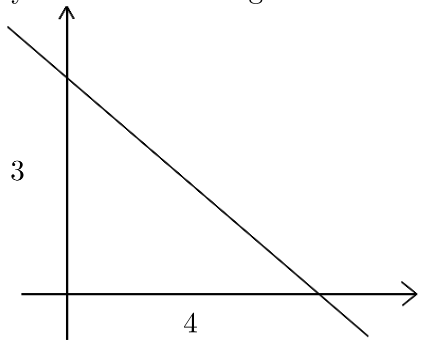
Abstract

abstract

1 Numerical Methods

1.1 Integration

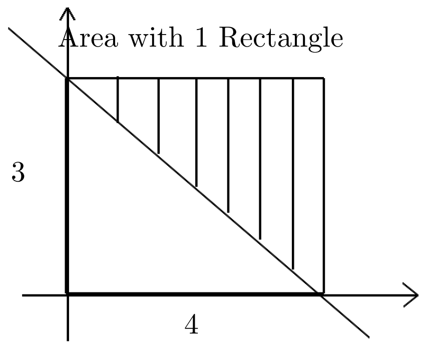
One of the key mathematical methods that both mathmeticians and scientists use is integration. When approached graphically integration simply means the area underneath a graph, bounded by the x-axis. This means that given a graph of a triangle, a rectangle, a parabola, or even a more complicated shape, a scientist can find the area underneath the graph by using integration. When looking at a triangle, if one happens to know the geometric method for computing the area, one can find the area underneath the graph exactly. Given the triangle as shown below, the method



that a mathematician would use is as follows:

$$\begin{aligned} A_{Triangle} &= \frac{1}{2}bh \\ &= \frac{1}{2}3 \times 4 \\ &= 6 \end{aligned}$$

This equation, however, has its drawbacks. Most importantly, one must have the previous knowledge of that the area of a triangle is defined by $\frac{1}{2}base \times height$. This severely limits mathematicians to shapes that they know the geometric equations for. Therefore, most people would be at a loss if they were asked to come up with the area under a parabola or another less common shape. Thus, integrals become an important tool for abstracting the area under any graph bounded to the x-axis. Continuing the triangle example, one can look at a triangle as roughly one rectangle. A rectangle has an area formula $base \times height$. But how can one find the values of the base and height. The base can be found very simply. By looking at the graph of a triangle, one would see the bases are the same, because there is only one rectangle for the triangle. This base can be short-handed as Δx or the amount that one tavels along the x axis per rectangle. In this example the "Δx" can be



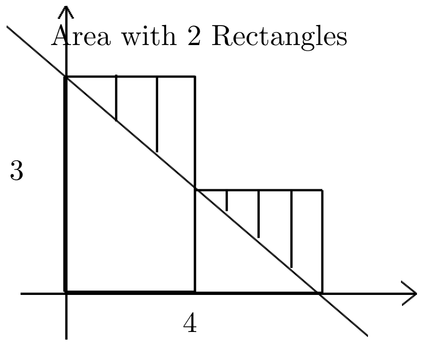
computed as follows:

$$\Delta x = \frac{4units}{1Rectangle} \tag{1}$$

The height at any point in a graph can also be short-handed as $f(x)$, or the value of the function that defines the graph at a given x value. In this case, the height will just be computed at one point for one rectangle. For the purpose of simplicity, one can find the height at $x = 0$. Using this knowledge, one can continue to compute an approximation of the area of the triangle.

$$\begin{aligned} A_{Rectangle} &= base \times height \\ &= \Delta x \times f(0) \\ &= 4 \times 3 \\ &= 12 \end{aligned}$$

This method, with just one Rectangle, is obviously not nearly as accurate as the geometric sum, in fact, it is 100% off the actual value. However, realizing that this error is caused by the inability of a single rectangle to approximate the area of a triangle, one can divide the graph into two rectangles. Now the base of each rectangle changes. The base of the triangle now must accomodate two rectan-



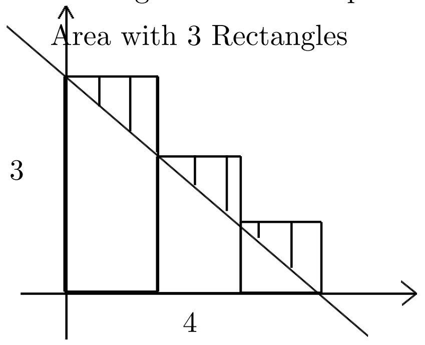
gles. Thus one divides the base of the triangle into two rectangles.

$$\Delta x = \frac{base}{2} = \frac{4}{2} = 2 \tag{2}$$

Now for the approximation to be computed, one must add up each of the rectangles and compute the height of two different rectangles. Each one of these rectangles are one base apart or one "*Deltax*. The computation would be as follows:

$$\begin{aligned} A_{2Rectangles} &= Rectangle_1 + Rectangle_2 \\ &= [\Delta x \times f(0)] + [\Delta x \times f(0 + \Delta x)] \\ &= [2(3)] + [2(1.5)] \\ &= 6 + 3 \\ &= 9 \end{aligned}$$

This approximation is much closer to the actual area of the triangle. This value of 9 is only 50% away from the actual value of 6. This method can be continued and the triangle can be even more closly approximated by three rectangles. The base and heights can be computed as before. The base would now be divided into three rectangles:

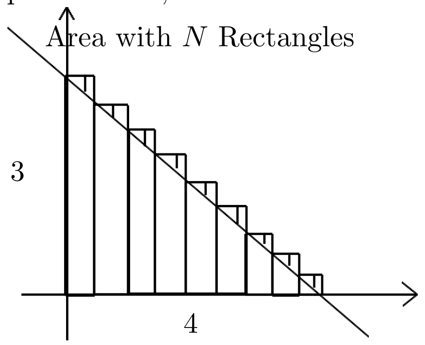


$$\Delta x = \frac{base}{3} = \frac{4}{3} \tag{3}$$

The heights would be computed at each x starting at 0 "Δx" apart, the start of each rectangle.

$$\begin{aligned}
A_{3Rectangles} &= R_1 + R_2 + R_3 \\
&= [\Delta x \times f(0)] + [\Delta x \times f(0 + \Delta x)] + [\Delta x \times f(0 + 2\Delta x)] \\
&= \frac{4}{3}(3) + \frac{4}{3}(2) + \frac{4}{1} \\
&= \frac{12}{3} + \frac{8}{3} + \frac{4}{3} \\
&= 8
\end{aligned}$$

This approximation is even closer to the actual value of 6. But one also notices that this jump is not by as much as the previous one. Now seeing the trend towards a more close approximation mathematicians add more rectangles, until their percent errror becomes marginal. To display this trend towards a much more accurate approximation, one can divide the triangle into 100 rectangles. The computation would be as follows:



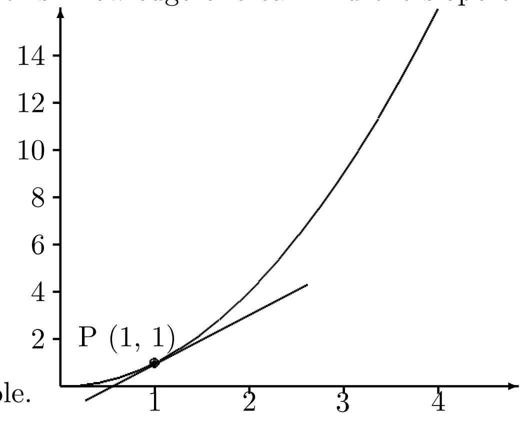
$$\Delta x = \frac{base}{100} = \frac{4}{100} = \frac{1}{4} \tag{4}$$

$$\begin{aligned}
A_{100Rectangles} &= \sum_{i=1}^{100} f(x)\Delta x \\
&= R_1 + R_2 + R_3 + \ldots + R_{98}R_{99}R_{100} \\
&= [\Delta x \times f(0)] + [\Delta x \times f(0+\Delta x)] + [\Delta x \times f(0+2\Delta x)] \ldots \\
&\quad [\Delta x \times f(97\Delta x)] + [\Delta x \times f(98\Delta x)] + [\Delta x \times f(99\Delta x)] \\
&= \frac{1}{25}(3) + \frac{1}{25}(\frac{297}{300}) + \frac{1}{25}\frac{294}{300} + \ldots + \frac{9}{25}\frac{6}{300}\frac{1}{25}\frac{3}{300} \\
&\approx 6
\end{aligned}$$

Now the approximation is much closer to the actual value, and depending on the purpose of the integration, might be close enough. This method for finding the area underneath a graph can obviously be very useful when one does not know the geometric equation for the area of the shape and perfers a more abstract approach. It is also clear, however, that this method is not only computationally demanding, but also approximate. Computers play an important role in making these integrals faster to compute and a more feasible approach to finding the area under graphs.

