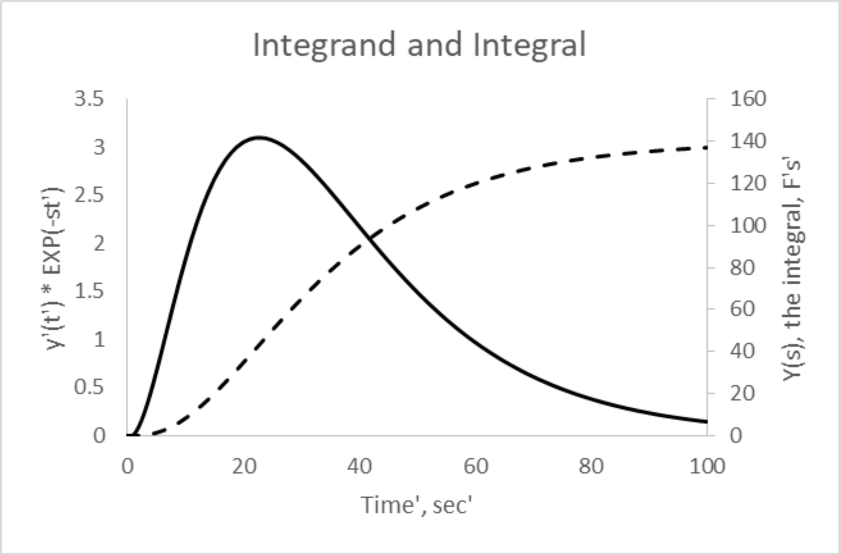


Figure 2.3 – Integrand and Integral

The integral symbol, simply means the area under the product curve with respect to time, from the start (at zero) to the end. The end would be at time of infinity. The integral is an area, a product of the height times time. Note: The units of the area are not the same as the process variable. If the PV represents temperature in °F and time is represented in seconds, then the integral has the units of Fahrenheit'-seconds' (F's'). Although both time and temperature have a physical meaning, there is no physical meaning to the transformed variable, Y, or its units. The dashed line in Figure 2.3 is how the integral increases with time (with $s = 0.05$ s^{-1}). For example, the area under the solid curve from $t' = 0$ to $t' = 60$ is about 120 F's', so the value of the dashed curve at $t' = 60$ is about 120 F's'. The value of the Laplace transform with $s = 0.05$ s^{-1} is about 140 F's'.

The s-coefficient could have any value. And, for each value, there is a different total area, asymptotic value of the area at $t' = \infty$. The total area is represented by the capital Y, and depends on the value of s. The value of the total area can be graphed w.r.t. s as in Figure 2.4.

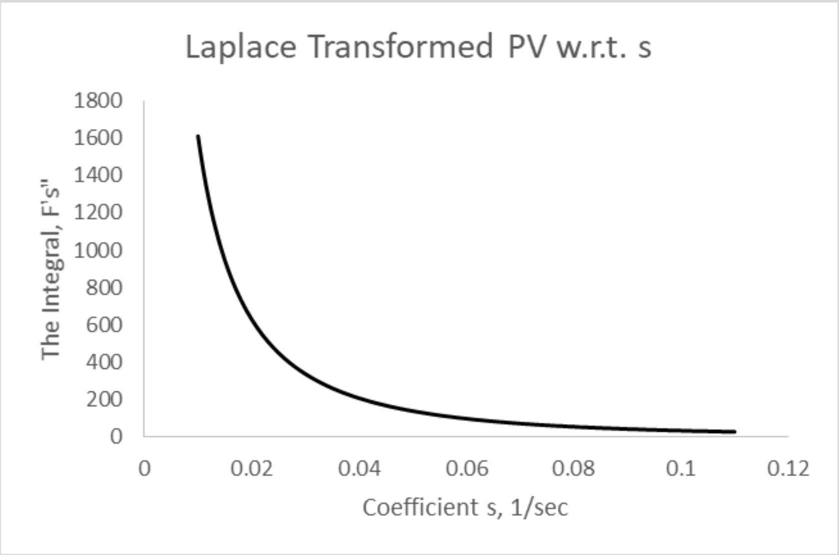


Figure 2.4 – The value of Y(s) w.r.t. s for a second-order response

Note: In Figure 2.4, at $s = 0.05$ s^{-1}, the total area is about 140 F's', consistent with the dashed line in Figure 2.3.

The $Y(s)$ w.r.t. s graph is unique to the $y(t)$ response, but it is not the $y(t)$ response, it does not have the same units, and it has no physical meaning. Like a fingerprint, it is unique to the person, but it is not the person. The fingerprint does not provide facial or voice recognition, nor have the same characterization of the person it uniquely represents.

In this case, the equation for the fingerprint, for the second-order trend in Figure 2.4, which is the Laplace transform of the PV trend in Figure 2.1 can be analytically derived. It is:

$$Y(s) = \frac{21.3}{(14s+1)(16s+1)s} \tag{2.2}$$

Note: The asymptotic value of 21.3 and the time-constants of 14 and 16 are both explicitly revealed within the s-functionality. The algebraic pattern of the Laplace transform indicates the second order response, and the coefficients in the transform are the specifics.

The $Y(s)$ function is a unique fingerprint to the $y(t)$ response, and can be represented by a formula as well as the graph of the function. For a ramp change in a PV (in which $y(t') = a + bt'$), the initial value is $y(t' = 0) = a$, the y-deviation is $y'(t') = bt'$, and the Laplace transform of $y'(t')$, the unique fingerprint of the ramp, the analytical integration of Equation (2.1), is $Y(s) = b/s^2$. Long ago in calculus class, you might have been able to derive that. But, there is no need now; unless, like doing Sudoku puzzles, you enjoy the intellectual exercise of analytically solving the integral. Noteworthy, is that the numerator in the Laplace representation is the ramp rate of change, b, and the s-squared functionality in the denominator is just another language for "ramp". If you see b/s^2 read it as "The PV makes a ramp change at a rate of b from its initial value."

The Laplace transform is just another language representation. If this English line, "The PV makes a ramp change at a rate of b from its initial value" was written in Spanish or Chinese or Laplace, it would represent the same trend. If you understood the language, you could sketch the trend. Alternately, if you saw a graph of the trend, you could describe it in English, or Laplace, or any language of communication.

