

Train **P**ropulsion **S**imulator

Table of Contents

Introduction.....	1
1. Create a TPS Input File.....	2
Track.....	3
Coupler.....	9
Car.....	10
Locomotive.....	16
Locomotive Operator.....	19
Train Consist.....	24
Simulation Parameters.....	28
Forced Speed File.....	30
Important TPS Quirks and Limitations.....	32
2. Run a TPS Simulation and View Results.....	34
Open TPS.....	34
Run Simulation.....	35
Format of Result Files.....	37
View Results.....	41
3. TPS Mathematical Algorithms.....	46
Track Model.....	46
Coordinate system.....	46
Grade.....	46
Curvature.....	47
Superelevation.....	49
Coupler Model.....	50
Car Model.....	52
Local Coordinate System.....	52
Overview of Forces Acting On a Car.....	53
Gravitational Force.....	53
Reactive Centrifugal Force.....	54
Curving Resistance.....	56
Propulsion Resistance.....	57
Inter-Car Coupler Forces.....	58
Analytical Formulation of the Relative Angle Between Adjacent Cars.....	61
Practical Implementation of the Relative Angle Between Adjacent Cars.....	63
Longitudinal Component of Inter-Car Forces.....	64
Lateral Component of Inter-Car Forces.....	66
L/V Ratio.....	71
Locomotive Model.....	78
Tractive Effort.....	78
Dynamic Brake.....	79
Rail Vehicle Hand Brake Model.....	79
Locomotive Independent Brake Model.....	79
Locomotive Automatic Brake Valve Model.....	81

Car Air Brake System Model.....	81
Car Control Valve.....	81
Mass Flow Rates as a Function of the Mode.....	82
Mode Thresholds.....	84
Brake Cylinder and Rigging.....	85
Piston Force.....	86
Normal Force Between Brake Shoes and Wheels.....	86
Retarding Brake Force.....	88
Maximum Brake Cylinder Pressure Under Full-Service Braking.....	89
Brake Rigging Leverage Ratio.....	91
Brake Pipe Model.....	92
Governing Differential Equations.....	94
Continuity Equation.....	94
Momentum Equation.....	96
Finite Element Formulation.....	98
Weighted Residual Form of Continuity Equation.....	99
Weighted Residual Form of Momentum Equation.....	100
Shape Functions.....	101
Integral Evaluations for Finite Element Mass Matrix.....	101
Integral Evaluations for Finite Element Stiffness Matrix.....	104
Integral Evaluations for Finite Element Forcing Vector.....	106
Matrix Form of Differential Equation for Continuity Equation.....	108
Matrix Form of Differential Equation for Momentum Equation.....	110
Row of Global Matrix For Internal Node of Continuity Equation.....	112
Row of Global Matrix For Internal Node of Momentum Equation.....	113
Row of Global Matrix For First Node of Continuity Equation.....	113
Row of Global Matrix For First Node of Momentum Equation.....	115
Row of Global Matrix For Last Node of Continuity Equation.....	117
Row of Global Matrix For Last Node of Momentum Equation.....	118
Linear Interpolation.....	118
Appendix A. Visualizing User-Defined Functions.....	121
Running TPS Function Visualizer.....	121
Creating a TPS Function Visualizer Input File.....	123
Appendix B. Mass Flow Rate Equation.....	125
Appendix C. Brake Shoe Temperature Modeling.....	126
References.....	128

Introduction

The Train Propulsion Simulator (TPS) is a program that simulates longitudinal train dynamics. This document provides details on how to use TPS. It also describes the mathematical assumptions and algorithms used in TPS.

TPS is distributed under version 3 of the General Public License (GPLv3). A copy of the GPL can be found at <http://www.gnu.org/licenses/gpl-3.0.html>. TPS is written in the C++ programming language, and the source code is available at its GitHub repository, which can be found at www.github.com/lnaz01/train-propulsion-simulator. Note that testing has only been performed on Microsoft Windows-based computers, and throughout this manual, it will be assumed the user is running Microsoft Windows. However, only minor changes, if any at all, would likely be needed to run TPS on a Linux machine.

Chapter 1 describes the proper formatting of a TPS user input file. TPS does not have a graphical user interface. Rather, TPS is text-based and runs from a command prompt. This results in the user having to put in the effort to generate a text input file with the proper formatting.

Chapter 2 describes how to run a simulation once the user input file has been written. It also explains the format of the resulting files output by TPS and discusses potential options for viewing these numerical results.

Chapter 3 puts forth the models and assumptions behind TPS. Understanding these models and assumptions will help the user provide valid input to and receive valid output from TPS.

1. Create a TPS Input File

TPS does not include a graphical user interface. Rather, TPS users must create a text input file with the proper formatting. The formatting of the input file as well as the threshold constraints for various data fields is detailed in this section.

In order to create an input file, the built-in Windows Notepad or an equivalent type of text editor should be used. A good option is the Notepad++ text editor as it shows line numbers and provides other features that may aid in the creation of a TPS input file.

Notepad++ can be downloaded at <https://notepad-plus-plus.org>.

A TPS input file requires the creation of the following components:

- 1 track
- 1 or more couplers
- 1 or more rail vehicles¹
- 1 or more locomotive operators (if there is 1 or more locomotives in the train consist)
- 1 train consist
- 1 simulation parameters section

There are ordering rules when defining the above components. The following precise ordering rules must be followed when creating a TPS input file:

- Exactly 1 track must be defined
- 1 or more couplers must be defined, and all couplers must be defined before the train consist is defined
- 1 or more rail vehicles must be defined, and all rail vehicles must be defined before the train consist is defined
- If 1 or more locomotives are defined, then at least 1 locomotive operator must be defined, and all locomotive operators must be defined before the train consist is defined and before the track is defined
- Conversely, if 1 or more locomotive operators is defined, then at least 1 locomotive must be defined

¹ A “rail vehicle” encompasses both cars and locomotives.

- Exactly 1 train consist must be defined
- Exactly 1 simulation parameters section must be defined, and the simulation parameters section must be defined after the train consist is defined

The above list of ordering rules is fairly complicated. The following list of ordering rules provides less flexibility for writing the input file but is easier to follow:

- Define the track first
- After defining the track, then define all couplers, rail vehicles, and locomotive operators. Note that if one or more locomotives is defined, then at least one locomotive operator must be defined, and vice versa.
- After defining all couplers, rail vehicles, and locomotive operators, then define the train consist
- After defining the train consist, then define the simulation parameters section

The formats for defining each of the components (track, coupler, car, locomotive, locomotive operator, and simulation parameters section) are put forth in the following subsections.

Track

Only one track may be defined in a TPS input file. A track is defined by three functions². The three functions and the allowable values for the independent and dependent components of each function are summarized in Table 1.

Table 1. Track functions

Function	Behavior	Independent Variable (IV) Units	IV Minimum Value	IV Maximum Value	Dependent Variable (DV) Units	DV Minimum Value	DV Maximum Value
1. Grade	Piecewise-smooth	Feet	0.0	1,056,000.0	Percent	-5.0	5.0
2. Curvature	Piecewise-linear	Feet	0.0	1,056,000.0	Degrees	-10.0	10.0
3. Super-elevation	Piecewise-linear	Feet	0.0	1,056,000.0	Inches	-5.0	5.0

The convention is adopted that positive curvature refers to curving to the right, and

- 2 Throughout this chapter, the term “function” is used in the mathematical sense. Specifically, a function is assumed to represent a physical variable, and each function is assumed to have one independent variable and one dependent variable. The terms “function” and “physical variable” are used interchangeably.

negative curvature refers to curving to the left. This is summarized in Table 2.

Table 2. Curvature sign convention

Curvature Sign	Physical Meaning
Positive	Curving to right
Negative	Curving to left

In addition, the convention is adopted that positive superelevation refers to the case where the right rail (when looking forward in the direction of motion) is raised above the left rail, and negative superelevation refers to the case where the left rail is raised above the right rail. Therefore, positive superelevation is properly associated with negative curvature, and negative superelevation is properly associated with positive curvature. This sign convention is summarized in Table 3.

Table 3. Superelevation sign convention

Superelevation Sign	Curvature Sign	Physical Meaning
Positive	Positive	Improper superelevation (curving to right with right rail raised above left rail)
Positive	Negative	Proper superelevation (curving to right with left rail raised above right rail)
Negative	Positive	Proper superelevation (curving to left with right rail raised above left rail)
Negative	Negative	Improper superelevation (curving to left with left rail raised above right rail)

All functions defined in a TPS input file must consist of one or more intervals, with a maximum of 2000 intervals. This applies to all track functions as well as other functions defined for other components (coupler, car, etc.) in the following sections. Intervals may be piecewise-smooth, piecewise-linear or step intervals. Piecewise-smooth intervals consist of between two to 30 points with cubic spline interpolation. Piecewise-linear intervals consist of exactly two points with linear interpolation. Step intervals consist of two points where the dependent variable values of the two points are equal. Step intervals are only used in the definition of a locomotive operator (see the “Locomotive Operator” section beginning on page 19).

In addition to the requirements outlined in Table 1, the three track functions (grade, curvature, and superelevation) must be at least 52,800.0 feet (10.0 miles) long, and as documented in the table, the maximum allowable track length is 1,056,000.0 feet (200.0 miles). The plot in Figure 1 shows the valid function space for the grade and

superelevation functions. This valid function space is shown by the dashed gray line. Note that no label is shown for the Y axis. In the case of the grade function, the Y axis should be labeled “Grade (%)”, and in the case of the superelevation function, the Y axis should be labeled “Superelevation (inches)”. The plot in Figure 1 further shows the valid space for a first data point in green as well as the valid space for a final data point in red. As can be seen, the independent value of the first point must be zero, and the independent value of the final point must be greater than or equal to 52,800 feet (10.0 miles) and less than or equal to 1,056,000.0 feet (200.0 miles).

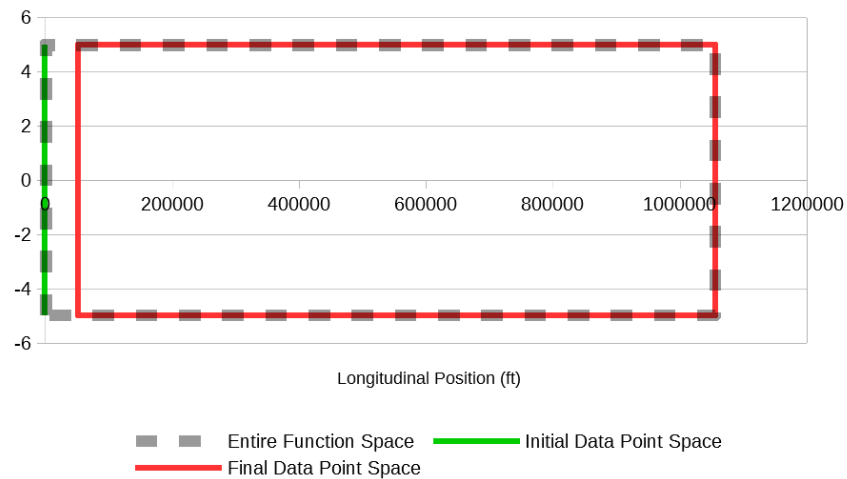


Figure 1. Function space for track grade and track superelevation functions

The plot in Figure 2 shows similar green and red portions for the curvature function.

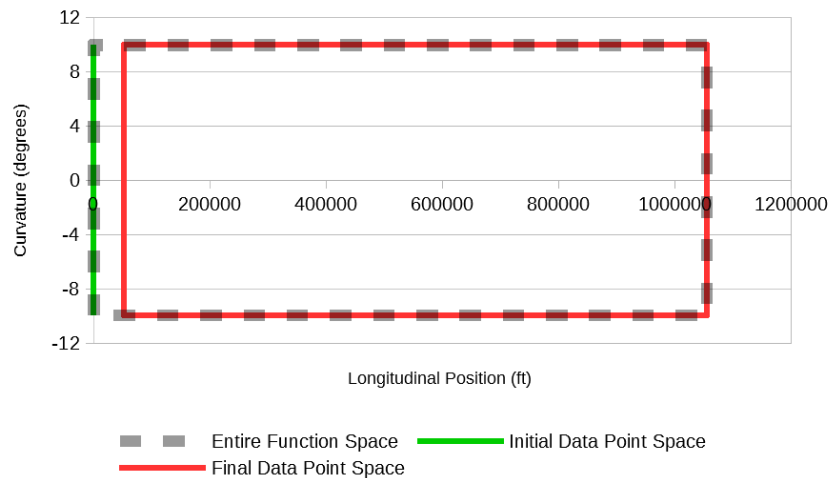


Figure 2. Function space for track curvature function

Figure 3 is a screen capture showing an example of valid syntax for the track component portion of the input file. The text in Figure 3 will be explained line-by-line as much of the syntax is applicable to other components (couplers, cars, locomotives, and locomotive operators) described in the following sections. The first line represents a comment. Any text in the same line after (i.e., to the right of) the # symbol is ignored by the TPS input file processor. In addition to a comment taking up an entire line, TPS also allows inline comments. An example of inline comments is shown in Figure 15 on page 26.

One other important point is that the TPS input file processor removes all the spaces in a line before processing the line. This means that the user may enter as many spaces within a line as they wish. Spaces may even be entered to separate the digits of a number. This may be useful if a number is large and has many digits. For example, writing the number 1,000,000.0 without commas can be confusing as the number of zeros is harder to count. TPS does not allow commas to separate the digits in a number because commas are used to separate the independent and dependent values of a function, which will be discussed in more detail in the coming paragraphs. However, the user may use spaces to replace the traditional commas. For example, instead of writing 1000000.0, the user may choose to enter 1 000 000.0 as a representation of 1,000,000.0. The space between the 1 and the first 0 as well as the space between the third 0 and fourth 0 will be ignored by TPS.

The second line in Figure 3 consists of the keyword `Track_` which indicates the start of the definition of the track component. The third line is simply blank. In order to make the text more readable, users may include as many blank lines as they wish. The fourth line is a comment line which indicates that the following lines define the functions of a track. The fifth line is another comment which indicates that the first function to be defined is the track grade function. The track functions must be defined in the order presented in Table 1, i.e., the grade function must be defined first, the curvature function second, and the superelevation function third. The sixth line is the keyword `Function_` which indicates the start of a function.

```

1  # Fifteen mile track segment
2  Track_
3
4  # FUNCTIONS
5  # 1. track grade
6  Function_
7  0.0, 0.0; 5280.0, 1.0; 21120.0, 0.0
8  21120.0, 0.0; 26400.0, -0.5; 29040.0, -1.0; 47520.0, 0.0
9  47520.0, 0.0; 79200.0, 2.5
10 _Function
11
12 # 2. track curvature
13 Function_
14 0.0, 0.0; 10560.0, 0.0
15 10560.0, 0.0; 10660.0, 2.0
16 10660.0, 2.0; 15840.0, 2.0
17 15840.0, 2.0; 15940.0, 0.0
18 15940.0, 0.0; 79200.0, 0.0
19 _Function
20
21 # 3. track superelevation
22 Function_
23 0.0, 0.0; 10560.0, 0.0
24 10560.0, 0.0; 10660.0, -1.0
25 10660.0, -1.0; 15840.0, -1.0
26 15840.0, -1.0; 15940.0, 0.0
27 15940.0, 0.0; 79200.0, 0.0
28 _Function
29
30 _Track
31

```

Figure 3. Example of valid track portion of input file

Functions consist of intervals, and intervals consist of two-dimensional points. A point's first and second dimensions represents its independent and dependent variable values, respectively. Each line forms an interval, and within the interval, two-dimensional points are separated by a semicolon (;), and the independent and dependent values of a point are separated by a comma. For example, line 7 in Figure 3 defines a first interval for the track grade function. The interval consists of three points. The first point has an independent value of 0.0 feet, which is a requirement for all three functions, and a dependent value of 0.0 percent; the second point has an independent value of 5280.0 feet and a dependent value of 1.0 percent; and the third point has an independent value of 21120.0 feet and a dependent value of 0.0 percent. Every interval must consist of at least two points. The track grade function is a piecewise-smooth function (Table 1), and as previously mentioned, piecewise-smooth functions can have up to 30 points per interval. Cubic spline interpolation is used to calculate interpolated values.

Line 8 defines the second interval for the grade function. In this example, the second interval consists of four points. The independent value of the first point of the second interval (in this case 21120.0 feet) must be equal to the independent value of the last point of the first interval in order to ensure a function without “gaps”.

Line 9 defines a third interval for the grade function. In this example, the third interval consists of only two points.

Line 10 has the keyword `_Function` which indicates the end of a function. Note that additional intervals could have been defined. A function can consist of as many as 2000 intervals. However, only three intervals are defined in this example.

The user may wish to visualize the spline interpolated values of a function, such as the track grade function defined in Figure 3. Appendix A on page 120 shows how this can be done using TPS’s function visualizer auxiliary program.

Lines 13-19 define the curvature function. The syntax is basically identical to the syntax for the grade function. However, the curvature function is a piecewise-linear function, as indicated in Table 1, rather than a piecewise-smooth function. This means that each interval must consist of exactly two points. In the example shown, the curvature function consists of five intervals. Note that for the first interval, the value of the independent variable must be equal to 0.0 feet which, as stated previously, is a requirement. For all other intervals, the value of the independent variable of the first point is equal to the value of the independent variable of the second point of the preceding interval.

Lines 22-28 define the superelevation function. This function is also a piecewise-linear function (Table 1), so each interval only consists of two points. This function consists of five intervals, like the curvature function.

Note that the independent component of the final point of the final interval for each of the three track functions must be equal. In this case the independent component of the final point of the final interval of each of the three functions is equal to 79,200.0 feet. This indicates that the track being defined as an input to TPS is 79,200.0 feet (i.e., 15.0 miles) long. This value of 79200.0 feet is greater than the minimum allowable track length of 52,800 (i.e., 10.0 miles) put forth earlier in this section and is less than the maximum allowable track length of 1,056,000.0 feet (i.e., 200.0 miles) shown in Table 1.

Line 30 consists of the keyword `_Track` which indicates the end of the definition of the track component.

Coupler

A coupler definition consists of one function which represents the force-deflection (i.e., the force-displacement) curve of the coupler. The allowable values of this function's independent and dependent components are summarized in Table 4. Note that in TPS when defining a coupler, positive displacement indicates tension, and negative displacement indicates compression.

Table 4. Coupler functions

Function	Behavior	Independent Variable (IV) Units	IV Minimum Value	IV Maximum Value	Dependent Variable (DV) Units	DV Minimum Value	DV Maximum Value
1. Force-deflection curve	Piecewise-linear	Inches	-5.5	5.5	Kips	-550.0	550.0

In addition to the requirements outlined in Table 4, the coupler function must have an initial data point that has an independent value less than or equal to -3.5 inches and a final data point that has an independent variable value greater than or equal to 3.5 inches. In addition, the initial data point must have a dependent variable value less than -350.0 kips, and the final data point must have a dependent variable value greater than 350.0 kips. These requirements for the loading function are summarized graphically in Figure 4.

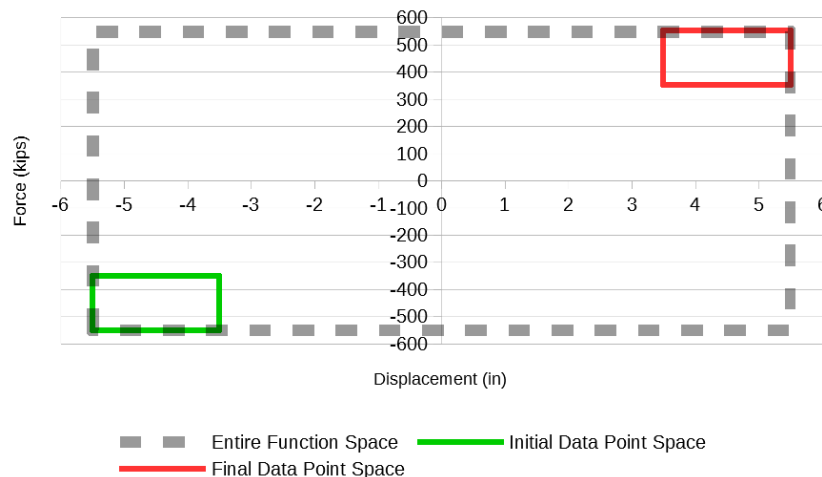


Figure 4. Function space for coupler force-deflection function

In addition to all the requirements put forth above, the slope of each piecewise-linear portion of both the coupler force-deflection function must be greater than 1.0 kips per inch and less than 1000.0 kips per inch.

Figure 5 is a screen capture showing an example of valid syntax for a coupler definition portion of the input file. The coupler definition is shown as beginning at line 37 with the keyword `Coupler_`. This is done for illustration purposes since it is assumed that the definition of this coupler is being placed in the same input file after the track that was defined in Figure 3 on page 7. It can be seen in Figure 3 that the definition of the track ends on line 30. In this example, several blank spaces were left between the end of the definition of the track and the beginning of the definition of the coupler in order to clearly separate the definition of the track and the definition of the coupler, thereby making the input file easier to read.

The text format defining a coupler is similar to the text format defining a track, as described in the previous section. However, the definition of a coupler begins with the keyword `Coupler_` (line 37) and ends with the keyword `_Coupler` (line 49). A coupler is defined by a single function, as shown in Table 4. The coupler force-deflection function is piecewise-linear which means that each interval must consist of exactly two points. In Figure 5, lines 41-47 define the force-deflection function.

```

36 # Standard friction coupler (coupler index 1)
37 Coupler_
38
39 # FUNCTIONS
40 # 1. force-deflection function
41 Function_
42 -4.0, -400.0; -3.0, -325.0
43 -3.0, -325.0; -0.75, -200.0
44 -0.75, -200.0; 0.75, 200.0
45 0.75, 200.0; 3.0, 325.0
46 3.0, 325.0; 4.0, 400.0
47 _Function
48
49 _Coupler
50

```

Figure 5. Example of valid coupler portion of input file

Car

In TPS, a car has 11 physical constants. The maximum and minimum allowable values of these physical constant parameters are summarized in Table 5.

Table 5. Car physical constants

Physical Constant	Units	Minimum Value	Maximum Value
1. Weight	Kips	30.0	600.0
2. Length	Feet	40.0	110.0
3. Number of Axles	Unitless	4	6
4. Cross-Sectional Area	Square Feet	20.0	200.0
5. Streamlining Coefficient	Unitless	1.0	30.0
6. Maximum Net Braking Ratio	Unitless	0.01	0.2
7. Hand Brake Status	Unitless	0 (released)	1 (applied)
8. Hand Brake Efficiency Ratio	Unitless	0.01	0.2
9. Truck center spacing	Feet	50% of car length	95% of car length
10. Coupler height	Feet	1.0	5.0
11. Center-of-gravity height	Feet	1.0	15.0

Most of the terms in Table 5 are self-explanatory. However, a couple points should be made. The second physical constant in Table 5 refers to the length from the end of one coupler to the end of the other coupler; it does not refer to simply the length of the car body.

The eighth physical constant (i.e., hand brake efficiency ratio) may need a little explanation. The hand brake model for a car (and a locomotive) is a simplified model that applies a retarding force proportional to the weight of the car. This retarding force is assumed to be constant and therefore is independent of the speed of the car. The proportionality factors allowed in TPS range from 0.01 to 0.2 (Table 5). For example, if a proportionality factor of 0.02 is inputted by the user, a car weighing 200.0 kips would experience a retarding force of 200.0 kips multiplied by 0.02, which equals 4.0 kips. Note that if the hand brake status physical constant in Table 5 is set to 0, which indicates the handbrake is released, then the hand brake efficiency ratio entered by the user will be ignored and will be assumed to be equal to 0.0 since the handbrake is released.

With regard to the fourth physical constant (i.e., cross-sectional area) and the fifth physical constant (i.e., streamlining coefficient) listed in Table 5, typical values for these two parameters are given in Table 6 for common car types.

Table 6. Cross-sectional area and streamlining coefficients for common car types

Type of car	Cross-sectional area (ft ²)	Streamlining coefficient
Boxcar	140	4.9
Bulkhead flatcar (loaded)	140	5.3
Bulkhead flatcar (empty)	140	12.0
Coal gondola (loaded)	105	4.2
Coal gondola (empty)	105	12.0
Covered hopper	125	7.1
Multi-level auto transporter (open)	150	12.3
Multi-level auto transporter (closed)	170	7.1
Standard flatcar (without trailers)	25	5.0
Standard flatcar (with trailers)	125	5.0
Tank car	95	5.5

In TPS, a car also consists of two functions. Characteristics of these two functions are given in Table 7.

Table 7. Car functions

Function	Behavior	Independent Variable (IV) Units	IV Minimum Value	IV Maximum Value	Dependent Variable (DV) Units	DV Minimum Value	DV Maximum Value
1. Brake Rigging Efficiency	Piecewise-smooth	PSI	15.0	105.0	Unitless	0.01	1.0
2. Brake Shoe Friction Coefficient	Piecewise-smooth	MPH	0.0	90.0	Unitless	0.01	1.0

In addition to the requirements outlined in Table 7, the first function (i.e., brake rigging efficiency) must have a first data point with an independent variable value equal to 15.0 psi which, rounded to the nearest whole number, is the approximate atmospheric pressure at sea level.³ In addition, this function must have a final data point with an independent variable value greater than or equal to 85.0 psi and less than or equal to 105 psi. Similarly, the second function must have a first data point with an independent variable value equal to 0.0 mph and a final data point with an independent variable value greater than or equal to 70.0 mph and less than or equal to 90 mph.

³ Actual atmospheric pressure at sea level is more precisely 14.7 psi.

The plot in Figure 6 shows the valid function space for the brake rigging efficiency function.

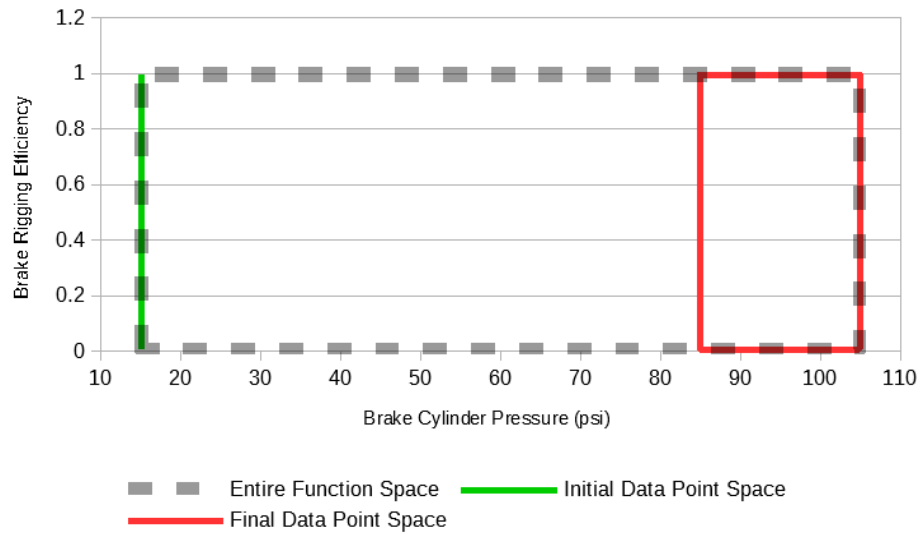


Figure 6. Function space for brake rigging efficiency function

Similarly, the plot in Figure 7 shows the valid function space for the brake shoe friction coefficient function. The independent variable for the brake shoe friction coefficient function is a car's velocity. This is in contrast to the brake rigging efficiency function in Figure 6 which has a car's brake cylinder pressure as its independent variable.

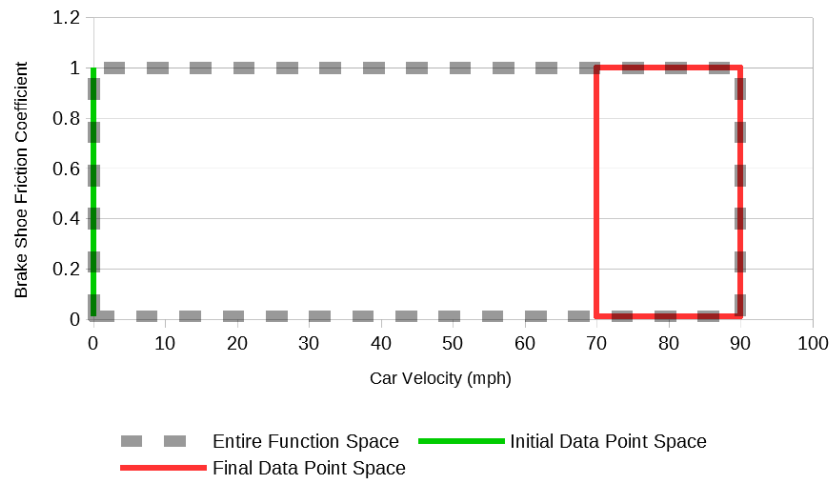


Figure 7. Function space for brake shoe friction coefficient function

Figure 8 shows an example of valid syntax for a car definition portion of the input file. The car definition is shown as beginning on line 56 with the keyword Car_. This is done

for illustration purposes since it is assumed that the definition of this car is being placed in the same input file after the track that was defined in Figure 3 on page 7 and the coupler that was defined in Figure 5 on page 10. It can be seen in Figure 5 that the definition of the coupler ends on line 49. In this example, several blank spaces were left between the end of the definition of the coupler and the beginning of the definition of the car in order to clearly separate the definition of the coupler and the definition of the car, thereby making the input file easier to read and understand.

The text format defining a car is similar to the text format defining a coupler, as described in the previous section. However, the definition of a car begins with the keyword `Car_` (line 56) and ends with the keyword `_Car` (line 84). A car has 11 physical constants, as shown in Table 5, which must be defined first. These physical constants are entered on line 79. Note that commas (,) are used to separate the 11 physical constants. Next the car functions are defined. The car functions must be defined in the order shown in Table 7. This means that the brake rigging efficiency function must be defined first followed by the brake shoe friction coefficient function. Both functions are piecewise-smooth which means they may contain between 2 and 30 points per interval. In the example shown in Figure 8, lines 74-76 define the brake rigging efficiency function. This function has a single interval (line 75). The interval appears on multiple lines if the “word wrap” feature of the text editor is used but this does not indicate that multiple intervals are being defined. Another interval is created by hitting enter and obtaining a new line number. This is demonstrated in the definition of the brake shoe friction coefficient function on lines 79-82. Line 80 contains the first interval which consists of two points, and line 81 contains the second interval which consists of 12 points. Line 84 with the keyword `_Car` indicates the end of the car definition.

```

55 # Loaded gondola (car index 1)
56 Car_
57
58 # PHYSICAL CONSTANTS
59 # 1. mass (kips)
60 # 2. length (feet)
61 # 3. number of axles
62 # 4. cross-sectional area (feet^2)
63 # 5. stream-lining coefficient
64 # 6. maximum net braking ratio
65 # 7. hand brake status
66 # 8. hand brake effectiveness ratio
67 # 9. truck center spacing (feet)
68 # 10. coupler height (feet)
69 # 11. center-of-gravity height (feet)
70 280.0, 60.0, 4, 105.0, 4.2, 0.1, 0, 0.02, 42.0, 2.7, 7.0
71
72 # FUNCTIONS
73 # 1. brake rigging efficiency function
74 Function_
75 15.0,0.46322; 16.0,0.47420; 17.0,0.48391; 18.0,0.49261; 19.0,0.50048; 20.0,0.50766; 21.0,0.51425; 22.0,0.52035;
23.0,0.52600; 24.0,0.53128; 25.0,0.53623; 30.0,0.55710; 35.0,0.57344; 40.0,0.58683; 45.0,0.59814; 50.0,0.60791;
55.0,0.61649; 60.0,0.62414; 65.0,0.63103; 70.0,0.63729; 75.0,0.64303; 80.0,0.64832; 85.0,0.65323; 90.0,0.65780
76 _Function
77
78 # 2. brake shoe friction coefficient function (assuming a brake shoe temperature of 300 degrees F)
79 Function_
80 0.0, 0.5568; 1.0, 0.5568
81 1.0, 0.5568; 3.0, 0.5028; 5.0, 0.4700; 7.0, 0.4479; 10.0, 0.4249; 20.0, 0.3820; 30.0, 0.3585; 40.0, 0.3425; 50.0,
0.3305; 60.0, 0.3210; 70.0, 0.3131; 80.0, 0.3065
82 _Function
83
84 _Car
85

```

Figure 8. Example of valid car portion of input file

A second car is defined in Figure 9. This definition of the second car starts at line 91 since the definition of the first car ended on line 84, as shown in Figure 8, and it is assumed that the second car is being defined in the same file. This second car is being defined for illustration purposes to show that defining more than one car is allowed. Note that defining more than one coupler, more than one locomotive, and more than one locomotive operator is also allowed. The definition of this second car will be used in the train consist definition portion of the input file (see “Train Consist” section on page 24).

```

90 # Unloaded gondola (car index 2)
91 Car_
92
93 # PHYSICAL CONSTANTS
94 # 1. mass (kips)
95 # 2. length (feet)
96 # 3. number of axles
97 # 4. cross-sectional area (feet^2)
98 # 5. stream-lining coefficient
99 # 6. maximum net braking ratio
100 # 7. hand brake status
101 # 8. hand brake effectiveness ratio
102 # 9. truck center spacing (feet)
103 # 10. coupler height (feet)
104 # 11. center-of-gravity height (feet)
105 60.0, 60.0, 4, 105.0, 12.0, 0.1, 0, 0.02, 42.0, 2.7, 3.5
106
107 # FUNCTIONS
108 # 1. brake rigging efficiency function
109 Function_
110 15.0,0.46322; 16.0,0.47420; 17.0,0.48391; 18.0,0.49261; 19.0,0.50048; 20.0,0.50766; 21.0,0.51425; 22.0,0.52035;
23.0,0.52600; 24.0,0.53128; 25.0,0.53623; 30.0,0.55710; 35.0,0.57344; 40.0,0.58683; 45.0,0.59814; 50.0,0.60791;
55.0,0.61649; 60.0,0.62414; 65.0,0.63103; 70.0,0.63729; 75.0,0.64303; 80.0,0.64832; 85.0,0.65323; 90.0,0.65780
111 _Function
112
113 # 2. brake shoe friction coefficient function (assuming a brake shoe temperature of 300 degrees F)
114 Function_
115 0.0, 0.5568; 1.0, 0.5568
116 1.0, 0.5568; 3.0, 0.5028; 5.0, 0.4700; 7.0, 0.4479; 10.0, 0.4249; 20.0, 0.3820; 30.0, 0.3585; 40.0, 0.3425; 50.0,
0.3305; 60.0, 0.3210; 70.0, 0.3131; 80.0, 0.3065
117 _Function
118
119 _Car
120

```

Figure 9. Second example of valid car portion of input file

Locomotive

In TPS, a locomotive has 12 physical constants. The maximum and minimum allowable values of these physical constant parameters are summarized in Table 8. Comparing Table 8 to Table 5, it can be seen that the first eleven locomotive physical constants are equivalent to the eleven car physical constants. The twelfth physical constant “Engine Effectiveness Ratio” can be used to factor in the age of a locomotive; as a locomotive ages, its engine may degrade resulting in a decrease in its tractive effort.