1.2 Derivatives

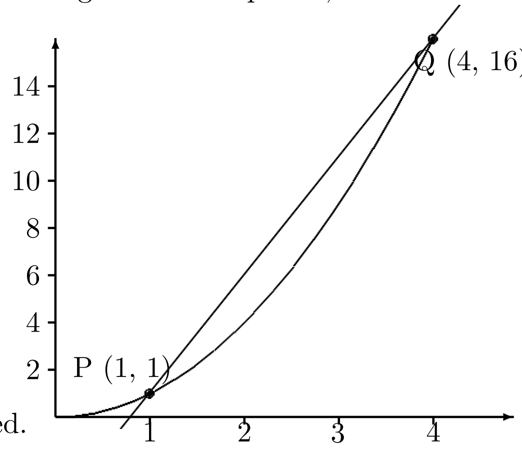
Derivatives, like integrals, are an important tool for mathmeticians and scientists alike. The objective of a derivative is to find the slope of a line at a specific point. If one is looking at a non-vertical line, with no curves, finding the slope is simple. The basic slope formula can be written $m = \frac{\Delta y}{\Delta x}$ where m is the slope $\Delta y \Delta x$ is the change in height over the change in the horizontal distance. Using this knowledge one can find the slope of line easily as demonstrated through the following example.



$$\begin{aligned}
m &= \frac{\Delta y}{\Delta x} \\
&= \frac{y_f - y_i}{x_f - x_i} \\
&= \frac{f(x_f) - f(x_i)}{x_f - x_i} \\
&= \frac{0-3}{4-0} \\
&= \frac{-3}{4}
\end{aligned}$$

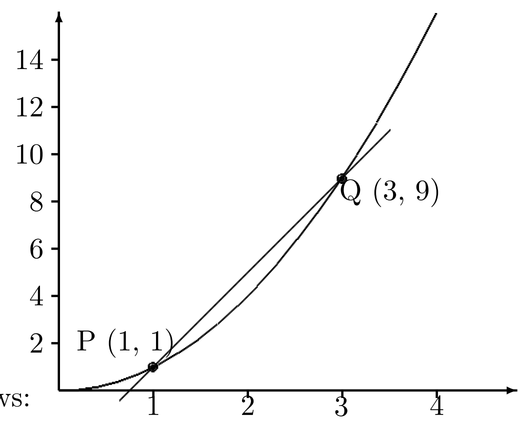
This method finds the slope at any point along this line because the slope of a line is constant, therefore the slope of the whole line is the same as the slope at any one point. This is effective for any non-vertical line with a constant slope. Finding the slope of a line on a line with a non-constant slope is much more complicated, however. To display this issue look at a parabola with defined by the equation of $y = x^2$. One could do this visually by drawing a tangent line, a line that just touches the graph at that one spot, then finding the slope of the tangent line. This would look like as follows for the point $P = 1$:

The slope of this line would be about 2 depending on how accurately one drew the tangent line. This, however, is not an effective method, it is only as accurate as the line one draws and is also not able to be abstracted to the point where a computer could do it. How does one exactly draw the line? It is intuitive and not exact. Therefore, one must find a more exact and computable method for finding the slope at a single point. When using the traditional slope formula one runs into a problem, how does one define the x_i and the x_f, if one wants to find the value at just one point, P. Let one assume the values P as the starting point and another point Q as the ending point, which is a horizontal distance h away from P. To find the slope of the line running through those two points, one would follow the same procedure that the previous example employed.



$$\begin{aligned} m &= \frac{f(x_Q) - f(x_P)}{x_Q - x_P} \\ &= \frac{f(x_P + h)}{x_P + h - x_P} \\ &= \frac{f(x_P + h)}{h} \\ &= \frac{f(3) - f(0)}{3} \\ &= \frac{9 - 0}{3} \\ &= \frac{9}{3} \\ &= 3 \end{aligned}$$

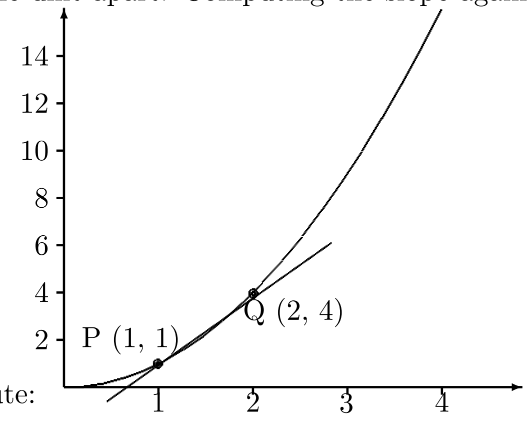
This gives a slope of 3 which is definately not the slope of the line at that point. But what would happen if one brought Q closer to P? Since P and Q are closer, one would compute a more accurate slope. Let h be set to 2, with Q being shifted left one unit. Following the same procedure as the



previous example one would compute the slope as follows:

$$\begin{aligned} m &= \frac{f(x_P+h)}{h} \\ &= \frac{f(3)-f(1)}{2} \\ &= \frac{9-1}{2} \\ &= \frac{8}{2} \\ &= 4 \end{aligned}$$

This answer is clearly closer to the expected value of 6 but is still not there. Thus let Q be brought even closer to P to the point where they are only one unit apart. Computing the slope again but



this time with the value of Q as 1, one would compute:

$$\begin{aligned} m &= \frac{f(x_P+h)}{h} \\ &= \frac{f(3)-f(2)}{1} \\ &= \frac{9-4}{1} \\ &= \frac{5}{1} \\ &= 5 \end{aligned}$$

This value computed for this slope is 5 which is only one off from the expected value. Thus one sees the trend that as h gets smaller (Q is brought closer to P), the value of the computed slope gets closer to the expected slope value. Mathematically that can be translated to be:

$$\lim_{h\to 0} \frac{f(x_P+h)}{h} = Slope_P$$

With more generalized variables, x for x_P and $\frac{dx}{dy}$ for the slope at a point this equation becomes:

$$\frac{dx}{dy} = \lim_{h \to 0} \frac{f(x+h)}{h}$$

2 Physical Systems

A physical system is made up of elements and these elements transfer or receive energy to other elements in the system. The total energy in a system never changes.

Simple systems may be linear, electrical, rotational or a combination of these. Linear systems (often called mechanical systems) are made up of elements that move in a straight line. Examples include springs, levers, and any object with a mass, such as a box, a cat, or a computer.

Electric systems are made up of elements connected by electricity. Examples include batteries, resistors, capacitors, and other electric components. Rotational systems are made up of elements that move in a circle. Examples include wheels and gears.

Frequently, systems involve elements from more than one type of system. A battery wired to an electric motor that drives a rack and pinion is one example. The system begins electric with the battery, changes to rotational in the motor, and becomes mechanical after the rack and pinion.

Certain systems, called static systems, never change. Energy is always transferred from one object to another at exactly the same rate. Other systems, called dynamic systems, change over time. Objects can speed up or slow down, capacitors can gain or lose charge, and many other aspects of the system change. The rate at which energy moves between objects changes.

The rate of energy transfer is given its own name: power. In a dynamic system, the power of each energy transfer changes over time. Here are some examples of power. Each example has the power formula appropriate for the situation. All values are given in SI units for consistency. A guide to all variables and units used in this section can be found at the end.

Mechanical systems: $l = F * v$

An engine winds up a rope, lifting a box into the air. The engine exerts a constant force of 100 N and the box moves at 2 m/s. What is the power of the engine?

$$P = F * v = (100N) * (2m/s) = 200W$$

The same situation as above, but this time the engine is a 400 W engine exerting 100 N of force. How fast does the box move?

$$v = P/F = (400W)/(100N) = 4m/s$$

This time, the 400 W engine is moving the box at 5 m/s. What force is it exerting?

$$F = P/v = (400W)/(5m/s) = 80N$$

Electrical systems: P=V*i

A battery is wired to an electric circuit. The battery provides 12 V and has a current of 2 A passing through it. What is the power of the battery?

$$P = V * i = (12V) * (2A) = 24W$$

Now, the same battery is working with a power of 36 W. What current does the battery produce?

$$i = P/V = (36W)/(12V) = 3A$$

A different battery is connected to a circuit. This one works at 45 W when 5 A of current pass through it. What is this battery's voltage?

$$V = P/i = (45W)/(5A) = 9V$$

Rotational systems: $P = \tau * \omega$

A person spins a flywheel by winding a crankshaft. She exerts a constant torque of 150 N*m at an angular velocity of 3 rad/s. What is her power?

$$P = \tau * \omega = (150N * m) * (3rad/s) = 450W$$

This time, she works at a power of 500 W while exerting a torque of 100 N*m. With what angular velocity does the wheel spin?

$$\omega = L/\tau = (500W)/(100N * m) = 5rad/s$$

Her weaker friend, capable of only 250 W of power, takes over and rotates the wheel at 2 rad/s. What torque does he exert?

$$\tau = L/\omega = (250W)/(2rad/s) = 125N * m$$

The power formulas for each system are very similar. Each is the product of two factors. The first factor, force, torque, or voltage, measures how hard the moving object is being pushed, whether in a straight line, in a circle, or as an electric charge being pushed through a circuit. The second factor, velocity, angular velocity, or current, measures how fast the moving object is moving, whether in a straight line, in a cricle, or as an electric charge being pushed through a circuit. Thus, an abstract power formula can be written that reflects each individual system:

$$P = e * f \tag{5}$$

In this abstract formula, the e is a factor called "effort" (how hard something is being pushed) and the f is a factor called "flow" (how fast something moves).

Elements in a system typically constrain these two power variables (effort and flow) in some way. There are three important constraining elements: resistive elements, capacitive elements, and inductive elements. These elements, which transfer power through only one connection to the rest of the system, are called 1-port elements.

2.1 Resistive Elements

Resistive elements act as energy dissipators in a system. They convert energy into thermal energy, which can never be returned to the rest of the system.

The most familiar resistive element is the electrical resistor. Resistors constrain the effort (voltage) and flow (current) through them by Ohm's Law:

$$\begin{aligned} V &= i * R \\ &\text{or} \\ e &= R * f \end{aligned}$$

Here are a few examples of how this relation is used:

A 10-ohm resistor has 2 A of current passing through it. What is the voltage across the resistor, and what is its power?

$$\begin{aligned} V &= i * R = (2A) * (10\Omega) &&= 20V \\ P &= e * f = V * i = (20V) * (2A) &&= 40W \end{aligned}$$

A 24-ohm resistor is wired to a 12 V battery. What is the current through the resistor, and what is its power?

$$\begin{aligned} i &= V/R = (12V)/(24\Omega) &&= 0.5A \\ P &= e * f = V * i = (12V) * (0.5A) &&= 6W \end{aligned}$$

The mechanical/rotational analogue to a resistor is a device called a dashpot (or damper). A dashpot typically has some sort of viscous fluid inside that resists motion, producing a force (or torque) proportional to the input (angular) velocity.