Some $y(t)$ responses are relatively simple, and the corresponding $Y(s)$ equation can be “easily” obtained by analytical solution of Equation (2.1). They’ve already done it, and placed the formulae in tables, so you don’t have to re-do it.

Transfer Function

The presentation above describes the Laplace transform of a signal, but it does not indicate why the signal was changing. For control, of more importance is the Laplace transfer function, which relates how one signal reacts to another. Consider a relation such as $y(t) = f(u(t))$, meaning that y is a response and u is the influence. The response behaviors have names such as gain, or integrating, or first-order; and time-domain (calculus and differential equation) representations that match the Laplace transfer function notation. Although the y-response depends on the input behavior, the function of how y responds to u is the same whether u makes a step up, or down, or a ramp or a sinusoidal change.

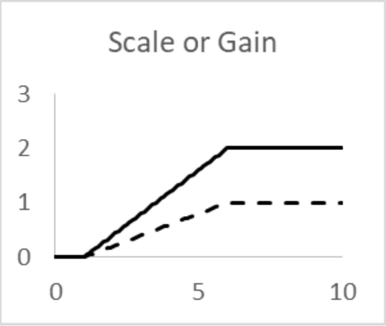
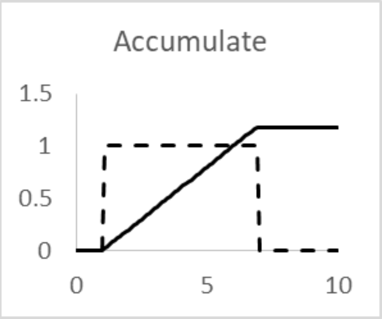
The transfer function describes the y response to any influence event pattern. In the words of Carlos A. Smith, “… it tells us how the input (influence) affects, or transfers, to the output.”

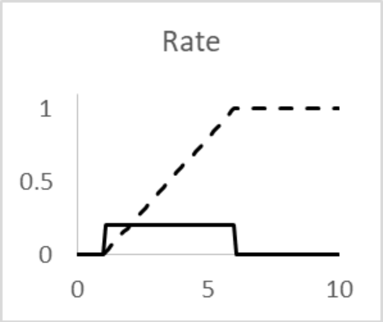
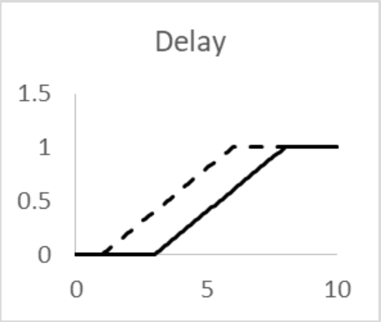
Key Fingerprints

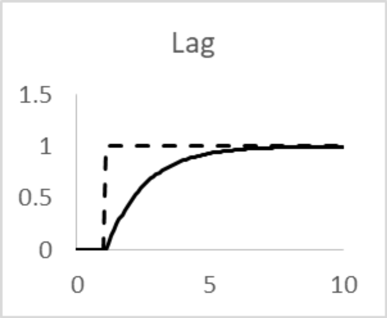
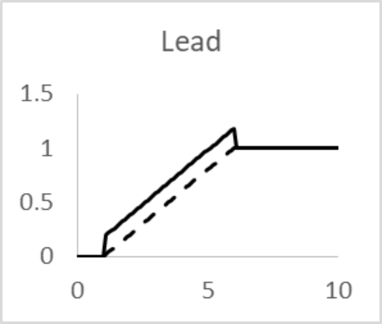
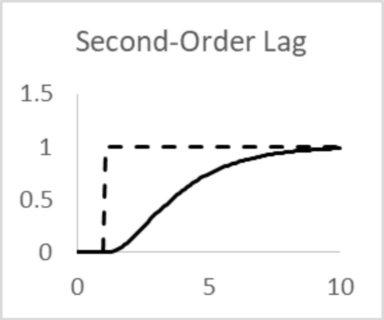
Table 2.1, reveals some key transfer functions. You should recognize the patterns, and what they mean. The left column is an English language description and some commentary. The middle column reveals the time domain (algebraic or differential equation) representation of the operation, and an illustration. In the illustration the dashed line is the influence, u', and the solid line is the process response, y'. The rightmost column reveals the Laplace Transfer Function of the operation, and includes the equation explicitly revealing what is the influence and what is the result.

Note: The time domain equations explicitly reveal that the variables are deviation variables. The transfer function representation, does not explicitly indicate deviation variables, but it is based on them.

Table 2.1 – Some common transfer functions

Name	Time Domain Equation	Laplace Transfer Function
Scaling, or Gain, by the multiplier factor K. y does exactly the same thing as u at the same time, but it is attenuated by the factor K. K may change both units and value, like a conversion of °C to °F. K may have a negative sign, indicating that y does exactly the opposite of u. Read it as "y is the scaled u," or as "y is a ratio of u."	$y'(t) = Ku'(t)$ 	K meaning $\hat{y}(s) = K\hat{u}(s)$ or simply $\hat{y} = K\hat{u}$
Integrating, or Accumulating. y progressively accumulates u. Like flow rate (in gpm) might fill a tank and volume (gallons) accumulates. Read it as "y is the time-accumulation of u."	$y'(s) = \int_{t'=0}^{t'=\infty} u'(t')dt'$ 	$1/s$ meaning $\hat{y}(s) = \frac{1}{s}\hat{u}(s)$ or simply $\hat{y} = \frac{1}{s}\hat{u}$

Rate, or Derivative. y is the rate of change of u. Like speed is the rate of change of distance or acceleration is the rate of change of speed. Read it as “y is the time rate of change of u.”	$y'(t') = \frac{du'(t')}{dt}$  	s meaning $\hat{y}(s) = s\hat{u}(s)$ or simply $\hat{y} = s\hat{u}$
Delay, or Deadtime, by an amount θ. y does exactly the same thing as u, but delayed by a time of θ minutes (or seconds). Like the 10 sec delay on a broadcast of a live televised event. Read is as “y is the same as u, but time-delayed by theta.”	$y'(t') = u'(t' - \theta)$  	$e^{-\theta s}$ meaning $\hat{y} = e^{-\theta s}\hat{u}$