Table 8. Locomotive physical constants

Physical Constant	Units	Minimum Value	Maximum Value
1. Weight	Kips	150.0	600.0
2. Length	Feet	40.0	110.0
3. Number of Axles	Unitless	4	6
4. Cross-Sectional Area	Square Feet	20.0	200.0
5. Streamlining Coefficient	Unitless	1.0	30.0
6. Maximum Net Braking Ratio	Unitless	0.01	0.2
7. Hand Brake Status	Unitless	0 (released)	1 (applied)
8. Hand Brake Efficiency Ratio	Unitless	0.01	0.2
9. Truck center spacing	Feet	50% of locomotive length	95% of locomotive length
10. Coupler height	Feet	1.0	5.0
11. Center-of-gravity height	Feet	1.0	15.0
12. Engine Effectiveness Ratio	Unitless	0.5	1.0

With regard to the fourth physical constant (i.e., cross-sectional area) and the fifth physical constant (i.e., streamlining coefficient) listed in Table 8, typical values for these two parameters for a typical leading and trailing locomotive are given in Table 9.

Table 9. Cross-sectional area and streamlining coefficients for typical leading and trailing locomotives

Type of car	Cross-sectional area (ft ²)	Streamlining coefficient
Freight locomotive (leading)	160	24.0
Freight locomotive (trailing)	160	5.5

In TPS, a locomotive also consists of four functions. Requirements for these four functions are summarized in Table 10. The first two functions in Table 10 are equivalent

to the two functions in Table 7.

Table 10. Locomotive functions

Function	Behavior	Independent Variable (IV) Units	IV Minimum Value	IV Maximum Value	Dependent Variable (DV) Units	DV Minimum Value	DV Maximum Value
1. Brake Rigging Efficiency	Piecewise-smooth	PSI	15.0	105.0	Unitless	0.01	1.0
2. Brake Shoe Friction Coefficient	Piecewise-smooth	MPH	0.0	90.0	Unitless	0.01	1.0
3. Full Throttle	Piecewise-smooth	MPH	0.0	90.0	Kips	0.0	400.0
4. Full Dynamic Braking	Piecewise-smooth	MPH	0.0	90.0	Kips	0.0	400.0

In addition to the requirements outlined in Table 10, the first function (brake rigging efficiency) must have a first data point with an independent variable value equal to 15.0 psi and a final data point with an independent variable value greater than or equal to 85.0 psi and less than or equal to 105.0 psi. This is the same as the case for a car's brake rigging efficiency function as put forth in the "Car" section beginning on page 10. The second function (brake shoe friction coefficient) must have a first data point with an independent variable value equal to 0.0 mph and a final data point with an independent variable value greater than or equal to 70.0 mph and less than or equal to 90 mph. This also is the same as the case for a car's brake rigging efficiency function.

The third function (i.e., full throttle) and fourth function (i.e., full dynamic braking) have the same requirements as the second function (brake shoe friction coefficient), namely both must have a first data point with an independent variable value equal to 0.0 mph and a final data point with an independent variable value greater than or equal to 70.0 mph and less than or equal to 90 mph.

The function space for the brake rigging efficiency was shown previously in Figure 6 on page 13. Similarly, the function space for brake shoe friction coefficient was shown previously in Figure 7 on page 13. The plot in Figure 10 shows the function space for full throttle and full dynamic braking.

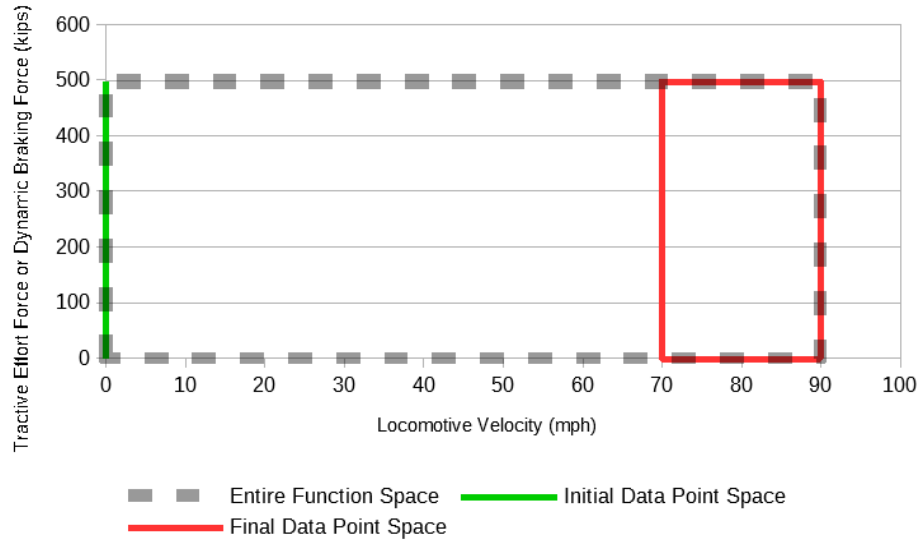


Figure 10. Function space for locomotive full throttle function and locomotive dynamic braking function

Figure 11 is a screen capture which shows an example of valid syntax for a locomotive definition portion of the input file. For illustration purposes, the locomotive definition is shown as beginning at line 126 with the keyword `Locomotive_`. This is due to the fact that, for this example, it is assumed that a track (Figure 3), a coupler (Figure 5), and two cars (Figure 8 and Figure 9) were previously defined in the input file.

The text format defining a locomotive is nearly identical to that of a car, except a locomotive has 12 physical constants (instead of 11 for a car) and four functions (instead of two for a car). Therefore, the syntax in Figure 11 will not be explained line-by-line. One important distinction is that the keyword `Locomotive_` (line 126) and the keyword `_Locomotive` (line 167) indicate the beginning and end of the definition of a locomotive, respectively.

```

125 # 4400 horsepower locomotive (locomotive index 1)
126 Locomotive_
127
128 # PHYSICAL CONSTANTS
129 # 1. mass
130 # 2. length
131 # 3. number of axles
132 # 4. cross-sectional area
133 # 5. stream-lining coefficient
134 # 6. maximum net braking ratio
135 # 7. hand brake status
136 # 8. hand brake effectiveness ratio
137 # 9. truck center spacing (feet)
138 # 10. coupler height (feet)
139 # 11. center-of-gravity height (feet)
140 # 12. engine effectiveness ratio
141 450.0, 75.0, 6, 160.0, 24.0, 0.1, 0, 0.02, 52.5, 2.7, 7.0, 0.95
142
143 # FUNCTIONS
144 # 1. brake rigging efficiency function
145 Function_
146 15.0,0.0,0.0322; 16.0,0.0,0.4749; 17.0,0.0,0.4839; 18.0,0.0,0.4926; 19.0,0.0,0.5048; 20.0,0.0,0.5076; 21.0,0.0,0.5142; 22.0,0.0,0.5203; 23.0,0.0,0.5260; 24.0,0.0,0.5318; 25.0,0.0,0.5362;
147 36.0,0.0,0.5578; 35.0,0.0,0.5734; 40.0,0.0,0.5863; 45.0,0.0,0.5984; 50.0,0.0,0.6079; 55.0,0.0,0.6169; 60.0,0.0,0.6244; 65.0,0.0,0.6310; 70.0,0.0,0.6372; 75.0,0.0,0.6430; 80.0,0.0,0.6492;
148 85.0,0.0,0.6533; 90.0,0.0,0.6578
149 _Function
150
151 # 2. brake shoe friction coefficient function (assuming a brake shoe temperature of 300 degrees F)
152 Function_
153 0.0, 0.5568; 1.0, 0.5568
154 1.0, 0.5568; 3.0, 0.5828; 5.0, 0.4780; 7.0, 0.4479; 10.0, 0.4249; 20.0, 0.3820; 30.0, 0.3585; 40.0, 0.3425; 50.0, 0.3305; 60.0, 0.3210; 70.0, 0.3131; 80.0, 0.3065
155 _Function
156
157 # 3. full throttle function
158 Function_
159 0.0,100.0; 7.6,100.0
160 7.6,100.0; 8.0,171.19; 10.0,136.95; 12.0,114.13; 15.0,91.3; 20.0,68.48; 25.0,54.78; 30.0,45.65; 40.0,34.24; 50.0,27.39; 60.0,22.83; 70.0,19.56; 80.0,17.12
161 _Function
162
163 # 4. full dynamic braking function
164 Function_
165 0.0,100.0; 12.6,100.0
166 12.6,100.0; 15.0,91.3; 20.0,68.48; 25.0,54.78; 30.0,45.65; 40.0,34.24; 50.0,27.39; 60.0,22.83; 70.0,19.56; 80.0,17.12
167 _Function
168 _Locomotive
169

```

Figure 11. Example of valid locomotive portion of input file

Locomotive Operator

In TPS, a locomotive operator has one physical constant.⁴ This physical constant takes on a value of 0 or 1, as shown in Table 11. A value of 0 indicates that the locomotive operator commands are based on distance (specifically, the location of the first rail vehicle in the train consist), and a value of 1 indicates that the locomotive operator commands are based on time (specifically, the elapsed time since start of simulation).

Table 11. Locomotive operator physical constants

Physical Constant	Units	Minimum Value	Maximum Value
1. Distance-Based Versus Time-Based Indicator	Unitless	0 (for distance-based)	1 (for time-based)

In TPS, a locomotive operator has four functions. The independent variable for these functions depend on the value of the physical constant entered; if a value of 0 is entered, then the independent variable is distance (in feet), and if a value of 1 is entered, then the independent variable is time (in seconds). Requirements for these four functions are given in Table 12.

4 Note that in this case the “physical constant” label perhaps is not an accurate label but is used in order to provide consistency with the “physical constant” term used in the “Car” section beginning on page 10 and the “Locomotive” section beginning on page 16.

Table 12. Locomotive operator functions

Function	Behavior	Independent Variable (IV) Units	IV Minimum Value	IV Maximum Value	Dependent Variable (DV) Units	DV Minimum Value	DV Maximum Value
1. Automatic Air Brake Setting	Step	Feet or Seconds	0.0	Track Length (feet) or 10800.0 (seconds)	PSI	15 (read paragraphs below for important details)	105
2. Locomotive Independent Brake Setting	Step	Feet or Seconds	0.0	Track Length (feet) or 10800.0 (seconds)	PSI	15 (read paragraphs below for important details)	105
3. Throttle Ratio	Piecewise-linear	Unitless	0.0	Track Length (feet) or 10800.0 (seconds)	Unitless	0.0	1.0
4. Dynamic Brake Ratio	Piecewise-linear	Unitless	0.0	Track Length (feet) or 10800.0 (seconds)	Unitless	0.0	1.0

In addition to the requirements shown in Table 12, if the physical constant put forth in Table 11 is equal to 0, the independent value of the last point of the last interval of each of the four functions must be equal to the track length. However, if the physical constant is equal to 1, the independent value of the last point of the last interval of each of the four functions must be equal to the maximum allowable simulation time of 10,800 seconds (which is equivalent to 180 minutes).

Values of 15 psi and 105 psi are shown in Table 12 as being the minimum and maximum dependent values for the first two functions (automatic air brake setting and locomotive independent brake setting). These numbers may not be familiar to those in the railroad industry that have experience with railroad brake systems. This is due to the fact that in TPS units of psi are used whereas in the railroad industry in the United States units of psig are standard. The typical brake pipe operating pressure in freight operations in the United States is 90 psig which corresponds to 105 psi. Similarly, atmospheric pressure is equal to 0 psig which corresponds to approximately 15 psi.⁵

Furthermore, in TPS the assumption is made that the automatic air brake setting applies only to cars in the consist. Therefore, in order to apply brakes to a locomotive, the

⁵ Actual atmospheric pressure at sea level is more precisely 14.7 psi.

locomotive independent brake setting must be used. This is different from real-world operations in which an application of the automatic air brakes results in braking of both cars and the locomotive. In real-world operations, a locomotive operator can “bail” on the automatic air brake for a locomotive, which means that the cars in the train keep braking, but the locomotive does not. Rather than having a “bailing” function, in TPS the locomotive independent air brake function is used to apply and release the friction brakes for a locomotive.

A more detailed explanation of the allowable (and advisable) values for the automatic air brake setting function in Table 12 will now be given. The dependent variable values of this function can be set to any integer value from 105 psi to 15 psi. However, in real-world operations at least a 5 psi reduction is needed to initiate a minimal service brake. In addition, at a value of approximately 79 psi (a 26 psi reduction), the brake cylinder and the auxiliary reservoir are in equilibrium (in other words, they have the same air pressure), which indicates a “full-service” brake application. Therefore, the values for service braking would be from 100 psi to 79 psi (or from 85 psig to 64 psig). These values are summarized in Table 13.

Table 13. Advisable values for dependent variable of automatic air brake function and locomotive independent brake setting

Dependent Variable Value	Meaning
15	Emergency brake application
Integers from 79 to 100	Brake application (with 79 indicating a full service brake application)
105	Brake release

In TPS, gradual release of the automatic air brake setting is allowed. For example, if the automatic air brake setting is at 79 psi, the user may reduce the braking application by raising the automatic air brake setting to 85 psi. Such gradual release operations are typically only possible in passenger operations since the car control valves on passenger cars are designed to handle such a gradual release application. However, performing such gradual release operations in TPS is not advised as TPS is largely intended to simulate freight train operations. As such, TPS uses a simplified control valve (see “Car Control Valve” section beginning on page 80) that is meant to mimic a basic freight car control valve. These standard freight car valves are not designed to handle a gradual release of the automatic air brake setting. Rather than performing a gradual release, in freight typical operations, if the automatic brake valve setting is at 79 psi, for example, but less braking is desired, then the automatic air brake setting must be fully released to a value of 105 psi (which is equivalent to 90 psig) and then re-applied to a desired value of

85 psi, for example.

As shown in Table 13, assigning a value of 15 psi (which is approximately the atmospheric pressure at sea level) to a locomotive operator's automatic air brake function initiates emergency braking. This means that if the last or first rail vehicle in the train consist is a car (rather than a locomotive) and if the end-of-train device (EOT) is a two-way device (see "Train Consist" section beginning on page 24 for assigning a two-way or one-way EOT device to a train consist), then the end-of-train (EOT) device will open and release air to atmosphere. However, this venting of the EOT device to atmosphere will only occur if the locomotive operator assigned to the nearest locomotive to the last (or first) car initiates emergency braking. For details on how to assign a locomotive operator to a locomotive, see the section titled "Train Consist" on page 24.

Figure 12 shows an example train consist. Note that the term "Loc" in this figure refers to a locomotive. This train consist which contains only seven rail vehicles is not realistic (since most train consists typically contain 100 rail vehicles or more) and is meant only for illustration purposes. In this train consist, if the last car in the consist (Car 7) has a two-way EOT, then it will vent to atmosphere only when the locomotive operator assigned to locomotive Loc 5 initiates emergency braking; in other words, when the locomotive operator model assigned to locomotive Loc 5 has its automatic air brake function assigned a value of 15 psi.



Figure 12. First example train consist to demonstrate emergency braking initiation and two-way EOT venting to atmosphere

Figure 13 shows a similar train consist to the one shown in Figure 12. However, in Figure 13, the leading locomotive is removed. In this case, a car (not a locomotive) is the leading rail vehicle in the train consist. This is an unlikely scenario, especially in a freight train consist. However, in such a scenario, if there was a two-way EOT on Car 1 (see section "Train Consist" beginning on page 24 to see how to assign a two-way EOT device to the first and last cars of a train consist), then this EOT would vent to atmosphere only when the locomotive operator assigned to locomotive Loc 3 in Figure 13 initiates emergency braking; in other words, when the locomotive operator model assigned to locomotive Loc 3 has its automatic air brake function assigned a value of 15 psi.



Figure 13. Second example train consist to demonstrate emergency braking initiation and two-way EOT venting to atmosphere

Figure 14 is a screen capture showing an example of valid syntax for a locomotive operator definition portion of the input file. Note that if a locomotive is defined in a TPS input file, then at least one locomotive operator must be defined. For illustration purposes, the locomotive operator definition is shown as beginning at line 174 with the keyword `LocomotiveOperator_`. For illustration purposes, it is assumed that a track (Figure 3), a coupler (Figure 5), two cars (Figure 8 and Figure 9), and a locomotive (Figure 11) were previously defined in the same input file and that the lines defining the locomotive operator in Figure 14 are placed after the lines defining the locomotive in Figure 11 on page 19.

```

173 # Locomotive operator (locomotive operator index 1)
174 LocomotiveOperator_
175
176 # PHYSICAL CONSTANTS
177 # '0' for distance-based; '1' for time-based
178 0
179
180 # FUNCTIONS
181 # 1. automatic air brake
182 Function_
183 0.0, 105; 42240.0, 105
184 42240.0, 79; 79200.0, 79
185 _Function
186
187 # 2. independent air brake
188 Function_
189 0.0, 105; 42240.0, 105
190 42240.0, 79; 79200.0, 79
191 _Function
192
193 # 3. throttle
194 Function_
195 0.0, 0.5; 2640.0, 0.5
196 2640.0, 0.5; 3640.0, 0.75
197 3640.0, 0.75; 42240.0, 0.75
198 42240.0, 0.75; 42540.0, 0.0
199 42540.0, 0.0; 79200.0, 0.0
200 _Function
201
202 # 4. dynamic brake
203 Function_
204 0.0, 0.0; 79200.0, 0.0
205 _Function
206
207 _LocomotiveOperator
208

```

Figure 14. Example of valid locomotive operator portion of input file

The text format defining a locomotive operator is similar to that of the other components already defined (track, coupler, car, and locomotive). Therefore, the syntax in Figure 14 will not be explained line-by-line. However, there are a couple points that should be mentioned. First, the keyword `LocomotiveOperator_` on line 174 indicates the beginning of the definition of a locomotive operator, and the keyword `_LocomotiveOperator` on line 207 indicates the end of the definition of a locomotive operator.

Another important point is that the first two functions (automatic air brake setting and independent air brake setting) are step functions. This requires that each interval consist of exactly two points and that the dependent variable value of both points be equal. As can be seen in Figure 14, this requirement is satisfied. Furthermore, it should be noted that the dependent variable values of the automatic air brake and independent air brake functions must be integers. However, on line 192, for example, if `105.0` was entered instead of `105`, TPS treats that as being valid syntax even though `105.0` technically is not an integer.

A final important point is to explain the throttle and dynamic brake functions. Typically, a locomotive has eight discrete throttle notches. In TPS, only one notch is defined (see “Full Throttle” function in Table 10 on page 17), and this notch represents maximum throttle. The other throttle notches are simulated by setting the locomotive operator throttle ratio function (third function in Table 12) to the appropriate value. For example, to simulate throttle notch 4, a value of 0.5 can be entered, as shown on line 195 in Figure 14. Throttle notch 4 corresponds to a TPS input value of 0.5 for the locomotive operator throttle ratio function because 4 divided by 8 (which is typically the maximum number of throttle notches) is equal to 0.5. Similarly, 0.75 on line 197 in Figure 14 represents throttle notch 6 since 6 divided by 8 is equal to 0.75.

Train Consist

In TPS, a train consist definition has two physical constants. The maximum and minimum allowable values of these two physical constant parameters are summarized in Table 14.

Table 14. Train consist physical constants

Physical Constant	Units	Minimum Value	Maximum Value
1. Air Temperature	Fahrenheit	-40.0	140.0
2. One-Way Or Two-Way End-Of-Train (EOT) Device	Unitless	1 (for one-way EOT)	2 (for two-way EOT)

The second physical constant in Table 14 may need some explanation for those unfamiliar with end-of-train (EOT) devices. A two-way EOT device is capable of receiving a signal from the locomotive when emergency braking is initiated by the locomotive operator. The two-way EOT device will subsequently open an orifice and vent the brake pipe to atmosphere. However, a one-way EOT device is not capable of performing this function of venting the brake pipe to atmosphere. Therefore, if a locomotive operator model performs emergency braking by having its automatic air brake function set to a value of 15 psi (see “Locomotive Operator” section on page 19) and if there is a corresponding EOT device (in other words, if there is a car rather than a locomotive at the end of the train), then the EOT device will only vent to atmosphere if the train consist’s second physical constant is equal to 2, which indicates that the train consist is equipped with a two-way EOT device, as shown in Table 14.

In TPS, unlike the definitions of a track, coupler, car, locomotive, and locomotive operator, the definition of a train consist does not have any functions. Rather, the second portion of a train consist definition (after the physical constants) consists of defining the order of the rail vehicles in the train consist as well as several parameters associated with each rail vehicle. This is best illustrated by going through an example train consist definition, as shown in Figure 15. The keyword `TrainConsist_` on line 214 indicates the start of the train consist definition. In this example, the definition of the train consist starts on line 214 because it is assumed that a track (Figure 3), a coupler (Figure 5), two cars (Figure 8 and Figure 9), a locomotive (Figure 11), and a locomotive operator (Figure 14) were all defined in the same input file before the train consist. As can be seen in Figure 14 on page 23, the definition of the locomotive operator ended on line 207.

The physical constants shown in Table 14 are defined on line 219 in Figure 15. The rail vehicles in the train consist are defined on lines 229-258. Each line represents a rail vehicle, so in this example, there are 30 rail vehicles in the train consist. A train consist may consist of up to 300 rail vehicles. Note that there is a comment symbol `#` contained in each of the lines 229-258. All input after the `#` symbol on each line is treated as a comment and is ignored by TPS. In other words, TPS allows inline comments, as was explained previously in the “Track” section beginning on page 3. In Figure 15, the comments at the end of lines 229-258 are simply there to make it easier to see the position of each rail vehicle in the train consist.

```

213 # Train consist with one leading locomotive and loaded gondolas followed by trailing, unloaded gondolas
214 TrainConsist_
215
216 # PHYSICAL CONSTANTS
217 # 1. air temperature
218 # 2. end-of-train device capability
219 75.0, 2
220
221 # RAIL VEHICLES
222 # 1. rail vehicle type indicator ('C' for car or 'L' for locomotive)
223 # 2. rail vehicle index
224 # 3. coupler index
225 # 4. rail vehicle velocity
226 # 5. brake pipe pressure (for car) or locomotive operator index (for locomotive)
227 # 6. auxiliary reservoir pressure (for car)
228 # 7. emergency reservoir pressure (for car)
229 L, 1, 1, 50.0, 1 # rail vehicle index 1
230 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 2
231 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 3
232 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 4
233 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 5
234 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 6
235 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 7
236 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 8
237 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 9
238 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 10
239 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 11
240 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 12
241 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 13
242 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 14
243 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 15
244 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 16
245 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 17
246 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 18
247 C, 1, 1, 50.0, 105, 105, 105 # rail vehicle index 19
248 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 20
249 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 21
250 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 22
251 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 23
252 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 24
253 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 25
254 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 26
255 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 27
256 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 28
257 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 29
258 C, 2, 1, 50.0, 105, 105, 105 # rail vehicle index 30
259
260 _TrainConsist
261

```

Figure 15. Example of valid train consist portion of input file

Table 15 summarizes the parameters of a rail vehicle line in a train consist, such as those in lines 229-258 in Figure 15.

Table 15. Parameters of rail vehicle in train consist definition

Parameter	Units	Minimum Value	Maximum Value
1. L for locomotive or C for car	Unitless	N/A	N/A
2. Locomotive or car index	Unitless	1	Maximum number of defined locomotives or cars (depending on whether the first parameter was an L or a C)
3. Coupler index	Unitless	1	Maximum number of defined couplers
4. Rail vehicle velocity	MPH	0.0	90.0
5. Locomotive operator (for locomotive)	Unitless	1	Number of defined locomotive operators
5. Brake pipe pressure (for car)	PSI	15	105
6. Auxiliary reservoir pressure (for car)	PSI	15	105
7. Emergency reservoir pressure (for car)	PSI	15	105

The first parameter of each line consists of an L for a locomotive or a C for a car. Therefore, it is clear in the example in Figure 15 that the first rail vehicle is a locomotive, and the trailing 29 rail vehicles are cars. The second parameter represents the locomotive index or car index. The only locomotive (line 229) has locomotive index 1. This means that only one locomotive had to be previously defined in the input file. The definition of this locomotive is shown in Figure 11 on page 19. The first eighteen cars (lines 230-247) have a car index of 1 which means that they are copies of the first car defined previously in the input file, as shown in Figure 8 on page 15. Similarly, the last eleven cars (lines 248-258) have a car index of 2 which means that they are copies of the second car defined previously in the input file, as shown in Figure 9 on page 15. Therefore, two cars must have been previously defined in the input file, or an error will be generated.

The third parameter is the coupler index. As shown in Figure 15, all the rail vehicles have a coupler index of 1. This means that only one coupler must have been previously defined in the input file. In this example, the coupler definition can be found in Figure 5 on page 10.

The fourth parameter is the initial rail vehicle velocity. It is advisable to enter equal (or nearly equal) velocities for all rail vehicles in the train consist.

The first four parameters represent the same type of physical initial condition for both a

car and locomotive. However, the physical parameter represented by the fifth parameter depends on the rail vehicle type. If the rail vehicle is a locomotive, then fifth parameter represents the locomotive operator index. This represents the index of the locomotive operator that is being assigned to the locomotive. In this example, only one locomotive operator was defined. The definition of this locomotive operator is in Figure 14 on page 23. Therefore, 1 is the only valid locomotive operator index that can be assigned to the locomotive on line 229 of the train consist definition in Figure 15.

However, if the rail vehicle is a car, then the fifth parameter represents the car's brake pipe pressure. Note that typical brake pipe operating pressure is 90 psig in freight operations. This value of 90 psig corresponds to 105 psi. In TPS, brake pipe pressure inputs are entered in units of psi. Therefore, a fully charged brake pipe is represented by 105 psi. It is advisable to enter equal (or nearly equal) brake pipe pressures for all cars in the train consist. In addition, it is advisable to have these initial car brake pipe pressures be equal to (or nearly equal to) the initial automatic brake valve settings for any locomotives (and respective locomotive operators) used in the train consist.

The sixth and seventh parameters apply only to a car. The sixth parameter represents the initial auxiliary reservoir pressure of a car, and the seventh parameter represents the initial emergency reservoir pressure of a car. A fully charged auxiliary reservoir and emergency reservoir each have a pressure of 105 psi.

Simulation Parameters

In TPS, the final portion of an input file consists of defining three simulation parameters. The first simulation parameter specifies an integration method for modeling the dynamics of the rail vehicles in a train consist. For this first parameter, 0 represents a fixed time step Runge-Kutta integration method, and 1 represents an adaptive time step Runge-Kutta-Fehlberg integration method. Both of these are explicit (rather than implicit) integration methods. The Runge-Kutta integration method with a fixed time step results in faster simulation times, but the Runge-Kutta-Fehlberg integration method results in more accurate results. For the fixed time step Runge-Kutta integration method, a time interval of 0.004 seconds is used which is equivalent to 250 hertz. Note that this integration method specified by the user (either a 0 for the fixed time step Runge-Kutta method or a 1 for the adaptive time step Runge-Kutta-Fehlberg method) refers only to the integration method used to model rail vehicle dynamics. For the modeling of the fluid dynamics of the brake pipe, a finite element implicit integration method is used, and that integration method has a fixed time step of 0.02 seconds, which is equivalent to 50 hertz. This finite element implicit integration method is described in detail in the "Brake Pipe

Model” section beginning on page 91.

The second simulation parameter is an integer that represents sampling frequency, which must be greater than or equal to 5 hertz and less than or equal to 1000 hertz. For example, if the sampling frequency is set to 20 hertz, then there will be approximately 20 samples (rows of data) saved per simulated second.

A couple points should be made regarding the sampling frequency. If a sampling frequency of 1000 hertz (which equates to a 0.001 second time step), for example, is specified in the input file and the Runge-Kutta fixed time step integration method (which has an integration time step of 0.004 seconds) is also specified, then effectively a sampling rate of 250 hertz (equivalent to a 0.004 second time step), rather than 1000 hertz, will result.

Furthermore, due to rounding errors, the sampling frequency specified by the user is meant to be approximate. The screen capture in Figure 16 shows some time values output by TPS when a sampling frequency of 50 hertz was specified in a TPS input file. As can be seen, the difference between most adjacent time values is 0.02 seconds, which equates to 50 hertz. However, between rows 13 and 14, there is a time gap of 0.024 seconds.

	A
1	Time (s)
2	0.02
3	0.04
4	0.06
5	0.08
6	0.1
7	0.12
8	0.14
9	0.16
10	0.18
11	0.2
12	0.22
13	0.24
14	0.264
15	0.284
16	0.304
17	0.324
18	0.344
19	0.364
20	0.384
21	0.404

Figure 16. Example time values resulting from a 50 hertz sampling frequency

The third and final parameter is a list of between one and twenty integers that represent the position in the train consist of the rail vehicles that the user wishes to save. TPS simulation results consist of a large number of data associated with each rail vehicle.

Saving all this data for every rail vehicle would result in a large amount of required storage space since many train consists consist of 150 rail vehicles or more. To avoid this requirement, TPS does not store data for every rail vehicle; rather, it allows the user to choose up to twenty rail vehicles in the train consist to save.

An example simulation parameters portion of an input file is shown in Figure 17. The simulation parameters portion of the input file starts on line 267 as, for illustration purposes, it is assumed that this portion of the input file is being placed after the train consist definition which ended on line 260, as shown in Figure 15. The keyword `Simulation_` on line 267 and the keyword `_Simulation` on line 278 indicate the start and end, respectively, of the simulation parameters portion of the input file. Line 270 specifies the integration method. Since line 270 has a 1, this indicates that the selected integration method is the Runge-Kutta-Fehlberg adaptive time step integration method. Line 273 specifies the sampling rate. In this example, a sampling rate of 50 hertz is specified. Line 276 indicates the rail vehicles to be saved. Since the train consist shown in Figure 15 on page 26 has 30 rail vehicles, the valid values to enter on this line are 1 through 30. Line 276 indicates that the results for the first, second, fifteenth, and thirtieth rail vehicles in the train consist should be saved. In other words, the results for the single locomotive (line 229 in Figure 15) should be saved; the results for the first car adjacent to the locomotive (line 230 in Figure 15) should be saved; the results for the car at the fifteenth position in the train consist (line 243 in Figure 15) should be saved; and the results for the last car in the train consist (line 258 in Figure 15) should be saved.

```
266 # Simulation parameters
267 Simulation_
268
269 # Integration method ('0' for fixed time step; '1' for variable time step)
270 1
271
272 # Sampling rate
273 50
274
275 # Rail vehicle indices to be saved
276 1, 2, 15, 30
277
278 _Simulation
279
```

Figure 17. Example of valid simulation parameters portion of input file

Forced Speed File

The user may wish to run a simulation based partially on event recorder data obtained from a locomotive. In such cases, the speed of the locomotive is known. In addition, other settings, such as throttle notch and automatic brake valve settings are known. However, due to the fact that track characteristics, locomotive characteristics, and train

consist characteristics are never known with full precision, there may be differences between simulated speeds and event recorder speeds. This difference may become large when several miles of train operations are simulated. As a result, the user may wish to “force” or reset the speed of the entire train consist at certain locations or at certain times in order to avoid the simulated train speeds from diverging too far from the event recorder train speeds. The forced speed file must be named `forced_speed.fsf`.⁶

Unlike the TPS input file described in the previous sections, no blank lines are allowed in the forced speed file, and no lines with only comments are allowed. However, inline comments, such as those shown in lines 229-258 in Figure 15 on page 26 are allowed.

The first line of the file must contain either a 0 if the forced speeds are based on distance or a 1 if the forced speeds are based on time.

Each line of the text file after the first line must contain two numbers separated by a comma. If the first line of the file contains a 0, then the first number in each of the subsequent lines represents the longitudinal position in feet from the starting point of the simulation. However, if the first line of the file contains a 1, then the first number in each of the subsequent lines represents the elapsed time in seconds from the starting time of the simulation. The second number represents the desired speed (for example, the known event recorder speed) in miles per hour at that location (if the first line contains a 0) or at that time (if the first line contains a 1).

The first number of each line must be greater than zero and less than the length of the respective track model (if there is a 0 in the first line of the file) or less than the maximum simulation time of 10,800 seconds (if there is a 1 in the first line of the file). In addition, the first number of each line must be in increasing order. In other words, the value of the first number on the third line must be greater than the value of the first number on the second line, and the value of the first number on the fourth line must be greater than the value of the first number on the third line, etc.

With regards to the second number of each line, valid values are considered to be from 0.0 miles per hour to 90.0 miles per hour. Obviously, the second number of each line does not have to be in increasing order as the speed will increase and decrease in a typical train handling operation.

Once the file is created, it must be saved and placed in the same directory as the user’s main TPS input file. Figure 18 shows a screenshot of an example forced speed file. As stated previously, there must be no blank lines. So the example shown below does not

⁶ The “fsf” extension stands for “forced speed file”.

have a sixth line; in other words, Enter is not pressed at the end of the fifth line. In addition, no lines with only comments are allowed, but inline comments are allowed. This is demonstrated on line 1 which has an inline comment.

```
1 0 # '0' indicates that the first number in the following lines represents position (in feet) on track
2 7920.0, 27.0
3 18480.0, 32.0
4 23760.0, 36.0
5 44880.0, 34.0
```

Figure 18. Example of valid syntax for forced speed file

A forced speed file is not required to run a simulation. When a simulation is initiated, TPS looks for a forced speed file in the same directory as the input file and will run the simulation without forcing (i.e., resetting) the speed at any point if a forced speed file does not exist.

Important TPS Quirks and Limitations

Several important points are discussed in the numbered list below regarding train handling quirks or oddities in TPS.