Mechanical dashpots meet the following constraint:

$$\begin{aligned} F &= b * v \\ &\text{or} \\ e &= b * f \end{aligned}$$

Rotational dashpots obey the following equation:

$$\begin{aligned} \tau &= b * \omega \\ &\text{or} \\ e &= b * f \end{aligned}$$

Here are a few examples of each form of dashpot:

A dashpot with damping coefficient 5 N*s/m is pushed at a constant 2 m/s. With what force does it resist being pushed, and what is its power?

$$\begin{aligned} F &= b * v = (5N * s/m) * (2m/s) &&= 10N \\ P &= e * f = F * v = (10N) * (2m/s) &&= 20W \end{aligned}$$

A dashpot with damping coefficient 10 N*s/m is pushed by a 40 N force to constant velocity. How fast does it move, and what is its power?

$$\begin{aligned} v &= F/b = (40N)/(10N * s/m) &&= 4m/s \\ P &= e * f = F * v = (40N) * (4m/s) &&= 160W \end{aligned}$$

A rotational dashpot in a door with damping coefficient 3 N*m*s opens at 2 rad/s. With what torque does it oppose this motion, and what is its power?

$$\begin{aligned} \tau &= b * \omega = (3N * m * s) * (2rad/s) &&= 6N * m \\ P &= e * f = \tau * \omega = (6N * m) * (2rad/s) &&= 12W \end{aligned}$$

A rotational dashpot in a door with damping coefficient 15 N*m*s is pushed by a 45 N*m torque to constant angular velocity. How fast does it move, and what is its power?

$$\begin{aligned} \omega &= \tau/b = (45N * m)/(15N * m * s) &&= 3rad/s \\ P &= e * f = \tau * \omega = (45N * m) * (3rad/s) &&= 135W \end{aligned}$$

The equations governing the behavior of efforts and flows through all three resistive elements are very similar. Thus, they can be abstracted to form a single equation:

$$e = R * f \tag{6}$$

From this equation, if either the effort or flow through a resistor is known, the other can be calculated, which in turn will give power.

Note: The following assumptions are made about resistive elements:

The function relating effort and flow is linear. This means all electric resistors are Ohmic, and non-electrical resistors are the mechanical/rotational equivalent of Ohmic. In theory, a non-linear relationship could be modeled, but this requires substantially higher-level math and might not be possible.

Friction is not a resistive element. Kinetic fricton does technically dissipate energy into thermal energy, but supplies a constant opposing effort, rather than scaling based on flow. For a full explanation and a simple workaround, see the Limitations section below.

2.2 Capacitive Elements

Capacitive elements are one of two types of energy storage elements. They are governed by what is called a state variable that reflects how much energy is stored. Capacitors store energy as an effort across them, such as the force of a spring or the voltage across an electrical capacitor. The state variable, either position or charge, is proportional to the effort on a capacitor.

Mechanical Capacitor (spring):

$$F = k * x$$

A spring with constant 25 N/m is compressed by 2 m. How much force does this compression require?

$$F = k * x = (25N/m) * (2m) = 50N$$

A spring with constant 15 N/m is compressed by a 45 N force. How far is it compressed?

$$x = F/k = (45N)/(15N/m) = 3m$$

Electrical Capacitor:

$$V = q/C$$

A 2 F capacitor has 6 C of charge on each plate. What is the voltage across it?

$$V = q/C = (6C)/(2F) = 3V$$

A .1 F capacitor is charged to 12 V. What is the charge on each plate?

$$q = V * C = (12V) * (.1F) = 1.2C$$

Rotational Capacitor (torsion spring):

$$\tau = \kappa * \theta$$

A 20 N*m/rad torsion spring is twisted 3 radians. What torque does it exert?

$$\tau = \kappa * \theta = (20N * m/rad) * (3rad) = 60N * m$$

A 10 N*m/rad torsion spring is twisted by a torque of 40 N*m. How far does it twist?

$$\theta = \tau/\kappa = (40N * m)/(10N * m/rad) = 4rad$$

These equations are all similar, and can be abstracted as follows:

$$\begin{aligned} e &= q/C && (7)\\ &\text{or} && (8)\\ q &= e * C && (9)\end{aligned}$$

where q is the general state variable 'position' for capacitors.

Note that the constant is the inverse of the constant in mechanical and rotational spring formulas. This means that, when working with springs, the constant must be inverted to use the abstract formula.

One variable is conspicuously absent from this abstract equation: flow. Without knowledge of flow, power cannot be determined.

Flow shows up in the time-varying nature of capacitive elements. Flow is the time rate of change of position, which is calculated by the product of average flow and time. For instance, the position of a spring is the product of the average speed it compresses and the duration that it compresses, and the charge on a capacitor is the product of charging current and duration of charging. The following examples demonstrate this:

A spring (k=10 N/m) is compressed at an average velocity of 2 m/s for 3 s. Assuming the average force it exerts is half of the final force, what is the average power used to compress the spring?

$$\begin{aligned} C =& \quad 1/k &&= 1/(10N/m) = 0.1m/N\\ q =& \quad f * t &&= (2m/s) * (3s) = 6m\\ e =& \quad q/C &&= (6m)/(0.1m/N) = 60N\\ e_{\text{average}} =& \quad e/2 &&= (60N)/2 = 30N\\ P =& \quad e * f &&= (30N) * (2m/s) = 60W\end{aligned}$$

A capacitor (C=2 F) is charged to 12 V in 10 s by a constant current. Assuming the average voltage is half of the final voltage, what is the average power used to charge it?

$$\begin{aligned} q =& \quad e * C &&= (12V) * (2F) = 24C\\ f =& \quad q/t &&= (24C)/(10s) = 2.4A\\ e_{\text{average}} =& \quad e/2 &&= (12V)/2 = 6V\\ P =& \quad e * f &&= (6V) * (2.4A) = 14.4W\end{aligned}$$

Capacitors are one of the two elements that allow systems to vary over time, because their properties vary over time. When modeling, very tiny steps are taken and assumed to be constant. Using flow information, the new state value can be calculated, which produces new effort information. This new effort information interacts with the rest of the system to produce new flow information, which allows the next time step to be calculated.

2.3 Inductive Elements

Inductive elements are the second type of energy storage elements. Like capacitors, they have a state variable that reflects how much energy is stored. Inductors store energy as a flow, such as the kinetic energy of a moving mass or the current through an electrical inductor. The state variable, either momentum or flux linkage, is proportional to the flow through a capacitor.

Mechanical Inductor (moving mass):

$$p = m * v$$

A block of mass 200 kg moves at a speed of 5 m/s. What momentum does it have?

$$p = m * v = (200kg) * (5m/s) = 1000kg * m/s$$

The same block has a momentum of 600 kg*m/s. How fast is it moving?

$$v = p/m = (600kg * m/s)/(200kg) = 3m/s$$

Electrical Inductor:

$$\lambda = L * i$$

A 12 H inductor has a current of 0.5 A passing through it. What is the flux linkage?

$$\lambda = L * i = (12H) * (0.5A) = 6Wb$$

A 16 H inductor has a flux linkage of 4 Wb. What current is passing through it?

$$i = \lambda/L = (4Wb)/(16H) = 0.25A$$

Rotational Inductor (flywheel):

$$p = I * \omega$$

A flywheel with a moment of inertia $50kg*m^2$ spins at 3 rad/s. What is its angular momentum?

$$p = I * \omega = (50kg * m^2) * (3rad/s) = 150kg * m^2/s$$

The same flywheel now spins with an angular momentum of $200kg * m^2/s$. What is its angular velocity?

$$omega = p/I = (200kg * m^2/s)/(50kg * m^2) = 4rad/s$$

Much like capacitors, the equations governing inductance are all similar, and can be abstracted to:

$$p = L * f$$

where p is the general state variable 'momentum' for capacitors.

Similarly to capacitors, one power variable, effort, does not show up in the abstract equation.

Effort is related to the time-varying nature of inductive elements. Effort is the time rate of change of momentum, which is calculated by the product of average effort and time. For instance, the momentum of a block is the product of the average force accelerating it and the duration that it accelerates, and the flux linkage of an inductor is the product of the charging voltage and the duration of charging. The following examples demonstrate this:

A block (m=50 kg) is pushed with a force of 20 N for 5 s. Assuming its average velocity is half of its final velocity, what is the average power used to get it moving?

$$\begin{aligned}
p = &\quad e * t &&= (20N) * (5s) = 100N * s \\
f = &\quad p/L &&= (100N * s)/(50kg) = 2m/s \\
f_{\text{average}} = &\quad f/2 &&= (2m/s)/2 = 1m/s \\
P = &\quad e * f &&= (20N) * (1m/s) = 20W
\end{aligned}$$

An inductor (L=15 H) is charged to a current of 4 A over 6 s by a constant voltage. Assuming the average current is half of the final current, what is the average power used to charge it?

$$\begin{aligned}
p = &\quad L * f &&= (15H) * (4A) = 60Wb \\
e = &\quad p/t &&= (60Wb)/(6s) = 10V \\
f_{\text{average}} = &\quad f/2 &&= (4A)/2 = 2A \\
P = &\quad e * f &&= (10V) * (2A) = 20W
\end{aligned}$$

Inductors, like capacitors, have time-varying properties that allow systems to vary over time. When modeling, very tiny steps are taken and effort information is used to calculate a new approximate state value. This in turn produces new flow information, which interacts with the rest of the system to determine the new effort across the inductor, which allows the next change in state to be calculated.