First-Order, or a Lag, with a time-constant of τ. y lags behind u in a first-order (exponential) manner. Like car speed would rise and asymptotically approach a steady value when the accelerator pedal is changed. If u makes a step-and-hold the y-response settles to the new value in about 5 time-constants. It gets about 63% through the transient when $= \tau$, and 95% when $t = 3\tau$, and 99% when $t = 5\tau$. Read it as “y is the tau-lagged response to u.”	$\tau\frac{dy'}{dt}+y'=u'$  	$\frac{1}{\tau s+1}$ meaning $\hat{y}=\frac{\hat{u}}{\tau s+1}$
Lead by a time-constant of τ. y leads u. Like throwing the ball in front of a running receiver. u represents the receiver position and y the place to aim the ball. $\frac{du\prime}{dt}$ is the runner speed, and τ is the time for the ball to get to the target. Read it as “y is the tau-leaded u.”	$y'=\tau\frac{du'}{dt}+u'$  	$\tau s+1$ meaning $\hat{y}=(\tau s+1)\hat{u}$
Second-Order, with time-constant of τ_1 and τ_2. A first-order lag follows a first-order lag, and y is the second lagged response to u. A common example is two tanks in series. The first is being filled, while it empties into the second which also has an outflow. Read it as “y2 is the tau2 time lagged y1, which is the tau1 lagged u.”	$(\tau_1\tau_2)\frac{d^2y'}{dt^2}+(\tau_1+\tau_2)\frac{dy'}{dt}+y'=u'$  	$\frac{1}{(\tau_1 s+1)(\tau_2 s+1)}$ meaning $\hat{y}=\frac{\hat{u}}{(\tau_1 s+1)(\tau_2 s+1)}$

If not based on deviation variables with an initial steady-state, the functions in Table 2.1 are a bit more complicated.

Table 2.1 is an essential subset of behaviors. Less frequently you may encounter open loop unstable, oscillatory, inverse response, and such. You can find translation keys, similar to Table 2.1 in many process control textbooks.

Interpreting Transform Notation

And, how do you read a transfer function? Recognize that the mathematical representation is algebra. Use algebra rules to regroup the terms into recognizable transfer function elements, then use Table 2.1 to interpret each element, then state the whole. The objective is not to get the value of the transformed variable (which has no physical meaning, which is the fingerprint), the objective is to use the transfer function to define the procedure to get the value of the real process variable. You can manipulate transfer functions like algebra, but you don't do the arithmetic operations on the transfer functions to get the desired value. Here are some examples.

Example 2.1

This is a transfer function of an elementary filter. The input is the noisy process signal, y, and the output is the filtered value y_f. The transfer function, however, describes the mathematical relation between the non-physically-meaningful variables $\hat{y}_f$ and $\hat{y}$.

$$\hat{y}_f = \left(\frac{1}{\tau_f s+1}\right)\hat{y} \tag{2.3}$$

Using Table 2.1 y_f is the first-order lagged value of y with a time-constant of τ_f. You would get the value of $y(t)$ by your preferred method of solving the differential equation $\tau\frac{dy\prime}{dt} + y' = u'$. My preferred method is numerical, which Part 4 of this monograph shows.

Example 2.2

This is a Laplace Transfer Function description of the standard PID controller. How can one translate it to describe the $m(t)$ response to an $e(t)$ influence?

$$\hat{m} = K\left(1 + \frac{1}{\tau_i s} + \tau_d s\right)\hat{e} \tag{2.4}$$

Recognize that the mathematical representation is algebra. First, regroup the terms into recognizable elements. Multiply $\hat{e}$ to each term in the parenthesis.

$$\hat{m} = K\left(\frac{1}{\tau_i}\frac{\hat{e}}{s} + (\tau_d s + 1)\hat{e}\right) = K(\hat{T}_1 + \hat{T}_2) \tag{2.5}$$

The first term, $\widehat{T}_1$ should be recognizable from Table 2.1 as the integral of $e(t)$ (the accumulation of $e(t)$, note not $\hat{e}$) divided by the scale factor (a gain) τ_i. The second term is the leaded $e(t)$, how far in front to target its value (to forecast its τ_d future value) based on the current value of $e(t)$ (not $\hat{e}$) and its rate of change. Then, the sum of the two terms is scaled to get the value of m'. Here, the English language translation of the Laplace notation is, "m' is the K-scaled sum of the τ_d-leaded and τ_i-integrated value of e; then $m = m' + m_{ref}$."

Again, you do use algebra to solve Equation (2.5). You do not multiply τ_d times s then add 1 to the sum then multiply that by the value of $\hat{e}$ to get the value for $\widehat{T}_2$. Physically, the variable $\hat{e}$ is meaningless, and s could have any value. To determine the leaded value of e, use any method you prefer to take the derivative of e, multiply it by τ_d and add e to it. My preferred method is numerical.

Example 2.3

This is one of several possible descriptions of the rate-before-reset, or interactive PID controller.

$$\widehat{m} = K\left(1 + \tau_d s + \frac{1+\tau_d s}{\tau_i s}\right)\hat{e} \tag{2.6}$$

Again, simplify by multiplying each term in the parenthesis by $\hat{e}$

$$\widehat{m} = K\left((1 + \tau_d s)\hat{e} + \frac{(1+\tau_d s)\hat{e}}{\tau_i s}\right) \tag{2.7}$$

Now consider the $\frac{(1+\tau_d s)\hat{e}}{\tau_i s}$ term. The numerator is the leaded actuating error, the value $e(t)$ would have if extrapolated τ_d time into the future. Then the denominator indicates this value should be integrated and scaled by τ_i. So, $\frac{(1+\tau_d s)\hat{e}}{\tau_i s}$ represents the leaded-integrated-scaled actuating error. Then add this to the leaded e and scale by K, so that, "m' is the K-scaled sum of the τ_d-leaded actuating error plus the τ_d-leaded, integrated and τ_i-scaled actuating error. Finally, $m = m' + m_{ref}$."