1. Many of TPS's quirks are related to the locomotive operator, and therefore, the "Locomotive Operator" section on page 19 should be reviewed carefully.
2. TPS does not simulate the details of the locomotive automatic brake valve. Mathematical models of the locomotive automatic brake valve (commonly referred to as the 26C automatic brake valve) are put forth in Abdol-Hamid (1986) and Specchia (2012). TPS takes the simplified approach of using the automatic air brake setting (function 1 in Table 12 on page 20) as the air pressure boundary condition for the differential equation used to model the brake pipe. Therefore, the assumption is made that the compressor and main reservoir present on a locomotive have the capacity to produce the automatic brake valve pressure setting input by the user. In TPS, it is assumed that the automatic brake valve setting is reduced at a rate of 2 psi per second for a service brake application. In other words, if the automatic brake valve setting is changed from 105 psi to 85 psi, that amounts to a 20 psi reduction. It would take 10 seconds (since 20 psi divided by 2 psi per second equals 10 seconds) for the boundary condition to change from 105 psi to 85 psi. However, for an emergency brake application, the automatic brake valve setting is reduced at a faster rate of 20 psi per second.
3. TPS does not currently model specific car control valves, such as the ABD or ABDW control valves which are common in the freight railroad industry. Rather,

- TPS includes a generic car control valve that has the very basic operations intrinsic to all car control valves used in the railroad industry. Details of this generic car control valve are put forth in the “Car Control Valve” section on page 80.
4. TPS does not currently simulate retainers. Retainers are devices that allow gradual release of air from the brake cylinder. Retainers are commonly used when a train is traversing a downhill slope to ensure the air brakes are not released too quickly.
 5. Typically, cars have an “empty load device”. This device acts as a binary on/off switch to indicate if the car is empty or loaded depending on whether the car’s weight exceeds a fixed threshold, such as 100 kips. The brake rigging leverage ratio is adjusted based on whether the on/off weight threshold has been exceeded. For example, if the car’s weight is below the threshold, then the brake rigging leverage ratio may be 4.0, but if the car’s weight exceeds the threshold, then the brake rigging leverage ratio is a higher value, such as 8.0. This binary empty load device functionality is not built into TPS. Instead, TPS calculates an exact brake rigging leverage ratio based on the weight of the car and the desired net braking ratio. Therefore, the empty load device in TPS can be thought of as having a continuously variable brake rigging leverage ratio. Calculation of this brake rigging leverage ratio is detailed in the section “Brake Rigging Leverage Ratio” on page 90.
 6. TPS does not model the temperature of a car’s brake shoes. The brake shoe friction coefficient is partially dependent on the brake shoe temperature, as shown in the “Normal Force Between Brake Shoes and Wheels” section beginning on page 85. Potential methods for modeling the temperature of a car’s brake shoes are presented in Appendix C.
 7. TPS does not model wheel slippage. It is assumed that gross wheel slippage does not occur. In other words, the condition of the rail, whether it is wet or sanded, is not currently modeled in TPS.
 8. TPS currently allows for a single force-deflection curve in the definition of a coupler, as discussed in the “Coupler” section beginning on page 9. However, typical friction couplers have a loading force-deflection curve and unloading force-deflection curve, as discussed in the “Coupler Model” section beginning on page 50.

2. Run a TPS Simulation and View Results

Open TPS

TPS can typically be started by double-clicking on the TPS executable file. If this does not work, then TPS can be started by using the command prompt. On a computer running a Windows operating system, the user must open the command prompt and then change the current directory to the location where the TPS executable file is located. In this example, it will be assumed that the TPS executable file is located in the directory C:\Users\Leith\Desktop. The current directory can be set using the `cd` command as shown in Figure 19.

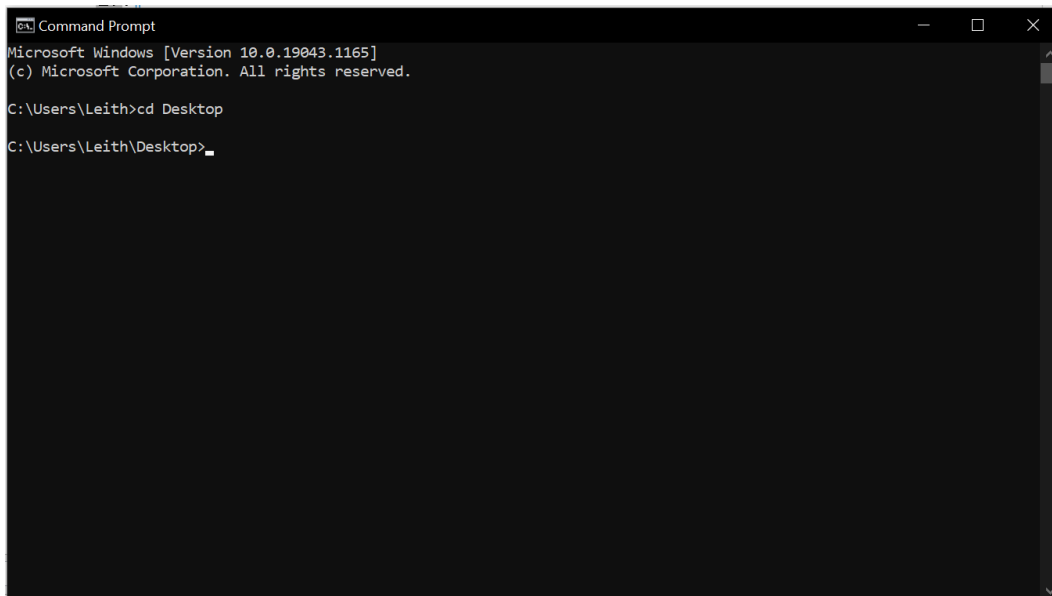
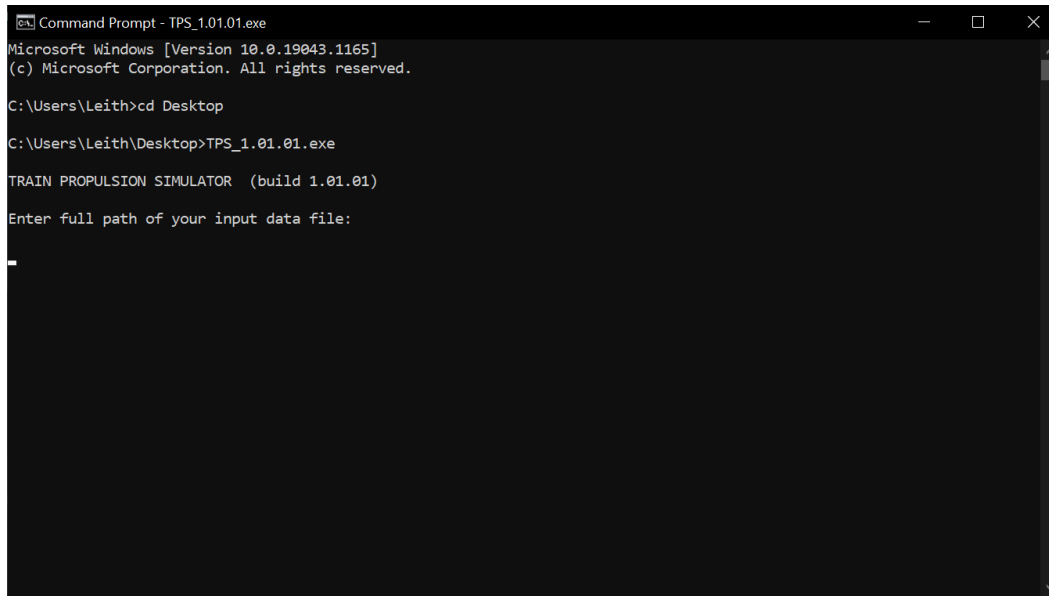


Figure 19. Change current directory in command prompt

Once the current directory is properly set, TPS can be run by simply typing the name of the executable file and pressing Enter, as shown in Figure 20. Make sure the executable file extension `.exe` is included at the end of the file name. In this example, the TPS executable file is located in the C:\Users\Leith\Desktop directory, and the name of the TPS executable file is `TPS_1.01.01.exe`.



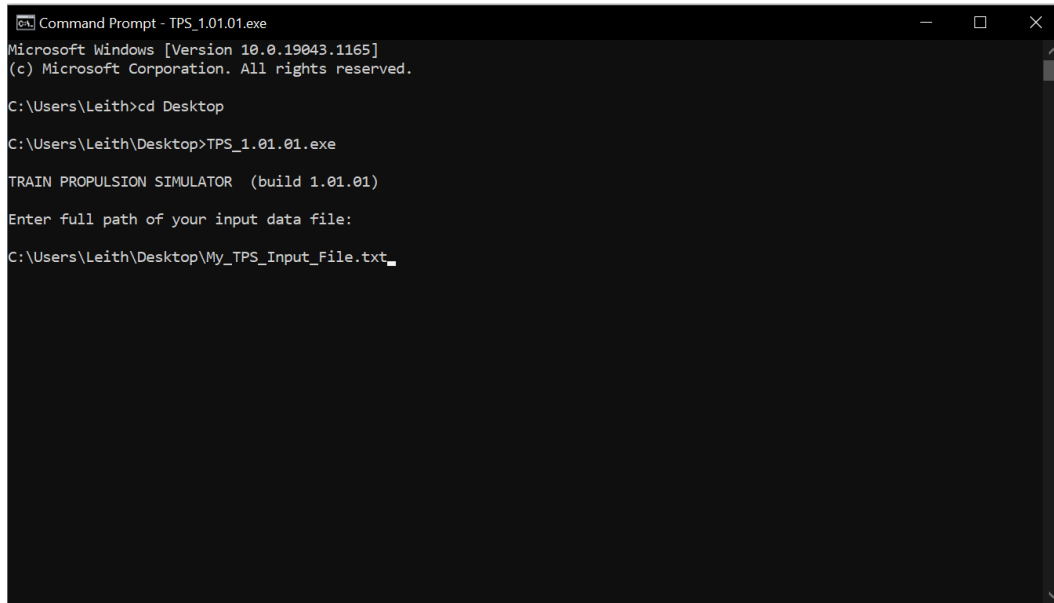
```
Command Prompt - TPS_1.01.01.exe
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Leith>cd Desktop
C:\Users\Leith\Desktop>TPS_1.01.01.exe
TRAIN PROPULSION SIMULATOR (build 1.01.01)
Enter full path of your input data file:
_
```

Figure 20. Run TPS file in command prompt

Run Simulation

In order to run a simulation, the full file path of the user's input file must be entered. For example, in the current case, it will be assumed the input file is in the same directory as the TPS executable file and the input file is titled `My_TPS_Input_File.txt`. Therefore, `C:\Users\Leith\Desktop\My_TPS_Input_File.txt` is typed, as shown in Figure 21. The format of the input file is detailed in Chapter 1. Next, press Enter. If there are no errors in the input file, then the simulation will begin. However, if an error exists in the input file, then the user will be provided with a prompt that tries to provide feedback on where the error is located in the file.



```
Command Prompt - TPS_1.01.01.exe
Microsoft Windows [Version 10.0.19043.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Leith>cd Desktop
C:\Users\Leith\Desktop>TPS_1.01.01.exe
TRAIN PROPULSION SIMULATOR (build 1.01.01)
Enter full path of your input data file:
C:\Users\Leith\Desktop\My_TPS_Input_File.txt_
```

Figure 21. Enter user input file path

Note that a simulation may take a significant amount of time (potentially several hours) to complete depending on the user's computer processing power, the length of the train consist, and the number of seconds that are simulated.

The simulation ends when one of the following conditions occurs:

1. 10,800 simulated seconds (i.e., 180 simulated minutes) elapses
2. the first car of the train consist approaches the end of the track segment or the last car in the train consist approaches the beginning of the track segment
3. the first car in the train consist has a velocity of 1 miles per hour or less for 1800 simulated seconds (i.e., 30 simulated minutes)
4. the first car in the train consist has a velocity that exceeds 150 miles per hour
5. the integration time step is too small
6. one or more of the couplers in the train consist has excessive displacement

The first three bullet points in list above are self-explanatory. Regarding the fourth bullet point, 150 miles per hour might be considered a low speed at which to terminate a simulation. However, TPS is largely geared towards freight train simulations, and 150 miles per hour is an excessive speed for freight operations. For example, one way that TPS is geared towards freight train simulations is that TPS assumes a single brake pipe that acts as a brake signal as well as a recharging line, as is typical in freight operations in

the United States. Passenger trains typically have two brake pipes; one that acts as a braking signal and the other that constantly recharges the auxiliary and emergency reservoirs on each car.

Regarding the fifth bullet point, if the user selects the Runge-Kutta-Fehlberg adaptive time step algorithm (see “Simulation Parameters” section beginning on page 28), then there is the potential that the time step could get so small that it would take an inordinate amount of time for the simulation to complete. TPS sets a minimum time step threshold in order to avoid this situation. If the adaptive time step is smaller than this value, then the simulation terminates.

Regarding the sixth bullet point, if any of the couplers in the train consist have excessive displacement (i.e., excessive tension or compression), then the simulation will terminate.

Format of Result Files

TPS results are outputted to multiple comma-separated value (CSV) files. These CSV files are saved in the same folder as the user’s input file. Five CSV files are automatically created that contain brake pipe pressures, auxiliary reservoir pressures, emergency reservoir pressures, trailing coupler forces, and trailing coupler displacements for the entire train consist. In addition to these five files, there will be up to 20 additional files depending on how many individual rail vehicles the user has chosen to save in the simulation parameters section of the input file (for example, see line 276 in Figure 17).

An example of the list of files outputted by running the portions of the input file detailed in the previous sections and ending with the simulation parameters put forth in Figure 17 on page 30 is given in Table 16. It can be seen that the five standard files related to brake pipe pressures, auxiliary reservoir pressures, emergency reservoir pressures, coupler forces, and coupler displacements have been created. In addition, the first, second, fifteenth, and thirtieth rail vehicles have been saved. The IFN portion of the file names in Table 16 stand for “input file name”. So if the name of the input file was `MyFirstSimulation`, then the first file would actually be named `MyFirstSimulation_1_locomotive` and so on for the rest of the files. Notice that the number portion added to the input file name represents the point in the train consist of the rail vehicle, and the last portion (after the second underscore) represents the type of the rail vehicle (locomotive or car) being saved.

Table 16. Example of output files created by running a simulation with simulation parameters put forth in Figure 17

File Name	File Contents
IFN_1_locomotive.csv	Results of simulation for locomotive at position 1 in train consist
IFN_2_car.csv	Results of simulation for car at position 2 in train consist
IFN_15_car.csv	Results of simulation for car at position 15 in train consist
IFN_30_car.csv	Results of simulation for car at position 30 in train consist
IFN_brake_pipe_pressures.csv	Brake pipe pressure results for each rail vehicle in the train consist
IFN_auxiliary_reservoir_pressures.csv	Auxiliary reservoir pressure results for each rail vehicle in the train consist
IFN_emergency_reservoir_pressures.csv	Emergency reservoir pressure results for each rail vehicle in the train consist
IFN_coupler_forces.csv	Trailing longitudinal coupler force results for each rail vehicle in the train consist
IFN_coupler_displacements.csv	Trailing longitudinal coupler displacement results for each rail vehicle in the train consist

In TPS, it is assumed that cars have an auxiliary reservoir and an emergency reservoir. However, it is not assumed that locomotives have an auxiliary reservoir and an emergency reservoir. Therefore, in Table 16, for the standard files that have auxiliary reservoir pressures and emergency reservoir pressures, in the case of locomotives the auxiliary reservoir and emergency reservoir pressure values are simply reported as the brake pipe operating pressure value of 105.0 psi.

Table 17 puts forth the file format for a car results file. Most of the fields are self-explanatory. However, a couple points must be made. The L/V ratio (column index 13) refers to wheel lateral force over wheel vertical force. The L/V ratio is commonly used in the railroad industry to predict wheel climb and rail rollover. With regard to the maximum L/V ratio (column index 13), a value of 100,000,000.0 indicates wheel unloading. In a wheel unloading scenario, the vertical force in the denominator goes to zero, which results in an undefined value for L/V. In such scenarios, a value of

100,000,000.0 is assigned.

With regard to the control valve operating mode (column index 14), values of 0, 1, 2, or 3 are possible. Lap mode is represented by a 0, service brake application mode is represented by a 1, brake release mode is represented by a 2, and emergency brake application mode is represented by a 3. These modes are explained in more detail in the “Car Control Valve” section beginning on page 80.

Table 17. File format for car results file

Column Index	Physical Variable (Units)
1	Time (s)
2	Position (ft)
3	Velocity (mph)
4	Track grade (%)
5	Track curvature (deg)
6	Track superelevation (in)
7	Deflection of trailing coupler (in)
8	Deflection of leading coupler (in)
9	Longitudinal force applied by trailing coupler (lb)
10	Longitudinal force applied by leading coupler (lb)
11	Lateral force applied by trailing coupler (lb)
12	Lateral force applied by leading coupler (lb)
13	Maximum L/V ratio
14	Control valve operating mode
15	Brake pipe pressure (psi)
16	Auxiliary reservoir pressure (psi)
17	Emergency reservoir pressure (psi)
18	Brake cylinder pressure (psi)

Similarly, Table 18 puts forth the file format for a locomotive results file. The first 13 parameters are equivalent to the first 12 parameters for a car results file (Table 17).

Table 18. File format for locomotive results file

Column Index	Physical Variable (Units)
1	Time (s)
2	Position (ft)
3	Velocity (mph)
4	Track grade (%)
5	Track curvature (deg)
6	Track superelevation (in)
7	Deflection of trailing coupler (in)
8	Deflection of leading coupler (in)
9	Longitudinal force applied by trailing coupler (lb)
10	Longitudinal force applied by leading coupler (lb)
11	Lateral force applied by trailing coupler (lb)
12	Lateral force applied by leading coupler (lb)
13	Maximum L/V ratio
14	Automatic air brake pressure setting (psi)
15	Independent brake pressure setting (psi)
16	Throttle setting (unitless)
17	Dynamic brake setting (unitless)

The sign convention used for the coupler longitudinal forces (column indices 9 and 10 in Table 17 and Table 18) is explained in Table 19.

Table 19. Sign convention for longitudinal component of coupler forces

Coupler Position	Sign of Longitudinal Force on Rail Vehicle Due to Coupler	Physical Meaning
Trailing	Positive	Longitudinal force directed toward forward direction of travel, which means trailing coupler is in compression (and has negative displacement)
Trailing	Negative	Longitudinal force directed against forward direction of travel, which means trailing coupler is in tension (and has positive displacement)
Leading	Positive	Longitudinal force directed toward forward direction of travel, which means leading coupler is in tension (and has positive displacement)
Leading	Negative	Longitudinal force directed against forward direction of travel, which means leading coupler is in compression (and has negative displacement)

The sign convention used for the coupler lateral forces (column indices 11 and 12 in Table 17 and Table 18) is explained in Table 20.

Table 20. Sign convention for lateral component of coupler forces

Curvature Sign	Sign of Lateral Force on Car Due to Coupler	Physical Meaning
Positive	Positive	Lateral force directed away from center of curvature
Positive	Negative	Lateral force directed towards center of curvature
Negative	Positive	Lateral force directed towards center of curvature
Negative	Negative	Lateral force directed away from center of curvature

View Results

TPS does not have a graphical user interface or plotting tools for viewing the results of a simulation. Instead, the user is expected to use external tools for plotting the results output by TPS. The user may choose to use a spreadsheet program, such as LibreOffice Calc or Microsoft Excel. However, TPS output files can be large. This is particularly true if a high sampling rate, such as 1000 hertz, is desired by the user (see “Simulation Parameters” section on page 28 for information on the user-defined sampling rate). Such large files are oftentimes not easily handled by spreadsheet programs. However, if a lower sampling rate is input by the user, such as a sampling rate of 10 hertz, then opening the CSV files generated by TPS using a spreadsheet program is more feasible.

An alternative solution is to make use of some of the free programs available for plotting large data sets. One such program is named gnuplot, which can be downloaded at www.gnuplot.info. Gnuplot is a powerful tool but takes some time to learn since it has its own language syntax for generating plots. There are many good resources on the internet available for learning gnuplot, including the official user manual, which is available as a PDF download. Therefore, a complete description of the intricacies of gnuplot will not be presented here. Instead, Figure 22 shows example syntax for generating a simple plot. The following paragraph will step through this file line-by-line to explain some of the basic gnuplot commands.

```

1 # Read comma-delimited data from file
2 set datafile separator comma
3
4 # First row contains column headers
5 set datafile columnheaders
6
7 # Set graph title
8 set title 'Car 15'
9
10 # Set label of x-axis
11 set xlabel 'Time (s)'
12
13 # Set label of y-axis
14 set ylabel 'Velocity (mph)'
15
16 # Use a line graph
17 set style data line
18
19 # Plot all rows of a single column
20 plot 'C:/Users/Leith/Desktop/Simulation_TEST_15_car.csv' \
21     using 1:3 title columnheader linewidth 4
22

```

Figure 22. Example gnuplot file

In gnuplot, # represents a comment, so all lines beginning with # are comment lines. Line 2 tells gnuplot that the file format of the data file is a comma-separated value file (as this is the type of file output by TPS). Line 5 tells gnuplot that the first row consists of column headers instead of data. Line 8 sets the title of the plot. Lines 11 and 14 set the x axis and y axis labels of the plot, respectively. Line 17 sets the style of plot to a line plot. Lines 20 and 21 are actually a single command. The backslash symbol (\) at the end of line 20 indicates that the command is not complete and continues on the next line. Line 20 begins with the keyword `plot` which is the main plotting command in gnuplot for generating 2D plots. The absolute file path of the data file is placed in quotes immediately following the keyword `plot`. In this example, the file path is `C:/Users/Leith/Desktop/Simulation_TEST_15_car.csv`. Line 21 begins with the keyword `using`. Following the keyword `using` are two numbers separated by a colon. The first number indicates the column number of the independent (x axis) variable, and the second number indicates the column number of the dependent (y axis) variable. In this case, `1:3` is indicated. Assuming that the data file is that of a car, this means that the independent variable is time, and the dependent variable is velocity. This can be verified by looking back at Table 17 on page 39 which shows that time is recorded in the first column and velocity is recorded in the third column. The rest of line 21 tells gnuplot to include the title, column header information, and sets the line width of the plot.

Once the user installs gnuplot and runs the file shown in Figure 22, the plot in Figure 23 is produced. This data represents with data for car 15 in the train consist example put forth throughout Chapter 1.

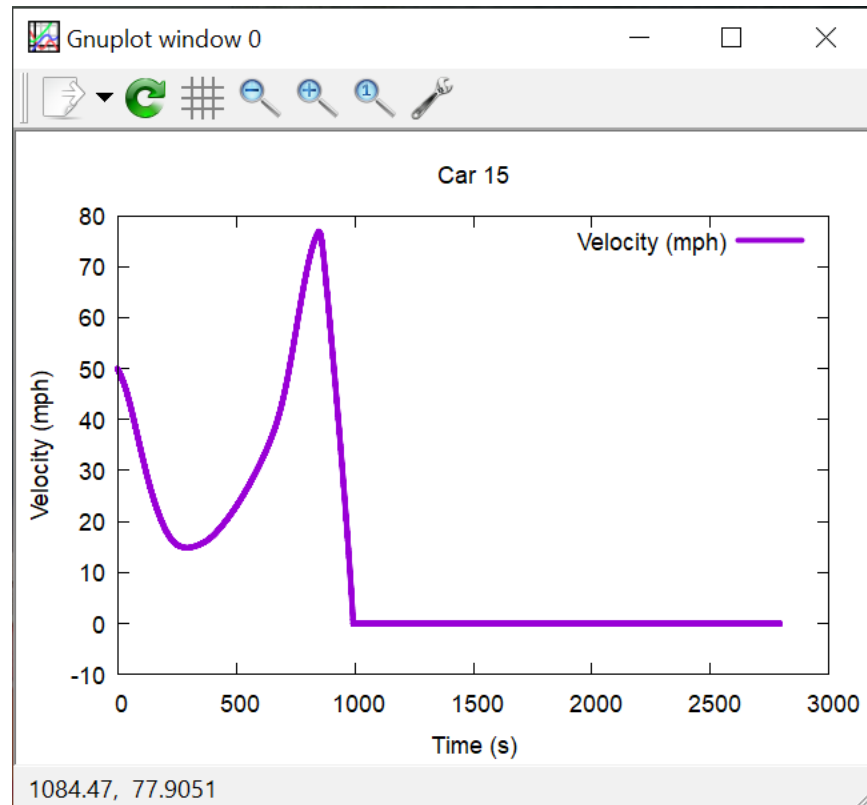


Figure 23. Example plot output by gnuplot

From the plot in Figure 23, it is clear that the car has a speed of approximately zero miles per hour after about 1100 seconds. The plot of the data when the car is stopped is not useful and does not need to be plotted. The code on lines 20 and 21 of the gnuplot input file shown in Figure 22 can be modified to plot data only for the rows when the time value in the first column is less than 1100 seconds. This can be accomplished by replacing lines 20 and 21 in Figure 22 with the lines 20 and 21 shown in Figure 24.

```

1 # Read comma-delimited data from file
2 set datafile separator comma
3
4 # First row contains column headers
5 set datafile columnheaders
6
7 # Set graph title
8 set title 'Car 15'
9
10 # Set label of x-axis
11 set xlabel 'Time (s)'
12
13 # Set label of y-axis
14 set ylabel 'Velocity (mph)'
15
16 # Use a line graph
17 set style data line
18
19 # Plot specific rows (based on row value) of a single column
20 plot 'C:/Users/Leith/Desktop/Simulation_TEST_15_car.csv' \
21     using (($1>0) & ($1<1100)) ? $1 : NaN:3 title columnheader linewidth 4
22

```

Figure 24. Gnuplot command for plotting specific rows of data

Lines 1 through 20 in Figure 24 are actually the same as lines 1 through 20 in Figure 22. It is only line 21 which has changed. Line 21 in Figure 24 uses the gnuplot ternary operator “?”. The code before the ternary operator ? gives the condition that the data in column 1 should be greater than 0 and less than 1100. If this applies, then the data should be plotted, as indicated by the \$1 immediately following the ternary operator ?. However, if the condition does not apply, then the data should not be plotted, which is indicated by the NaN value after the colon. This is a very brief explanation of the gnuplot ternary operator ?. A more detailed explanation can be found in the official user manual. The plot that results from running the code with the new line 21 is shown in Figure 25. As can be seen, only data for a time value of less than 1100 is now plotted.

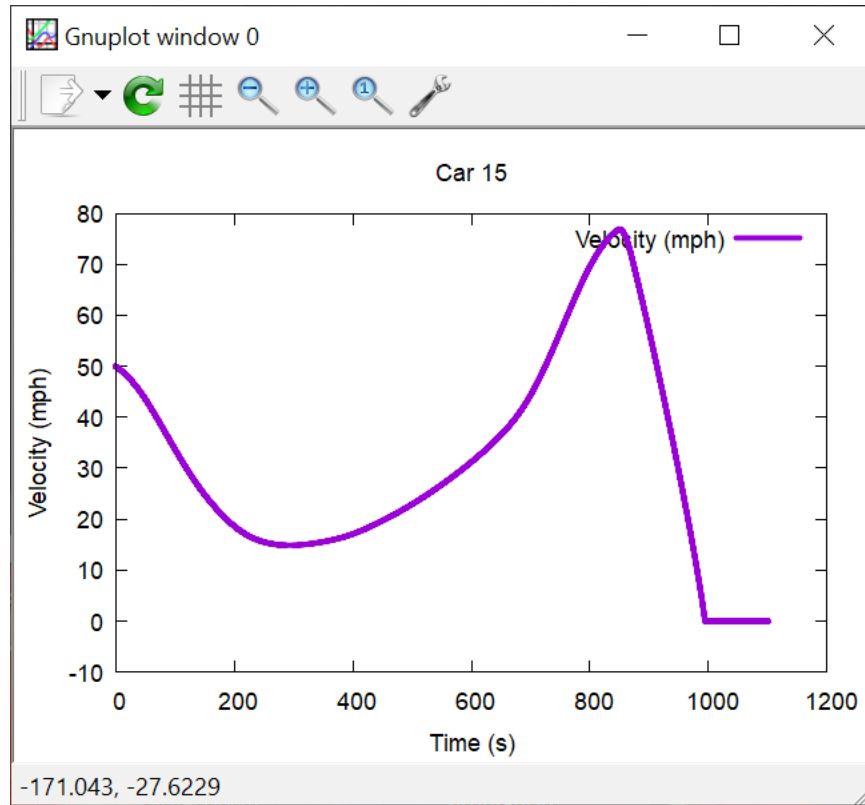


Figure 25. Example plot output by gnuplot with only a subset of data plotted

3. TPS Mathematical Algorithms

This chapter discusses the following components and how they are modeled in TPS:

- Track
- Coupler
- Car
- Locomotive tractive effort and dynamic braking
- Car brake components
- Brake pipe

Throughout this chapter, commentary is made on where the variables and calculations being discussed are found and implemented in the TPS source code.

Track Model

Coordinate system

The local coordinate system of the track is shown in Figure 26. The sign convention for rotation about each of these axes will be discussed in more detail in the following subsections.

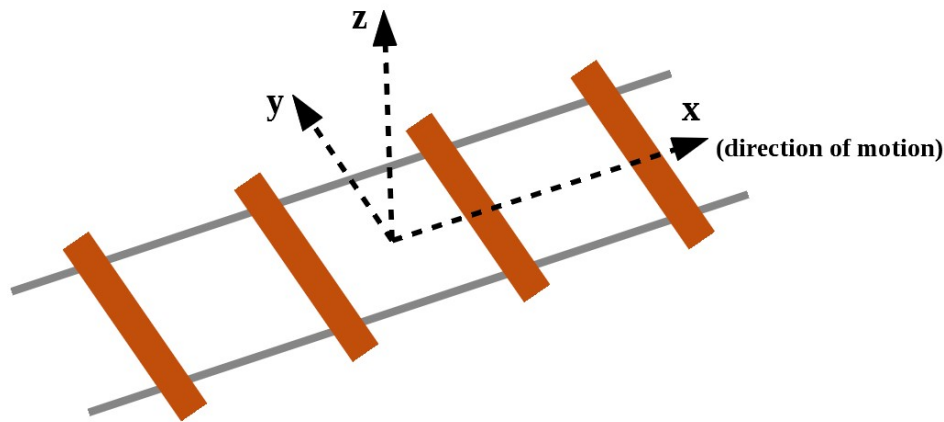


Figure 26. Local coordinate system of track

Grade

Track grade is defined as rotation about the y axis shown in Figure 26. The convention is

adopted that positive rotation about the y axis (with the y axis is pointing out of the page) is defined as clockwise rotation, as shown by the red arrow in Figure 27. This means that a positive rotation about the y axis equates to a positive grade, such as moving up a hill.

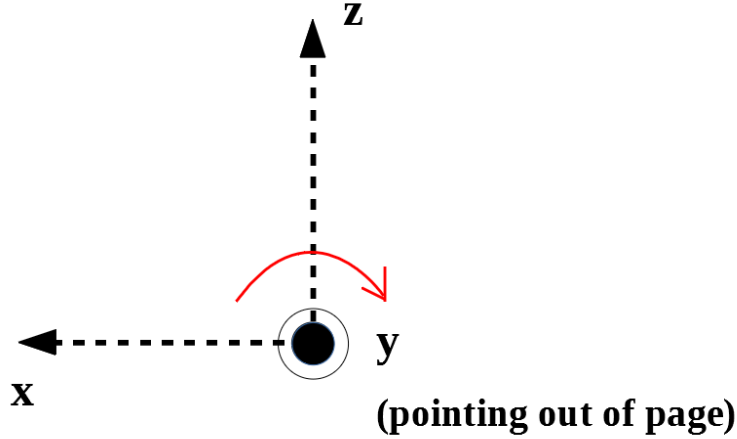


Figure 27. Sign convention for track grade (rotation about y axis)

Track grade is a function of the longitudinal position along the track and is defined as a percentage. A one percent grade is equal to a one foot rise over 100 feet. Similarly, a negative one percent grade represents a one foot drop over 100 feet. In TPS, the minimum and maximum allowable grade values are -5 percent and 5 percent, respectively (see Table 1 on page 3).

Grade is sometimes measured in degrees. Therefore, the mathematical relationship between percent g and degrees g_d is provided in Equation 1. If the user possesses track grade data in degrees, they can use Equation 1 to convert from degrees to percent.

$$g = 100 \tan\left(g_d \left(\frac{\pi}{180}\right)\right) \quad (1)$$

Curvature

Track curvature is defined as rotation about the z axis shown previously in Figure 26 on page 46. The convention is adopted that positive rotation about the z axis (with the z axis is pointing out of the page) is defined as clockwise rotation, as shown by the red arrow in Figure 28. This means that a positive rotation about the z axis (i.e., positive curvature) equates to curving to the right. This convention was mentioned previously in Table 2 on page 4.

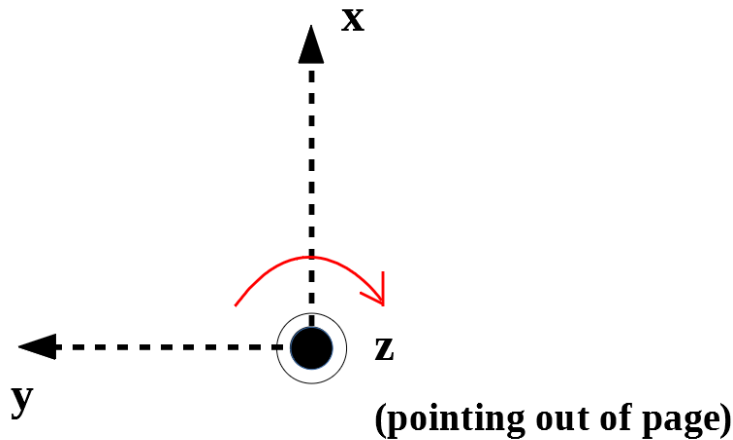


Figure 28. Sign convention for track curvature (rotation about z axis)

Track curvature is a function of the longitudinal position along the track and has units of degrees.⁷ The track curvature function is continuous and consists of piecewise linear segments. In TPS, the minimum and maximum allowable curvature values are -10 degrees and 10 degrees, respectively (see Table 1 on page 3).

In the United States, the railroad industry defines track curvature κ_r as the central angle (in degrees) that is subtended by a 100 foot chord, as shown in Figure 29. A track curvature κ_r approaching zero degrees (and radius r approaching infinity) represents tangent (straight) track.