2.4 The Tetrahedron of State

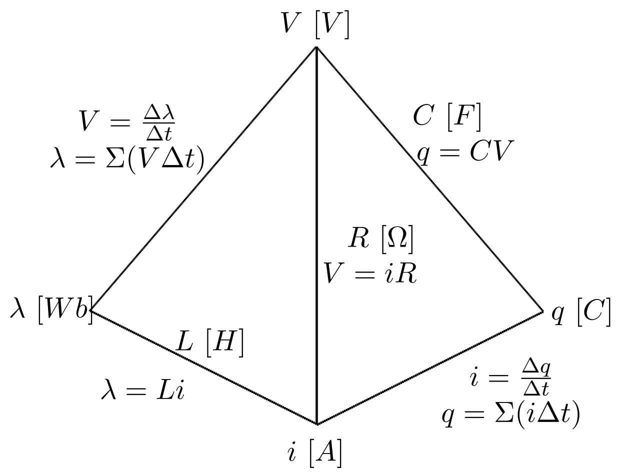
The most important variables used to model systems are effort and flow, which between them give the power of an energy transfer. Three simple elements common to all three domains relate the effort and flow of a bond ot each other.

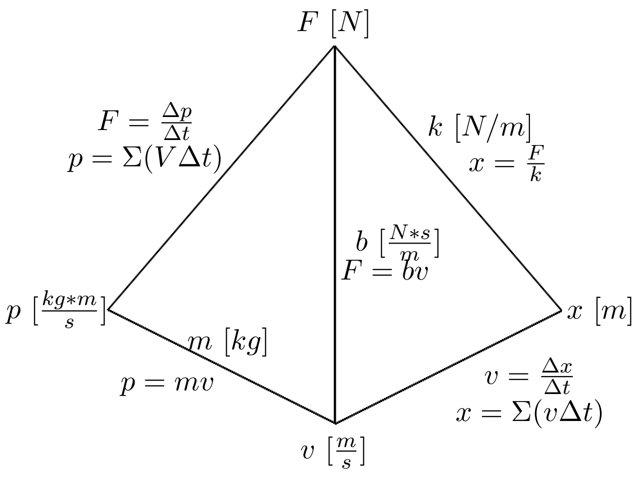
The resistor is a dissipative element that linearly relates effort and flow. Once either one is provided by the system, the resistor determines the other, and thus determines power.

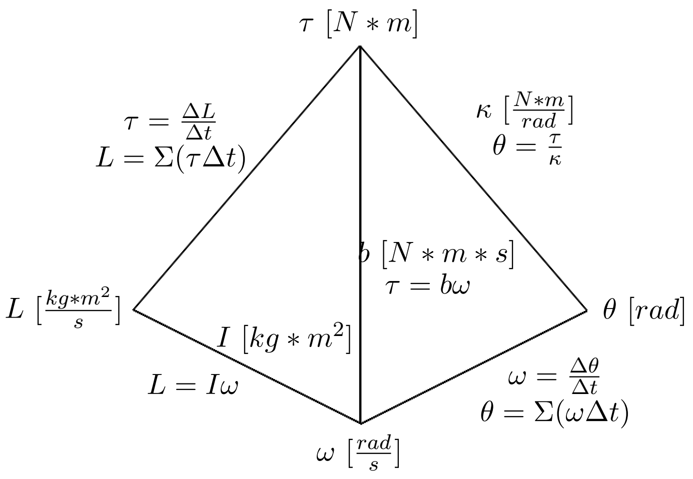
The capacitor is a storage element that stores energy as an effort. The energy stored is represented by the state variable q, which is linearly related to the effort across a capacitor. The flow through a capacitor is the rate at which q, and thus the stored energy, changes.

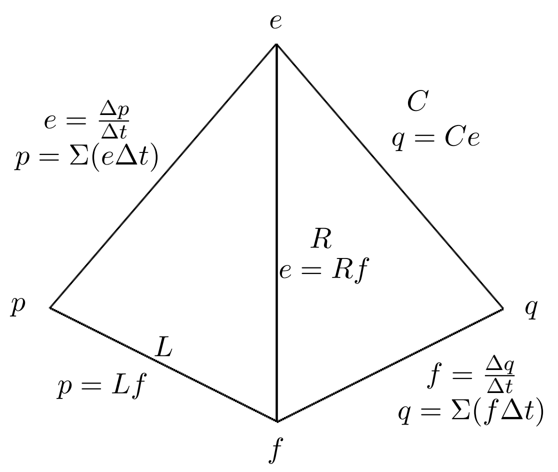
The inductor is a storage element that stores energy in the form of a flow. This storage is reflected by the state variable p, which is linearly related to the flow through the inductor. The effort across the inductor is the rate at which p, and thus the stored energy, changes.

These relationships between the two power variables, e and f, and the two state variables, can be summed up in a diagram called the "Tetrahedron of State." The tetrahedron below is an abstract one, representing an abstract system; the tetrahedron reflecting any specific physical system can be obtained by substituting the corresponding variables in for each abstract one.


$V\ [V]$
$V = \frac{\Delta\lambda}{\Delta t}$
$\lambda = \Sigma(V\Delta t)$
$C\ [F]$
$q = CV$
$R\ [\Omega]$
$V = iR$
$\lambda\ [Wb]$
$q\ [C]$
$L\ [H]$
$\lambda = Li$
$i = \frac{\Delta q}{\Delta t}$
$q = \Sigma(i\Delta t)$
$i\ [A]$


$F\ [N]$
$F = \frac{\Delta p}{\Delta t}$
$p = \Sigma(V\Delta t)$
$k\ [N/m]$
$x = \frac{F}{k}$
$b\ [\frac{N*s}{m}]$
$F = bv$
$p\ [\frac{kg*m}{s}]$
$x\ [m]$
$m\ [kg]$
$p = mv$
$v = \frac{\Delta x}{\Delta t}$
$x = \Sigma(v\Delta t)$
$v\ [\frac{m}{s}]$


$\tau\ [N*m]$
$\tau = \frac{\Delta L}{\Delta t}$
$L = \Sigma(\tau\Delta t)$
$\kappa\ [\frac{N*m}{rad}]$
$\theta = \frac{\tau}{\kappa}$
$b\ [N*m*s]$
$\tau = b\omega$
$L\ [\frac{kg*m^2}{s}]$
$\theta\ [rad]$
$I\ [kg*m^2]$
$L = I\omega$
$\omega = \frac{\Delta\theta}{\Delta t}$
$\theta = \Sigma(\omega\Delta t)$
$\omega\ [\frac{rad}{s}]$



3 Processing Bond Graphs

In order to model a bond graph once it has been created, either through the GUI or by hard coding, it is desirous to develop equations describing the system. This process is completed in three steps: causality definition, equation extraction, and equation postprocessing.

$$\begin{tikzcd}
\text{Se} \arrow[r] & 1 \arrow[r] \arrow[d] & \text{C} \\
& \text{R} &
\end{tikzcd}$$

The graph pictured above is one of the simplest possible that represents a realistic time-varying system. It consists of a source of effort, Se, powering a constant flow through a resistor and capacitor. Electrically, this would be a battery connected to a resistor and capacitor, one of the first circuits introduced in all calculus-based physics courses. Mechanically, this would be someone pushing a side-by-side spring and dashpot at the same speed. This graph will be used as the primary example throughout the section; equations will be derived for it and compared to capacitor-charging equations, and the steps a computer takes to derive equations will be explained.

3.1 The Structure of Equations

The first problem in modeling a system is determining which variables are needed to completely describe the system's behavior. Most variables in a system, such as the effort and flow through each bond, can be calculated from other elements in the system and need not be known to describe the system's state. The only necessary variables are the state variables for inductors (p) and capacitors (q), which describe how much energy each storage element is storing; all other variables can ultimately be calculated from these. In fact, sometimes not all state variables are needed; if derivative causality (discussed below) occurs, some are unnecessary.

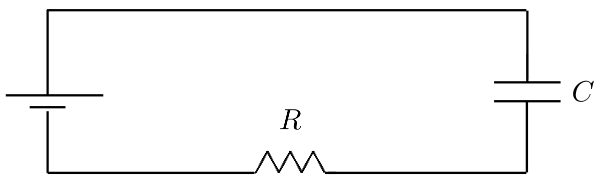
Using state variables to describe the behavior of a system also makes it convenient to calculate the future behavior of the system. The instantaneous effort or flow through a storage element is the rate of change of its state variable, which lets the values of each state variable in the next time step be approximated. These efforts and flows are algebraically determined by the state variables themselves. Thus, a computer can model the behavior of a system once equations for these efforts and flows are determined.

This observation leads to the desired structure of equations describing a system. To be solvable, a system needs one equation for the effort through each inductor and one equation for the flow through each capacitor. These equations can be functions of state variables and input constants (such as from a flow or effort source). Sometimes, it is impossible to create an equation without using effort or flow variables. If these efforts or flows do not already have an equation, then an auxiliary equation describing it is needed.

This type of set of equations is called a system of differential equations. Differential equations are equations made up of numbers, variables, and rates of change of variables. They can be solved to give behavior over time by a number of different algorithms.

3.2 Example Equation Derivation

$$\begin{tikzcd}
\text{Se (0)} \arrow[r, "(1)"'] & \text{1 (2)} \arrow[r, "(5)"] \arrow[d, "(3)"] & \text{C (6)} \\
& \text{R (4)} &
\end{tikzcd}$$



Here is the bond graph from above, pictured with the electrical circuit it represents. This is a standard RC circuit, and its charging equation can be found online or in any college or advanced high school physics textbook. The process for deriving equations for a graph will be demonstrated with this particular system, and compared to the equations for the RC circuit.