Example 2.4

Here, we'll take an English language description and create the Laplace transfer function. If you describe a process as being first-order plus deadtime, it means that the response is the scaled value of the delayed and lagged influence deviation. If $u(t)$ is the influence, then the deviation is $u'(t') = u(t') - u(t' = 0) = u(t') - u_{ref}$. In Laplace notation the delayed u is represented as $\hat{u}_d = e^{-\theta s}\hat{u}$. The lagged delayed u is represented as $\hat{u}_{ld} = \frac{1}{\tau s+1}\hat{u}_d = \frac{e^{-\theta s}}{\tau s+1}\hat{u}$. And, finally the response is the scaled lagged delayed u, which is $\hat{y} = \hat{u}_{sld} = K\hat{u}_{ld} = \frac{K}{\tau s+1}\hat{u}_d = \frac{Ke^{-\theta s}}{\tau s+1}\hat{u}$. So, the Laplace transform description of a FOPDT response is:

$$\hat{y} = \frac{Ke^{-\theta s}}{\tau s+1}\hat{u} \tag{2.8}$$

Summary

For most of us, there is no need to use calculus to derive the Laplace transformed equation, and no need to use partial fractions to invert a transform. Once one learns what the symbols represent, it is as easy to state a cause-and-effect relationship in Laplace transfer function notation as in English, or any language that you might be accustomed to.

Of course, there has been much value provided to automatic control, by those who have used Laplace transform analysis to prove that the integral-mode of a controller can remove steady state offset, to generate tuning rules, to discover reset feedback as a method to prevent integral windup, etc. However, once that has been done by one theoretician, the rest of us don't need to re-do it on a daily basis. Undergraduates should know that techniques of process control have a theoretical underpinning, but preparation of undergraduates to be able to do the theory is a misdirection of what 99% of them need to know in the practice.

So, understand how to read Laplace transfer functions. Use Table 2.1 as your "Rosetta Stone" translation guide.

Part 3 Interpreting Block Diagrams of Functions

Introduction

You are familiar with algebraic notation for a sequence of mathematical operations. Unfortunately, this familiarity can be a barrier to reading and presenting Laplace Transfer functions. For instance, you might think that the "+" sign means "add". I'll illustrate block diagram notation and the familiarity traps with three examples.

Example 3.1

Here is a simple algebraic example.

$$y = 3(x - 1) \tag{3.1}$$

In English language, Equation (3.1) means, "Get the value of x, subtract 1 from it, multiply that result by 3, and that is the value of y."

Note: This may seem to be a trivial point, but you need to explicitly recognize this issue, because it will not be so obvious when interpreting Laplace transforms. You recognize that the parenthesis is not an operation, nor is it part of the symbols. For instance, "$3x$" would mean multiply 3 times the value of x, but the "$3($$x$" in Equation (3.1) does not mean multiply 3 times the value of "(" times the value of x. By practice, you understand how to execute this procedure and are not confounded by the symbolic representation. The parenthesis is a direction to the order of the execution of operations, and not an element in the execution.

We could illustrate this procedure of Equation (3.1) with a block diagram, as in Figure 3.1. I think that you can see that it directs the same procedure as does the equation or as does the English language statement.

The circle with the four segments indicates either addition or subtraction, as the signs for this application indicate. The square with the × indicates that the two inputs need to be multiplied.

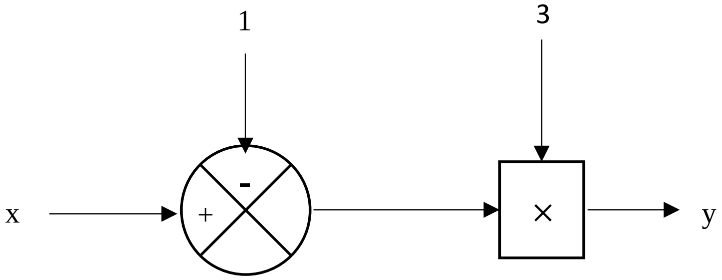


Figure 3.1 – Block Diagram Representation of Equation (3.1)

Note: Although the convention for algebraic equations and computer code is to execute the operations on the right of the equal sign and then to assign that to the variable on the left; in a block diagram, the flow goes from left to right.

Note: The lines in a block diagram represent information flow, and the circles and boxes represent operations. Unfortunately those of us grounded in process flow diagrams, see the shapes as processing units and the lines as process flow lines.

Example 3.2

The block diagram notation can also be used for differential and integral equations: For instance:

$$y = a\frac{du}{dt} + bu + c\int udt \qquad (3.2)$$

Note: There are algebra operations indicated in the equation. These include multiply the values of b and u, and the addition and equate operations. If Equation (3.2) were all algebra, you would consider $\frac{du}{dt}$ as a fraction, recognize the common d coefficients in the numerator and denominator, cancel them, and then divide $\frac{u}{t}$ to evaluate that term. But, you wouldn't, because you understand that the $\frac{du}{dt}$ symbol means, "Take the rate of change of u". It is an operation, it is neither algebra nor arithmetic. It could be more explicitly represented as $\frac{d}{dt}(u)$, which could as easily have been defined by the symbol $rate(u)$. Not strange, considering that we have other functions such as $Ln(x)$, and $sin(\pi + \theta)$. Here the argument of the function is in parenthesis. When you see $sin(\pi + \theta)$, you know it is not the product of the three variables s, i, n, multiplied by the sum of $\pi + \theta$. However, one unfamiliar with functional representation, might want to cancel the d in the numerator and denominator, or multiply the s, i, n, and $\pi + \theta$. I write this because I find students who are unfamiliar with Laplace notation will often try to solve for the real variable represented by the transform with the algebra indicated in the fingerprint.

The procedure to solve for the value of y given the value of u in Equation (3.2) could be represented by the block diagram notation in Figure 3.2.

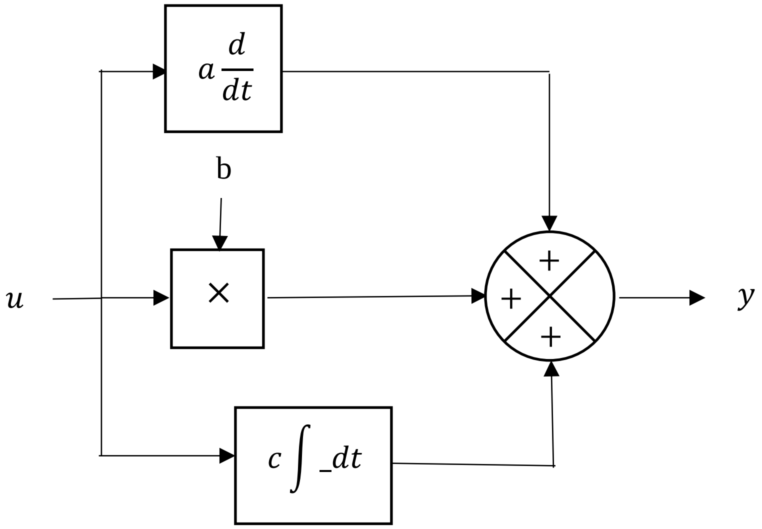


Figure 3.2 – Block Diagram Procedure for Equation (3.2)

Note: On the left side of Figure 3.2, the value of u goes three ways. This is not 1/3rd of the value of u going each way. Like one person talking to three who are listening, each listener hears 100% of the message. Again, the lines represent information, not a conserved material, as it would in a process flow diagram. Even if the signal u means three people, the signal splitting does not mean that one person went each direction. The concept is that each box is told there are three people.