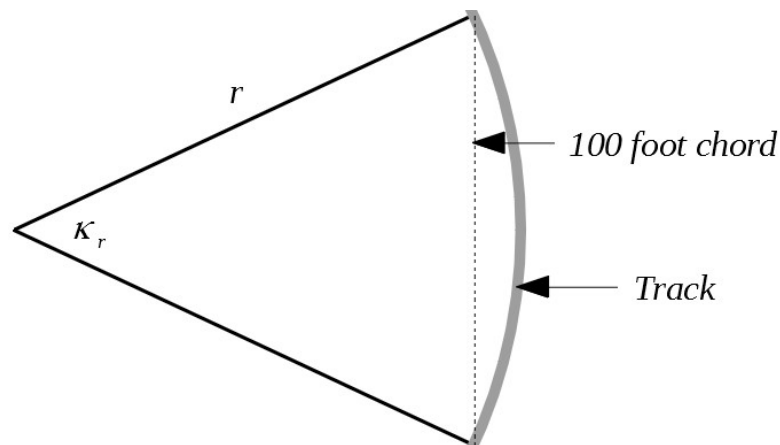


Figure 29. Railroad industry's definition of track curvature

⁷ This is in contrast to the geometric definition of curvature which has units of inverse distance (such as inverse meters in SI units). The geometric definition of curvature is discussed in more detail in the “Analytical Formulation of the Relative Angle Between Adjacent Cars” section beginning on page 60.

The relationship between track curvature κ_r and radius of curvature r is put forth in Equation 2. Note that in Equation 2, the factor $\pi/180$ converts κ_r from degrees to radians.

$$\sin\left(\frac{\kappa_r}{2}\left(\frac{\pi}{180}\right)\right) = \frac{50}{r} \quad (2)$$

Figure 30 demonstrates that the relationship in Equation 2 results from using two right triangles to approximate the area shown in Figure 29.

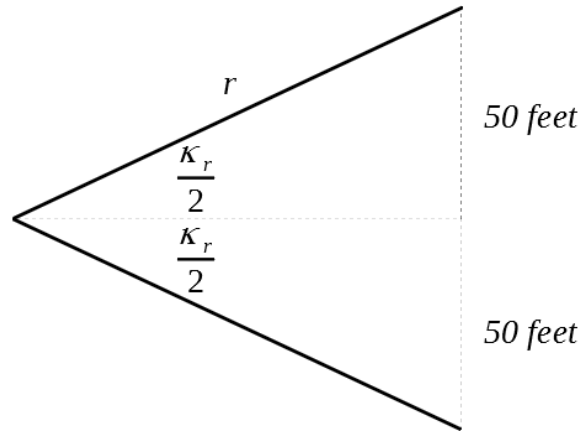


Figure 30. Division of Figure 29 into two right triangles

Solving for r in Equation 2 results in Equation 3.

$$r = \frac{50}{\sin\left(\frac{\kappa_r}{2}\left(\frac{\pi}{180}\right)\right)} \quad (3)$$

If a track curvature κ_r is known, Equation 3 can be used to calculate the radius of curvature r .

Superelevation

Track superelevation is defined as rotation about the x axis shown previously in Figure 26 on page 46. The convention is adopted that positive rotation about the x axis (with the x axis is pointing out of the page) is defined as clockwise rotation, as shown by the red arrow in Table 25. This means that a positive rotation about the x axis equates to the right rail being elevated above the left rail when looking forward in the direction of motion. This convention was mentioned previously in Table 3 on page 4.

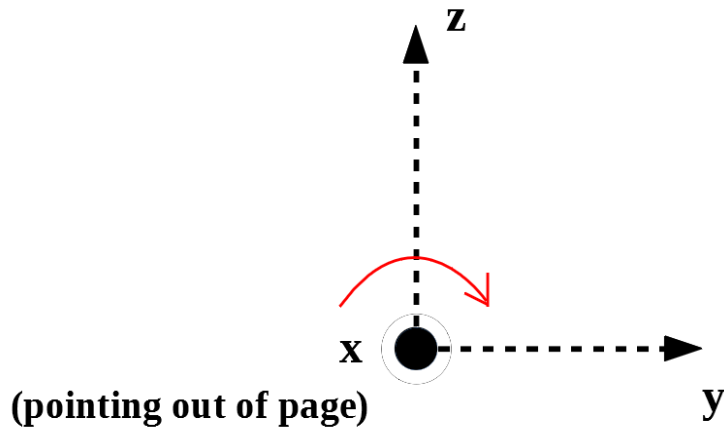


Figure 31. Sign convention for track superelevation
(rotation about x axis)

Track superelevation is a function of the longitudinal position along the track and has units of inches. Like track curvature, track superelevation function is continuous and consists of piecewise linear segments. In TPS, the minimum and maximum allowable superelevation values are -5 inches and 5 inches, respectively (see Table 1 on page 3).

Coupler Model

An example force-displacement function is shown in Figure 32.

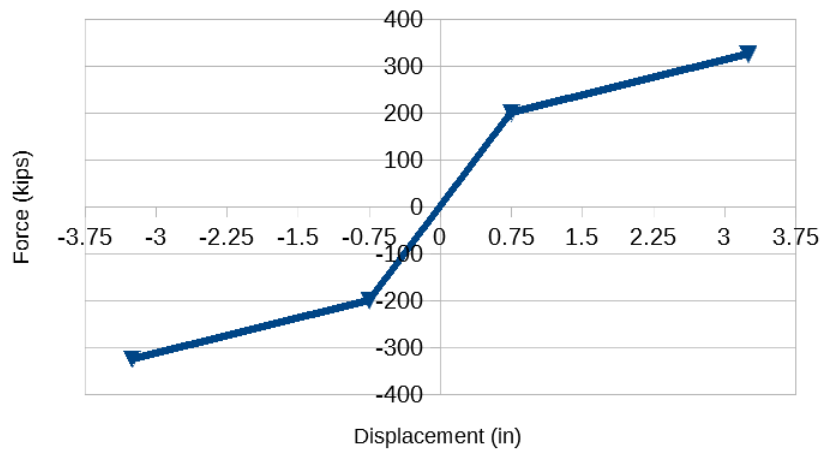


Figure 32. Example coupler force-displacement function

In TPS, the sign convention is adopted that negative displacement represents compression, and positive displacement represents tension. As shown in Figure 32, the

force-displacement function of a coupler is continuous and consists of piecewise linear segments. The minimum and maximum allowable slopes for any linear segment are 1.0 kips per inch and 1000.0 kips per inch, respectively. In the TPS source code, these minimum and maximum allowable slope values are stored in the `MINS_US` and `MAXS_US` variables, respectively, which are member of the `Coupler` class. Furthermore, in TPS, a small damping value is assumed. In the TPS source code, this damping value is stored in the `DAMPING_CONST` variable which is also a member of the `Coupler` class.

Typical real-world couplers have a loading force-deflection curve and an unloading force-deflection curve, as shown in Figure 33. However, in TPS, a single force-deflection curve is defined, as described in the “Coupler” section on page 9. This single force-deflection curve represents the loading force-deflection curve.

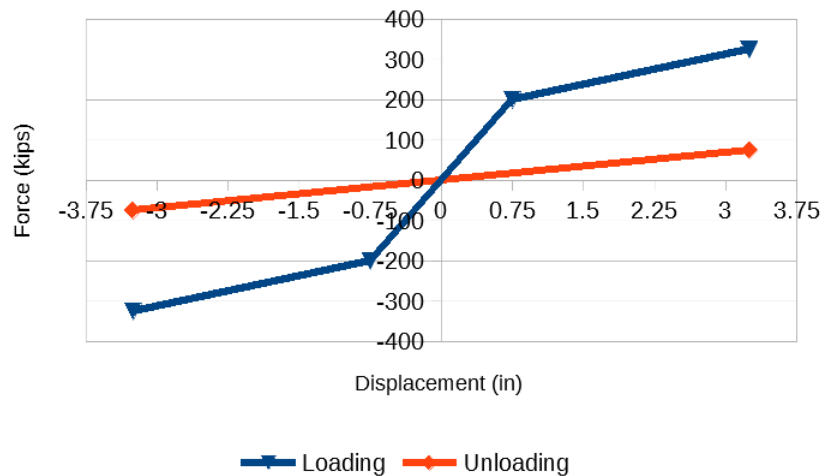


Figure 33. Example coupler loading and unloading functions

Car Model

In TPS, a car is modeled as a single lumped mass with one degree of freedom which accounts for longitudinal movement along the track. This single lumped mass represents the carbody, underframe, trucks, wheelsets, and all other associated masses. The minimum and maximum allowable values for the weight of a car are 30.0 kips and 600.0 kips, respectively, as shown in Table 5 on page 11.

It is assumed that a car does not deviate from the track centerline. In other words, a car does not experience any displacement in the lateral direction with respect to the track. Despite this assumption of no lateral displacement, there are known lateral and vertical forces acting on a car that is traversing a curved section of track. The following

subsections discuss these lateral and vertical forces as well as longitudinal forces which result in movement of a car on the track.

Local Coordinate System

The local coordinate system of a car is depicted in Figure 34 which is simply a new depiction of the coordinate system shown in Figure 26 on page 46 but now showing the outline of a car body. As stated in the previous in the “Track Model” section, track superelevation causes rotation about the x axis, track grade causes rotation about the y axis, and track curvature causes rotation about the z axis.

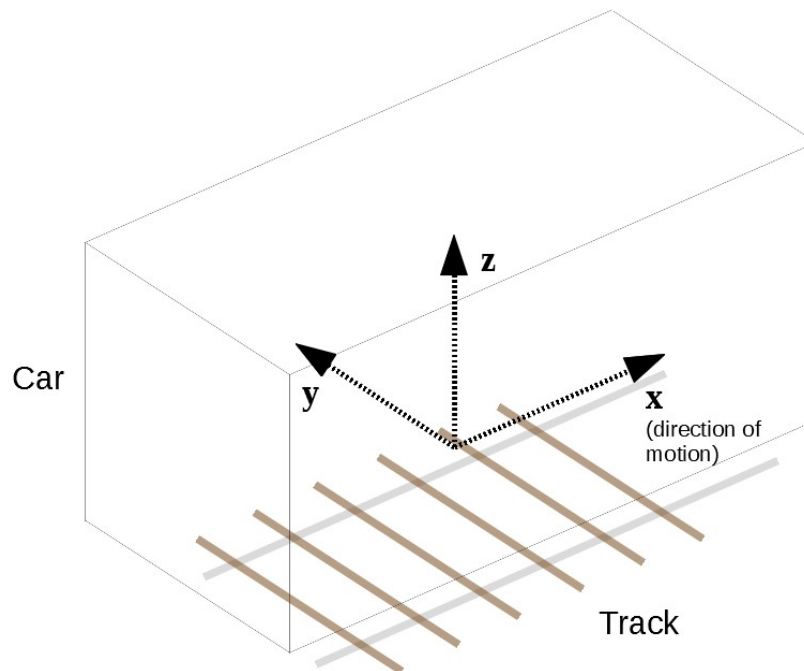


Figure 34. Local coordinate system of a car

A vector in global coordinates can be transformed into a car's local coordinate system using Euler angles and rotation matrices. For example, a vector \mathbf{W} representing the weight of a car has only a z axis component in global coordinates. However, if a car's local coordinate system is rotated about the x axis (due to track superelevation) or rotated about the y axis (due to track grade), then the \mathbf{W} vector will have x and/or y axis components as well as a z axis component.

Overview of Forces Acting On a Car

The primary physical forces acting on a car are the gravitational force, centripetal forces, curving resistance forces, propulsion resistance forces, and the forces due to the leading

coupler and the trailing coupler. Each of these forces will be discussed in more detail in the following subsections.

Gravitational Force

Gravitational force constitutes a constant vertical force in global coordinates. This vertical force f_g is given by the formula in Equation 4. In Equation 4, m is the mass of a rail vehicle, and g is the gravitational acceleration at the surface of the Earth which is approximately 9.8 m/s^2 .

$$f_g = -m g \quad (4)$$

However, the local rotation of a rail vehicle due to track grade and/or track superelevation causes the gravitational force to be transformed into a local force with a vertical component as well as a longitudinal component and a lateral component.

In the TPS source code, transformation of the global gravitation force into the local coordinate system shown in Figure 34 occurs in the function named `calcLocal3DVecFromGlobalGravitationalForce` which is a member of the `RailVehicle` class. This function calls the `rotateVector` function which is a member of the `VectorRotator` class.

Reactive Centrifugal Force

When a car traverses a curve, there is a resulting inertial force, called the reactive centrifugal force, that is applied on the car. This reactive centrifugal force f_{rcf} is a lateral force, as shown in Figure 35. The equal and opposite force, referred to as the centripetal force f_{cf} , is also shown.

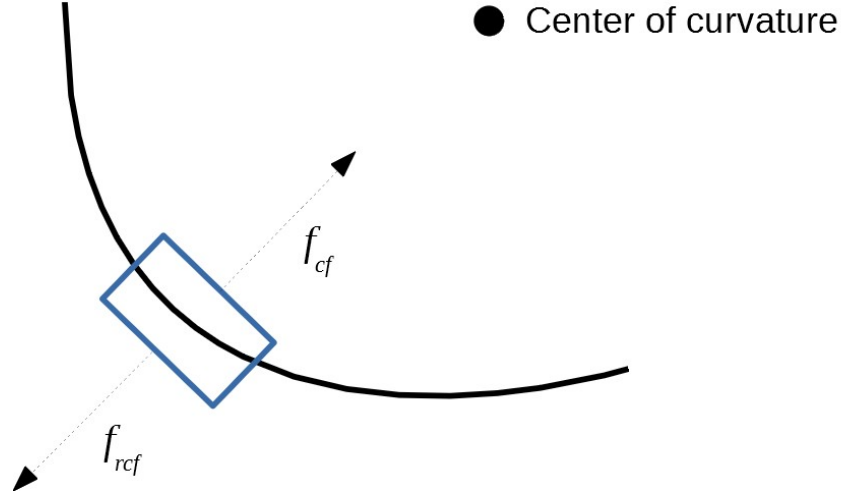


Figure 35. Diagram of reactive centrifugal force

The reactive centrifugal force can be calculated using Equation 5. M is the constant mass of the car, v is the instantaneous velocity of the car, r is the instantaneous radius of the curve.

$$f_{rcf} = \frac{M v^2}{r} \quad (5)$$

The radius of curvature r is positive when the track curvature κ_r is positive and is negative when the track curvature κ_r is negative (see Equation 3 on page 49). Therefore, the reactive centrifugal force f_{rcf} is positive when a car is traversing track with positive curvature (curving to the right) and is negative when traversing track with negative curvature (curving to the left). This information is summarized in Table 21.

Table 21. Reactive centrifugal force sign convention

Curvature Sign	Radius Sign	Reactive Centrifugal Force Sign
Positive (curving to right)	Positive	Positive
Negative (curving to left)	Negative	Negative

It can be seen visually that this sign convention is consistent with the local coordinate system shown previously in Figure 34 on page 52. Overlaying that local coordinate system on the car shown in Figure 35 results in Figure 36, where the local coordinate system is represented by the red arrows. In this configuration, the car is curving to the right, and the f_{rcf} force points in the direction of the y axis and is therefore positive.

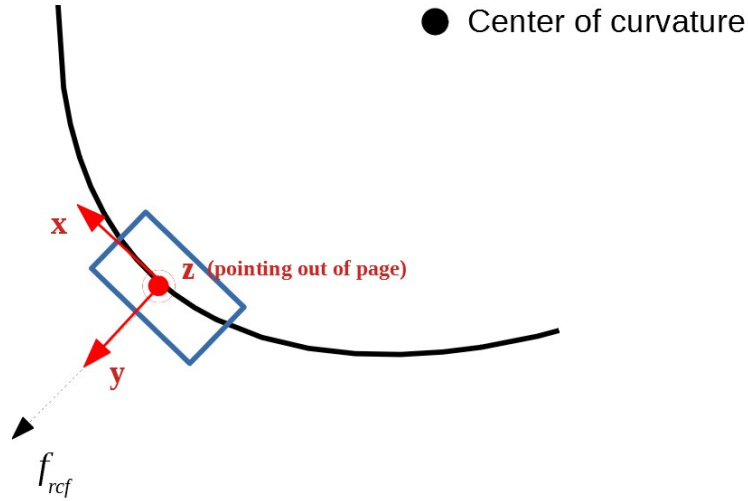


Figure 36. Diagram showing positive reactive centrifugal force resulting from curving to the right

Alternatively, Figure 37 shows the case for curving to the left. As can be seen, in this case, the reactive centrifugal force f_{rcf} points opposite to the direction of the y axis and therefore is negative.

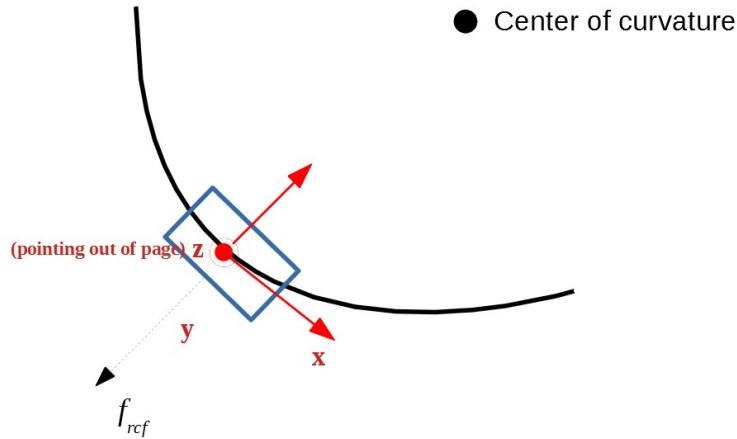


Figure 37. Diagram showing negative reactive centrifugal force resulting from curving to the left

The diagrams shown in Figure 36 and Figure 37 show that the sign convention put forth in Table 21 is consistent with the local coordinate system shown in Figure 34.

Note that if superelevation is present, the reactive centrifugal force has a vertical component as well as a lateral component. In the TPS source code, transformation of the global reactive centrifugal force given in Equation 5 into the local coordinate system shown in Figure 34 occurs in the `calcLocal3DVecFromGlobalReactiveCentrifugalForce`

function, which is a member of the `RailVehicle` class. This function calls the `rotateVector` function, which is a member of the `VectorRotator` class.

Curving Resistance

Equation 6 puts forth a formula that is used in the technical literature (Garg & Dukkipati 1984) for calculation of the resistance forces on a car due to track curvature. f_{crf} is the curving resistance force (in Newtons) of a rail vehicle, m is the mass of the rail vehicle (in kilograms), g is the gravitational acceleration at the surface of the Earth which is approximately 9.8 m/s^2 , and κ_r is the track curvature in degrees (see “Curvature” section beginning on page 47). It is assumed that the curving resistance force f_{crf} only has a longitudinal component. This formula assumes standard three-piece trucks without wheel/rail lubrication.

$$f_{crf} = -(0.0004 m g |\kappa_r|) \frac{v}{|v|} \quad (6)$$

In the TPS source code, curving resistance f_{crf} is calculated in the function named `calc_sumExternalTangentialForces` which is a member of the `RailVehicle` class.

Propulsion Resistance

Propulsion resistance includes aerodynamic resistance, flanging resistance, bearing resistance, and rolling friction at the wheel-rail contact patch. Propulsion resistance is calculated using the Canadian National Railway’s version of the Davis formula⁸ which is put forth in Equation 7. The propulsion resistance force f_{prf} is the resistance to movement of a car in pounds per ton, N is the number of axles, W is weight in tons, v is velocity in miles per hour, A is cross-sectional area in square feet, and C is the streamlining coefficient. It is assumed that the propulsion resistance force f_{prf} in Equation 7 only has a longitudinal component.

$$f_{prf} = -\left(1.5 + \frac{18 N}{W} + 0.03|v| + \frac{A C}{10000 W} v^2\right) \frac{v}{|v|} \quad (7)$$

In the TPS source code, propulsion resistance f_{prf} is calculated in the `calc_sumExternalTangentialForces` function which is a member of the `RailVehicle` class.

Cross-sectional area and streamlining coefficient values for common car types are given

8 The Davis formula (or Davis equation) is a well-known resistance equation used in the railroad industry.

in Table 22.

Table 22. Cross-sectional area and streamlining coefficients for common car types

Type of car	Cross-sectional area (ft ²)	Streamlining coefficient
Boxcar	140	4.9
Bulkhead flatcar (loaded)	140	5.3
Bulkhead flatcar (empty)	140	12.0
Coal gondola (loaded)	105	4.2
Coal gondola (empty)	105	12.0
Covered hopper	125	7.1
Freight locomotive (leading)	160	24.0
Freight locomotive (trailing)	160	5.5
Multi-level auto transporter (open)	150	12.3
Multi-level auto transporter (closed)	170	7.1
Standard flatcar (without trailers)	25	5.0
Standard flatcar (with trailers)	125	5.0
Tank car	95	5.5

Inter-Car Coupler Forces

At the beginning of each integration step, the inter-car coupler forces must be calculated. These forces are needed as they are part of the forcing term in the governing differential equation, which will be discussed in more detail in the following subsections.

Calculating the inter-car coupler forces begins by considering the difference between the instantaneous and unstressed center-to-center distance between adjacent rail vehicles.

For purposes of this explanation, this difference will be referred to as $\Delta_{stressed}$. The unstressed center-to-center distance is known since the length of each car is known. In the TPS source code, the unstressed center-to-center distance is stored in the `centerToCenterDistance_Unstressed` variable and is calculated in the `calc_centerToCenterDistance_Unstressed` function both of which are members of the `CouplingSystem` class.

The next step in the calculation of the inter-car coupler forces is to realize that when two cars are coupled together they can be abstracted as two springs in series. If two springs are in series and there is a force applied at one or both ends of this spring system, then the resulting force is the same on each of the two springs (or couplers in this case). With this in mind, TPS uses an iteration technique to approximate the force that is applied on each

of the couplers. This technique starts by assuming zero displacement of the first coupler and a displacement of $\Delta_{stressed}$ for the second coupler. Clearly, the forces in these two couplers will not be equal since a spring with no displacement should have zero force being applied to it while the second spring will have a non-zero force being applied since it has a displacement of $\Delta_{stressed}$. This is represented graphically in Figure 38. In Figure 38, the blue plot represents the force-displacement curve of the first coupler, and the orange plot represents the force-displacement curve of the second coupler which is assumed to be linear just for illustration purposes and to make a clear distinction from the first coupler represented by the blue plot. The blue plot represents a more realistic force-displacement curve for real-world couplers.

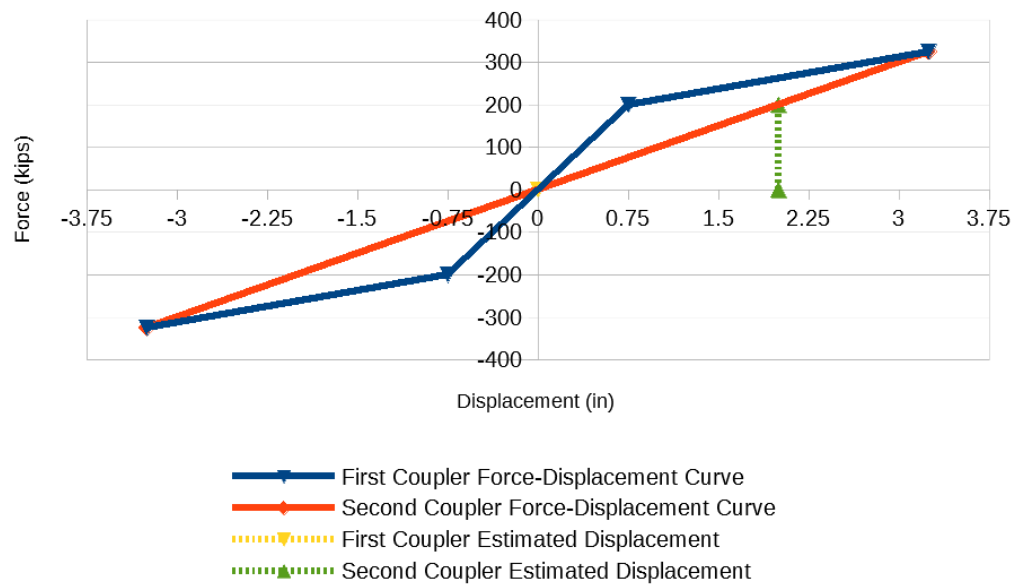


Figure 38. Initial assumption of zero displacement for first coupler and $\Delta_{stressed}$ displacement for second coupler

A zoomed-in view of the positive portion of the x axis of Figure 38 is shown in Figure 39. The yellow plot represents the initial assumed displacement of the first coupler which is zero, and the green plot shows the initial displacement of the second coupler which is equal to $\Delta_{stressed}$. In this example, $\Delta_{stressed}$ is assumed to be equal to 2.0 inches.

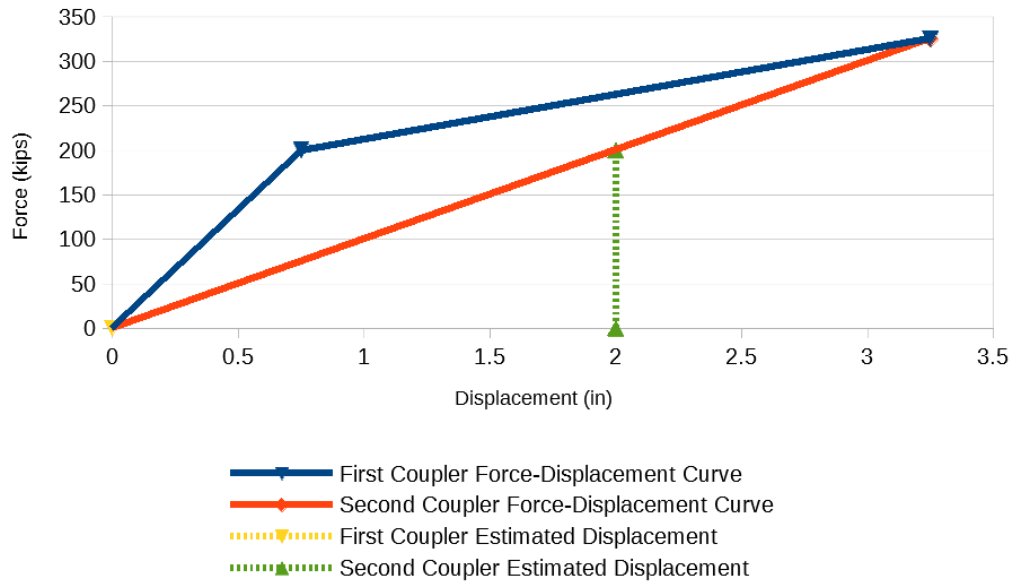


Figure 39. Zoomed-in view of Figure 38

Next, the displacement of the first coupler is increased by 0.25 inches while the displacement of the second coupler is decreased by 0.25 inches. This is represent graphically in Figure 40. It can be seen that the forces in the couplers are still not equal.

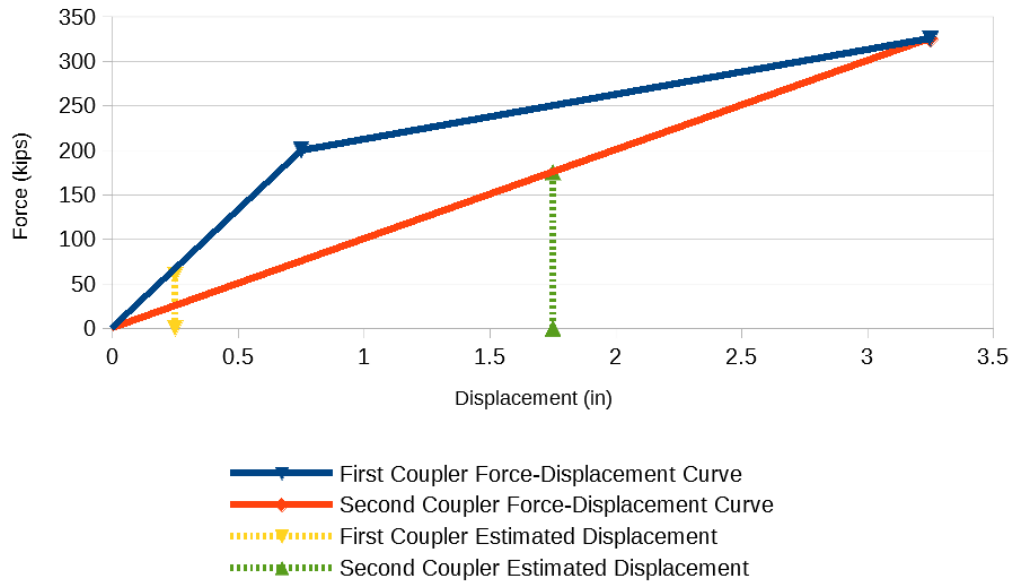


Figure 40. Updated assumption for displacement of first coupler and second coupler

This process is repeated until the two couplers have approximately the same force, as shown in Figure 41. In Figure 41, it can be seen that the first coupler has a displacement

of approximately 0.55 inches, and the second coupler has a displacement of 1.45 inches. It is important to note that the sum of these two displacements must be equal to Δ_{stressed} which in this example was assumed to be 2.0 inches. It is also clear that if the two couplers have identical force-displacement curves, then clearly the resulting displacements will be equivalent for each coupler.

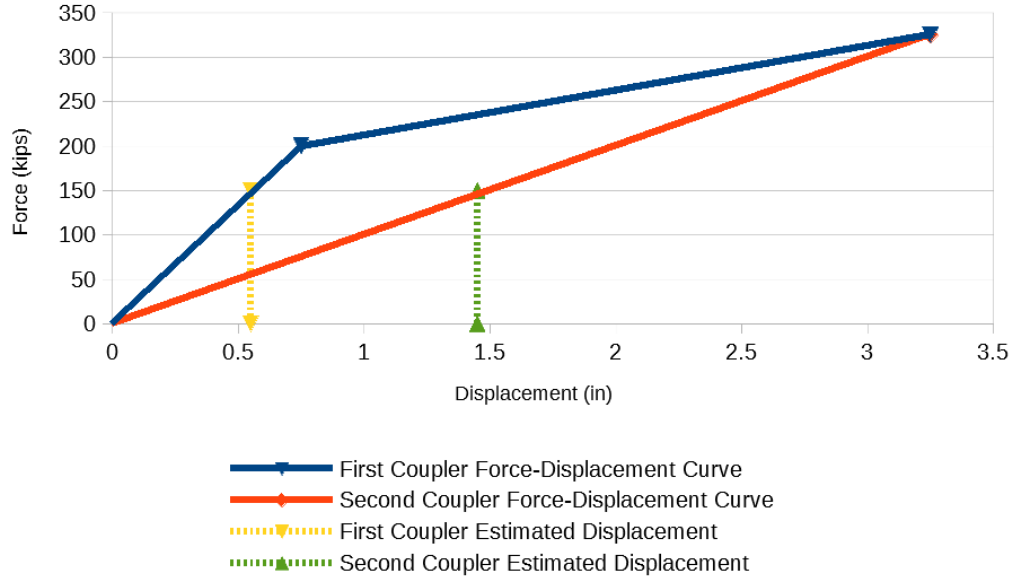


Figure 41. Final estimated displacement of first coupler and second coupler

In the TPS source code, this iterative process is performed in the function named `calculateForceOnLeadingRVDueToStiffnessAndCalculateCouplerDisplacements` which is a member of the `CouplingSystem` class.

Once the total inter-car coupler force is calculated using the process outlined above, the next step is to decompose the total inter-car coupler force into a longitudinal component and a lateral component. This decomposition depends on the angle between adjacent cars. Calculation of the angle between adjacent cars and the subsequent calculation of the longitudinal and lateral components of coupler forces will be discussed in the following subsections.

Analytical Formulation of the Relative Angle Between Adjacent Cars

On curved track, a non-zero angle exists between adjacent cars in a train consist. The angles $\alpha_{i,i-1}$ and $\alpha_{i,i+1}$ between car i and leading car $i-1$ and trailing car $i+1$ can be calculated if the track curvature κ_r is known (see “Curvature” section on page 47 for the railroad industry’s definition of track curvature).

In geometry, the curvature κ_g of a plane curve is defined as the inverse of the instantaneous radius r of curvature. κ_g can be approximated by the second derivative of the plane curve y with respect to the independent variable x .

$$\kappa_g = \frac{1}{r} \approx \frac{d^2 y}{dx^2} \quad (8)$$

The relationship between the railroad industry's definition of track curvature κ_r and the geometric definition of curvature κ_g can be established by plugging Equation 3 on page 49 into Equation 8. The result of this substitution is shown in Equation 9 which provides a value for κ_g . In Equation 9, the numerator is unitless, and the denominator has units of feet (since "50" refers to 50 feet as shown in Figure 30). Therefore, κ_g in Equation 9 has units of inverse feet (ft^{-1}).

$$\kappa_g = \frac{\sin\left(\frac{\kappa_r}{2}\left(\frac{\pi}{180}\right)\right)}{50} \approx \frac{d^2 y}{dx^2} \quad (9)$$

Taking the integral of Equation 9 results in the slope θ of the plane curve y , as shown in Equation 10.

$$\theta = \int \kappa_g(x) dx = \int \frac{\sin\left(\frac{\kappa_r(x)}{2}\left(\frac{\pi}{180}\right)\right)}{50} dx \approx \frac{dy}{dx} \quad (10)$$

This slope θ is a function of longitudinal position along the track. In the TPS source code, this value is calculated using the `angle` function which is a member of the `Track` class. The value returned by the `angle` function is stored in the `theta` variable which is a member of the `RailVehicle` class.

The angle $\alpha_{i,i-1}$ can then be defined by the difference in the slope θ_i at car i and the slope θ_{i-1} at its immediately adjacent leading car $i-1$, as shown in Equation 11. As was discussed in the "Curvature" section on page 47, the convention is adopted that curving to the right is positive curvature. This results in increasingly positive slopes θ , and the slope of the leading car θ_{i-1} will be greater than the slope θ_i at car i . This means $\alpha_{i,i-1}$ will be negative on sections with positive curvature. In the case of negative curvature (curving to the left), $\alpha_{i,i-1}$ will be positive.

$$\alpha_{i,i-1} = \theta_i - \theta_{i-1} \quad (11)$$

Similarly, the angle $\alpha_{i,i+1}$ can be defined by the difference in the slope θ_i at car i and the slope θ_{i+1} at car $i+1$ (Equation 12). Mirroring the explanation in the previous paragraph, in the case of positive curvature (curving to the right), θ_i will be greater than θ_{i+1} , and therefore, $\alpha_{i,i+1}$ will be positive in the case of positive curvature. In the case of negative curvature (curving to the left), $\alpha_{i,i+1}$ will be negative.

$$\alpha_{i,i+1} = \theta_i - \theta_{i+1} \quad (12)$$

In Equation 11 and Equation 12, $\alpha_{i,i-1}$ and $\alpha_{i,i+1}$ are in radians. Multiplying by $180/\pi$ will convert their respective values to degrees.