First, note that everything in the graph has received a number in parentheses. All elements and bonds in a graph receive a unique identification number. This number can be used as a subscript to variables for clarity. For instance, f1 refers to the flow in bond 1, e3 refers to the effort in bond 3, and $q6$ refers to the position of capacitor 6. Efforts and flows are typically numbered by bond, while state variables receive the number of their storage element.

In this graph, there is one storage element, the capacitor. It is in integral causality (what this means will be described below), so it provides an equation and state variable to the system of equations. There are no other storage elements, so this single equation is adequate. The equation is intended to represent the rate of change of $q6$ (which is $f5$), so is begun as follows:

$$f5 = f5$$

This flow comes from the 1 junction in the center of the graph. A 1 junction has equal flows through all bonds, giving the following equation:

$$f5 = f1 = f3$$

The flow in bond 1 does not help continue the equation. Bond 1 connects to a source of effort, which places no constraint on flow. Thus, the equation must use $f3$. (There is a method to indicate exactly which bond to follow in this situation, called causality. Causality works even on complex graphs, where it is not intuitive which bond should be used. It will be introduced as the first step of how a computer determines the equation).

$$f5 = f3$$

On the other end of bond 3 is a resistor. Resistors constrain the power variables on their bond using the relationship $e = R * f$, so the equation can become the following:

$$f5 = e3/R4$$

The next element met is the 1 junction again, but this time an effort must be determined. 1 junctions constrain efforts to add to zero when signed (that is, efforts into the junction minus efforts out of the junction result in zero). In this case, the equation is $e1 = e3 + e5$. Substituting;

$$f5 = (e1 - e5)/R4$$

The effort in bond 1 is an input constant; no further action must be taken with it. The effort in bond 5 is determined by the capacitor, and is governed by the equation $q = e * C$. Thus,

$$f5 = (e1 - (q6/C6))/R4$$

Now, all variables on the right side are constants, inputs to the system, or state variables, so the equation is finished.

To check the equation, it is rearranged as follows:

$$f5 * R4 = e1 - (q6/C6)$$
$$e1 - (q6/C6) - (f5 * R4) = 0$$

This last equation represents an application of Kirchhoff's second law (the sum of directed voltages around a closed circuit is always zero) to the electrical circuit. The voltage of the battery, minus the voltage of the capacitor, minus the voltage across the resistor, yields 0 regardless of the charge on the capacitor. Thus, the equation derived is valid.

3.3 Causality

The following simple thought experiment demonstrates the principle of causality.

Consider the simple graph below:

$$\begin{tikzcd} S \arrow[r, harpoon] & R \end{tikzcd}$$

This represents a very simple system, where energy is directly converted into thermal energy by a resistor. For the purposes of this thought experiment, this graph represents the system of Alice pushing Bob across a frictional surface (technically, friction does not work this way, but it is easiest to visualize). Bob does not appear in the graph because it is assumed that equilibrium is reached; that is, he is moving at a constant velocity. At constant velocity, he receives no effort (Alice's push works only on friction), so can be removed from the graph.

In the first scenario, the coefficient of friction, R, is 20. If Alice pushes with an effort of 80, Bob ends up moving with a flow of 4 (based on the resistor). In this scenario, Alice has a power of 320.

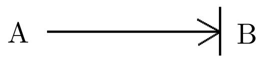
Now Bob puts on his sticky shoes. In these shoes, the coefficient of friction, R, becomes 40. Alice begins to push with the same effort of 80, and Bob only moves with a flow of 2. Alice has a power of 160 in this case.

This isn't fast enough for Bob, so Alice pushes him until he reaches a flow of 4. In order to do this, Alice pushes with an effort of 160, so uses a power of 640 to push Bob.

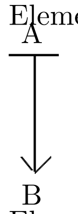
Alice is not satisfied with either scenario while Bob wears his sticky shoes. In one case, he does not move as fast as without them. In the other, she must push harder. Alice now attempts to push Bob with the same effort and flow as with his normal shoes. She pushes him with an effort of 80 at a flow of four. However, friction generates an effort backwards of 160, which attempts to decelerate him. Alice's only options to reach equilibrium (and maintain the same graph structure) are changing her effort to match that of friction, or changing the her flow so that the effort of friction matches hers. In short, Alice cannot always choose both the effort and the flow of the energy transfer. She can only pick one, and the other is set by the resistor on the other end of the transfer.

This simple example illustrates the concept of causality: one side of a power bond determines effort, and the other side determines flow. Neither side can determine both for the same energy transfer. When Alice tried to define both effort and flow while Bob wore his sticky shoes, it did not work. She could define one, but had to allow the resistance she was transferring energy to define the other.

In a bond graph, the causality of a bond is indicated by a causal stroke. A causal stroke is a vertical line (on a horizontal bond; causal strokes are horizontal on a vertical bond) on the side of a bond that determines the flow. The other side determines effort. Here are some examples:



Element A determines the effort on the bond; element B determines the flow.



Element A determines the flow on the bond; element B determines the effort.

The structures used to represent bonds to a computer program all have a parameter defining causality. This parameter indicates whether the element on the input (non-arrow) end of the bond causes effort or flow. The element on the other end causes the other of the two power variables.

3.3.1 Causality Constraints

Almost all elements in a bond graph constrain causality on their connecting bond(s) in some way.

Effort sources provide effort to the system. Therefore, they must cause effort on their bond.

Flow sources provide flow to the system. Therefore, they must cause flow on their bond. Transformers couple the efforts on each bond to each other and the flows on each bond to each other. Therefore, both bonds must have the same causality.

Gyrators couple the effort on one bond to the flow on the other. Therefore, both bonds must have opposite causality.

All bonds on a 0 junction have the same effort. Therefore, only one bond can provide effort information to a 0 junction. All other bonds provide flow information. All bonds on a 1 junction have the same flow. Therefore, only one bond can provide flow information to a 1 junction. All other bonds provide effort information.

Resistors can accept any causality. This is the only element type that does not constrain causality.

Capacitors prefer to receive flow information and provide effort information. If the capacitor receives effort information, its state variable would be forced upon it by the rest of the system, rather than varying freely.

Inductors prefer to receive effort information and provide flow information. If the inductor receives flow information, its state variable would be forced upon it by the rest of the system, rather than varying freely.

The preferred causal state for storage elements (inductors and capacitors) is called integral causality. This is because, in the preferred causal state, the power variable the storage element provides is based on the integral (accumulation over time) of the other power variable. The other causal state is called differential or derivative causality, because in this state the power variable the storage element provides is based on the derivative (rate of chang) of the other power variable. Because a storage element in differential causality has its state variable forced upon it by the rest of the system, it does not add another equation to the system of equations. Integrally causal storage elements are the only ones to receive an equation.

3.3.2 Assigning Causality

causality assignment in a bond graph follows a simple algorithm, which can be automated. The steps are as follows:

1. Choose a source element and assign it proper causality. Use junctions and 2-port elements to extend causal implications as far as possible (e.g. A 1 junction with flow information provided must provide flow information to its other bonds; a 0 junction with effort information provided must provide effort information to its other bonds; a 2port element with one bond assigned defines the other)

2. Repeat step 1 until all source elements have causality assigned.

3. Choose an unassigned storage (C or L) element and assign it its preferred causality. Extend causal implications as far as possible as in step 1.

4. Repeat step 3 until all storage elements have causality assigned. Constraints by junctions and 2ports might assign differential causality to some.

5. Choose an unassigned resistive (R) element (if any remain) and assign it an arbitrary causality. Extend causal implications as far as possible.

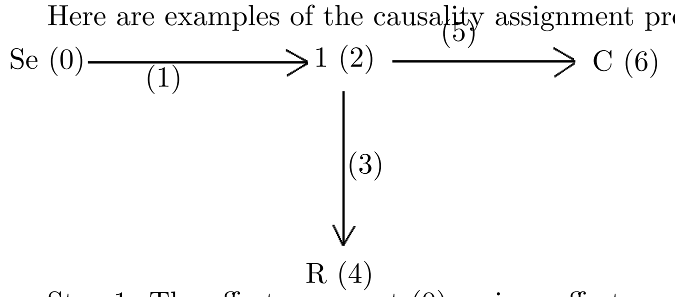
6. Repeat step 5 until all resistive elements have causality assigned.

7. Choose an unassigned bond (if any remain) and assign it an arbitrary causality. Extend causal implications as far as possible.

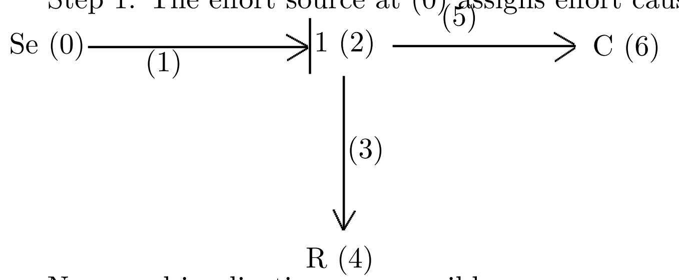
8. Repeat step 7 until all bonds have causality assigned.

If step 5 or later is reached, causality cannot be assigned in exactly one way. This causes an algebraic loop to arise, in which one flow or effort variable is algebraically dependent upon itself. To resolve this issue, computers require auxiliary functions for these variables.