In the times block (the × operation) I explicitly indicate that the multiplication of the entering signal is by the value of b. However, I included the multiply-by coefficients a and c within the blocks that indicate “take the derivative” and “integrate”. Often, the multiplication operation is just illustrated with the coefficient in the block. There are diverse conventions for block diagram structures depending on the legacy and training of the individual’s origin. But, to me what convention one uses is inconsequential. Except, you should use the convention preferred by your boss or customer.

Note: The argument of the integral and of the derivative are not explicitly acknowledged, but implicitly implied by the signal that enters the box.

Note: The box containing the $a\frac{d}{dt}$ label, could have contained this English Language statement, “To calculate the value of the output, take the derivative of the input and multiply it by the value of a.” The function described is not algebra even though it looks like a fraction.

Example 3.3

Finally, here is a Laplace transfer function relation representing a PI controller, but with the proportional action filtered to temper noise.

$$\hat{u} = K\left(\frac{1}{\tau_f s+1} + \frac{1}{\tau_i s}\right)\hat{e} \tag{3.3}$$

Note: Recall, the values of $\hat{u}$ and $\hat{e}$ have no physical meaning. Further, the value of s is not specified. So, for the second term in the parenthesis, we do not multiply τ_f times s then take its reciprocal and add it to the first term. The $\frac{1}{\tau_i s}$ pattern represents an operation. It says, "Integrate the variable and divide by tau-i." Similar to the $\frac{du}{dt}$ operation discussed above, you don't solve Equation (3.3) by arithmetic operations. You would solve for u (not $\hat{u}$) by following the procedure outlined in Part 4 of this monograph. You would translate Equation (3.3) into English by using Table 2.1.

However, the algebraic form of a Laplace transformed relation, such as Equation (3.3), can be represented in a block diagram. See Figure 3.3.

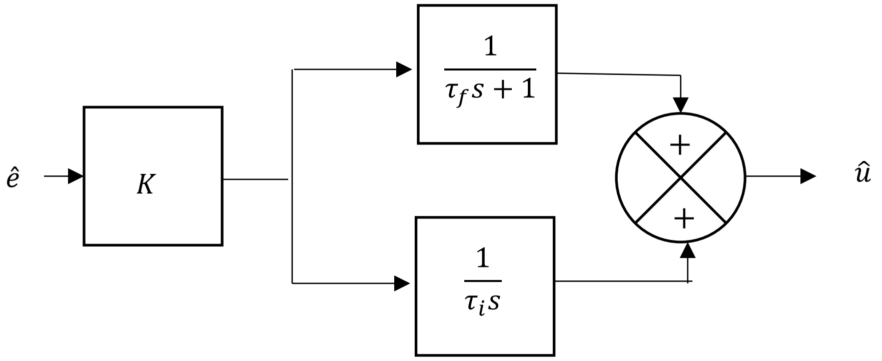


Figure 3.3 – Block Diagram Representation of Equation (3.3) – Laplace Transfer Function Notation

Again, the operation indicated by the $\frac{1}{\tau_f s+1}$ in the upper block does not mean multiply τ_f times s then add that to 1, then take the reciprocal. It means, "The outgoing value is the τ_f-lagged response to the incoming value." The $\frac{1}{\tau_f s+1}$ symbol means determine the value of the response, the output, z, when it is governed by this first order relation $\tau\frac{d}{dt}(z) + z = x$, where x is the influence.

A block diagram of Equation (3.3) could as easily have been indicated in the time domain (or any written language). See Figure 3.4.

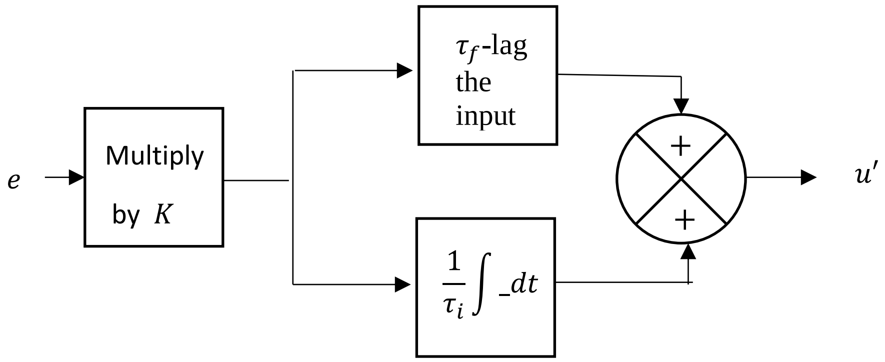


Figure 3.4 – Block Diagram Representation of Equation (3.3) – Mixed English and Calculus Notation

Note: The input is e, not $\hat{e}$, and the output is u' not $\hat{u}$.

Note: There is no difference in instruction between Figures 3.3 and 3.4. Both indicate the same operations with the same coefficient values.

Part 4 Creating Code for Simulators
(Solving from the Laplace Representation)

Introduction

Some people will want to create simulators from Laplace transfer function descriptions of a process and controller. These can be useful to explore what might happen if tuning, filtering, or controller structure is changed; or if the process structure or flowrates are changed. Some may want to create simulators for training, or process capability analysis. I guided my own understanding of control techniques by creating simulators to test and explore if what "they" told me was true (for instance does reset feedback really prevent windup), and felt that similar investigations by students would help them learn and understand. Most people will not be exploring options, but a few will find this fourth, and final, part in the series to be a useful guide.