In the TPS source code, $\alpha_{i,i-1}$ and $\alpha_{i,i+1}$ are stored in the `alpha_leadingRailVehicle` and `alpha_trailingRailVehicle`, respectively, which are both members of the `RailVehicle` class.

Practical Implementation of the Relative Angle Between Adjacent Cars

As was stated in the “Track Model” section on page 46, the curvature function κ_g put forth in Equation 9 is made up of piecewise-linear segments. These piecewise-linear segments can be individually integrated and combined in order to produce a piecewise-smooth slope function θ put forth in Equation 10.

Assume the starting point and ending point of piecewise-linear segment n of the curvature function are given by $(x_{n,0}, y_{n,0})$ and $(x_{n,f}, y_{n,f})$. The variables $x_{n,0}$ and $y_{n,0}$ are the initial longitudinal position along the track and the initial geometric curvature value, respectively, for the n -th segment of the piecewise-linear function κ_g . Similarly, $x_{n,f}$ and $y_{n,f}$ are the final longitudinal position along the track and the final geometric curvature value, respectively. This n -th segment of the piecewise-linear function κ_g is given by Equation 13.

$$\kappa_{g,n}(x) - y_{n,0} = \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) (x - x_{n,0}) \quad (13)$$

Solve for $\kappa_{g,n}(x)$ in Equation 13 and distribute, as shown in Equation 14.

$$\kappa_{g,n}(x) = \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) x + \left(-x_{n,0} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) + y_{n,0} \right) \quad (14)$$

Integrating Equation 14 results in Equation 15, where y_n is an integration constant.

$$\theta_{g,n}(x) = \frac{1}{2} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) x^2 + \left(-x_{n,0} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) + y_{n,0} \right) x + y_n \quad (15)$$

The integration constant y_n in Equation 15 can be solved for by setting x equal to $x_{n,0}$ and setting the initial slope value of the current segment $\theta_{g,n}(x_{n,0})$ equal to the final slope value of the previous segment $\theta_{g,n-1}(x_{n-1,f})$ as shown in Equation 16.

$$\theta_{g,n-1}(x_{n-1,f}) = \frac{1}{2} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) x_{n,0}^2 + \left(-x_{n,0} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) + y_{n,0} \right) x_{n,0} + y_n \quad (16)$$

Solving for y_n results in Equation 17.

$$y_n = \theta_{g,n-1}(x_{n-1,f}) - \frac{1}{2} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) x_{n,0}^2 - \left(-x_{n,0} \left(\frac{y_{n,f} - y_{n,0}}{x_{n,f} - x_{n,0}} \right) + y_{n,0} \right) x_{n,0} \quad (17)$$

Equation 15 in combination with the value for the integration constant y_n given in Equation 17 can be used to find the slope any point in the interval n . As a result, the angle between adjacent cars can be found.

In the TPS source code, the above equations are implemented in the `slopeEquationCoefficients` function which is a member of the `Track` class.

Longitudinal Component of Inter-Car Forces

In the longitudinal direction along the track, a car and coupler together constitutes a single-degree-of-freedom harmonic oscillator. Therefore, a train consist is comprised of multiple harmonic oscillators in series. Figure 42 shows a series of n harmonic oscillators with index values for the masses m ranging from 0 to $n-1$ and index values for the spring force functions k ranging from 0 to $n-2$.

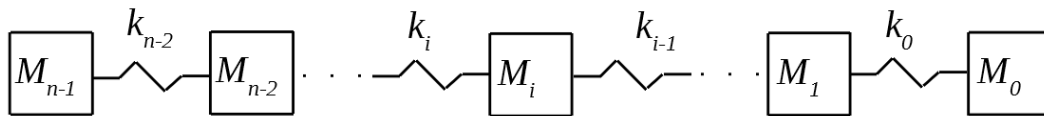


Figure 42. Harmonic oscillators in series

The differential equation governing the longitudinal motion of the i th car (mass M_i) in a consist is put forth in Equation 18. $\tau_{i,i-1}$ is the longitudinal component of the force on car i due to the adjacent leading car $i-1$, and $\tau_{i,i+1}$ is the longitudinal component of the force on car i due to the adjacent trailing car $i+1$. τ_{gf} , τ_{rcf} , τ_{crf} , τ_{prf} are the longitudinal component of

the global gravitational force, the longitudinal component of the global reactive centrifugal force, the longitudinal curving resistance force, and the longitudinal propulsion resistance force acting on car i , respectively.

$$M_i \frac{d^2 x}{dt^2} = \tau_{i,i-1} + \tau_{i,i+1} + \tau_{gf} + \tau_{rcf} + \tau_{crf} + \tau_{prf} \quad (18)$$

Formulas for $\tau_{i,i-1}$ and $\tau_{i,i+1}$ are put forth in Equation 19. In the TPS source code, these two variables are represented by the variables `tangentialForceDueToLeadingCoupler` and `tangentialForceDueToTrailingCoupler` which are members of the `RailVehicle` class and are calculated in the `calc_tangentialForceDueToLeadingCoupler` and `calc_tangentialForceDueToTrailingCoupler` functions, respectively. These functions are also members of the `RailVehicle` class.

$$\begin{aligned} \tau_{i,i-1} &= k_{i-1} \cos(\alpha_{i,i-1}) \\ \tau_{i,i+1} &= -k_i \cos(\alpha_{i,i+1}) \end{aligned} \quad (19)$$

k_{i-1} is the force of the spring in coupler $i-1$ and is a function f of the longitudinal position x_i of car i , the longitudinal position x_{i-1} of car $i-1$ (mass M_{i-1}), and the natural (unstressed) distance $d_{i,i-1}$ between the longitudinal center of car i and the longitudinal center of car $i-1$, as shown in Equation 20.

$$k_{i-1} = f((x_{i-1} - x_i) - d_{i,i-1}) \quad (20)$$

Similarly, k_i in Equation 19 is the force of the spring in coupler i and is a function f of the longitudinal position x_i of car i , the longitudinal position x_{i+1} of car $i+1$ (mass M_{i+1}), and the natural (unstressed) distance $d_{i,i+1}$ between car i and car $i+1$, as shown in Equation 21.

$$k_i = f((x_i - x_{i+1}) - d_{i,i+1}) \quad (21)$$

The method for calculating k_{i-1} and k_i was discussed previously in the section titled “Inter-Car Coupler Forces” on page 57.

Equation 18 can be converted into state-space form. First divide by mass M_i , as shown in Equation 22.

$$\frac{d^2 x}{dt^2} = \frac{1}{M_i} (\tau_{i,i-1} + \tau_{i,i+1} + \tau_{gf} + \tau_{rcf} + \tau_{crf} + \tau_{prf}) \quad (22)$$

The state-space variables for car i (mass M_i) are position x_i and velocity dx_i/dt , as shown

in Equations 23.

$$\begin{aligned} u_{2i} &= x_i \\ u_{2i+1} &= \frac{dx_i}{dt} \end{aligned} \quad (23)$$

The state-space variables shown in Equation 23 can be used to restate Equation 22 in state-space form, as shown in Equations 24. In the TPS source code, these state-space equations are implemented in the `calc_ssvDot` function in the `RailVehicle` class.

$$\begin{aligned} \frac{du_{2i}}{dt} &= \frac{dx_i}{dt} = u_{2i+1} \\ \frac{du_{2i+1}}{dt} &= \frac{d^2x_i}{dt^2} = \frac{1}{M_i}(\tau_{i,i-1} + \tau_{i,i+1} + \tau_{gf} + \tau_{rcf} + \tau_{crf} + \tau_{prf}) \end{aligned} \quad (24)$$

In TPS, the state-space equations in Equations 24 are solved using either the Runge-Kutta method (commonly known as the RK4 method) or the Runge-Kutta-Fehlberg method (commonly known as the RKF45 method). Both the RK4 method and the RKF45 method are explicit integration methods. However, the RK4 method has a fixed time step size, and the RKF45 method has an adaptive time step size. In the TPS source code, both the RK4 algorithm and RKF45 algorithms are implemented in the `ExplicitSSComponent` class.

Lateral Component of Inter-Car Forces

Let $p_{i,i-1}$ be the lateral component of the force on car i due to the adjacent leading car $i-1$. $p_{i,i-1}$ can be calculated using Equation 25. In Equation 25, $\tau_{i,i-1}$ is the longitudinal component of the force on car i due to car $i-1$, and $\alpha_{i,i-1}$ is the relative angle between car i and car $i-1$. In the TPS source code, this variable $p_{i,i-1}$ is represented by the variable `lateralForceDueToLeadingCoupler` which is a member of the `RailVehicle` class and is calculated in the `calc_lateralForceDueToLeadingCoupler` function which is also a member of the `RailVehicle` class.

$$p_{i,i-1} = \tau_{i,i-1} \tan(\alpha_{i,i-1}) \quad (25)$$

It is important to verify that the polarity of the lateral force $p_{i,i-1}$ produced by Equation 25 is consistent with the local coordinate system shown previously in Figure 34 on page 52. First assume that car i and the adjacent leading car $i-1$ are traversing a positive curve (curving to the right), as shown in Figure 43. A similar analysis for traversing a negative curve will not be presented in detail here but can be easily verified by the reader, if

desired. Note that the amount of curvature is exaggerated for full illustration purposes. The red arrows represent the local coordinate system for car i .

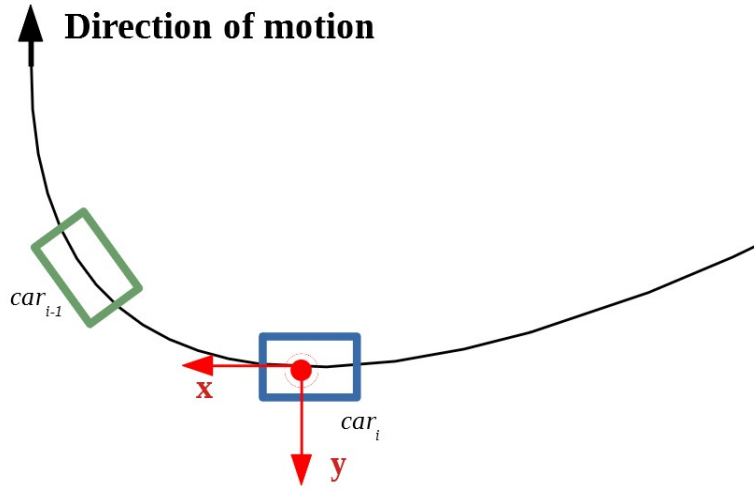


Figure 43. Car i and adjacent leading car $i-1$ traversing a positive curvature (curving to the right)

First, assume that there is tension in the coupler between car i and car $i-1$, as is shown in Figure 44, which shows a close-up of the force between car i and car $i-1$ on the left-hand side of the figure and a decomposition of the forces into a longitudinal and lateral component on the right-hand side. The longitudinal force $\tau_{i,i-1}$ on car i is positive, as it is in the same direction as the x axis, which indicates the direction of motion. Since the curvature is positive, $\alpha_{i,i-1}$ is negative, as explained in the discussion of the definition of $\alpha_{i,i-1}$ put forth in Equation 11 on page 61. Plugging in a positive value for $\tau_{i,i-1}$ and a negative value for $\alpha_{i,i-1}$ in Equation 25 results in a negative value for $p_{i,i-1}$. This is consistent with the resulting $p_{i,i-1}$ shown in Figure 44 since $p_{i,i-1}$ points in the opposite direction to the y axis of the local coordinate system which is represented by the red arrows.

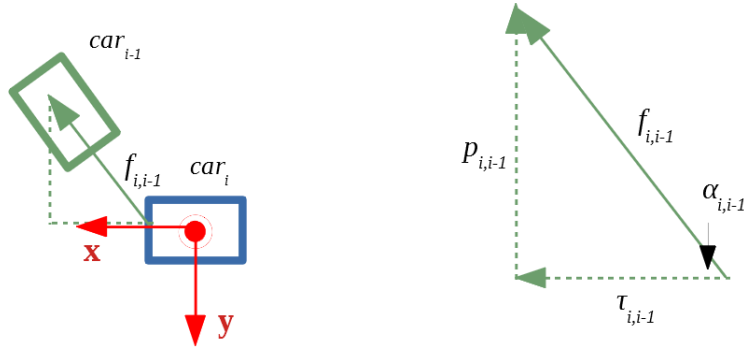


Figure 44. Decomposition of forces when couplers between car i and car $i-1$ (shown in Figure 43) are in tension

Next, assume that there is compression in the coupler between car i and car $i-1$, as is shown in Figure 45, which shows a close-up of the force between car i and car $i-1$ on the left-hand side of the figure and a decomposition of the forces into a longitudinal and lateral component on the right-hand side. The longitudinal force $\tau_{i,i-1}$ on car i is negative, as it is in the opposite direction of the x axis, which indicates the direction of motion. Since the curvature is positive, $\alpha_{i,i-1}$ is negative, as explained in the discussion of the definition of $\alpha_{i,i-1}$ put forth in Equation 11 on page 61. Plugging in a negative value for $\tau_{i,i-1}$ and a negative value for $\alpha_{i,i-1}$ in Equation 25 results in a positive value for $p_{i,i-1}$. This is consistent with the resulting $p_{i,i-1}$ shown in Figure 45 since $p_{i,i-1}$ points in the same direction as the y axis of the local coordinate system which is represented by the red arrows.

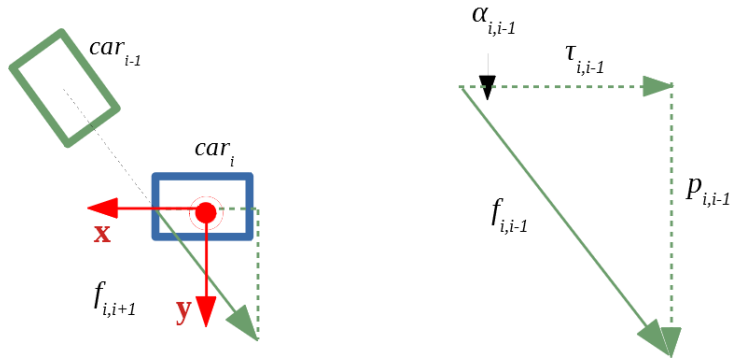


Figure 45. Decomposition of forces when couplers between car i and car $i-1$ (shown in Figure 43) are in compression

The preceding paragraphs determined that the lateral component $p_{i,i-1}$ of the coupler forces between car i and the adjacent leading car $i-1$ produced by Equation 25 are consistent

with the sign convention for the local coordinate system shown in Figure 34. A similar analysis will now be performed for the lateral component $p_{i,i+1}$ of the coupler forces between car i and the adjacent trailing car $i+1$. The lateral component of the coupler forces between these two cars can be calculated using Equation 26. In Equation 26, $\tau_{i,i+1}$ is the longitudinal component of the force on car i due to car $i+1$, and $\alpha_{i,i+1}$ is the relative angle between car i and car $i+1$. In the TPS source code, this variable $p_{i,i+1}$ is represented by the variable `lateralForceDueToTrailingCoupler` which is a member of the `RailVehicle` class and is calculated in the `calc_lateralForceDueToTrailingCoupler` function. This function is also a member of the `RailVehicle` class.

$$p_{i,i+1} = \tau_{i,i+1} \tan(\alpha_{i,i+1}) \quad (26)$$

It is important to verify that the polarity of the lateral forces $p_{i,i+1}$ produced by Equation 26 are consistent with the local coordinate system shown previously in Figure 34 on page 52. First assume that car i and the adjacent trailing car $i+1$ are traversing a positive curve (curving to the right), as shown in Figure 46. A similar analysis for traversing a negative curve will not be presented in detail here but can be easily verified by the reader, if desired. Note that the amount of curvature is exaggerated for full illustration purposes. The red arrows represent the local coordinate system for car i .

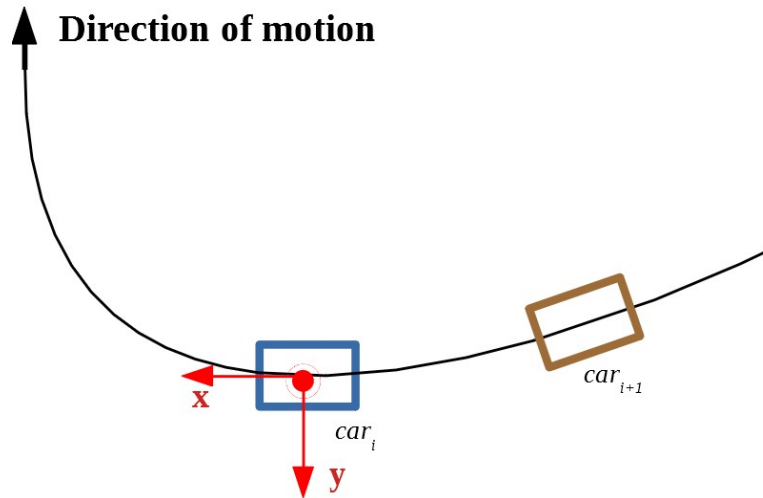


Figure 46. Car i and adjacent trailing car $i+1$ traversing a positive curvature (curving to the right)

First, assume that there is tension in the coupler between car i and car $i+1$, as is shown in Figure 47, which shows a close-up of the force between car i and car $i+1$ on the left-hand side of the figure and a decomposition of the forces into a longitudinal and lateral component on the right-hand side. The longitudinal force $\tau_{i,i+1}$ on car i is negative, as it is

in the opposite direction as the x axis, which indicates the direction of motion. Since the curvature is positive, $\alpha_{i,i+1}$ is positive, as explained in the discussion of the definition of $\alpha_{i,i+1}$ put forth in Equation 12 on page 62. Plugging in a negative value for $\tau_{i,i+1}$ and a positive value for $\alpha_{i,i+1}$ in Equation 26 results in a negative value for $p_{i,i+1}$. This is consistent with the resulting $p_{i,i+1}$ shown in Figure 47 since $p_{i,i+1}$ points in the opposite direction to the y axis of the local coordinate system which is represented by the red arrows.

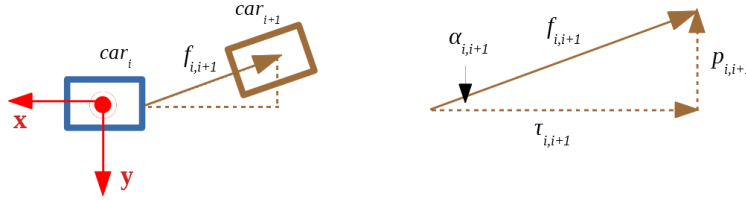


Figure 47. Decomposition of forces when couplers between car i and car $i+1$ (shown in Figure 46) are in tension

Next, assume that there is compression in the coupler between car i and car $i+1$, as is shown in Figure 48, which shows a close-up of the force between car i and car $i+1$ on the left-hand side of the figure and a decomposition of the forces into a longitudinal and lateral component on the right-hand side. The longitudinal force $\tau_{i,i+1}$ on car i is positive, as it is in the same direction as the x axis, which indicates the direction of motion. Since the curvature is positive, $\alpha_{i,i+1}$ is positive, as explained in the discussion of the definition of $\alpha_{i,i+1}$ put forth in Equation 12 on page 62. Plugging in a positive value for $\tau_{i,i+1}$ and a positive value for $\alpha_{i,i+1}$ in Equation 25 results in a positive value for $p_{i,i+1}$. This is consistent with the resulting $p_{i,i+1}$ shown in Figure 48 since $p_{i,i+1}$ points in the same direction as the y axis of the local coordinate system which is represented by the red arrows.

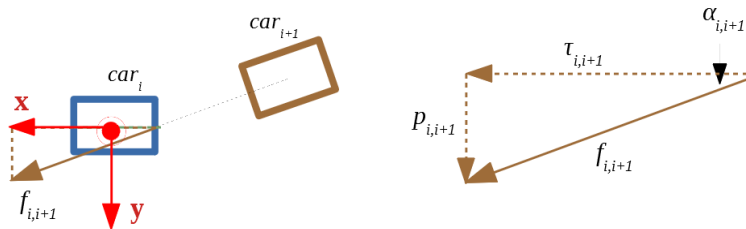


Figure 48. Decomposition of forces when couplers between car i and car $i+1$ (shown in Figure 46) are in compression

The above analyses only considered the sign of the lateral component of coupler forces when traversing positive curvatures (curving to the right). Performing a similar analysis for the case of negative curvature results in the the sign conventions put forth previously in Table 20 on page 41.

L/V Ratio⁹

In the railroad industry, the L/V ratio is a common metric used for predicting a derailment scenario due to wheel climb or rail rollover. The L/V ratio represents the lateral truck-side force over the vertical truck-side force.¹⁰ In TPS, four L/V ratios are calculated for a car:¹¹

- L/V ratio for left side of leading truck
- L/V ratio for right side of leading truck
- L/V ratio for left side of trailing truck
- L/V ratio for right side of trailing truck

The first step to calculate these four L/V ratios for car i is to use the leading lateral coupler force $p_{i,i-1}$, the trailing lateral coupler force $p_{i,i+1}$, the total car length L_i , and the truck center spacing D_i to calculate the resulting reactive forces r_{lead} and r_{trail} that the track applies to the leading and trailing trucks, respectively, of car i . These forces, which result in a yawing moment equal to zero, are shown in Figure 49.

9 The formulas and derivations in this section are based largely on those put forth in Garg & Dukkipati 1984.

10 There is also an individual wheel L/V ratio which is an independent metric and is not calculated by TPS.

11 The maximum of the four L/V ratios is output at each time step, as shown in Table 17 on page 39.

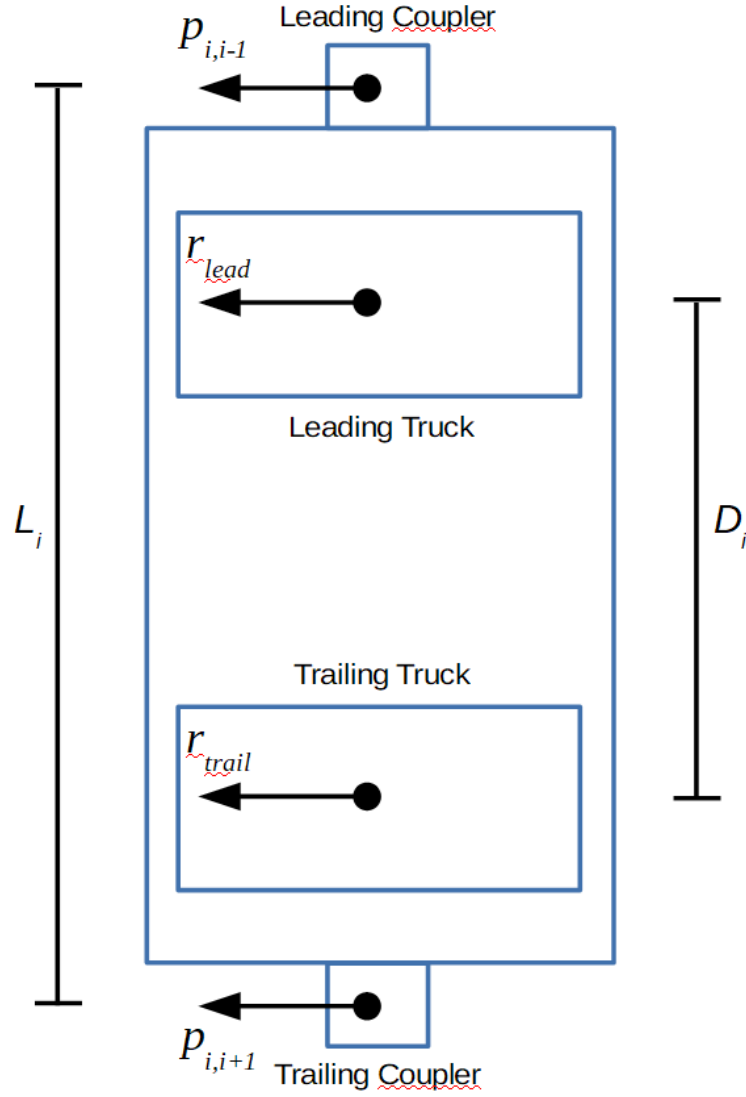


Figure 49. Car lateral forces which result in a yawing moment equal to zero

The leading lateral coupler force $p_{i,i-1}$ and the trailing lateral coupler force $p_{i,i+1}$ are known. The force r_{lead} can be solved for by formulating a moment equation with the axis at the trailing truck. This results in the moment equation in shown in Equation 27.

$$p_{i,i+1} \left(\frac{L_i - D_i}{2} \right) - r_{lead} D_i - p_{i,i-1} \left(D_i + \frac{L_i - D_i}{2} \right) = 0 \quad (27)$$

Solving for r_{lead} in Equation 27 results in Equation 28.

$$r_{lead} = \frac{p_{i,i+1} \left(\frac{L_i - D_i}{2} \right) - p_{i,i-1} \left(D_i + \frac{L_i - D_i}{2} \right)}{D_i} \quad (28)$$

The force r_{lead} in Equation 28 is the reactive force that the track applies to the leading truck. To get the force that the leading truck applies to the track f_{lead} , the reactive force r_{lead} must be multiplied by -1, as shown in Equation 29.

$$f_{lead} = (-1)r_{lead} \quad (29)$$

In the TPS source code, this force f_{lead} is represented by the variable `forceOnTrackDueToLeadingTruck` which is a member of the `RailVehicle` class. This variable is calculated in the `calc_forceOnTrackDueToLeadingTruck` function which is also a member of the `RailVehicle` class.

The force f_{trail} and reactive force r_{trail} representing the force that the trailing truck applies to the track and the reactive force that the track applies to the trailing truck, respectively, can be calculated in a similar manner, namely by formulating a moment equation with the axis at the leading truck. In the TPS source code, this force f_{trail} is represented by the variable `forceOnTrackDueToTrailingTruck` which is a member of the `RailVehicle` class. This variable is calculated in the `calc_forceOnTrackDueToTrailingTruck` function which is also a member of the `RailVehicle` class.

Next the reactive vertical forces that the left rail applies to the leading truck $v_{left,lead}$ and that the right rail applies to the leading truck $v_{right,lead}$ must be calculated. Similarly, the reactive vertical forces that the left rail applies to the trailing truck $v_{left,trail}$ and that the right rail applies to the trailing truck $v_{right,trail}$ must also be calculated. Figure 50 shows a car cross-sectional view at the trailing truck and the applicable forces. H_{cog} is the center of gravity height, H_c is the coupler height, and D_g is the track gage. p_{tot} and v_{tot} are the total lateral force and vertical force, respectively, applied by the car to the rail due to the reactive centrifugal force and gravitational force.

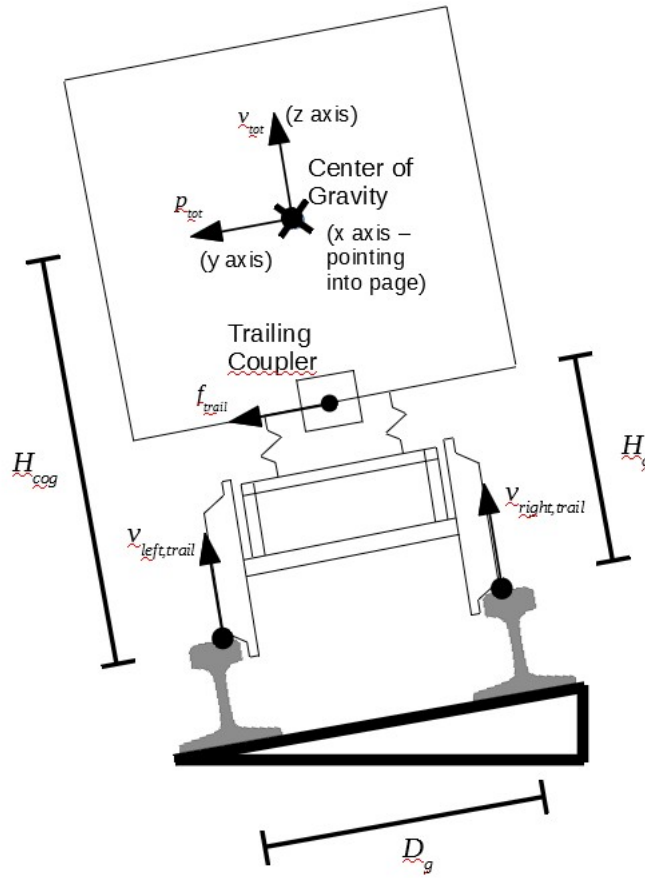


Figure 50. Car cross-sectional forces at trailing truck which result in rocking moment equal to zero

An equation for the reactive force $v_{left, trail}$ can be formulated by setting the sum of the moments around the right rail equal to zero, as shown in Equation 30.

$$\begin{aligned}
 D_g v_{left, trail} - \sqrt{\left(\left(D_g/2\right)^2 + \left(H_c\right)^2\right)} f_{trail} \sin(\alpha) - \\
 \sqrt{\left(\left(D_g/2\right)^2 + \left(H_{cog}\right)^2\right)} \frac{P_{tot}}{2} \sin(\beta) + \\
 \sqrt{\left(\left(D_g/2\right)^2 + \left(H_{cog}\right)^2\right)} \frac{v_{tot}}{2} \sin\left(\frac{\pi}{2} - \beta\right) = 0
 \end{aligned} \tag{30}$$

Equation 30 applies to the left rail of the trailing truck. However, replacing f_{trail} with f_{lead} would result in an equation $v_{left, lead}$ that applies to the left rail of the leading truck. The square root terms in Equation 30 represent the distance between the top of rail and bolster

and the distance between the top of rail and coupler and are arrived at by using the Pythagorean theorem. Solving for $v_{left, trail}$ results in Equation 31. Note that based on the direction of the arrow for reactive force $v_{left, trail}$ in Figure 50, when $v_{left, trail}$ is equal to or less than zero, this indicates wheel uplift since the left rail is not applying a positive reactive force to the left side of the trailing truck.

$$\begin{aligned}
 v_{left, trail} = & \frac{\sqrt{((D_g/2)^2 + (H_c)^2)} f_{trail} \sin(\alpha)}{D_g} + \\
 & \frac{\sqrt{((D_g/2)^2 + (H_{cog})^2)} \frac{P_{tot}}{2} \sin(\beta)}{D_g} - \\
 & \frac{\sqrt{((D_g/2)^2 + (H_{cog})^2)} \frac{v_{tot}}{2} \sin\left(\frac{\pi}{2} - \beta\right)}{D_g}
 \end{aligned} \tag{31}$$

In the TPS source code, the reactive force applied by the track to the left side of the trailing truck $v_{left, trail}$ is represented by the `verticalForceLeftSideTrailingTruck` variable which is a member of the `RailVehicle` class. This variable is calculated in the `calc_verticalForceLeftSideTrailingTruck` function which is also a member of the `RailVehicle` class. Similarly, the reactive force applied by the track to the left side of the leading truck $v_{left, lead}$ is represented by the `verticalForceLeftSideLeadingTruck` variable which is a member of the `RailVehicle` class. This variable is calculated in the `calc_verticalForceLeftSideLeadingTruck` function which is also a member of the `RailVehicle` class.

As was the case with $v_{left, trail}$, an equation for the reactive force $v_{right, trail}$ can be formulated by setting the sum of the moments around the left rail equal to zero, as shown in Equation 32.

$$\begin{aligned}
 -D_g v_{right, trail} - & \sqrt{((D_g/2)^2 + (H_c)^2)} f_{trail} \sin(\alpha) - \\
 & \sqrt{((D_g/2)^2 + (H_{cog})^2)} \frac{P_{tot}}{2} \sin(\beta) - \\
 & \sqrt{((D_g/2)^2 + (H_{cog})^2)} \frac{v_{tot}}{2} \sin\left(\frac{\pi}{2} - \beta\right) = 0
 \end{aligned} \tag{32}$$

Equation 32 applies to the right rail of the trailing truck. However, replacing f_{rail} with f_{lead} would result in an equation $v_{right,lead}$ that applies to the right rail of the leading truck. Solving for $v_{right,trail}$ results in Equation 33. Note that based on the direction of the arrow for the reactive force $v_{right,trail}$ in Figure 50, when $v_{right,trail}$ is equal to or less than zero, this indicates wheel uplift since the right rail is not applying a positive reactive force to the right side of the trailing truck.

$$v_{right,trail} = -\frac{\sqrt{((D_g/2)^2 + (H_c)^2)} f_{trail} \sin(\alpha)}{D_g} - \frac{\sqrt{((D_g/2)^2 + (H_{cog})^2)} \frac{P_{tot}}{2} \sin(\beta)}{D_g} - \frac{\sqrt{((D_g/2)^2 + (H_{cog})^2)} \frac{v_{tot}}{2} \sin\left(\frac{\pi}{2} - \beta\right)}{D_g} \quad (33)$$

In the TPS source code, the reactive force applied by the track to the right side of the trailing truck $v_{right,trail}$ is represented by the `verticalForceRightSideTrailingTruck` variable which is a member of the `RailVehicle` class. This variable is calculated in the `calc_verticalForceRightSideTrailingTruck` function which is also a member of the `RailVehicle` class. Similarly, the reactive force applied by the track to the right side of the leading truck $v_{right,lead}$ is represented by the `verticalForceRightSideLeadingTruck` variable which is a member of the `RailVehicle` class. This variable is calculated in the `calc_verticalForceRightSideLeadingTruck` function which is also a member of the `RailVehicle` class.

Formulas for angles α and β in Equation 30 through Equation 33 are given in Equation 34 and shown graphically in Figure 51.

$$\begin{aligned} \alpha &= \tan^{-1}\left(\frac{H_c}{0.5 D_g}\right) \\ \beta &= \tan^{-1}\left(\frac{H_{cog}}{0.5 D_g}\right) \end{aligned} \quad (34)$$

In the TPS source code, α is represented by the variable named `angleBetweenTopOfRailAndCoupler` and β is represented by the variable named `angleBetweenTopOfRailAndCenterOfGravity`. Both of these variable are members of the

RailVehicle class. These variables are calculated in the function named `calc_angleBetweenTopOfRailAndCoupler` and the function named `calc_angleBetweenTopOfRailAndCenterOfGravity`, respectively. Both of these functions are also members of the RailVehicle class.