Here are examples of the causality assignment process:



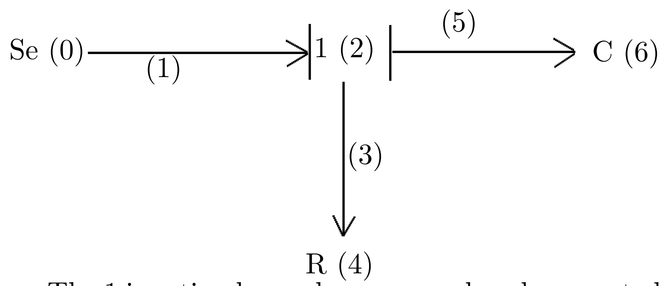
Step 1: The effort source at (0) assigns effort causality to bond 1:



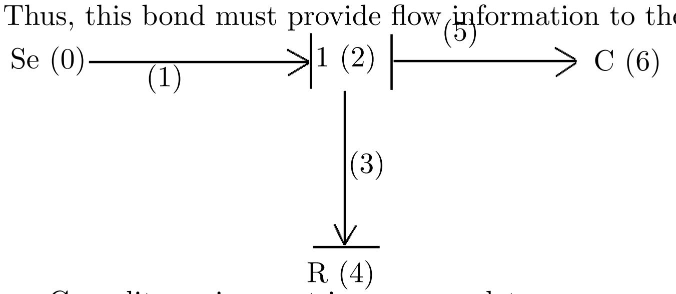
No causal implications are possible.

Step 2: All sources are assigned.

Step 3: The capacitor at (6) receives integral causality. Bond 5 receives flow causality

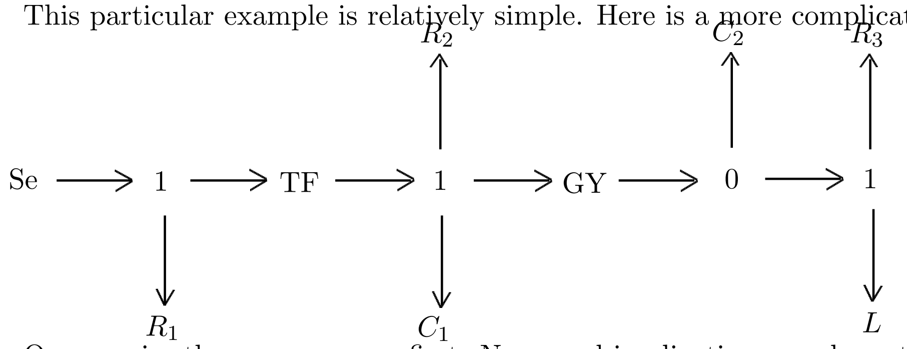


The 1 junction has only one more bond connected to it, but has not yet received flow information. Thus, this bond must provide flow information to the junction, and must have effort causality.

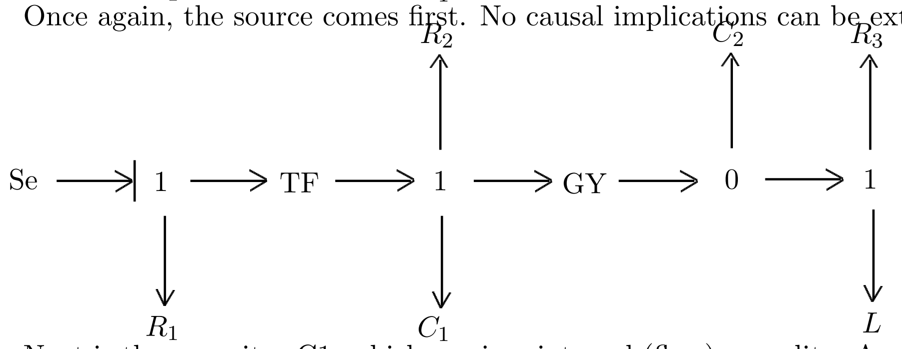


Causality assignment is now complete.

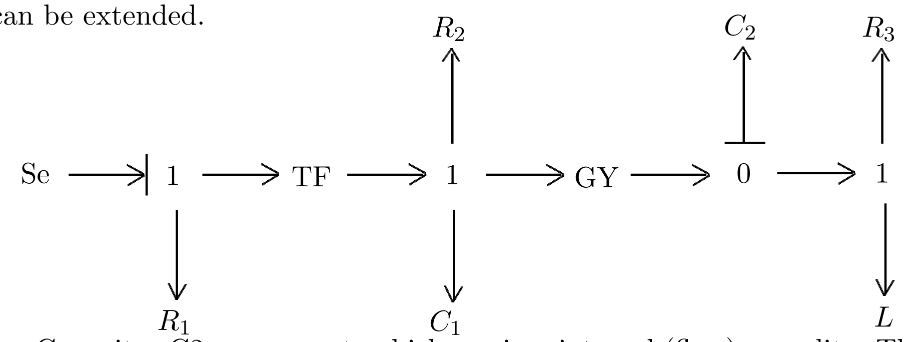
This particular example is relatively simple. Here is a more complicated one:



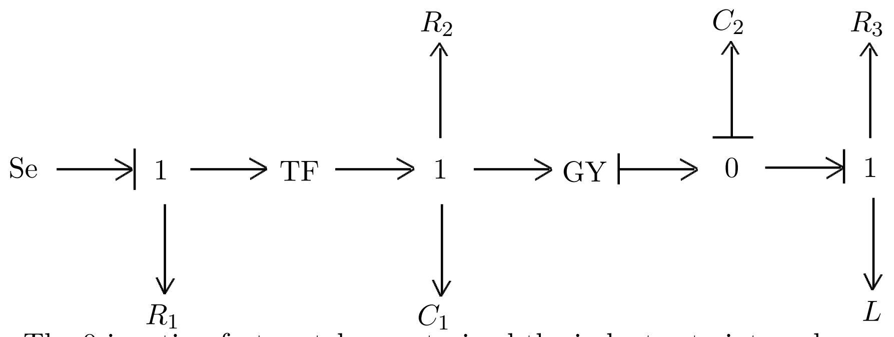
Once again, the source comes first. No causal implications can be extended.



Next is the capacitor C1, which receives integral (flow) causality. Again, no causal implications can be extended.

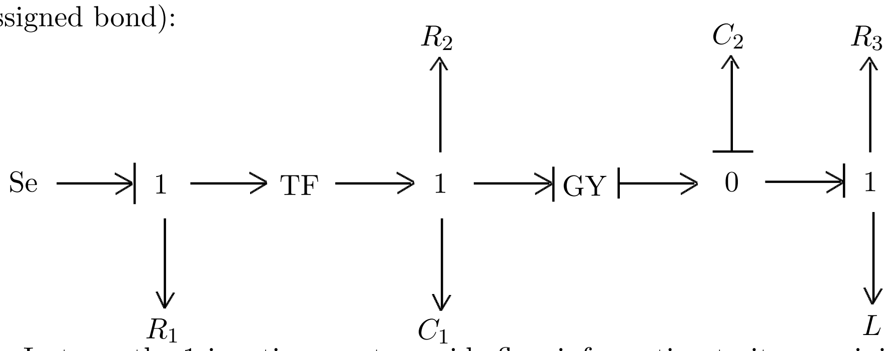


Capacitor C2 comes next, which receives integral (flow) causality. This provides effort information to the 0 junction, which must provide effort information to its other bonds.

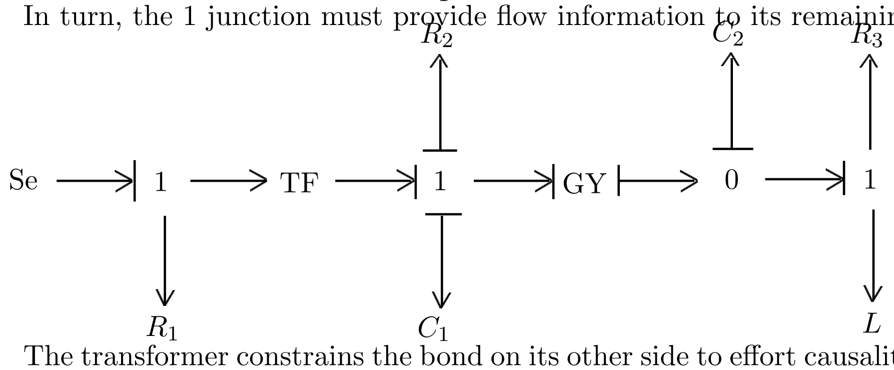


The 0 junction fortunately constrained the inductor to integral causality. If there were another capacitor in place of the inductor, it would be constrained to differential causality.

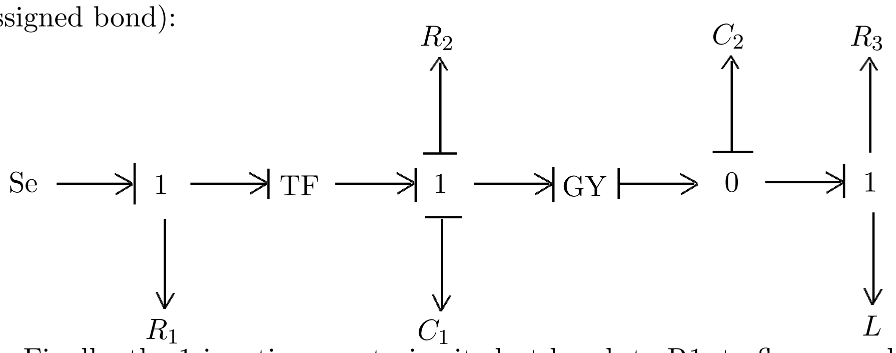
The gyrator constrains the bond on its other side to effort causality (opposite the already assigned bond):



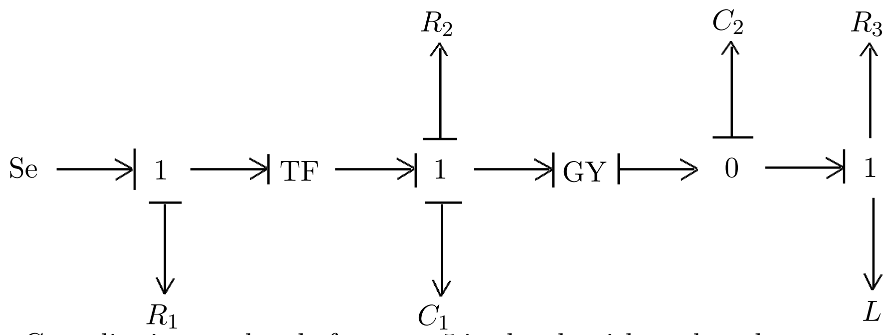
In turn, the 1 junction must provide flow information to its remaining bonds.