The tables in this fourth part extend the correspondence in Table 2.1 to several representations – Function name, Laplace, calculus, finite difference, and computer code. The finite difference representations included here are elementary, simple algebraic numerical method versions of the calculus equation. Certainly, one could be more sophisticated in numerical techniques, but I see no need. Because I commonly use Excel and VBA (Visual Basic for Applications) macros, my computer code is a VBA representation of the numerical solution. If you are coding in Fortran, or C, or spreadsheet cells, or any other environment, it should be easy for you to translate my VBA code into your language.

Recall that the prime index on a variable means that it is a deviation from a reference value, $y'(t) = y(t) - y_{ref}$. The reference value, y_{ref}, is usually, but not necessarily, an initial steady state value. So, if you calculate the y' value, you need to convert it to the process variable (PV) value by using $y(t) = y'(t) + y_{ref}$.

Translating Laplace Notation to Executable Code

Table 4.1 – Gain

$\hat{y} = K\hat{u}$	$y'(t) = Ku'(t)$

$y_i{}' = Ku_i{}'$	udev := u - uref
	ydev := K * udev
	y := ydev + yref

In Table 4.1 (and those that follow) the upper left cell contains the Laplace transfer function, with the redundant function-of-s representation omitted. The upper right cell contains the calculus representation with the prime explicitly indicating that these are deviation variables. Often this will be a differential equation, and often called a time-domain relation, because these variables represent those that change in time. With the Gain operation presented in Table 4.1, it is a simple scaling, ratio, or multiplication.

The lower left cell represents the algebraic form of a digital or numerical solution of the time domain equation. I will be using simple numerical methods, which I find fully adequate when the time increment in a simulations is adequately small. Good practice in control is to have 30 (or more) control actions in an open loop transient. This means that the time interval for control is at least about one-tenth of a time-constant, which permits use of the simple numerical methods of rectangle rule of integration and finite difference approximation to the derivative. The subscript i represents the sampling interval.

Finally, in the lower right cell is the VBA code. Note that there are two deviation variables, u' and y'. To make sense in the relations of calculating y from u, their reference values (yref and uref) need to represent the same steady state case.

Note: You can derive $y_{ref} = Ku_{ref}$ from the equation in the upper right of the table, by the definitions $y'(t) = y(t) - y_{ref}$ and $u'(t) = u(t) - u_{ref}$.

Note: The equal sign in the code is not an algebraic equality, but indicates an assignment. The convenience of this means that the subscript is often not needed. There is no need to store or recall the past thousands of u- and y-values. In VBA code the assignment symbol is the equal sign. I have appropriated the VBA code by using the := symbol to explicitly indicate it is an assignment. Don't include the colon if you are going to run VBA.

Table 4.2 – Integrate

$\hat{y} = \frac{\hat{u}}{s}$	$y'(s) = \int_{t'=0}^{t'=\infty} u'(t')dt'$

$y_i' = y_{i-1}' + \Delta t\, u_i{}'$	udev := u - uref
	ydev := ydev + dt * udev
	y := ydev + yref

In the lower left cell, the numerical, finite difference, solution to the calculus integral uses the rectangle rule of integration. Here, the most recent value of the integral, the i^th^ value, y_i', is calculated by incrementally adding the most recent rectangle, $\Delta t\, u_i{}'$, to the integral value at the prior y-value, y_{i-1}'. Δt is the time increment for the simulation, and should less than about 1/10th of any time-constant in the simulation. Typically, because the event is in deviation variables, the integral is initialized at a value of zero, $y_0' = 0$.

The subscripts in the numerical representation of the lower left cell might make one think that the code must store all of the past values in an array. However, they are not present in the code in the lower right cell. The advantage of an assignment statement in this calculation is that you do not have to store all of the past values. The new value only needs to be incrementally updated from the recent past value.

As an aside: In algebraic notation, the equation $x = x + 1$ makes no sense. However, if it is an assignment in computer code, the operation indicated by $x := x + 1$ means, "Recall the value in the storage location called x, add 1 to it, and store that new value in the storage location called x." If x starts at zero, each execution of the assignment increments it by 1. The value of x is 0, then 1, then 2, then 3, 4, 5, 6, … Here the initial value for ydev is zero, since y starts at its initial yref value.

Table 4.3 – Rate

$\hat{y} = s\hat{u}$	$y'(t') = \frac{du'(t')}{dt}$

$y_i' = (u_i' - u_{i-1}')/\Delta t$	udev := u - uref
	ydev := (udev – udevold) / dt
	udevold := udev
	y := ydev + yref

Again, Δt in the finite difference method for the derivative should be less than about 1/10th of any time-constant in the simulation. This does require the immediately past value in the calculation; but, not all past values need to be stored, just the most recent past value. In the assignment statement the line "udevold := udev" makes the old value available each time the set of lines are executed. Here, two variables (udev and udevold) need to be initialized with values of zero, since u starts at its uref value and at steady state.

Table 4.4 – Lag

$\hat{y} = \frac{\hat{u}}{\tau s + 1}$	$\tau \frac{dy'}{dt} + y' = u'$

$y'_{i+1} = (\Delta t/\tau)u'_i + (1 - \Delta t/\tau)y'_i$	udev := u – uref
	ydev := (dt/tau)*udev + (1-dt/tau)*ydevold
	ydevold := ydev
	y := ydev + yref

The finite difference notation in the lower left can be derived by using the traditional forward difference approximation to the derivative when the rest of the terms in the differential equation are evaluated at the ith time. This is called an explicit finite difference representation, in which all terms in the calculus representation (upper right cell) are evaluated the common time, now. Validity of the method is dependent on Δt being less than about 1/10th of any time-constant in the simulation. There are many alternate finite difference solutions (including implicit and semi-implicit), but they all have similar forms, and equivalent calculated results if $\Delta t < \tau/10$. The finite difference representation suggests that past y-values need to be stored, but only the first past value is needed. Accordingly, this simplifies the code. The ydevold value should be initialized to zero indicating that the process starts at a steady state at the value of yref.