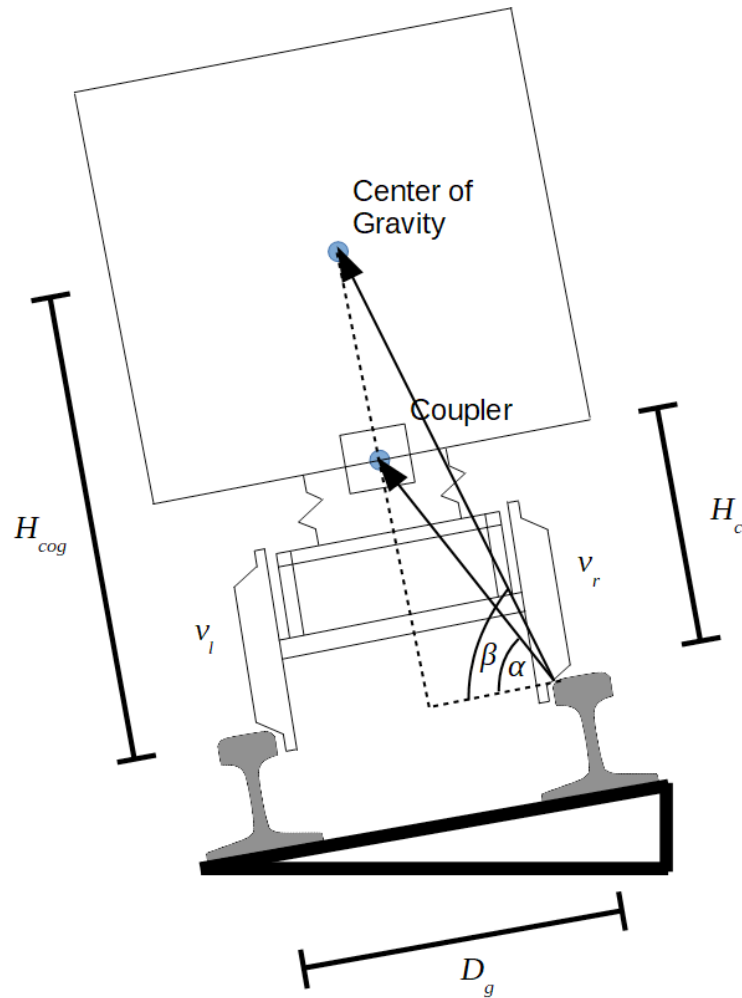


Figure 51. Angles α and β used in calculation of vertical reactive forces

The four L/V ratios can be calculated using the equations in Equation 35. In the TPS source code, these four L/V ratios are represented by the variables `lOverVForLeftSideOfLeadingTruck`, `lOverVForRightSideOfLeadingTruck`, `lOverVForLeftSideOfTrailingTruck`, and `lOverVForRightSideOfTrailingTruck`, which are all members of the RailVehicle class.

$$\begin{aligned}
\left(\frac{L}{V}\right)_{\text{leading truck, left rail}} &= \frac{f_{\text{lead}} + \frac{P_{\text{tot}}}{2}}{v_{\text{left,lead}}} \\
\left(\frac{L}{V}\right)_{\text{leading truck, right rail}} &= \frac{f_{\text{lead}} + \frac{P_{\text{tot}}}{2}}{v_{\text{right,lead}}} \\
\left(\frac{L}{V}\right)_{\text{trailing truck, left rail}} &= \frac{f_{\text{trail}} + \frac{P_{\text{tot}}}{2}}{v_{\text{left,trail}}} \\
\left(\frac{L}{V}\right)_{\text{trailing truck, right rail}} &= \frac{f_{\text{trail}} + \frac{P_{\text{tot}}}{2}}{v_{\text{right,trail}}}
\end{aligned} \tag{35}$$

Locomotive Model

A locomotive has the same modeling assumptions and constraints put forth in the “Car Model” section on page 51. In addition, a locomotive has the ability to produce tractive effort and dynamic braking, as discussed in the following subsections.

Tractive Effort

Tractive effort is the force that results at the track-locomotive interface due to the torque applied to the axles of a locomotive by its motors. Tractive effort is a function of the velocity of a locomotive. This function is continuous and consists of piecewise-smooth, spline-interpolated segments.

Typically, manufacturers provide empirical curves showing the relationship between speed and tractive effort for each throttle position of a locomotive model. If this empirical tractive effort data is not readily available, it can be approximated using Equation 36. f_{te} is tractive effort in pounds, v is velocity in mph, P is power in units of horsepower, and E is a constant representing the efficiency of the electromechanical drive system, which is typically in the range from 0.80 to 0.85 for diesel-electric locomotives.

$$f_{te} = \frac{375 P E}{v} \tag{36}$$

Equation 36 is a physical equation (with the exception of the empirical efficiency factor

E), as it is dividing power P by velocity v in order to obtain force f_{te} . The 375 factor is needed to convert power P from units of horsepower to units of pounds multiplied by miles per hour.

A typical diesel-electric locomotive has eight throttle positions, as well as an idle position. f_{te} in Equation 36 represents the tractive effort for the highest throttle position (position 8). Tractive effort for the other throttle positions can be estimated by simply scaling. For example, tractive effort for throttle position 7 can be estimated by multiplying Equation 36 by a factor of 7/8. Similarly, tractive effort for throttle position 6 can be estimated by multiplying Equation 36 by a factor of 6/8, and so forth.

Dynamic Brake

As was the case with tractive effort, dynamic braking is a function of the velocity of a locomotive. This function is continuous and consists of piecewise smooth, spline-interpolated segments.

Typically, manufacturers provide empirical curves showing the relationship between speed and dynamic braking for multiple throttle positions of a locomotive model. If this empirical tractive effort data is not readily available, it can be approximated using Equation 36, as was the case with tractive effort discussed in the previous subsection.

Rail Vehicle Hand Brake Model

The hand brake model for a rail vehicle (car or locomotive) is a simplified model that applies a retarding force proportional to the weight of the car. This retarding force is assumed to be constant and therefore is independent of the speed of the car. The proportionality factor used in TPS is entered by the user, and must be in the range from 0.01 to 0.2, as shown in Table 5 on page 11. This constant ratio set by the user is an obvious simplification and is meant to simply provide some means for estimating the effect of an applied handbrake to the dynamics of the train consist.

Locomotive Independent Brake Model

The locomotive independent brake model is a simplified model. Based on the setting input by the user for the locomotive independent brake function (see “Locomotive Operator” section on page 19), the locomotive brake cylinder pressure is calculated using the function represented as the solid orange plot in Figure 52 which defines the locomotive brake cylinder pressure as a function of the automatic brake valve setting. This orange plot was chosen since it closely resembles a car’s brake cylinder pressure,

which is a function of the local brake pipe pressure and is represented as the dashed blue plot in Figure 52. Note that in the case of the dashed blue plot, the horizontal axis should actually be labeled “Local Brake Pipe Pressure (psi)”. The sudden jump in the blue dashed plot at approximately 78 psi is the result of the car going into emergency braking and the orifice between the emergency reservoir and the brake cylinder opening up, which results in a sharp increase in the brake cylinder pressure.

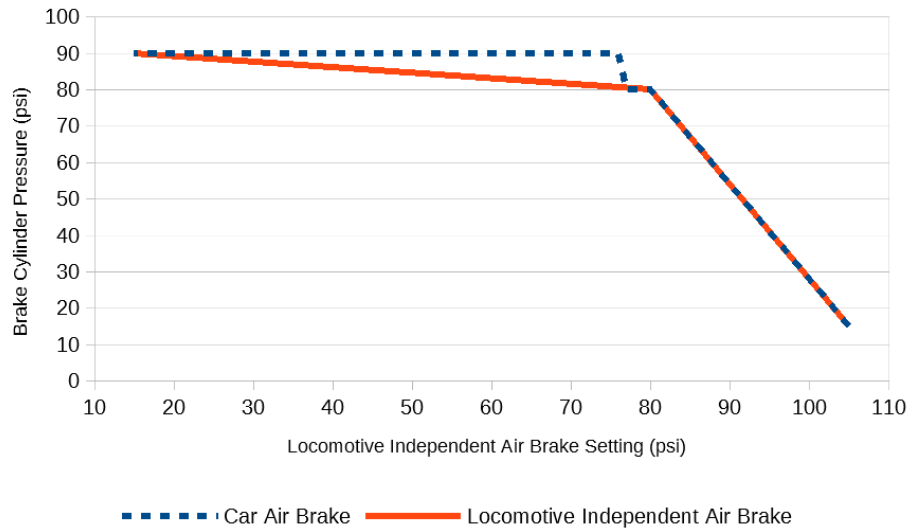


Figure 52. Locomotive independent air brake setting and resulting brake cylinder pressure

The orange plot shown in Figure 52 is defined in the constructor definition for the `ControlValve_Locomotive` class should the user wish to modify this function. Specifically, the values for the arrays named `brakeCylinderPressureFunction_Domain` and `brakeCylinderPressureFunction_Range` would have to be modified.

It is assumed that this brake cylinder is the same size brake cylinder as a typical car brake cylinder (see “Brake Cylinder and Rigging” section on page 84). The brake rigging leverage ratio is adjusted to a high (and unrealistic) value in order to achieve the desired maximum net braking ratio (see “Brake Rigging Leverage Ratio” section on page 90). The fact that a single car brake cylinder is assumed for a locomotive which results in an unrealistic required brake rigging leverage ratio is largely irrelevant since it is simply a numerical exercise performed in the TPS source code in order to calculate the brake shoe normal force (see “Normal Force Between Brake Shoes and Wheels” section on page 85) and retarding brake force (see “Retarding Brake Force” section on page 87).

Locomotive Automatic Brake Valve Model

The locomotive automatic brake valve model is detailed in Abdol-Hamid (1986) and Specchia et al (2012). A model of the locomotive automatic brake valve model is not included in TPS. Rather, the automatic air brake setting that the user defines as part of a locomotive operator (see “Locomotive Operator” section on page 19) simply acts as the air pressure boundary condition for the brake pipe model which is described in detail in the section titled “Brake Pipe Model” beginning on page 91.

In the TPS source code, transitions from one automatic air brake setting to another are not instantaneous but rather are gradual. This gradual change from one automatic air brake setting to another is represented by the variable `idealizedRelayValvePressure` which is a member of the `Locomotive` class. Calculation of this variable is performed in the `calc_idealizedRelayValvePressure` function which is also a member of the `Locomotive` class. By examining the code in this function, one can determine that the `idealizedRelayValvePressure` variable is changed at a rate of 2 psi per second for a service brake application or service brake release. In the TPS source code, this rate of change of 2 psi per second is represented by the `ABVROC_SB` variable in the `Locomotive` class. For an emergency brake application (which occurs when the user sets the automatic air brake function to 15 psi as detailed in the “Locomotive Operator” section beginning on page 19), the `idealizedRelayValvePressure` variable is decreased at a rate of 20 psi per second. In the TPS source code, this value is represented by the `ABVROC_EB` variable in the `Locomotive` class.

Car Air Brake System Model

Car Control Valve

Each car has a control valve that controls brake application and release. The control valve is often called a triple valve as it has three primary functions: charging the two reservoirs (auxiliary and emergency), applying the brakes, and releasing the brakes. In order to perform these functions, the control valve controls the flow of air between components of the air brake system, including the brake pipe, auxiliary reservoir, emergency reservoir, and brake cylinder.

There are several control valves commonly in use in the railroad industry, such as the ABD valve (U.S. Patent No. 3,175,869 to W. B. Kirk), the ABDW valve (U.S. Patent No. 3,716,276 to Wilson et al), and newer valves that include improvements to the ABDW valve (U.S. Patent No. 5,118,166 to Panebianco). Due to the number of control valve

types being used and the changes and improvements that are continually being made, models for specific control valves are not currently included in TPS. Instead, a simplified control valve is modeled. In the TPS source code, this simplified car control valve model is implemented in the `ControlValve_Car` class.

Mass Flow Rates as a Function of the Mode

The basic car control valve modeled in TPS has four primary modes of operation:

- Equilibrium mode
- Service brake application
- Emergency brake application
- Brake release mode

The equilibrium mode (commonly known as the “lap mode”) results at the end of each of the other three modes once an equilibrium condition is achieved. By “equilibrium condition”, what is meant is a condition where there is no air flow between various components of the car brake system.

Each of the other four modes is characterized by one or more air flows between car brake system components. Table 23, Table 24, and Table 25 are mass flow rate tables for the service brake application mode, emergency brake application mode, and brake release mode, respectively. Subscript abbreviation *a* stands for atmosphere, *x* stands for auxiliary reservoir, *e* stands for emergency reservoir, *c* stands for brake cylinder, and *bp* stands for brake pipe. The mass flow rates \dot{m} in the three tables can be calculated using the mass flow rate equation detailed in Appendix B. The subscript convention is explained in Appendix B but will be repeated here for convenience. Consider $\dot{m}_{x,bp}$, for example.

$\dot{m}_{x,bp}$ is the flow from the brake pipe into the auxiliary reservoir. Therefore, $\dot{m}_{x,bp}$ is positive if the air pressure in the brake pipe is greater than the air pressure in the auxiliary reservoir. Similarly, $\dot{m}_{e,bp}$ is the flow from the brake pipe to the emergency reservoir, and so on.

It can be seen from Table 23 that there is only one mass flow in a service brake application mode. This is a positive mass flow from the auxiliary reservoir to the brake cylinder.

Table 23. Mass flows for service brake application mode

$\dot{m}_{x,bp} = 0$
$\dot{m}_{e,bp} = 0$
$\dot{m}_{c,x} \geq 0$
$\dot{m}_{c,e} = 0$
$\dot{m}_{c,a} = 0$
$\dot{m}_{bp,a} = 0$

In the case of an emergency brake application, Table 24 shows that there are three mass flows. There are positive mass flows from the auxiliary reservoir to the brake cylinder and from the emergency reservoir to the brake cylinder. There is also a negative mass flow from the atmosphere to the brake pipe, or to state it more clearly, there is a positive mass flow from the brake pipe to the atmosphere.

Table 24. Mass flows for emergency brake application mode

$\dot{m}_{x,bp} = 0$
$\dot{m}_{e,bp} = 0$
$\dot{m}_{c,x} \geq 0$
$\dot{m}_{c,e} \geq 0$
$\dot{m}_{c,a} = 0$
$\dot{m}_{bp,a} \leq 0$

In the case of brake release, Table 25 shows that there are three mass flows. There is a positive mass flow from the brake pipe to the auxiliary reservoir. There is also a negative mass flow from the atmosphere to the brake cylinder, or in clearer terms, there is a positive mass flow from the brake cylinder to atmosphere. The third mass flow $\dot{m}_{e,bp}$ is the mass flow from the brake pipe to the emergency reservoir and is shown in the table as not being equal to zero. In the case of brake release after an emergency brake application, $\dot{m}_{e,bp}$ would be a positive mass flow from the brake pipe to the emergency reservoir; in other words, $\dot{m}_{e,bp}$ would be greater than zero. However, in the case of brake release after a service brake application, $\dot{m}_{e,bp}$ would be less than zero; in other words, there would be a positive mass flow from the emergency reservoir to the brake pipe. This positive flow from the emergency reservoir to the brake pipe during a service brake release may not be representative of real-world control valves such as the ABD and ABDW control valves. Such control valves have an additional “quick release” reservoir that vents to the brake pipe to keep the brake pipe’s air pressure elevated which allows for

faster service brake release. Rather than including this quick release reservoir, the basic car control valve modeled in TPS allows the emergency reservoir to vent to the brake during the service brake release mode.

Table 25. Mass flows for brake release mode

$\dot{m}_{x,bp} \geq 0$
$\dot{m}_{e,bp} \neq 0$
$\dot{m}_{c,x} = 0$
$\dot{m}_{c,e} = 0$
$\dot{m}_{c,a} \leq 0$
$\dot{m}_{bp,a} = 0$

In the TPS source code, these mass flow rates are represented by the variables `mdot_ar_bp`, `mdot_er_bp`, `mdot_bc_ar`, `mdot_bc_er`, `mdot_bc_atm`, and `mdot_bp_atm`, all of which are members of the `ControlValve_Car` class.

Mode Thresholds

The previous section put forth tables (Table 23, Table 24, and Table 25) outlining the mass mass flow rates based on the present mode. This section puts forth the requirements for triggering a particular mode. In the TPS source code, the current operating mode is calculated in the `calc_currentOperatingMode` function which is a member of the `ControlValve_Car` class.

The difference between the auxiliary reservoir pressure p_x and local brake pipe p_{bp} pressure is the metric that triggers mode changes. Table 26 shows four modes and the condition necessary for triggering the mode.

Table 26. Triggering condition for the four modes of the simplified car control valve assumed in TPS

Mode	Triggering Condition
Lap	$ p_x - p_{bp} < 0.25 \text{ psi}$
Service Brake Application	$p_x - p_{bp} > 0.75 \text{ psi}$
Emergency Brake Application	$p_x - p_{bp} > 2.75 \text{ psi}$
Release After Service Brake Application	$p_{bp} - p_x > 1.75 \text{ psi}$

In the TPS source code, values of the psi thresholds shown in the right-hand column of Figure 58 are defined by the variables LMIT, SBAMIT, EBAMIT, and SBRMIT, all of which are members of the ControlValve_Car class.

Brake Cylinder and Rigging

The differential equations governing the brake cylinder's piston displacement u_p and pressure p_c are discussed by Afshari et al (2012) and shown in Equation 37. M_p is the mass of the piston, P_a is atmospheric pressure, A_p is the cross-sectional area of the piston, K_c is the spring stiffness, L_c is the spring pre-load force, F_f is the friction force, f_p is the piston axial force, U_{con} is the piston displacement when contact is made between the brake shoes and the wheels (also known as “slack piston travel”), V_0 is the initial volume of the cylinder when the piston is at rest, A_c is the cross-sectional area of the cylinder, R_g is the specific gas constant of air, Θ is the air temperature, and h is the Heaviside step function.

$$\begin{aligned} M_p \frac{d^2 u_p(t)}{dt^2} &= (p_c(t) - P_a) A_p - K_c u_p(t) - L_c - F_f - f_p(t) h(u_p(t) - U_{con}) \\ \frac{dp_c(t)}{dt} &= \frac{1}{V_0 + A_c u_p(t)} \left(R_g \Theta \frac{dm_c(t)}{dt} - p_c(t) A_c \frac{du_p(t)}{dt} \right) \end{aligned} \quad (37)$$

Note that in TPS, a 10.0 inch diameter brake piston is assumed which results in a value of approximately 78.5 square inches for A_p . In addition, a 7.0 inch piston travel is assumed which results in a value of 7.0 for U_{con} .

If the independent value provided to the Heaviside step function h is greater than or equal to zero, h has a value of one. Otherwise h is equal to zero. Therefore, in the first equation in Equation 37, h is equal to zero when u_p is less than U_{con} and is equal to one when u_p is greater than or equal to U_{con} .

dm_c/dt is the mass flow rate into the brake cylinder. The mass flow rate between the brake cylinder and another component can be calculated using Equation 161 detailed in Appendix B.

The two coupled Equations 37 can be simplified into a single equation by making the assumption that the piston is fixed at the point U_{con} where contact is made between the brake shoes and wheels. This results in a fixed volume for brake cylinder, and the dynamics of the brake piston are no longer applicable. Therefore, the two Equations 37 are reduced to the single Equation 38.

$$\frac{dp_c(t)}{dt} = \frac{1}{V_0 + A_c U_{con}} \left(R_g \Theta \frac{dm_c(t)}{dt} \right) \quad (38)$$

In the TPS source code, Equation 38 is implemented in the `calc_pressure` function which is a member of the `BrakeCylinder` class.

Piston Force

Once the brake shoes contact the wheels, the acceleration and velocity of the piston is approximately equal to zero, and the Heaviside step function is equal to one (Afshari et al 2012). Therefore, in the first equation in Equations 37 set the term $d^2u_p(t)/dt^2$ equal to zero, the term $h(u_p(t) - U_{con})$ equal to one, and solve for the piston force f_p .

$$f_p(t) = (p_c(t) - P_a)A_p - K_c U_{con} - L_c \quad (39)$$

In Equation 39, when the first term $((p_c(t) - P_a)A_p)$ is greater than the sum of the second term involving the spring constant and the third term involving the spring preload force, the piston force f_p becomes positive which indicates that contact has been made between the brake shoes and wheels. The force that results between the brake shoes and wheels is discussed in the next section.

In the TPS source code, Equation 39 is implemented in the `calc_pistonForce` function which is a member of the `BrakeCylinder` class.

Normal Force Between Brake Shoes and Wheels

The total normal brake force for a car is the normal force applied by all the brake shoes to the respective wheels. The total normal brake force f_n is a function of the piston force f_p .

$$f_n = L\eta f_p \quad (40)$$

f_n is the total normal brake force, f_p is the piston force, L is the brake rigging leverage ratio, and η is the brake rigging efficiency. The brake rigging leverage ratio L is a constant magnification factor that is produced by the brake rigging configuration. The calculation of this constant L is put forth in the subsection titled “Brake Rigging Leverage Ratio” on page 90. The brake rigging efficiency η is defined as a function of the pressure p_c in the brake cylinder. This brake rigging efficiency function is continuous and consists of piecewise smooth, spline-interpolated segments and is one of the functions associated with the definition of a car (or locomotive) in TPS, as shown in Table 7 on page 12.

Low et al (1977) puts forth the following equation for conventional freight braking rigging efficiency η as a function of the brake cylinder pressure p_c in psi units.

$$\eta = \frac{\sqrt{\ln(p_c - 6.0)}}{3.2} \quad (41)$$

Figure 53 shows a plot resulting from plugging in numerical values for p_c in Equation 41.

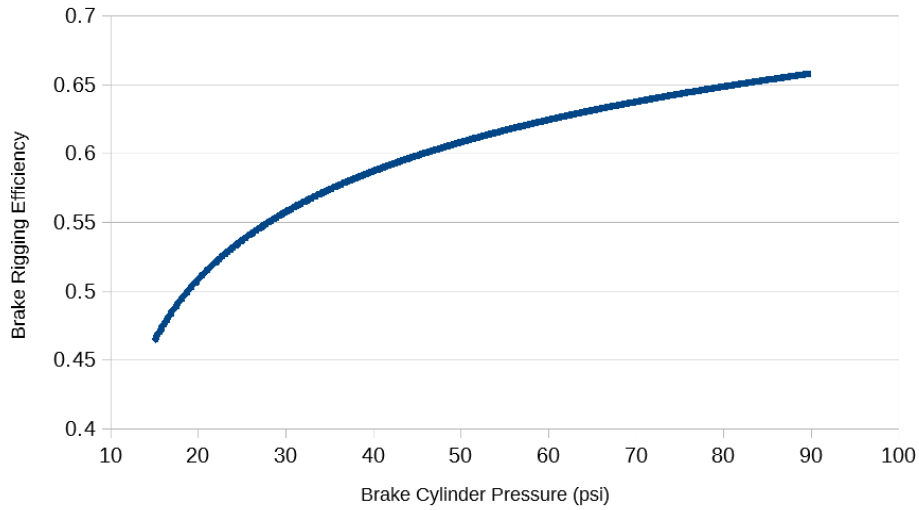


Figure 53. Plot of brake rigging efficiency as a function of brake cylinder pressure

In Low et al (1977), it states that the units for p_c in Equation 41 are psi. However, there are indications that the units may actually be psig. If this is the case, then p_c in units of psi can be converted to psig units by subtracting 15.0 psi since atmospheric pressure is approximately 15.0 psi. This is shown in Equation 42.

$$\eta = \frac{\sqrt{\ln((p_c - 15.0) - 6.0)}}{3.2} \quad (42)$$

The user can decide whether Equation 41 or Equation 42 is correct by doing their own research, or they could choose to use an entirely different function. Several discrete values from the user's chosen function can be plugged into the first function in the definition of a car, as shown in Figure 9 on page 15.

In the TPS source code, calculation of the normal force between the brake shoes and the

wheels is implemented in the `calc_normalShoeWheelForce` function which is a member of the `BrakeCylinder` class.

Retarding Brake Force

The total retarding brake force f_r for a car is a function of the normal brake force f_n .

$$f_r = \mu_{bs} f_n \quad (43)$$

f_r is the retarding brake shoe force, f_n is the normal brake shoe force, and μ_{bs} is the brake shoe friction coefficient. The brake shoe friction coefficient μ_{bs} is defined as a function of the velocity of the car. This brake shoe friction coefficient function is continuous and consists of piecewise smooth, spline-interpolated segments and is one of the functions associated with the definition of a car (or locomotive) in TPS, as shown in Table 7 on page 12.

Low et al (1977) puts forth Equation 44 and Equation 45 for the brake shoe friction coefficient of a plain composition brake shoe. Equation 44 applies to situations where the temperature of the wheel is less than 420 degrees Fahrenheit, and Equation 45 applies when the temperature of the wheel is greater than 420 degrees Fahrenheit. In both equations, v is the velocity of the car in miles per hour, and t_{bs} is the brake shoe temperature in degrees Fahrenheit.

$$\mu_{bs} = \frac{0.64}{v^{0.15} + \frac{0.1}{v}} - 0.0000025(t_{bs} - 200.0)^2 \quad (44)$$

$$\mu_{bs} = \frac{0.64}{v^{0.15} + \frac{0.1}{v}} - 0.00816(t_{bs} - 200.0)^{0.5} \quad (45)$$

Figure 54 shows a plot resulting from assuming a brake shoe temperature of 300 degrees Fahrenheit and plugging in numerical values for v in Equation 44. Several discrete values from this function could be plugged into the second function in the definition of a car, as shown in Figure 9 on page 15.

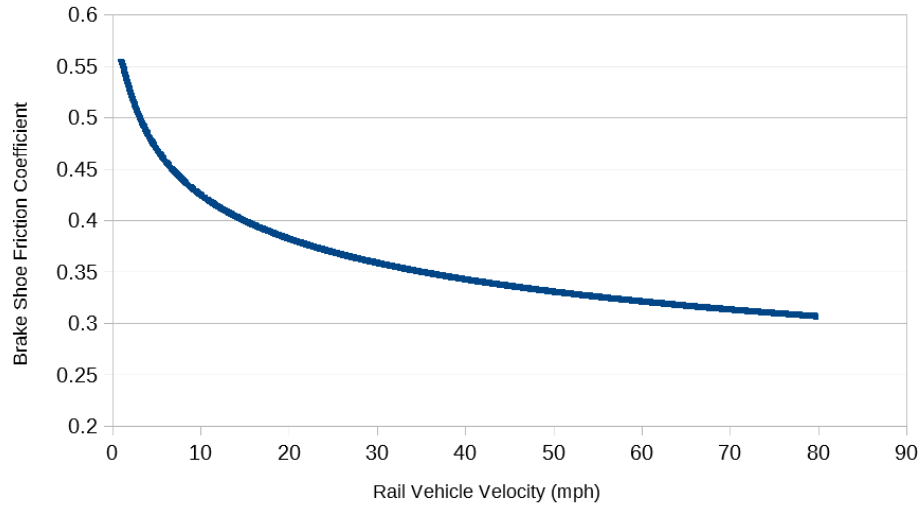


Figure 54. Plot of brake shoe friction coefficient as a function of car velocity

As shown in Equation 44 and Equation 45, the brake shoe friction coefficient is typically a function of the temperature of the brake shoe. However, TPS currently does not incorporate brake shoe temperature modeling and only allows the user to define the brake shoe friction coefficient as a function of car velocity. Incorporating brake shoe temperature modeling would not be difficult. Appendix C puts forth a brake shoe temperature model presented in Low et al (1977). In addition, Appendix C puts forth a secondary brief physical analysis of the equation governing the temperature of a lumped parameter body (such as a brake shoe). This equation or a similar type of equation could be used for modeling brake shoe temperatures and could be incorporated into TPS in the future.

In the TPS source code, calculation of the retarding brake force is implemented in the `calc_retardingBrakeForce` function which is a member of the `BrakeCylinder` class.

Maximum Brake Cylinder Pressure Under Full-Service Braking

In order to calculate the brake rigging ratio needed to achieve the maximum desired net braking ratio, which is a user input (see Table 5 on page 11), the maximum brake cylinder pressure under full-service braking must be calculated. This maximum attainable brake cylinder pressure under full service braking can be calculated using Boyle's law which assumes a constant gas temperature and states that the initial gas pressure P_0 multiplied by the initial volume V_0 is equal to the final gas pressure P_f multiplied by the final pressure V_f as shown in Equation 46.

$$P_0 V_0 = P_f V_f \quad (46)$$

The initial pressures and volumes (left-hand side of Equation 46) are the steady-state brake release pressures and volumes, and the final pressures and volumes (right-hand side of Equation 46) are the steady-state emergency braking pressures and volumes. To show this, the left-hand side and right-hand side of Equation 46 are restated in Equation 47.

$$P_c V_c + P_x V_x + P_{pipe} V_{pipe} = P_{equil} (V_c + V_x + V_{pipe}) \quad (47)$$

Descriptions of each variable and their assumed values are put forth in Table 27.

Table 27. Explanation of variables in Equation 47

Variable	Description	Value
P_c	Steady-state brake cylinder pressure under brake release application	15.0 psi ¹²
V_c	Brake cylinder volume	700.0 in ³
P_x	Steady-state auxiliary reservoir pressure under brake release application	105.0 psi
V_x	Auxiliary reservoir volume	2500.0 in ³
P_{pipe}	Steady-state piping pressure under brake release application	15.0 psi
V_{pipe}	Sum of piping volume between auxiliary reservoir and control valve and piping volume between brake cylinder and control valve	310 in ³
P_{equil}	Steady-state pressure (in brake cylinder, auxiliary reservoir, and piping) under full service brake application	(To be calculated)

Now all the variables in Equation 47 are known except for P_{equil} . To determine the value of P_{equil} , solve Equation 47 for P_{equil} .

$$P_{equil} = \frac{P_c V_c + P_x V_x + P_{pipe} V_{pipe}}{V_c + V_x + V_{pipe}} \quad (48)$$

Plugging in all the known values in the right-hand side of Equation 48 results in a value of approximately 79.1 psi for P_{equil} . This value is the maximum attainable brake cylinder pressure under a full service brake application.

In the TPS source code, P_{equil} in Equation 48 is represented by the variable `maxPressureForFullServiceBraking` which is a member of the `BrakeCylinder` class. This

¹² This is the approximate value of atmospheric pressure at sea level. A more precise value is 14.7 psi.

variable is calculated in the `calc_maxPressureForFullServiceBraking` which is also a member of the `BrakeCylinder` class.

Brake Rigging Leverage Ratio

The maximum net braking ratio R_n is equal to the maximum normal force F_n between the brake shoes and wheels divided by the car weight W , as shown in (49).

$$R_n = \frac{F_n}{W} \quad (49)$$

The maximum normal force F_n is equal to the brake rigging leverage ratio L multiplied by the maximum piston force F_p , as shown Equation 50.

$$F_n = L F_p \quad (50)$$

Plug the value of F_n in Equation 50 into the formula for R_n shown in Equation 49.

$$R_n = \frac{L F_p}{W} \quad (51)$$

Solve Equation 51 for the brake rigging leverage ratio L , as shown in Equation 52.

$$L = \frac{R_n W}{F_p} \quad (52)$$

Based on Equation 39 in the subsection titled “Piston Force” on page 85, the equation for the maximum piston force F_p is put forth in Equation 53.

$$F_p = (P_c - P_a) A_p - K_c U_{con} - L_c \quad (53)$$

In Equation 53, P_c is the maximum brake cylinder pressure which is put forth in Equation 48. Note that in Equation 48 the maximum brake cylinder pressure is labeled P_{equil} rather than P_c . Substitute the right-hand side of Equation 48 for the constant P_c in Equation 53, as shown in Equation 54.

$$F_p = \left(\frac{P_c V_c + P_x V_x + P_{pipe} V_{pipe}}{V_c + V_x + V_{pipe}} - P_a \right) A_p - K_c U_{con} - L_c \quad (54)$$

Plug the right-hand side of Equation 54 into Equation 52.

$$L = \frac{R_n W}{\left(\frac{P_c V_c + P_x V_x + P_{pipe} V_{pipe}}{V_c + V_x + V_{pipe}} - P_a \right) A_p - K_c U_{con} - L_c} \quad (55)$$

In TPS, Equation 55 is used to calculate the ideal brake rigging leverage ratio. In the TPS source code, this calculation is implemented in the `calc_brakeRiggingLeverageRatio` function which is a member of the `RailVehicle` class.

Brake Pipe Model¹³

In TPS, it is assumed that the brake pipe of every rail vehicle has a diameter of 1.25 inches. This value is stored in the `DIAMETER` variable. This variable is a member of the `BrakePipe_FiniteElement` class which will be discussed in more detail in the following paragraphs. Furthermore, it is assumed that for every rail vehicle, its brake pipe length is 10% longer than the car length entered by the user (see Table 5 on page 11). This is due to the slack in the brake pipe and the fact that the brake pipe may not run directly from one end of the car to the other due to obstructions from other car components. This ratio of the brake pipe length over the car length is stored in the `BRAKE_PIPE_LENGTH_OVER_CAR_LENGTH` variable which is a member of the `RailVehicle` class. The length of the brake pipe is stored in the `brakePipeLength` variable and is calculated in the function `calc_brakePipeLength` both of which are also members of the `RailVehicle` class.

Clearly, in the real world every rail vehicle has a brake pipe. In TPS, however, there is no `BrakePipe` class. In other words, every rail vehicle includes a brake cylinder created from the `BrakeCylinder` class, and every rail vehicle includes two couplers created from the `Coupler` class. However, rail vehicles in TPS do not include a brake pipe created from a `BrakePipe` class. Rather, there exists a variable `brakePipeLength` in the `RailVehicle` class which stores the length of the respective rail vehicle's brake pipe as explained in the previous paragraph.

Instead of having a brake pipe object associated with each rail vehicle, TPS includes a `BrakePipe_Cumulative` class. This class refers to a cumulative brake pipe which abstractly consists of the brake pipes of multiple rail vehicles. In TPS, a train consist includes a vector of cumulative brake pipes, which is represented by the `brakePipes_Cumulative` variable, which is a member of the `TrainConsist` class.

A cumulative brake pipe stretches from one locomotive to another, if there are one or

¹³ The brake pipe formulas detailed in this section were originally developed by Abdol-Hamid (1986).

more cars between the locomotives; however, a cumulative brake pipe does not exist between adjacent locomotives. A cumulative brake pipe can also stretch from a car at the end of the train to the nearest locomotive. Such examples of cumulative brake pipes are shown in Figure 55. Note that “Loc” in Figure 55 refers to a locomotive. This example train consist consists of two cumulative brake pipes.