The transformer constrains the bond on its other side to effort causality (the same as the already assigned bond):



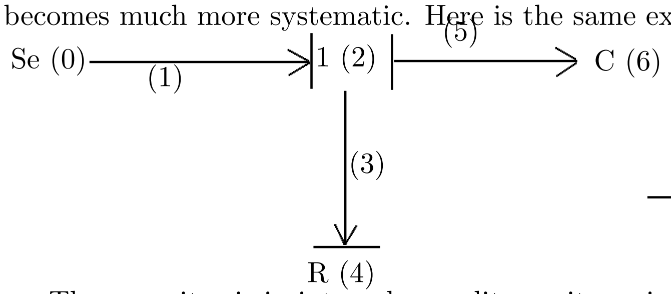
Finally, the 1 junction constrains its last bond, to R1, to flow causality.

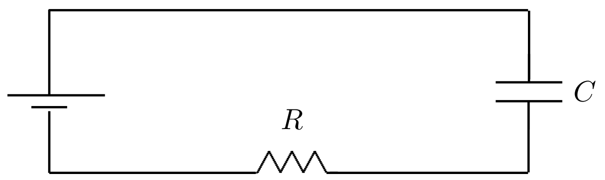


Causality is complete before step 5 in the algorithm, thus there are no algebraic loops. There are three storage elements in integral causality, so there will be three differential equations describing the behavior of this system.

3.3.3 The Importance of Causality

Causality serves to direct a person or computer around a bond graph while deriving equations. The most important role is at junctions. For instance, in the example above of deriving equations, the process required a flow coming from a 1 junction, which meant guessing which flow to use. Rather than guess (likely incorrectly), it is easier to follow causality. The correct bond is the one that provides flow information to the 1 junction. By following causality, derivation of equations becomes much more systematic. Here is the same example as above, but with causality assigned.





The capacitor is in integral causality, so it receives an equation.

$$f5 = f5$$

Flow is determined on (5) from the junction end. The junction receives flow information from f3

$$f5 = f3$$

Using the resistor's equation,

$$f5 = e3/R4$$

The next step goes back to the junction, but requests effort. This is a 1 junction, so the signed sum of efforts must be zero. Substituting,

$$f5 = (e1 - e5)/R4$$

The effort on (1) is from a source and is constant. The effort on (5) comes from the capacitor:

$$f5 = (e1 - (q6/C6))/R4$$

The result is the same equation as before, but this time there is no guessing. Especially in larger, more complicated systems, absolute knowledge of which bond to follow out of junctions makes deriving equations significantly easier. It also makes the process into an algorithm that can be followed by a computer.

3.3.4 Differential Causality

Differential causality is one of two troublesome issues that can arise when getting equations from a bond graph. The other, algebraic loops, can be resolved by auxiliary functions.

The problem with differential causality is that the storage element does not provide a state variable. Thus, if a power variable requiring a state variable (effort on a capacitor or flow on an inductor) is requested, the request must bounce back through the system rather than pick up information from the storage element.

3.3.5 Wrong Requests From Storage Elements

Capacitors prefer to provide effort information and inductors prefer to provide flow information. When flow information is requested from a capacitor, it can only be provided as the time derivative of position. In this case, which can occur during differential causality or when connected to a 0 junction, the capacitor equation is rearranged into q=e*C and its derivative is analytically calculated (which is fairly easy for a computer). This derivative is then substituted into the equation. The same process occurs when effort information is requested from an inductor.

3.3.6 Summary: Extracting Equations from Bond Graphs by a Human

To extract an equation from a bond graph, a human must take the following steps:

1. Assign causality throughout the graph. Identify differential causality on storage elements and algebraic loops.
2. Identify integral causality on storage elements and create an equation for each such element.
3. Propagate these equations through the graph, following causality, until the equation consists only of valid state variables, constants, inputs, power variables with equations, and power variables with an algebraic loop.
4. Create necessary auxiliary functions to resolve algebraic loops.
5. Simplify the system of equations.
6. Solve the system of differential equations, either analytically or numerically. The results indicate the behavior of the system.

Later sections detail how each of these steps can be automated.

4 Automating Equation Production

The method for extracting equations from a bond graph is fairly simple for a human, consisting of the following steps:

1. Assign causality throughout the graph. Identify differential causality on storage elements and algebraic loops.
2. Identify integral causality on storage elements and create an equation for each such element.
3. Propagate these equations through the graph, following causality, until the equation consists only of valid state variables, constants, inputs, power variables with equations, and power variables with an algebraic loop.
4. Create necessary auxiliary functions to resolve algebraic loops.
5. Simplify the system of equations.

6. Solve the system of differential equations, either analytically or numerically. The results indicate the behavior of the system.

An automated system uses a similar procedure, with a few modifications for processing:

1. Sort a list of bond graph elements in order, to simplify referring to specific elements.

2. Check whether arrow conventions are all followed (this one is a check against human error while entering the graph).

3. Assign causality throughout the graph.

4. Identify integral causality on storage elements. Create an equation for each element.

5. Propagate equations through the graph until only valid state variables, constants, etc. occur.

The computer must keep track of power variables already passed through. If one is met again, then an algebraic loop is occuring and evaluation on this branch stops.

6. Check the equations for variables arising from loops. Generate an auxiliary equation for each, and propagate through the graph.

7. Simplify the system of equations (if possible or desired).

8. Solve the system using a differential equation solving algorithm (analytical or numerical).

The Structure of a Graph (and Equations)

The following structures are used to create a graph:

(define-struct bond (id in out causality))

A bond has an id number, in and out (id numbers of the elements it connects), and causality (the word 'effort' or 'flow').

(define-struct 1port (id type bond constant))

A 1-port element has an id number, type (either 'SF, 'SE, 'R, 'L, or 'C), id number of the bond it connects to, and a constant (flow provided, effort provided, resistance, inductance, or capacitance).

(define-struct 2port (id type in out constant))

A 2-port element has an id number, type (either 'TF or 'GY), id numbers of the two bonds it connects to (in and out), and the constant from its appropriate formula.

(define-struct junction (id type bonds))

A junction has an id number, type (either 0 or 1), and a list of bonds that it connects to.

A graph is a list (or temporarily an array) of these elements, ordered by increasing id number. Placeholders are inserted as needed so that every element's index (counting from 0) matches its id number.

The equations manipulated in steps 4-7 are also represented as structures. The following structures form an equation:

(define-struct num (n))

A num represents a number and has a single field for its numerical value.

(define-struct var (s))

A var represents a variable and has a single field for its name (a symbol).

(define-struct op (name left right)

The op structure is a binary operation and the basis of all equations. Its name can be 'equals, 'add, 'sub, 'mul, or 'div. Another equation (possibly another op structure) is nested within left and right.

(define-struct deriv (eqn))

The deriv structure is only used when the wrong variable is requested from a storage element (flow from a capacitor or effort from an inductor). It is used to indicate that the symbolic derivate of its contents is needed to complete the equation. One step an automated system must take is evaluating all deriv structures.

(define-struct pvar (type n))

A pvar (short for power variable) represents the effort or flow on a bond. A pvar can occur in the final system of equations to refer to the rate of change of a state variable or a variable in an auxiliary equation.

(define-struct svar (type n))

A svar (short for state variable) represents the position on a capacitor or momentum of an inductor. Every integrally causal storage element will have an svar in the final system of equations.

(define-struct dpvar (type n dir))

A dpvar (short for directed power variable) is a structure used only in equation stepping. It represents an effort or flow on a bond being stepped a certain direction (either up or down) the bond. The test for completion of the propagation process is whether the equation contains a dpvar.

4.1 Step 1: Sorting a Graph

Sorting a graph by id numbers is the simplest of the eight steps. Any simple or native sorting algorithm can do this, and from there it is a simple matter to pass through the list and add necessary placeholders to space it out.

4.2 Step 2: Checking Arrow Conventions

The standard convention for arrow directions (direction of energy flows) is that arrows must point away from source elements and toward R, L, and C elements. In addition, 2-port elements must have one arrow point towards them and the other arrow point away from them.

To check the arrow convention, the elements on both ends of each bond are checked to see if they don't properly meet the power convention. If either of these elements violates the power convention, bond direction is reversed. This process is repeated on every bond in a graph.

4.3 Step 3: Assigning Causality

In this section, the graph is converted from a list to a mutable vector (an array). This is done to avoid completely rebuilding the graph multiple times. After causality assignment, the graph is converted back to a list.

Causality assignment consists of two substeps, which are alternated between until all bonds have causality. These steps are assignment on a new bond and propagation.

Causalilty assignment on a new bond is based on 1port elements, since these constrain causality based on their properties as elements. To assign causality on a new bond, the program first checks if any source elements have a non-causal bond. If so, causality is assigned on this bond. If no such source elements exist, the program checks storage elements, then resistive elements, and finally bonds, until an element with unassigned causality exists. If a bond receives causality in this step, the program begins causality propagation. If no bond receives causality, then this step is completed.

Causality propagation is based around 2port and junction elements, since these constrain causality based on the causality on other bonds. To propagate causality, the program checks whether enough causality has been defined on a 2port or junction to constrain the other bonds. If so, these bonds have their causality set. The propagation program continuously loops through a graph until it makes an entire pass without changing the graph.