Table 4.5 – Second-Order Lag (sequential first-order lags)

$\hat{y}_1 = \frac{\hat{u}}{\tau_1 s + 1}$ $\hat{y}_2 = \frac{\hat{y}_1}{\tau_2 s + 1}$ Or equivalently $\hat{y}_2 = \frac{1}{(\tau_1 s + 1)(\tau_2 s + 1)}\hat{u}$	$\tau_1 \frac{dy'_1}{dt} + y'_1 = u'$ $\tau_2 \frac{dy'_2}{dt} + y'_2 = y'_1$

$y'_{2,i+1} = (\Delta t/\tau_2)y'_{1,i} + (1 - \Delta t/\tau_2)y'_{2,i}$
$y'_{1,i+1} = (\Delta t/\tau_1)u'_i + (1 - \Delta t/\tau_1)y'_{1,i}$
	y2dev := (dt/tau2)*ydev1 + (1-dt/tau2)*ydevold2
	ydevold2 := ydev2
	y := ydev2 + yref
	udev := u – uref
	y1dev := (dt/tau1)*udev + (1-dt/tau1)*ydevold1
	ydevold2 := ydev2

A second-order lag can be modeled in a variety of ways; but usually, it refers to one first-order process influencing a subsequent first-order process. This is what the Laplace and calculus notations indicate. Each process has its own time-constant. You might note that the two processes are presented in reverse order in the lower left digital representation. This is because I am using the explicit finite difference numerical method. Here the subscripts $i+1$ and i on the y terms represent the end and the beginning of a time increment. The new value for y_1' does not happen until the end of the time step, what is influencing y_2' during the time increment Δt is the current effluent from process 1, not what it will be at the end of Δt. Similarly, the change in the second process is calculated in the assignment statements (in the lower right cell in the table) prior to the change in the first process. Again Δt should be less than 1/10th of any time-constant in the simulation.

Note: I have often observed students seeking to intuitively combine several of the computer code relations. Don't. These are assignment statements not algebraic relations.

Table 4.6 – Delay

$\hat{y} = e^{-\theta s}\hat{u}$	$y'(t') = u'(t' - \theta)$

${y'}_i = {u'}_{i-N}$
Where $N = Round(\frac{\theta}{\Delta t})$
	i = i + 1
	IF i > N then i = 1
	udev(i) = u - uref
	j = i+ 1
	If j > N then j = 1
	ydev = udev(j)
	y = ydev + yref

The delay is of time duration θ, however this might not be an integer number of simulation time intervals. So, the number of time intervals, N, is calculated as the $\frac{\theta}{\Delta t}$ value rounded to the nearest integer. Here the computer code does use an array of past values, but the array that holds past u-values only needs to be N elements in length. The indices i and j indicate where to place the latest measurement and where to find the Nth past measurement (often termed push and pull in computer-talk). Elements in the udev array should be initialized to zero, representing that u is at an initial steady state at the uref value.

Example 4.1

If you have a compound transfer function, use algebra to break it into recognizable elements, and then write the code for each. For example, here is a feedforward controller (alternately termed a dynamic compensator or decoupler).

$$\hat{y} = K\frac{\tau_1 s+1}{\tau_2 s+1}e^{-\theta s}\hat{u} \quad (4.1)$$

It indicates that $y'(t)$ is a leaded, lagged, delayed, scaled response to $u'(t)$. I would group recognizable terms as presented in Table 4.7, but there are many fully acceptable, equivalent ways.

Table 4.7 – Solution Decomposition Procedure to Equation (4.1)

Operation	New variable	Remaining Operations
v is the delayed u	$\hat{v} = e^{-\theta s}\hat{u}$	$\hat{y} = K\frac{\tau_1 s + 1}{\tau_2 s + 1}\hat{v}$
w is the scaled v	$\hat{w} = K\hat{v}$	$\hat{y} = \frac{\tau_1 s + 1}{\tau_2 s + 1}\hat{w}$
x is the lagged w	$\hat{x} = \frac{1}{\tau_2 s + 1}\hat{w}$	$\hat{y} = (\tau_1 s + 1)\hat{x}$
y is the leaded x, which makes it the scaled derivative of x added to x	$\hat{y} = \tau_1 s\hat{x} + \hat{x}$	

So, taking each operation one at a time, the VBA digital code is:

```
I = I + 1                              ‘increment put counter
IF I > N then I = 1                    ‘reset counter, $N = int(\frac{\theta}{\Delta t} + 0.5)$
u_prime(I) = u(I) – u_ref              ‘calculate u deviation
J = I + 1                              ‘read counter of Nth past value
IF J > N then J = 1                    ‘reset read counter
v = u_prime(J)                         ‘delayed value of influence
w = K * v                              ‘scaled delayed influence
x = (dt/tau2)*w + (1-dt/tau2)*xold     ‘lagged scaled delayed influence
y_prime = tau1*(x-xold)/dt +x          ‘leaded lagged scaled delayed influence
xold = x                               ‘need to remember old value
y = y_prime + y_ref                    ‘convert deviation back to PV
```

Figure 4.1 illustrates the execution of the code. The influence makes an ideal step-and-hold and the process responds with a gain of 2, delay of 10 s, lead of 60 s and lag of 20 s.

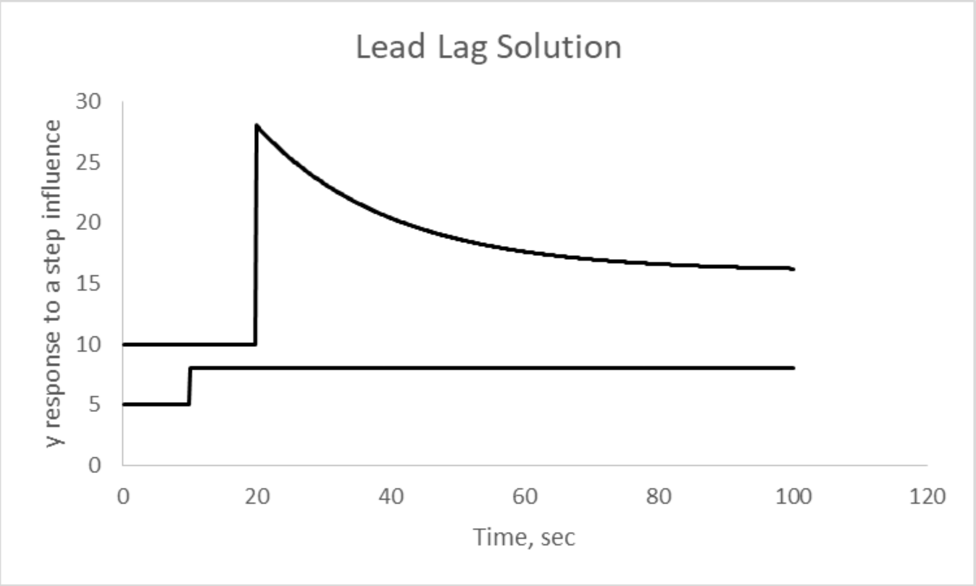


Figure 4.1 – The scaled-delayed-lead-lag response to a step-and-hold influence

Utility

That was easier than partial fractions! And now, as Figure 4.2 reveals, the digital code version of the solution to the transfer function can respond to any sort of influence pattern. The influence is the lower trace and the response is the upper trend. With partial fraction inversion of Laplace Transforms to an analytical model, an input varying such as this one would make the solution intractable.