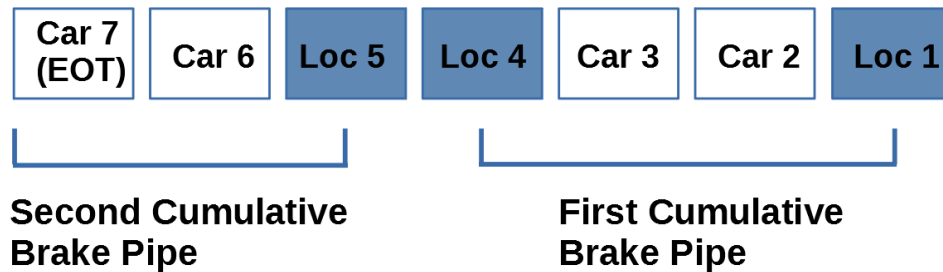


Figure 55. Example configuration of cumulative brake pipes in a train consist

The creation of cumulative brake pipes clearly depends on the makeup of the train consist and, specifically, the number of locomotives and their positions in the train consist. The creation of cumulative brake pipes for a train consist is performed in the `calc_brakePipes_Cumulative` function which is a member of the `TrainConsist` class.

A cumulative brake pipe consists of multiple smaller segments which will be referred to as brake pipe finite elements since these smaller discrete segments are used in a finite element formulation for solving the governing brake pipe differential equations. In TPS, these brake pipe finite elements are implemented in the `BrakePipe_FiniteElement` class. A brake pipe finite element consists of a maximum of three adjacent rail vehicles. This maximum value of three is stored in the `MAX_RAIL_VEHICLES_PER_BRAKE_PIPE_FE` variable which is a member of the `BrakePipe_FiniteElement` class. For example, if a cumulative brake pipe spanned 12 rail vehicles, then there would be four brake pipe finite elements, each including three rail vehicles. However, if a cumulative brake pipe spanned 11 rail vehicles, then there would still be four brake pipe finite elements, but the fourth brake pipe finite element would only consist of two rail vehicles. This vector of brake pipe finite elements is represented by the `brakePipe_FiniteElements` variable in the `BrakePipe_Cumulative` class. The calculation of this vector of brake pipe finite elements for a given cumulative brake pipe is performed in the `calc_brakePipe_FiniteElements` function which is also a member of the `BrakePipe_Cumulative` class.

The brake pipe finite element length calculation is performed in the `calc_brakePipeFELength` function which is a member of the `BrakePipe_FiniteElement`

class. If all of the rail vehicles associated with a brake pipe finite element are cars, then the length of the brake pipe finite element is simply the sum of the brake pipes of the cars. However, if the brake pipe finite element includes a locomotive, then only one half of the locomotive's brake pipe length is added to the brake pipe finite element length. This is shown visually in Figure 56 where the droopy lines are meant to represent brake pipes associated with each car or locomotive.

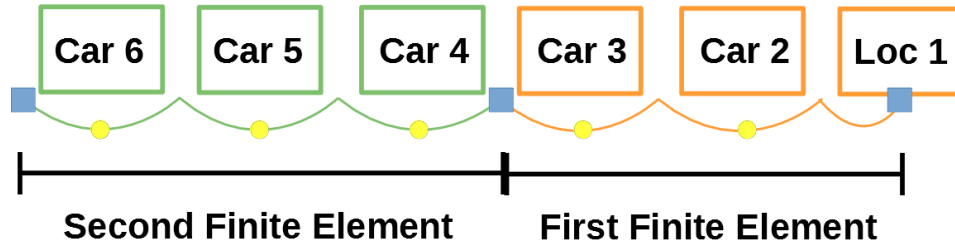


Figure 56. Example brake pipe finite element configurations

The following subsections will describe the finite element formulation of solving the brake pipe differential equations. This finite element formulation will solve for the brake pipe air pressure and velocity at discrete nodes which exist at the end of every third rail vehicle, as described above. Such nodes are represented by the three small blue squares in Figure 56. Note that the significance of the yellow dots in Figure 56 will be explained in the “Linear Interpolation” section beginning on page 117.

The above paragraphs can be largely summarized by stating that based on the positioning of locomotives in a train consist, a vector of cumulative brake pipes is created, and then a vector of brake pipe finite elements is created for each cumulative brake pipe.

Governing Differential Equations

The governing differential equations for the brake pipe are a pair of first order, coupled partial differential equations. These two equations are known as the continuity equation and the momentum equation. The derivation of these two equations is presented in Abdol-Hamid (1986).

Continuity Equation

The continuity equation is shown in Equation 56.

$$(A + v) \frac{\partial \rho}{\partial t} + \frac{\partial (\rho u A)}{\partial x} - \eta = 0 \quad (56)$$

In Equation 56, the variables t and x are the independent variables time and space,

respectively. The variables ρ and u are the dependent variables density and velocity, respectively. A is the cross-sectional area of the brake pipe, and v is the branch pipe volume per unit length. η is the mass flow into the brake pipe. If the mass flow is out of the brake pipe (in other words, if the pressure in the brake pipe is decreasing), then η is negative.

The branch pipe volume v is assumed to be negligibly small compared to the brake pipe cross-sectional area A . Therefore, the branch pipe volume v can be removed from Equation 56, as shown in Equation 57.

$$A \frac{\partial \rho}{\partial t} + \frac{\partial (\rho u A)}{\partial x} - \eta = 0 \quad (57)$$

Introduce a dummy variable m which is equal to the density ρ multiplied by the velocity u , as shown in Equation 58.

$$\rho u = m \quad (58)$$

Plug Equation 58 into Equation 57, as shown in Equation 59.

$$A \frac{\partial \rho}{\partial t} + \frac{\partial (mA)}{\partial x} - \eta = 0 \quad (59)$$

Isothermal flow is assumed, and therefore, there is a linear relationship between air density and air pressure, as shown in Equation 60. R_g is the gas constant of air, which is equal to 287.0 joules / (kilograms * kelvin). θ is the air temperature, which is assumed to be constant.

$$\rho = \frac{p}{R_g \theta} \quad (60)$$

Plug in Equation 60 into Equation 59. This results in Equation 61.

$$A \frac{\partial \left(\frac{p}{R_g \theta} \right)}{\partial t} + \frac{\partial (mA)}{\partial x} - \eta = 0 \quad (61)$$

Since R_g and θ are constants with respect to time, they can be moved outside of the partial differential term in Equation 61. This results in Equation 62.

$$\frac{A}{R_g \theta} \frac{\partial p}{\partial t} + \frac{\partial(m A)}{\partial x} - \eta = 0 \quad (62)$$

Equation 62 is the form of the continuity equation that will be used in the finite element formulation to be discussed in the coming sections.

Momentum Equation

The momentum equation is shown in Equation 63.

$$A \frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2 A)}{\partial x} + A \frac{\partial p}{\partial x} + F_s = 0 \quad (63)$$

In Equation 63, the variables t and x are the independent variables time and space, respectively. The variables ρ , u , and p are the dependent variables density, velocity, and pressure, respectively. A is the brake pipe area. F_s represents the shear force due to the wall of the brake pipe. The formula for F_s is shown in Equation 64.

$$F_s = \tau \pi \bar{d} \quad (64)$$

In Equation 64, \bar{d} represents the hydraulic diameter and is given by the formula in Equation 65.

$$\bar{d} = \sqrt{\frac{4}{\pi} A_{avg}} \quad (65)$$

For a circular cross-section, the hydraulic diameter \bar{d} is equivalent to the regular diameter d . Therefore, Equation 64 can be re-written as shown in Equation 66.

$$F_s = \tau \pi d \quad (66)$$

Equation 67 puts forth the formula for τ in Equation 64.

$$\tau = f \frac{\rho u^2}{8} \left[\frac{u}{|u|} \right] \quad (67)$$

In Equation 67, f represents the wall friction factor and is a function of the Reynolds number R_e , as shown in Equation 68. This formula states that the wall friction factor is obtained by raising the Reynolds number to the power b and then multiplying by a .

$$f = a R_e^b \quad (68)$$

The Reynolds number R_e in Equation 68 can be calculated using Equation 69.

$$R_e = \frac{\rho u d}{\mu} \quad (69)$$

In Equation 69, ρ is the local brake pipe air density, u is the local brake pipe air velocity, d is the local brake pipe diameter, and μ is the air dynamic viscosity. Air dynamic viscosity values are available in the technical literature or on several websites on the internet.¹⁴ Air dynamic viscosity is a function of the air pressure and air temperature. Remember that air temperature is inputted by the user (see Table 14 on page 24). However, the dependence on pressure in the small range of brake pressures from 15 psi to 105 psi (see “Locomotive Operator” section on page 19) is not great, so a constant atmospheric pressure is assumed. Therefore, it is assumed that the air dynamic viscosity is only a function of the air temperature.

In Equation 68, the variables a and b depend on the Reynolds number. These values of a and b are put forth in Table 28.

Table 28. Values of wall friction factor coefficients

Reynolds Number	Flow Regime	a	b
< 2,000	Laminar	64.0	-1.0
>= 2,000 and < 4,000	Transition	0.000137	0.717
>= 4,000 and < 40,000	Turbulent	0.13977	-0.11781
>= 40,000	Turbulent Limit	0.04	0.0

Plugging Equation 69 into Equation 68 results in Equation 70.

$$f = a \left(\frac{\rho u d}{\mu} \right)^b \quad (70)$$

Plugging Equation 70 into Equation 67 results in Equation 71.

$$\tau = a \left(\frac{\rho u d}{\mu} \right)^b \frac{\rho u^2}{8} \left[\frac{u}{|u|} \right] \quad (71)$$

14 For example, see the air dynamic viscosity data presented at the “The Engineering Toolbox” website at the following web address: https://www.engineeringtoolbox.com/air-absolute-kinematic-viscosity-d_601.html.

Plugging Equation 71 into Equation 66 results in Equation 72.

$$F_s = a \left(\frac{\rho u d}{\mu} \right)^b \frac{\rho u^2}{8} \left[\frac{u}{|u|} \right] \pi d \quad (72)$$

Next, define a dummy variable m which is equal to the density ρ multiplied by the velocity u , as shown in Equation 73.

$$\rho u = m \quad (73)$$

Plugging Equation 66 and Equation 73 into Equation 63 results in Equation 74.

$$A \frac{\partial m}{\partial t} + \frac{\partial (m u A)}{\partial x} + A \frac{\partial p}{\partial x} + \tau \pi d = 0 \quad (74)$$

Equation 74 is the form of the momentum equation that will be used in the finite element formulation to be discussed in the coming sections. Note that τ in Equation 74 is given by Equation 71.

Finite Element Formulation

This section demonstrates use of the finite element method to solve the pair of coupled differential equations shown in Equation 62 and Equation 74, which represent the continuity equation and momentum equation, respectively. Background on the finite element method, including weighted residuals and shape functions, can be found in introductory textbooks on the finite element method, such as Zienkiewicz and Morgan (2006). This finite element formulation of the continuity equation and momentum equation is put forth in Abdol-Hamid (1986).

In the TPS source code, the finite element formulation presented in the following sections is implemented in the `BrakePipe_Cumulative` class. For each cumulative brake pipe, the finite element global matrix is calculated in the `calc_systemMatrix` function, and the global forcing function is calculated in the `calc_forcingVector` function, both of which are members of the `BrakePipe_Cumulative` class. The resulting system of linear equations for each cumulative brake pipe is solved by calling the `gaussElimination` function from within the `simulate` function. This `simulate` function is a member of the `Simulation` class, and the `gaussElimination` function is a member of the `LinearSystem` class.

Weighted Residual Form of Continuity Equation

The weighted residual form of the continuity equation (Equation 62) is achieved by multiplying by a weight function w and integrating over the relevant domain, as shown in Equation 75.

$$\int_{\Omega} \left[w \frac{A}{R_g \theta} \frac{\partial p}{\partial t} + w \frac{\partial(mA)}{\partial x} - w \eta \right] dx = 0 \quad (75)$$

The time derivative term p and spatial derivative term mA in Equation 75 can be approximated as a sum of the product of the nodal values and shape functions S , as shown in Equation 76 and Equation 77. Note that NN refers to the number of nodes in the finite element mesh.

$$p_{app}(x) = \sum_{j=1}^{NN} p_j S_j(x) \quad (76)$$

$$(mA)_{app}(x) = \sum_{j=1}^{NN} (mA)_j S_j(x) \quad (77)$$

Plugging Equation 76 and Equation 77 into Equation 75 results in Equation 78.

$$\int_{\Omega} \left[w \left(\frac{A}{R_g \theta} \sum_{j=1}^{NN} \frac{dp_j}{dt} S_j \right) + w \left(\sum_{j=1}^{NN} m_j A_j \frac{dS_j}{dx} \right) - w \eta \right] dx = 0 \quad (78)$$

Next, set the weight functions w to be the same as the shape functions, as shown in Equation 79.

$$w(x) = S_i(x) \quad (79)$$

Plugging Equation 79 into Equation 78 results in Equation 80.

$$\int_{\Omega} \left[S_i \left(\frac{A}{R_g \theta} \sum_{j=1}^{NN} \frac{dp_j}{dt} S_j \right) + S_i \left(\sum_{j=1}^{NN} m_j A_j \frac{dS_j}{dx} \right) - S_i \eta \right] dx = 0 \quad (80)$$

Next, move the summation signs outside of the integral in Equation 80. This results in Equation 81.

$$\frac{A}{R_g \theta} \sum_{j=1}^{NN} \left[\frac{dp_j}{dt} \int_{\Omega} (S_i S_j dx) \right] + \sum_{j=1}^{NN} \left[m_j A_j \int_{\Omega} \left(S_i \frac{dS_j}{dx} \right) \right] - \eta \int_{\Omega} S_i dx = 0 \quad (81)$$

Weighted Residual Form of Momentum Equation

The weighted residual form of the momentum equation (Equation 74) is achieved by multiplying by a weight function w and integrating over the relevant domain, as shown in 82.

$$\int_{\Omega} \left[w A \frac{\partial m}{\partial t} + w \frac{\partial(muA)}{\partial x} + w A \frac{\partial p}{\partial x} + w \tau \pi d \right] dx = 0 \quad (82)$$

The time derivative term m , the spatial derivative term muA , and the spatial derivative term p in Equation 82 can be approximated as a sum of the product of the nodal values and shape functions S , as shown in Equation 83, Equation 84, and Equation 85. Note that NN refers to the number of nodes in the finite element mesh.

$$m_{app}(x) = \sum_{j=1}^{NN} m_j S_j(x) \quad (83)$$

$$(muA)_{app}(x) = \sum_{j=1}^{NN} (muA)_j S_j(x) \quad (84)$$

$$p_{app}(x) = \sum_{j=1}^{NN} p_j S_j(x) \quad (85)$$

Plugging Equation 83, Equation 84, and Equation 85 into Equation 82 results in Equation 86.

$$\int_{\Omega} \left[w \left(A \sum_{j=1}^{NN} \frac{dm_j}{dt} S_j \right) + w \left(\sum_{j=1}^{NN} m_j u_j A_j \frac{dS_j}{dx} \right) + w \left(A \sum_{j=1}^{NN} p_j \frac{dS_j}{dx} \right) + w(\tau \pi d) \right] dx = 0 \quad (86)$$

Next, set the weight functions w to be the same as the shape functions, as was shown previously in Equation 79. Plugging Equation 79 into Equation 86 results in Equation 87.

$$\int_{\Omega} \left[S_i \left(A \sum_{j=1}^{NN} \frac{dm_j}{dt} S_j \right) + S_i \left(\sum_{j=1}^{NN} m_j u_j A_j \frac{dS_j}{dx} \right) + S_i \left(A \sum_{j=1}^{NN} p_j \frac{dS_j}{dx} \right) + S_i(\tau \pi d) \right] dx = 0 \quad (87)$$

Next, move the summation signs outside of the integral in Equation 87. This results in

Equation 88.

$$\begin{aligned}
 & A \sum_{j=1}^{NN} \left[\frac{dm_j}{dt} \int_{\Omega} (S_i S_j dx) \right] + \sum_{j=1}^{NN} \left[m_j u_j A_j \int_{\Omega} \left(S_i \frac{dS_j}{dx} dx \right) \right] + \\
 & A \sum_{j=1}^{NN} \left[p_j \int_{\Omega} \left(S_i \frac{dS_j}{dx} dx \right) \right] + \\
 & \tau \pi d \int_{\Omega} (S_i dx) = 0
 \end{aligned} \tag{88}$$

Shape Functions

The definition for the two shape functions S on an element are given in Equation 89 and Equation 90.

$$S_1 = \frac{1}{2}(1 - \zeta) \tag{89}$$

$$S_2 = \frac{1}{2}(1 + \zeta) \tag{90}$$

In Equation 89 and Equation 90, the variable ζ ranges from -1 to 1. Then the variable x is defined as shown in Equation 91, where $x_1^{(e)}$ and $x_2^{(e)}$ are the values of x at the first and second nodes, respectively, of element e .

$$x = \frac{h^{(e)}}{2} \zeta + \frac{x_1^{(e)} + x_2^{(e)}}{2} \tag{91}$$

For use in future calculations, the derivative of x in Equation 91 with respect to the variable ζ is shown in Equation 92.

$$\frac{dx}{d\zeta} = \frac{h^{(e)}}{2} \tag{92}$$

Integral Evaluations for Finite Element Mass Matrix

The mass matrix for each finite element must be calculated. The elements of the mass matrix are determined by the integral term in Equation 81 and Equation 88 containing the $S_i S_j$ term. This integral must be evaluated for the four cases shown in Table 29.

Table 29. Four cases for determining finite element mass matrix

Index i	Index j
1	1
1	2
2	1
2	2

The integral in Equation 93 is the integral which must be evaluated with the four cases shown in Table 29. The values for S_1 and S_2 are given in Equation 89 and Equation 90, respectively.

$$\int_{\Omega} S_i S_j dx \quad (93)$$

The integral in Equation 93 must be evaluated from the position of the first node $x_1^{(e)}$ to the position of the second node $x_2^{(e)}$, as shown in Equation 94.

$$\int_{x_1^{(e)}}^{x_2^{(e)}} S_i S_j dx \quad (94)$$

The integration variable in Equation 94 must be changed from x to ζ . This is achieved in Equation 95.

$$\int_{x_1^{(e)}}^{x_2^{(e)}} (S_i S_j) dx = \int_{-1}^1 (S_i S_j) \frac{dx}{d\zeta} d\zeta \quad (95)$$

Equation 92 puts forth a value for the derivative of x with respect to ζ . Plugging Equation 92 into Equation 95 results in the integral term in Equation 96.

$$\int_{-1}^1 (S_i S_j) \frac{h^{(e)}}{2} d\zeta \quad (96)$$

In Equation 96, $h^{(e)}$ is the length of the finite element and is constant. Therefore, this term can be taken out of the integral term, as shown in Equation 97.

$$\frac{h^{(e)}}{2} \int_{-1}^1 (S_i S_j) d\zeta \quad (97)$$

The formula in Equation 97 must be evaluated for the four cases shown in Table 29. For the first case, S_1 given in Equation 89 is plugged in for both S_i and S_j in Equation 97. This is shown in Equation 98.

$$\frac{h^{(e)}}{2} \int_{-1}^1 \frac{1}{2}(1-\xi) \frac{1}{2}(1-\xi) d\xi \quad (98)$$

Multiplying the terms in the integral in Equation 98 results in Equation 99.

$$\frac{h^{(e)}}{8} \int_{-1}^1 (1-2\xi+\xi^2) d\xi \quad (99)$$

Integrating the formula in Equation 99 results in the formula in Equation 100.

$$\frac{h^{(e)}}{8} \left[\left(\xi - \xi^2 + \frac{1}{3}\xi^3 \right) d\xi \right]_{-1}^1 \quad (100)$$

Plugging in the domain end point values of -1 and 1, respectively, and simplifying results in a value of $\frac{h^{(e)}}{3}$.

The same integral calculations must be performed for the other three cases shown in Table 29. The details of these calculations will not be presented here. However, the final values for each of the four cases is shown in Table 30.

Table 30. Finite element mass matrix values

Index i	Index j	Mass Matrix Value
1	1	$\frac{h^{(e)}}{3}$
1	2	$\frac{h^{(e)}}{6}$
2	1	$\frac{h^{(e)}}{6}$
2	2	$\frac{h^{(e)}}{3}$

Equation 101 shows the values presented in Table 30 in matrix form.

$$[M] = \begin{bmatrix} \frac{h^{(e)}}{3} & \frac{h^{(e)}}{6} \\ \frac{h^{(e)}}{6} & \frac{h^{(e)}}{3} \end{bmatrix} \quad (101)$$

Integral Evaluations for Finite Element Stiffness Matrix

The stiffness matrix for each finite element must also be calculated. The elements of the stiffness matrix are determined by the integral term in Equation 81 and Equation 88

containing the $S_i \frac{dS_j}{dx}$ term. As was the case with the mass matrix in the previous section, this integral must be evaluated for the four cases shown previously in Table 29.

The integral in Equation 102 is the integral which must be evaluated with the four cases shown in Table 29. The values for S_1 and S_2 are given in Equation 89 and Equation 90, respectively.

$$\int_{\Omega} S_i \frac{dS_j}{dx} dx \quad (102)$$

The integral in Equation 102 must be evaluated from the position of the first node $x_1^{(e)}$ to the position of the second node $x_2^{(e)}$, as shown in Equation 103.

$$\int_{x_1^{(e)}}^{x_2^{(e)}} S_i \frac{dS_j}{dx} dx \quad (103)$$

The integration variable in Equation 103 must be changed from x to ξ . This is achieved in Equation 104.

$$\int_{x_1^{(e)}}^{x_2^{(e)}} \left(S_i \frac{dS_j}{dx} \right) dx = \int_{-1}^1 \left(S_i \frac{dS_j}{d\xi} \frac{d\xi}{dx} \right) \frac{dx}{d\xi} d\xi \quad (104)$$

Simplifying the right-hand side of Equation 104 results in the formula shown in Equation 105.

$$\int_{-1}^1 \left(S_i \frac{dS_j}{d\xi} \right) d\xi \quad (105)$$

The formula in Equation 105 must be evaluated for the four cases shown in Table 29. For

the first case, S_1 given in Equation 89 is plugged in for both S_i and S_j in Equation 105.

This is shown in Equation 106. Note that the term $\frac{dS_1}{d\xi}$ is equal to $-\frac{1}{2}$.

$$\int_{-1}^1 \frac{1}{2}(1-\xi) \left(-\frac{1}{2}\right) d\xi \quad (106)$$

Multiplying the terms in the integral in Equation 106 and moving the constants outside of the integral results in Equation 107.

$$-\frac{1}{4} \int_{-1}^1 (1-\xi) d\xi \quad (107)$$

Integrating the formula in Equation 107 results in the formula in Equation 108.

$$-\frac{1}{4} \left[\xi - \frac{1}{2} \xi^2 \right]_{-1}^1 \quad (108)$$

Plugging in the domain end point values of -1 and 1, respectively, and simplifying results in a value of $-\frac{1}{2}$.

The same integral calculations must be performed for the other three cases shown in Table 29. The details of these calculations will not be presented here. However, the final values for each of the four cases is shown in Table 31.

Table 31. Finite element stiffness matrix values

Index i	Index j	Stiffness Matrix Value
1	1	$-\frac{1}{2}$
1	2	$\frac{1}{2}$
2	1	$-\frac{1}{2}$
2	2	$\frac{1}{2}$

Equation 109 shows the values presented in Table 31 in matrix form.

$$[K] = \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \quad (109)$$

Integral Evaluations for Finite Element Forcing Vector

The forcing vector for each finite element must be calculated. The elements of the forcing vector are determined by the integral term in Equation 81 and Equation 88 containing the S_i term only. This integral must be evaluated for the two cases shown in Table 32.

Table 32. Two cases for determining finite element forcing vector

Index i
1
2

The integral in Equation 110 is the integral which must be evaluated with the two cases shown in Table 32. The values for S_1 and S_2 are given in Equation 89 and Equation 90, respectively.

$$\int_{\Omega} S_i dx \quad (110)$$

The integral in Equation 110 must be evaluated from the position of the first node $x_1^{(e)}$ to the position of the second node $x_2^{(e)}$, as shown in Equation 111.

$$\int_{x_1^{(e)}}^{x_2^{(e)}} S_i dx \quad (111)$$

The integration variable in Equation 111 must be changed from x to ζ . This is achieved in Equation 112.

$$\int_{x_1^{(e)}}^{x_2^{(e)}} S_i dx = \int_{-1}^1 S_i \frac{dx}{d\zeta} d\zeta \quad (112)$$

Equation 92 puts forth a value for the derivative of x with respect to ζ . Plugging Equation 92 into Equation 112 results in the integral term in Equation 113.

$$\int_{-1}^1 S_i \frac{h^{(e)}}{2} d\zeta \quad (113)$$

In Equation 113, $h^{(e)}$ is the length of the finite element and is constant. Therefore, this term can be taken out of the integral term, as shown in Equation 114.

$$\frac{h^{(e)}}{2} \int_{-1}^1 S_i d\zeta \quad (114)$$

The formula in Equation 114 must be evaluated for the two cases shown in Table 32. For the first case, S_1 given in Equation 89 is plugged in for S_i in Equation 114. This is shown in Equation 115.

$$\frac{h^{(e)}}{2} \int_{-1}^1 \frac{1}{2} (1-\zeta) d\zeta \quad (115)$$

Removing the constant terms from the integral in Equation 115 results in Equation 116.

$$\frac{h^{(e)}}{4} \int_{-1}^1 (1-\zeta) d\zeta \quad (116)$$

Integrating the formula in Equation 116 results in the formula in Equation 117.

$$\frac{h^{(e)}}{4} \left[\left(\zeta - \frac{1}{2} \zeta^2 \right) d\zeta \right]_{-1}^1 \quad (117)$$

Plugging in the domain end point values of -1 and 1, respectively, and simplifying results in a value of $\frac{h^{(e)}}{2}$.

The same integral calculations must be performed for the second case shown in Table 32. The details of these calculations will not be presented here. However, the final values for each of the two cases is shown in Table 33.

Table 33. Finite element forcing vector values

Index i	Forcing Vector Value
1	$\frac{h^{(e)}}{2}$
2	$\frac{h^{(e)}}{2}$

Equation 118 shows the values presented in Table 33 in matrix form.

$$[F] = \begin{bmatrix} \frac{h^{(e)}}{2} \\ \frac{h^{(e)}}{2} \end{bmatrix} \quad (118)$$

Matrix Form of Differential Equation for Continuity Equation

The differential form of the continuity equation (Equation 62 on page 95) can be written in matrix form as shown in Equation 119. Note that for clarity, square brackets are used for matrices, and curly brackets are used for vectors.

$$\frac{A_h}{R_g \Theta} [M] \frac{\partial \{p\}}{\partial t} + [K] [C_{K1}] \{m\} - c_{F1} \{F\} = 0 \quad (119)$$

In Equation 119, $[M]$ refers to the mass matrix (Equation 101), $[K]$ refers to the stiffness matrix (Equation 109), and $\{F\}$ refers to the forcing vector (Equation 118). The matrix $[C_{K1}]$ refers to the coefficients for the stiffness matrix. The elements of this matrix are shown in Equation 120. In Equation 120, A_1 and A_2 refer to the brake pipe cross-sectional area at the first and second nodes, respectively, of the finite element.

$$[C_{K1}] = \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} \quad (120)$$

A_h in Equation 119 is the harmonic mean of the brake pipe cross-sectional area at the first and second nodes, as shown in Equation 121.

$$A_h = \frac{2 A_1 A_2}{A_1 + A_2} \quad (121)$$

In Equation 119, c_{F1} is a scalar and is equal to the mass flow η into the brake pipe finite

element, as shown in Equation 122.

$$c_{F1} = \eta \quad (122)$$

The time differential term in Equation 119 can be approximated as shown in Equation 123. The terms s and $s+1$ do not represent the exponential operator. Rather, they refer to the value of $\{p\}$ at the current time step s and the next time step $s+1$, respectively.

$$\frac{\partial p}{\partial t} = \frac{\{p\}^{s+1} - \{p\}^s}{\Delta t} \quad (123)$$

Plugging Equation 123 into Equation 119 results in Equation 124.

$$\frac{A_h}{R_g \Theta} [M] \frac{\{p\}^{s+1} - \{p\}^s}{\Delta t} + [K][C_{K1}]\{m\}^{s+1} - c_{F1}^s \{F\} = 0 \quad (124)$$

Distributing the time differential term in Equation 124 results in Equation 125.

$$\frac{A_h}{R_g \Theta \Delta t} [M]\{p\}^{s+1} - \frac{A_h}{R_g \Theta \Delta t} [M]\{p\}^s + [K][C_{K1}]\{m\}^{s+1} - c_{F1}^s \{F\} = 0 \quad (125)$$

In Equation 125, moving all the terms containing values with an exponential s , which refers to the current time step, to the right-hand side of the equation results in Equation 126.

$$\frac{A_h}{R_g \Theta \Delta t} [M]\{p\}^{s+1} + [K][C_{K1}]\{m\}^{s+1} = \frac{A_h}{R_g \Theta \Delta t} [M]\{p\}^s + c_{F1}^s \{F\} \quad (126)$$

Plugging the matrix values representing a single finite element i into Equation 126 results in Equation 127.

$$\begin{aligned}
& \frac{1}{R_g \Theta \Delta t} \begin{bmatrix} \frac{A_h^{(i)} h^{(i)}}{3} & \frac{A_h^{(i)} h^{(i)}}{6} \\ \frac{A_h^{(i)} h^{(i)}}{6} & \frac{A_h^{(i)} h^{(i)}}{3} \end{bmatrix} \begin{bmatrix} p_1^{(i)} \\ p_2^{(i)} \end{bmatrix}^{s+1} + \begin{bmatrix} -\frac{A_1^{(i)}}{2} & \frac{A_2^{(i)}}{2} \\ -\frac{A_1^{(i)}}{2} & \frac{A_2^{(i)}}{2} \end{bmatrix} \begin{bmatrix} m_1^{(i)} \\ m_2^{(i)} \end{bmatrix}^{s+1} = \\
& \frac{1}{R_g \Theta \Delta t} \begin{bmatrix} \frac{A_h^{(i)} h^{(i)}}{R_g \theta 3} & \frac{A_h^{(i)} h^{(i)}}{R_g \theta 6} \\ \frac{A_h^{(i)} h^{(i)}}{R_g \theta 6} & \frac{A_h^{(i)} h^{(i)}}{R_g \theta 3} \end{bmatrix} \begin{bmatrix} p_i \\ p_j \\ p_k \end{bmatrix}^s + \begin{bmatrix} \frac{\eta^{(i)} h^{(i)}}{2} \\ \frac{\eta^{(i)} h^{(i)}}{2} \end{bmatrix}^s
\end{aligned} \tag{127}$$

Plugging the matrix values representing two finite elements i and j into Equation 126 results in Equation 128.

$$\begin{aligned}
& \frac{1}{R_g \Theta \Delta t} \begin{bmatrix} \frac{A_h^{(i)} h^{(i)}}{3} & \frac{A_h^{(i)} h^{(i)}}{6} & 0.0 \\ \frac{A_h^{(i)} h^{(i)}}{6} & \frac{A_h^{(i)} h^{(i)}}{3} + \frac{A_h^{(j)} h^{(j)}}{3} & \frac{A_h^{(j)} h^{(j)}}{6} \\ 0.0 & \frac{A_h^{(j)} h^{(j)}}{6} & \frac{A_h^{(j)} h^{(j)}}{3} \end{bmatrix} \begin{bmatrix} p_i \\ p_j \\ p_k \end{bmatrix}^{s+1} + \begin{bmatrix} -\frac{A_1^{(i)}}{2} & \frac{A_2^{(i)}}{2} & 0.0 \\ -\frac{A_1^{(i)}}{2} & \frac{A_2^{(i)}}{2} - \frac{A_1^{(j)}}{2} & \frac{A_2^{(j)}}{2} \\ 0.0 & -\frac{A_1^{(j)}}{2} & \frac{A_2^{(j)}}{2} \end{bmatrix} \begin{bmatrix} m_i \\ m_j \\ m_k \end{bmatrix}^{s+1} = \\
& \frac{1}{R_g \Theta \Delta t} \begin{bmatrix} \frac{A_h^{(i)} h^{(i)}}{3} & \frac{A_h^{(i)} h^{(i)}}{6} & 0.0 \\ \frac{A_h^{(i)} h^{(i)}}{6} & \frac{A_h^{(i)} h^{(i)}}{3} + \frac{A_h^{(j)} h^{(j)}}{3} & \frac{A_h^{(j)} h^{(j)}}{6} \\ 0.0 & \frac{A_h^{(j)} h^{(j)}}{6} & \frac{A_h^{(j)} h^{(j)}}{3} \end{bmatrix} \begin{bmatrix} p_i \\ p_j \\ p_k \end{bmatrix}^s + \begin{bmatrix} \frac{\eta^{(i)} h^{(i)}}{2} \\ \frac{\eta^{(i)} h^{(i)}}{2} + \frac{\eta^{(j)} h^{(j)}}{2} \\ \frac{\eta^{(j)} h^{(j)}}{2} \end{bmatrix}^s
\end{aligned} \tag{128}$$

Matrix Form of Differential Equation for Momentum Equation

The differential form of the momentum equation (Equation 74 on page 97) can be rewritten in matrix form as shown in Equation 129. Note that for clarity, square brackets are used for matrices, and curly brackets are used for vectors.

$$A_h [M] \frac{\partial \{m\}}{\partial t} + [K] [C_{K2}] \{m\} + A_h [K] \{p\} + \tau_a \pi d_h \{F\} = 0 \tag{129}$$

In Equation 129, $[M]$ refers to the mass matrix (Equation 101), $[K]$ refers to the stiffness matrix (Equation 109), and $\{F\}$ refers to the forcing vector (Equation 118). The matrix $[C_{K2}]$ refers to the coefficients for the stiffness matrix. The elements of this matrix are shown in Equation 130. In Equation 130, A_1 and A_2 refer to the brake pipe cross-sectional area at the first and second nodes, respectively, of the finite element. Similarly, u_1 and u_2 refer to the air velocity at the first and second nodes, respectively.

$$[C_{K2}] = \begin{bmatrix} u_1 A_1 & 0 \\ 0 & u_2 A_2 \end{bmatrix} \quad (130)$$

A_h in Equation 129 is the harmonic mean of the brake pipe cross-sectional area at the first and second nodes, as shown previously in Equation 121 on page 107. Similarly, d_h is the harmonic mean of the brake pipe cross-sectional diameter at the first and second nodes, as shown in Equation 131.