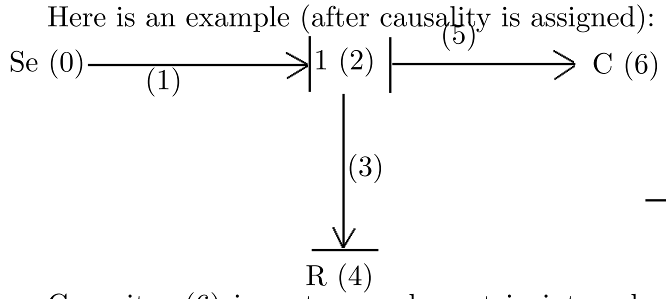
2ports constrain causality in the following patterns:

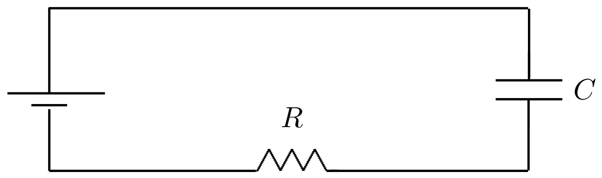
0 junctions can receive effort information from only a single bond, and 1 junctions can receive flow information from only a single bond. Thus, they look something like this:

4.4 Step 4: Identifying Integral Causality

To identify integral causality, a program loops through all elements in a graph and checks the bond connecting each storage element. If a storage element is in integral causality, it gets an equation consisting of an op structure reflecting equality. The left side of the op is the charging pvar on the bond, and the right side is the dpvar for the same variable going up the bond. This dpvar is later stepped through.

Here is an example (after causality is assigned):





Capacitor (6) is a storage element in integral causality, so it gets an equation. Its equation looks like this:

(make-op 'equals (make-pvar 'flow 5) (make-dpvar 'flow 5 'up))

4.5 Step 5: Propagation

Equation propagation is central to automating and possibly its most important step. All other steps mostly make life easier for a human user and reduce human error; this step actually models the graph as a system of equations. Naturally, this step is also one of the two most complicated portions of the entire program, due to the many different actions that must be taken depending on the current state.

Equation propagation is based around the function process-element. This function takes in a graph and a dpvar and returns the appropriate equation substituted in place of the dpvar. The following are examples of possible dpvars from the graph pictured in step 4, along with an equation equivalent to what process-element would produce (in some cases, these equations are simplified for clarity).

(make-dpvar 'effort 1 'up) -¿ (make-pvar 'effort 1)

This is an input variable and a constant, so does not need further processing. Transformation to a pvar indicates that this branch of the equation is done.

(make-dpvar 'flow 1 'up) -¿ (make-dpvar 'flow 1 'down)

The effort source provides no constraint on flow, so reflects any inquiries about flow information back to the rest of the system.

(make-dpvar 'effort 1 'down) -¿ (make-op 'add (make-dpvar 'effort 3 'down) (make-dpvar 'effort 5 'down))

In a 1 junction, directed efforts sum to zero. Thus, when an effort is requested from a 1 junction, a sum (or difference) of the other efforts results. The same holds for flows on a 0 junction.

(make-dpvar 'flow 1 'down) -¿ (make-dpvar 'flow 3 'down)

Since flows are all equal in a 1 junction, this step follows causality. The same holds for effort on a 0 junction.

(make-dpvar 'effort 3 'up) -¿ (make-op 'sub (make-dpvar 'effort 1 'up) (make-dpvar 'effort 5 'down))

(make-dpvar 'flow 3 'up) -¿ (make-dpvar 'flow 3 'down)

Junctions behave in the same manner, no matter which bond queries them for information.

(make-dpvar 'effort 3 'down) -¿ (make-op 'mul (make-dpvar 'flow 3 'up) (make-num R4))

(make-dpvar 'flow 3 'down) -¿ (make-op 'div (make-dpvar 'effort 3 'up) (make-num R4))

When requesting a power variable from a resistor, it applies the abstract resistance formula.

(make-dpvar 'effort 5 'up) -¿ (make-op 'sub (make-dpvar 'effort 1 'up) (make-dpvar 'effort 3 'down))

(make-dpvar 'flow 5 'up) -¿ (make-dpvar 'flow 3 'down)

Junctions behave in the same manner, no matter which bond queries them for information.

(make-dpvar 'effort 5 'down) -¿ (make-op 'div (make-svar 'q 6) (make-num C6))

Requesting an effort from a capacitor returns the abstract capacitance equation. There are no dpvars in this equation, so this branch of the equation has finished stepping.

(make-dpvar 'flow 5 'down) -¿ (make-deriv (make-op 'mul (make-dpvar 'effort 5 'up) (make-num C6)))

Requesting a flow from a capacitor returns the time derivative of its position (from the abstract equation). The stepper would continue stepping through the equation and evaluate the derivative afterwards.

The equation stepper takes in an equation made in step 4 and propagates it to completion. It calls process-element on every dpvar and substitutes the result into the position formerly occupied by the dpvar, then repeats until all dpvars have been evaluated.

Here is the sequence of steps taken to derive the equation for the graph pictured in step 4:

(make-op 'equals (make-pvar 'flow 5) (make-dpvar 'flow 5 'up))

(make-op 'equals (make-pvar 'flow 5) (make-dpvar 'flow 3 'down))

(make-op 'equals (make-pvar 'flow 5) (make-op 'div (make-dpvar 'effort 3 'up) (make-num R4)))

(make-op 'equals (make-pvar 'flow 5) (make-op 'div (make-op 'sub (make-dpvar 'effort 1 'up) (make-dpvar 'effort 5 'down)) (make-num R4)))

(make-op 'equals (make-pvar 'flow 5) (make-op 'div (make-op 'sub (make-pvar 'effort 1) (make-op 'div (make-svar 'q 6) (make-num C6))) (make-num R4)))

This last equation can be transcribed as follows:

f5 = (e1 - (q6 / C6)) / R4

which matches what was derived as the charging equation in a previous section.

The stepper has one more important role, which is watching out for algebraic loops. It does this by maintaining a list of all dpvars passed through on the current branch of the equation (op structures act like branches in a binary tree). If a dpvar that has already been encountered shows up again, the stepper replaces it with a pvar, terminating the branch and preventing an infinite loop. The extra pvar is dealt with in step 6.

4.6 Step 6: Proofreading (Resolving Algebraic Loops)

Once causality is assigned, equations for integral storage elements have been fully propagated, and derivatives have been evaluated, the next step is to proofread and ensure that only valid variables exist in the right side of the equations, which all have the structure (make-op 'equals left right). Valid variables consist of inputs from constant sources, state variables from integrally causal storage elements, and all variables that have their own equations (make up the right side of an equals-type op structure). The proofreader checks through all equations for invalid power variables. If none exist, it returns the equations as they are. Should the proofreader find an invalid power variable, it creates a new equation with a dpvar for that power variable directed to follow causality. All such new equations are fed back through the stepper, which completes the auxiliary equations. The proofreader then checks over these auxiliary equations again. If needed, it can create even more auxiliary equations; otherwise, it returns the equations from the stepper.

4.7 Step 7: Simplification

The equation derived in step 5 for the graph is technically a complete differential equation and can, in theory, immediately be solved. In practice, however, this equation is not ready to be solved. Steps 5 and 6 can easily output equations with a lot of redundancy, and a simplified representation is generally desirable.

This is the point at which the assumption, several sections back, that every element in a system scales linearly becomes important. If every single element affects the pvars specified by the equations either as a constant or by linearly scaling, then the system of differential equations is linear and can be expressed in matrix form. The preferred form for this representation is

$$AX = BU$$

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} pvar_1 \\ pvar_2 \\ \vdots \\ pvar_n \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & \dots & \dots & b_{1m} \\ b_{21} & b_{22} & \dots & \dots & b_{2m} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & \dots & b_{nm} \end{bmatrix} \begin{bmatrix} svar_1 \\ svar_2 \\ \vdots \\ svar_{m-1} \\ 1 \end{bmatrix}$$

This is a system of n equations for n power variables, which are stored in matrix X. Matrix A contains the coefficients for each power variable. Matrix U stores the state variables for the integrally causal storage elements, as well as the number 1 at the bottom. This 1 allows for constants to be represented. Matrix B contains the coefficient for the state variables and the constants.

In addition, matrices X and U are sorted, so the first (m-1) power variables in X are the rates of change of the state variables in order. This simplifies modeling the differential equations.

The first step of simplification is to replace all svar and pvar structures with var structures. A list of all var structures that occur in any of the equations is compiled, and regular expressions are used to separate them into replacements for state variables, the power variables which are the rates of change of state variables, and the auxiliary power variables. Each set is sorted in order based on the id number of the bond or element it is on in the graph. The state variables are then assembled into matrix U and the power variable lists are appended and then assembled into matrix X.

Next, the equations are algebraically rearranged. First, all subtraction is replaced by the addition of a negative. Then, the distributive property is repeatedly applied. This causes all sums to bubble up in the tree structure of an equation and all products and quotients to bubble down. After full distribution is done, each equation is broken into a list of terms that add together to create zero. The terms cannot have add-type ops in them, only mul-type and div-type. In addition, the linearity assumption ensures each term contains either a single variable or no variable (for the constant term).

Each of these product/quotient terms can be expressed as the product of a variable and a number (or just a number). The variable is obtained by searching all branches for a var structure, and the constant is obtained by substituting (make-num 1) for the variable, parsing the equation into actual numbers and operators (rather than structures), and evaluating.

To assemble coefficient matrices A and B, then, the program breaks every equation up into product/quotient terms. The row in the coefficient matrix each term goes in depends on which equation contained the term. The column in the coefficient matrix (and which matrix, A or B) each term goes in depends on what variable the term has (or doesn't have). The coefficient for each product/quotient term in every equation is determined and placed in the corresponding position in the coefficient matrices (which are sorted to match the ordering of matrices X and U). Finally, the program assembles the four matrices, A, X, B, and U, into a list, which is passed to the differential equation solver.