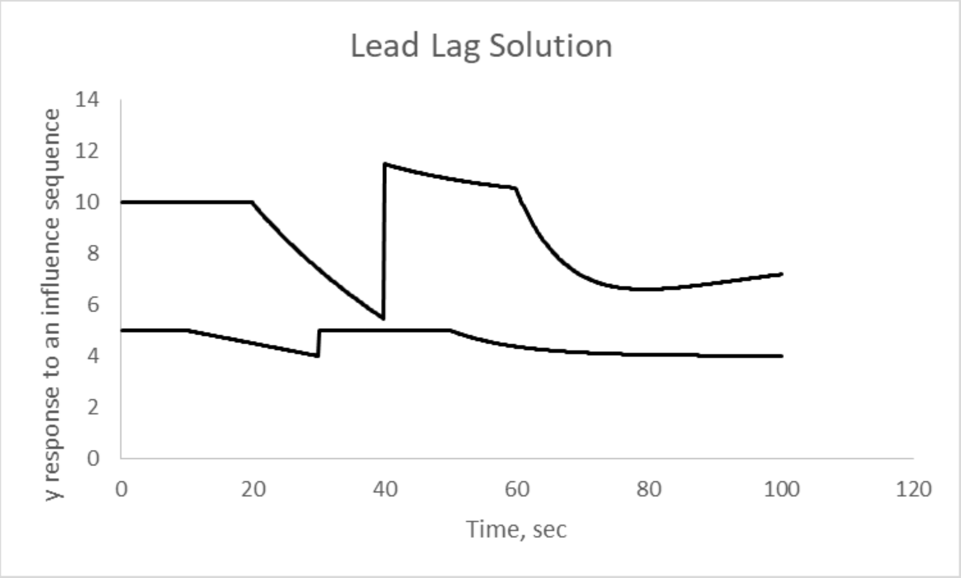


Figure 4.2 – The scaled-delayed-lead-lag response (upper trend) to a diversity of influence patterns (lower trend)

Although described by a Laplace transfer function, we did not solve the transfer function to obtain the process response to an input. We translated the Laplace transform notation to executable code, which represents the time-domain operation, which is indicated by the transfer function pattern. If an English language instruction would have been, “The process is a θ-delayed, K-scaled, τ_2-lagged, τ_1-leaded response to the influence” we would have translated it to the same code.

Example 4.2

Again, here is the Equation (3.3) Laplace transfer function relation representing a PI controller, but with the proportional action filtered to temper noise.

$$\hat{u} = K\left(\frac{1}{\tau_f s+1} + \frac{1}{\tau_f s}\right)\hat{e} \tag{4.2}$$

Multiply each term in the parenthesis by $\hat{e}$ to get familiar transfer functions.

$$\hat{u} = K\left(\frac{\hat{e}}{\tau_f s+1} + \frac{\hat{e}}{\tau_i s}\right) = K(\hat{T}_1 + \hat{T}_2) \tag{4.3}$$

Although Equation (4.3) is in transformed variable notation, it says, “u’ is the K-scaled sum of Term-1 and Term-2.” It says, “Term-1 is the τ_f-lagged actuating error, and that Term-2 is the time integral of the actuating error scaled by τ_i.” Finally, u, will be calculated by $u = u' + u_{ref}$. To start e' must be calculated. Since $e' = y'_{SP} - y' = (y_{SP} - y_{ref}) - (y - y_{ref}) = y_{SP} - y = e$, just use the actuating error.

So the VBA code is

```
e = y_SP – y                                    ‘calculate actuating error
T2 = T2 + e * dt / tau_i                        ‘integrate with rectangle rule
T1 = (dt / tau_f) * e + (1 - dt / tau_f) * T1   ‘first order lag
u_dev = K * (T1 + T2)                           ‘sum and scale
u = u_def + u_ref                               ‘convert to u.
```

Note: y_SP, T1, T2, and u_ref need to be initialized. This code represents the P-filtered PI controller that would be executed in the AUTO mode. My preference is to initialize variables in the MAN mode. Initialize y_SP with the current controlled variable value, and u_ref with the current manipulated variable value (the controller output). Initialize T1 and T2 as zero.

Note: This primitive controller code does not reveal the MAN operations, not does it have either an anti-windup feature, or an override to keep u within physical bounds (ideally 0 to 100%).

Acknowledgment

I appreciate the review and comments provided by Carlos A. Smith, coauthor of the textbook Principles and Practices of Automatic Process Control, but I accept full responsibility for any errors or oversights that

might remain. I appreciate the folks at Control Global for helping me disseminate this applied view of Laplace transforms. And certainly, am in awe of the mathematicians who envisioned the transform and its utility.

About

Russ Rhinehart started his 13-year industrial process career in the Carolinas, and greatly enjoyed the coaching of new engineers and the analysis methods of process modeling, control, and optimization. So, he returned to school for a PhD to be able to begin a 29-year academic career of coaching and investigation. He started at Texas Tech University, then became the Head of Chemical Engineering at Oklahoma State University for 13 years. Now, in "retirement", he enjoys having the time to organize material representing best practices in engineering techniques. He has published three books about "what I wish I knew when starting my career": Applied Engineering Statistics (Chapman & Hall/CRC, with R. M. Bethea), Nonlinear Regression Modeling for Engineering (John Wiley & Sons), and Engineering Optimization: Applications, Methods, and Analysis (John Wiley & Sons). He hosts the web site www.r3eda.com to provide open access to materials, and provides monthly "Develop Your Potential" articles for CONTROL magazine. His BS ChE degree is from the U of Maryland, and PhD is from North Carolina State U.

This monograph on Laplace transforms is an original work by R. Russell Rhinehart. The reader has the author's permission to use any or all of the contents in any manner, when the original source is appropriately acknowledged.