$$d_h = \frac{2 d_1 d_2}{d_1 + d_2} \quad (131)$$

τ_a in Equation 129 is the arithmetic mean of the value of τ at the first and second nodes, labeled τ_1 and τ_2 , respectively. The value of τ_1 and τ_2 is given by Equation 71 on page 96. The resulting value of τ_a is given in Equation 132.

$$\tau_a = \frac{a \left(\frac{\rho_1 u_1 d_1}{\mu} \right)^b \frac{\rho_1 u_1^2}{8} \left[\frac{u_1}{|u_1|} \right] + a \left(\frac{\rho_2 u_2 d_2}{\mu} \right)^b \frac{\rho_2 u_2^2}{8} \left[\frac{u_2}{|u_2|} \right]}{2} \quad (132)$$

In Equation 129, the time differential term can be approximated as shown in Equation 133. The terms s and $s+1$ do not represent the exponential operator. Rather, they refer to the value of $\{m\}$ at the current time step s and the next time step $s+1$, respectively.

$$\frac{\partial \{m\}}{\partial t} = \frac{\{m\}^{s+1} - \{m\}^s}{\Delta t} \quad (133)$$

Plugging Equation 133 into Equation 129 results in Equation 134.

$$A_h[M] \frac{\{m\}^{s+1} - \{m\}^s}{\Delta t} + [K][C_{K2}]\{m\}^{s+1} + A_h[K]\{p\}^{s+1} + \tau_a^s \pi d_h \{F\} = 0 \quad (134)$$

Distributing the time differential term in Equation 134 results in Equation 135.

$$\frac{A_h[M]\{m\}^{s+1}}{\Delta t} - \frac{A_h[M]\{m\}^s}{\Delta t} + [K][C_{K2}]\{m\}^{s+1} + A_h[K]\{p\}^{s+1} + \tau_a^s \pi d_h \{F\} = 0 \quad (135)$$

In Equation 135, moving all the terms containing values with an exponential s , which refers to the current time step, to the right-hand side of the equation results in Equation 136.

$$\frac{A_h}{\Delta t}[M]\frac{\{m\}^{s+1}}{\Delta t} + [K][C_{K2}]\{m\}^{s+1} + A_h[K]\{p\}^{s+1} = \frac{A_h}{\Delta t}[M]\frac{\{m\}^s}{\Delta t} - \tau_a^s \pi d_h \{F\} \quad (136)$$

Plugging the matrix values representing a single finite element i into Equation 136 results in Equation 137.

$$\begin{aligned} \frac{A_h^{(i)}}{\Delta t} \begin{bmatrix} \frac{h^{(i)}}{3} & \frac{h^{(i)}}{6} \\ \frac{h^{(i)}}{6} & \frac{h^{(i)}}{3} \end{bmatrix} \begin{bmatrix} m_1^{(i)} \\ m_2^{(i)} \end{bmatrix}^{s+1} + \begin{bmatrix} -\frac{u_1^{(i)} A_1^{(i)}}{2} & \frac{u_2^{(i)} A_2^{(i)}}{2} \\ -\frac{u_1^{(i)} A_1^{(i)}}{2} & \frac{u_2^{(i)} A_2^{(i)}}{2} \end{bmatrix} \begin{bmatrix} m_1^{(i)} \\ m_2^{(i)} \end{bmatrix}^{s+1} + A_h^{(i)} \begin{bmatrix} -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} p_1^{(i)} \\ p_2^{(i)} \end{bmatrix}^{s+1} = \\ \frac{A_h^{(i)}}{\Delta t} \begin{bmatrix} \frac{h^{(i)}}{3} & \frac{h^{(i)}}{6} \\ \frac{h^{(i)}}{6} & \frac{h^{(i)}}{3} \end{bmatrix} \begin{bmatrix} m_1^{(i)} \\ m_2^{(i)} \end{bmatrix}^s - \begin{bmatrix} \frac{\tau_a^{(i)} \pi d_h^{(i)} h^{(i)}}{2} \\ \frac{\tau_a^{(i)} \pi d_h^{(i)} h^{(i)}}{2} \end{bmatrix}^s \end{aligned} \quad (137)$$

Plugging the matrix values representing two finite elements i and j into Equation 136 results in Equation 138.

$$\begin{aligned} \frac{1}{\Delta t} \begin{bmatrix} \frac{A_h^{(i)} h^{(i)}}{3} & \frac{A_h^{(i)} h^{(i)}}{6} & 0.0 \\ \frac{A_h^{(i)} h^{(i)}}{6} & \frac{A_h^{(i)} h^{(i)}}{3} + \frac{A_h^{(j)} h^{(j)}}{3} & \frac{A_h^{(j)} h^{(j)}}{6} \\ 0.0 & \frac{A_h^{(j)} h^{(j)}}{6} & \frac{A_h^{(j)} h^{(j)}}{3} \end{bmatrix} \begin{bmatrix} m_i \\ m_j \\ m_k \end{bmatrix}^{s+1} + \begin{bmatrix} -\frac{u_1^{(i)} A_1^{(i)}}{2} & \frac{u_2^{(i)} A_2^{(i)}}{2} & 0.0 \\ -\frac{u_1^{(i)} A_1^{(i)}}{2} & \frac{u_2^{(i)} A_2^{(i)}}{2} - \frac{u_1^{(j)} A_1^{(j)}}{2} & \frac{u_2^{(j)} A_2^{(j)}}{2} \\ 0.0 & -\frac{u_1^{(j)} A_1^{(j)}}{2} & \frac{u_2^{(j)} A_2^{(j)}}{2} \end{bmatrix} \begin{bmatrix} m_i \\ m_j \\ m_k \end{bmatrix}^{s+1} + \\ \begin{bmatrix} -\frac{A_h^{(i)}}{2} & \frac{A_h^{(i)}}{2} & 0.0 \\ -\frac{A_h^{(i)}}{2} & \frac{A_h^{(i)}}{2} - \frac{A_h^{(j)}}{2} & \frac{A_h^{(j)}}{2} \\ 0.0 & -\frac{A_h^{(j)}}{2} & \frac{A_h^{(j)}}{2} \end{bmatrix} \begin{bmatrix} p_i \\ p_j \\ p_k \end{bmatrix}^{s+1} = \\ \frac{1}{\Delta t} \begin{bmatrix} \frac{A_h^{(i)} h^{(i)}}{3} & \frac{A_h^{(i)} h^{(i)}}{6} & 0.0 \\ \frac{A_h^{(i)} h^{(i)}}{6} & \frac{A_h^{(i)} h^{(i)}}{3} + \frac{A_h^{(j)} h^{(j)}}{3} & \frac{A_h^{(j)} h^{(j)}}{6} \\ 0.0 & \frac{A_h^{(j)} h^{(j)}}{6} & \frac{A_h^{(j)} h^{(j)}}{3} \end{bmatrix} \begin{bmatrix} m_i \\ m_j \\ m_k \end{bmatrix}^s - \begin{bmatrix} \frac{\tau_a^{(i)} \pi d_h^{(i)} h^{(i)}}{2} \\ \frac{\tau_a^{(i)} \pi d_h^{(i)} h^{(i)}}{2} + \frac{\tau_a^{(j)} \pi d_h^{(j)} h^{(j)}}{2} \\ \frac{\tau_a^{(j)} \pi d_h^{(j)} h^{(j)}}{2} \end{bmatrix}^s \end{aligned} \quad (138)$$

Row of Global Matrix For Internal Node of Continuity Equation

In the global matrix, a row representing an internal node for the continuity equation can be constructed using the middle row of of the matrices found in Equation 128 on page 109. The resulting row of the global matrix for this internal node j is shown in Equation

139.

$$\left[\frac{1}{R_g \Theta \Delta t} \frac{A_h^{(i)} h^{(i)}}{6} \frac{A_1^{(i)}}{2} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(i)} h^{(i)}}{3} + \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(j)} h^{(j)}}{3} \frac{A_2^{(i)}}{2} - \frac{A_1^{(j)}}{2} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(j)} h^{(j)}}{6} \frac{A_2^{(j)}}{2} \right] \begin{bmatrix} p_i \\ m_i \\ p_j \\ m_j \\ p_k \\ m_k \end{bmatrix}^{s+1} =$$

$$\left[\frac{1}{R_g \Theta \Delta t} \frac{A_h^{(i)} h^{(i)}}{6} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(i)} h^{(i)}}{3} + \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(j)} h^{(j)}}{3} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(j)} h^{(j)}}{6} \right] \begin{bmatrix} p_i \\ p_j \\ p_k \end{bmatrix}^s +$$

$$\left[\frac{\eta^{(i)} h^{(i)}}{2} + \frac{\eta^{(j)} h^{(j)}}{2} \right]^s$$
(139)

Row of Global Matrix For Internal Node of Momentum Equation

In the global matrix, a row representing an internal node for the momentum equation can be constructed using the middle row of of the matrices found in Equation 138 on page 111. The resulting row of the global matrix for this internal node j is shown in Equation 140.

$$\left[-\frac{A_h^{(i)}}{2} \frac{1}{\Delta t} \frac{A_h^{(i)} h^{(i)}}{6} - \frac{u_1^{(i)} A_1^{(i)}}{2} \frac{A_h^{(i)}}{2} - \frac{A_h^{(j)}}{2} \frac{1}{\Delta t} \frac{A_h^{(i)} h^{(i)}}{3} + \frac{1}{\Delta t} \frac{A_h^{(j)} h^{(j)}}{3} + \frac{u_2^{(j)} A_2^{(j)}}{2} - \frac{u_1^{(j)} A_1^{(j)}}{2} \frac{A_h^{(j)}}{2} \frac{1}{\Delta t} \frac{A_h^{(j)} h^{(j)}}{6} + \frac{u_2^{(j)} A_2^{(j)}}{2} \right] \begin{bmatrix} p_i \\ m_i \\ p_j \\ m_j \\ p_k \\ m_k \end{bmatrix}^{s+1} =$$

$$\left[\frac{1}{\Delta t} \frac{A_h^{(i)} h^{(i)}}{6} \frac{1}{\Delta t} \frac{A_h^{(i)} h^{(i)}}{3} + \frac{1}{\Delta t} \frac{A_h^{(j)} h^{(j)}}{3} \frac{1}{\Delta t} \frac{A_h^{(j)} h^{(j)}}{6} \right] \begin{bmatrix} m_i \\ m_j \\ m_k \end{bmatrix}^s -$$

$$\left[\frac{\tau_a^{(i)} \pi d_h^{(i)} h^{(i)}}{2} + \frac{\tau_a^{(j)} \pi d_h^{(j)} h^{(j)}}{2} \right]^s$$
(140)

Row of Global Matrix For First Node of Continuity Equation

In the global matrix, a row representing the first node for the continuity equation can be constructed using the first row of of the matrices found in Equation 128 on page 109. The resulting row of the global matrix for this first node is shown in Equation 141.

$$\left[\frac{1}{R_g \Theta \Delta t} \frac{A_h^{(1)} h^{(1)}}{3} - \frac{A_1^{(1)}}{2} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(1)} h^{(1)}}{6} \frac{A_2^{(1)}}{2} \right] \begin{bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \end{bmatrix}^{s+1} =$$

$$\left[\frac{1}{R_g \Theta \Delta t} \frac{A_h^{(1)} h^{(1)}}{3} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(1)} h^{(1)}}{6} \right] \begin{bmatrix} p_1 \\ p_2 \end{bmatrix}^s + \left[\frac{\eta^{(1)} h^{(1)}}{2} \right]^s$$
(141)

Since the first node is at the end (or boundary) of a cumulative brake pipe, it is necessary to specify a boundary condition. Two possible boundary conditions are of interest. The first requires specifying the air pressure at the boundary, and the second requires specifying the air velocity at the boundary. Which boundary condition is specified depends on the type of the first rail vehicle (since we are dealing with the first node in this subsection) associated with the cumulative brake pipe. An air pressure boundary condition applies if the first rail vehicle is a locomotive, and an air velocity boundary condition applies if the first rail vehicle is a car.¹⁵

In TPS, the continuity equation is used to specify the air pressure boundary condition. Since this subsection is addressing the continuity equation, then Equation 141 must be modified if the first rail vehicle is a locomotive. Namely, the air pressure must be set equal to the locomotive's idealized relay valve pressure. This can be done by using Equation 142.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \end{bmatrix}^{s+1} = P_{RelayValve} \quad (142)$$

As mentioned previously in the “Important TPS Quirks and Limitations” section beginning on page 32, TPS does not simulate the details of the locomotive automatic brake valve. Rather, in TPS, an “idealized” relay valve pressure is assumed.¹⁶ This idealized relay valve pressure $P_{RelayValve}$ shown in Equation 142 is stored in the `idealizedRelayValvePressure` variable in which is a member of the `Locomotive` class. This idealized relay valve pressure is calculated in the `calc_idealizedRelayValvePressure` function, which is also a member of the `Locomotive` class. The idealized relay valve pressure variable provides a smooth transition when the automatic air brake setting is changed (see “Locomotive Operator” section beginning on page 19). In TPS, it is assumed that this idealized relay valve pressure is reduced at a rate of 2 psi per second for a service brake application. In other words, if the automatic brake

¹⁵ This is a slight oversimplification. An air pressure boundary condition could apply if a car is the first rail vehicle associated with a cumulative brake pipe. This would occur if the train consist had a two-way EOT (see “Train Consist” section beginning on page 24) and if emergency braking is initiated. In this case, the two-way EOT vents the brake pipe to atmosphere, and an air pressure boundary condition would result. Namely, the air pressure boundary condition would be equal to atmospheric pressure, which is approximately 15 psi.

¹⁶ More details on a locomotive's automatic brake valve, including the relay valve and other components, can be found in Abdol-Hamid (1986) and Specchia et al (2012).

valve setting is changed from 105 psi to 85 psi, that amounts to a 20 psi reduction. It would take 10 seconds (since 20 psi divided by 2 psi per second equals 10 seconds) for the boundary condition to change from 105 psi to 85 psi. However, for an emergency brake application, the automatic brake valve setting is reduced at a faster rate of 20 psi per second. These values of 2 psi per second and 20 psi per second are stored in the ABVROC_SB and ABVROC_EB variables, respectively. Both of these variables are members of the Locomotive class.

The continuity equation could also be used to specify the air velocity boundary condition. However, in TPS, the momentum equation is used for this purpose, as explained in the following subsection “Row of Global Matrix For First Node of Momentum Equation”. Nevertheless, specifying the air velocity boundary condition using the continuity equation could be done by modifying Equation 141 if the first rail vehicle is a car. Namely, the air velocity must be set equal to zero, as shown in Equation 143.

$$\begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \end{bmatrix}^{s+1} = 0 \quad (143)$$

To summarize, Equation 141 is to be used in the global matrix if the first rail vehicle associated with the cumulative brake pipe is a car, and Equation 142 is to be used if the first rail vehicle associated with the cumulative brake pipe is a locomotive.

Row of Global Matrix For First Node of Momentum Equation

In the global matrix, a row representing the first node for the momentum equation can be constructed using the first row of of the matrices found in Equation 138 on page 111. The resulting row of the global matrix for this first node is shown in Equation 144.

$$\begin{bmatrix} -\frac{A_h^{(1)}}{2} & \frac{1}{\Delta t} \frac{A_h^{(1)} h^{(1)}}{3} - \frac{u_1^{(1)} A_1^{(1)}}{2} & \frac{A_h^{(1)}}{2} & \frac{1}{\Delta t} \frac{A_h^{(1)} h^{(1)}}{6} - \frac{u_2^{(1)} A_2^{(1)}}{2} \end{bmatrix} \begin{bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \end{bmatrix}^{s+1} = \begin{bmatrix} \frac{1}{\Delta t} \frac{A_h^{(1)}}{3} & \frac{1}{\Delta t} \frac{A_h^{(1)}}{6} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}^s - \left[\frac{\tau_a^{(1)} \pi d_h^{(1)} h^{(1)}}{2} \right]^s \quad (144)$$

As stated in the previous subsection (“Row of Global Matrix For First Node of Continuity Equation” beginning on page 112), since the first node is at the end (or

boundary) of a cumulative brake pipe, it is necessary to specify a boundary condition. Two possible boundary conditions are of interest. The first requires specifying the air pressure at the boundary, and the second requires specifying the air velocity at the boundary. Which boundary condition is specified depends on the type of the first rail vehicle (since we are dealing with the first node in this subsection) associated with the cumulative brake pipe. An air pressure boundary condition applies if the first rail vehicle is a locomotive, and an air velocity boundary condition applies if the first rail vehicle is a car.¹⁷

In TPS, the momentum equation is used to specify the air velocity boundary condition. Since this subsection is addressing the momentum equation, then Equation 144 must be modified if the first rail vehicle is a car. Namely, the air velocity must be set equal to zero. This can be done by using Equation 145.

$$\begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \end{bmatrix}^{s+1} = 0 \quad (145)$$

The momentum equation could also be used to specify the air pressure boundary condition. However, in TPS, the continuity equation is used for this purpose, as explained in the previous subsection “Row of Global Matrix For First Node of Continuity Equation” beginning on page 112. Nevertheless, specifying the air pressure boundary condition using the momentum equation could be done by modifying Equation 144 if the first rail vehicle is a locomotive. Namely, the air pressure must be set equal to the pressure of the locomotive’s idealized relay valve pressure $P_{relayValve}$, as shown in 146. This idealized relay valve pressure was discussed in more detail in the previous subsection “Row of Global Matrix For First Node of Continuity Equation”.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \end{bmatrix}^{s+1} = P_{RelayValve} \quad (146)$$

¹⁷ This is a slight oversimplification. An air pressure boundary condition could apply if a car is the first rail vehicle associated with a cumulative brake pipe. This would occur if the train consist had a two-way EOT device (see “Train Consist” section beginning on page 24) and if emergency braking is initiated. In this case, the two-way EOT device vents the brake pipe to atmosphere, and an air pressure boundary condition would result. Namely, the air pressure boundary condition would be equal to atmospheric pressure, which is approximately 15 psi.

To summarize, Equation 144 is to be used in the global matrix if the first rail vehicle associated with the cumulative brake pipe is a locomotive, and Equation 145 is to be used if the first rail vehicle associated with the cumulative brake pipe is a car.

Row of Global Matrix For Last Node of Continuity Equation

In the global matrix, a row representing the last node for the continuity equation can be constructed using the last row of of the matrices found in Equation 128 on page 109. The resulting row of the global matrix for this last node is shown in Equation 147. Note that if there are n finite elements, then there are $n+1$ nodes.

$$\begin{bmatrix} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(n)} h^{(n)}}{6} & -\frac{A_1^{(n)}}{2} & \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(n)} h^{(n)}}{3} & \frac{A_2^{(n)}}{2} \end{bmatrix} \begin{bmatrix} p_n \\ m_n \\ p_{n+1} \\ m_{n+1} \end{bmatrix}^{s+1} = \begin{bmatrix} \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(n)} h^{(n)}}{6} & \frac{1}{R_g \Theta \Delta t} \frac{A_h^{(n)} h^{(n)}}{3} \end{bmatrix} \begin{bmatrix} p_n \\ p_{n+1} \end{bmatrix}^s + \begin{bmatrix} \frac{\eta^{(n)} h^{(n)}}{2} \end{bmatrix}^s \quad (147)$$

A more complete discussion of boundary conditions for the continuity equation can be found in the “Row of Global Matrix For First Node of Continuity Equation” section beginning on page 112 and, therefore, will not be repeated here. In short, if the last rail vehicle associated with a cumulative brake pipe is a car, then Equation 147 is used. However, if the last rail vehicle associated with a cumulative brake pipe is a locomotive, then the air pressure boundary condition must be set equal to the locomotive’s idealized relay valve pressure. This can be done by replacing Equation 147 with Equation 148. Note that if there is a total of n brake pipe finite elements, then this results in a total of $n+1$ nodes in the finite element mesh.

$$\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_n \\ m_n \\ p_{n+1} \\ m_{n+1} \end{bmatrix}^{s+1} = P_{RelayValve} \quad (148)$$

Row of Global Matrix For Last Node of Momentum Equation

In the global matrix, a row representing the last node for the momentum equation can be constructed using the last row of of the matrices found in Equation 138 on page 111. The resulting row of the global matrix for this last node is shown in Equation 149. Note that

if there are n finite elements, then there are $n+1$ nodes.

$$\left[\begin{array}{cccc} -\frac{A_h^{(n)}}{2} & \frac{1}{\Delta t} \frac{A_h^{(n)} h^{(n)}}{6} - \frac{u_1^{(n)} A_1^{(n)}}{2} & \frac{A_h^{(n)}}{2} & \frac{1}{\Delta t} \frac{A_h^{(n)} h^{(n)}}{3} - \frac{u_2^{(n)} A_2^{(n)}}{2} \end{array} \right] \begin{bmatrix} p_n \\ m_n \\ p_{n+1} \\ m_{n+1} \end{bmatrix}^{s+1} = \left[\frac{1}{\Delta t} \frac{A_h^{(n)} h^{(n)}}{6} - \frac{1}{\Delta t} \frac{A_h^{(n)} h^{(n)}}{3} \right] \begin{bmatrix} m_n \\ m_{n+1} \end{bmatrix}^s - \left[\frac{\tau_a^{(n)} \pi d_h^{(n)} h^{(n)}}{2} \right]^s \quad (149)$$

A more complete discussion of boundary conditions for the momentum equation can be found in the “Row of Global Matrix For First Node of Momentum Equation” section beginning on page 114 and, therefore, will not be repeated here. In short, if the last rail vehicle associated with a cumulative brake pipe is a locomotive, then Equation 149 is used. However, if the last rail vehicle associated with a cumulative brake pipe is a car, then the air velocity boundary condition must be set to equal to zero. This can be done by replacing Equation 149 with Equation 150. Note that if there is a total of n brake pipe finite elements, then this results in a total of $n+1$ nodes in the finite element mesh.

$$\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_n \\ m_n \\ p_{n+1} \\ m_{n+1} \end{bmatrix}^{s+1} = 0 \quad (150)$$

Linear Interpolation

It will be assumed that the car control valve is in the middle of a car (as shown by the small yellow dots in Figure 56 on page 93), and therefore, the brake pipe pressure in the middle of each car must be approximated in order to determine the mass flow from the brake pipe into a car’s auxiliary or emergency reservoirs (see Appendix B. Mass Flow Rate Equation on page 124 for the equation that is used to calculate this mass flow). Linear interpolation is used to approximate the brake pipe pressure in between the finite element nodes. In other words, the brake pipe pressure P along a brake pipe finite element is assumed to be a linear function of the displacement x , as shown in Equation 151.

$$P(x) = a + bx \quad (151)$$

Therefore, the brake pipe pressure P_1 at the first node of the finite element and P_2 at the second node of the finite element are given by Equation 152 and Equation 153, respectively.

$$P_1 = a + bX_1 \quad (152)$$

$$P_2 = a + bX_2 \quad (153)$$

Solving for b in Equation 152 and Equation 153 results in Equation 154 and Equation 155, respectively.

$$b = \frac{P_1 - a}{X_1} \quad (154)$$

$$b = \frac{P_2 - a}{X_2} \quad (155)$$

Setting Equation 154 and Equation 155 equal to each other and solving for a results in Equation 156.

$$a = \frac{P_1 X_2 - P_2 X_1}{X_2 - X_1} \quad (156)$$

Next, solving for a in Equation 152 and Equation 153 results in Equation 157 and Equation 158, respectively.

$$a = P_1 - bX_1 \quad (157)$$

$$a = P_2 - bX_2 \quad (158)$$

Setting Equation 157 and Equation 158 equal to each other and solving for b results in Equation 159.

$$b = \frac{P_2 - P_1}{X_2 - X_1} \quad (159)$$

Substituting Equation 156 and Equation 159 into Equation 151 and rearranging terms

results in Equation 160.

$$P(x) = P_1 \left(\frac{X_2 - x}{X_2 - X_1} \right) + P_2 \left(\frac{x - X_1}{X_2 - X_1} \right) \quad (160)$$

In the TPS source code, the longitudinal location x of the center of each car associated with a brake pipe finite element is calculated in the `calc_railVehicleXLocations` function. Equation 160 is implemented in the `calculateInterpolatedValue` function. This function is called from the `calculateBrakePipePressureForRailVehicles` function. These functions are all members of the `BrakePipe_FiniteElement` class.

Appendix A. Visualizing User-Defined Functions

Running TPS Function Visualizer

This appendix explains how to use TPS's function visualizer program. This auxiliary program was created in order to help users visualize the interpolated values of the functions they are inputting into TPS. This is important since cubic spline interpolation can result in unexpected behavior with respect to interpolated values.

Running the TPS function visualizer executable should result in the command prompt output shown in Figure 57.

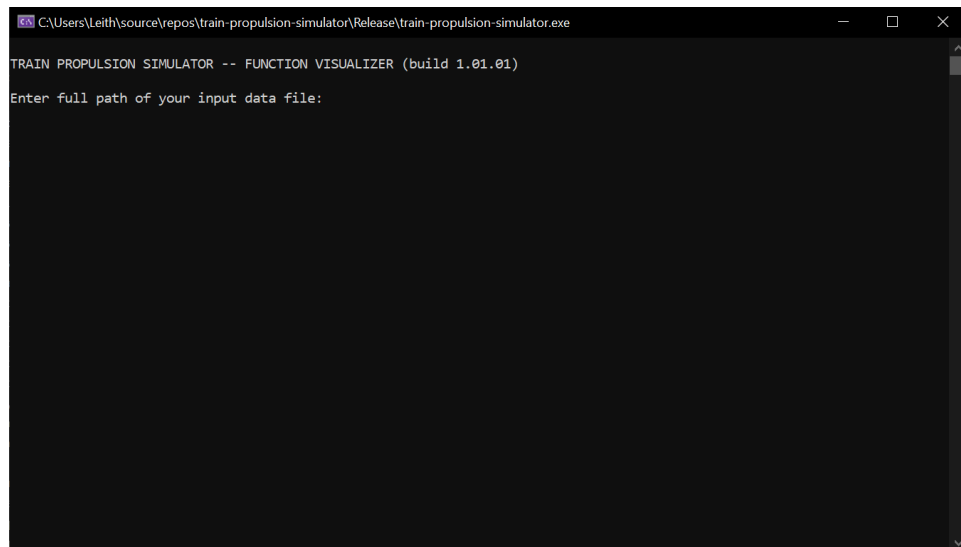
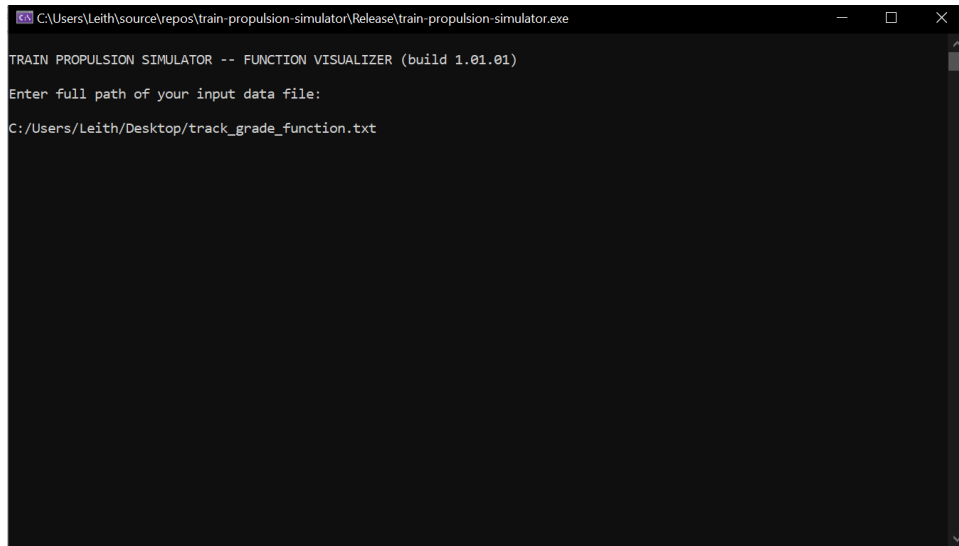


Figure 57. TPS function visualizer command prompt

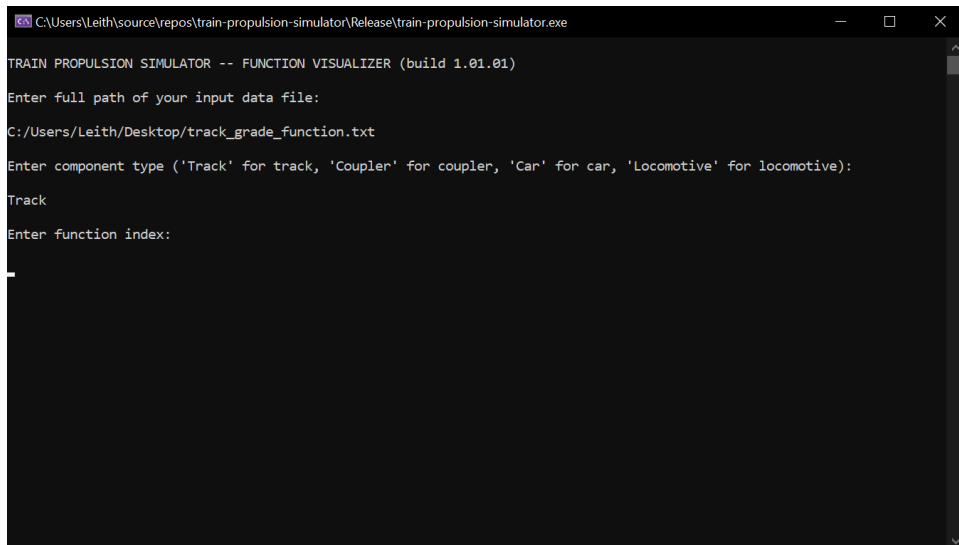
Next the user must enter the full path of the input file that contains the definition of their function. In the example shown in Figure 58, the full path of the input file is entered, which in this case is C:\Users\Leith\Desktop\track_grade_function.txt, and the Enter button is pressed. This results in the output shown in Figure 58 where the user is prompted for a component type.



```
C:\Users\Leith\source\repos\train-propulsion-simulator\Release\train-propulsion-simulator.exe
TRAIN PROPULSION SIMULATOR -- FUNCTION VISUALIZER (build 1.01.01)
Enter full path of your input data file:
C:/Users/Leith/Desktop/track_grade_function.txt
```

Figure 58. Prompt for component type

As the name of the input file (track_grade_function.txt) indicates, in this example the user wishes to plot a track grade function. Therefore, the user must type Track and press Enter. This results in the screen shown in Figure 59.

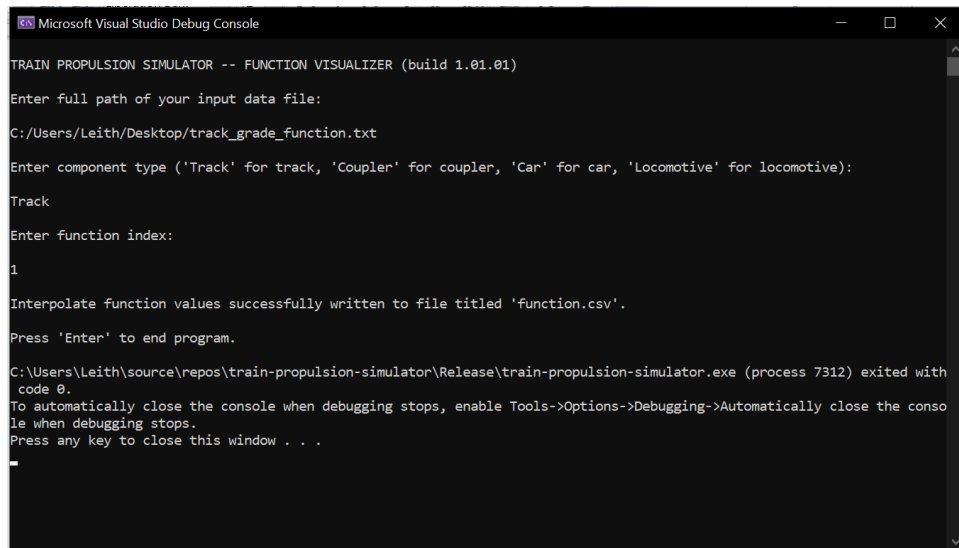


```
C:\Users\Leith\source\repos\train-propulsion-simulator\Release\train-propulsion-simulator.exe
TRAIN PROPULSION SIMULATOR -- FUNCTION VISUALIZER (build 1.01.01)
Enter full path of your input data file:
C:/Users/Leith/Desktop/track_grade_function.txt
Enter component type ('Track' for track, 'Coupler' for coupler, 'Car' for car, 'Locomotive' for locomotive):
Track
Enter function index:
1
```

Figure 59. Prompt for function index

A track consists of three functions: a grade function, a curvature function, and a superelevation function. Therefore, the user can enter 1 to indicate a grade function, 2 to indicate a curvature function, or 3 to indicate a superelevation function (Table 1 on page 3). As mentioned previously, for this example a grade function will be assumed, and therefore, the number 1 is typed and then Enter is pressed, which results in the screen

shown in Figure 60 if there are no errors in the input file. If there are errors in the input file, then the input file processor indicates the approximate line number where the error is located.



```
Microsoft Visual Studio Debug Console

TRAIN PROPULSION SIMULATOR -- FUNCTION VISUALIZER (build 1.01.01)

Enter full path of your input data file:
C:/Users/Leith/Desktop/track_grade_function.txt

Enter component type ('Track' for track, 'Coupler' for coupler, 'Car' for car, 'Locomotive' for locomotive):
Track

Enter function index:
1

Interpolate function values successfully written to file titled 'function.csv'.

Press 'Enter' to end program.

C:\Users\Leith\source\repos\train-propulsion-simulator\Release\train-propulsion-simulator.exe (process 7312) exited with
code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Figure 60. Prompt indicating function values successfully written to file

As indicated by the prompt in Figure 60, a file named `function.csv` is output to the same directory as the user's input file. This file can be opened in a spreadsheet program, such as LibreOffice Calc or Microsoft Excel. The first column of the file has the independent values of the function, and the second column has the dependent values of the function. The function can then be plotted by using the spreadsheet's built-in plotting tools.

Creating a TPS Function Visualizer Input File

As was stated in the "Create a TPS Input File" section beginning on page 2, TPS does not include a graphical user interface, and therefore, an input file must be created with the appropriate format and data. This applies to the auxiliary TPS function visualizer program, just as it does to the primary TPS simulation program. In the previous section, it was assumed that a track grade function was being entered. Therefore, in this section, an input file defining a track grade function will be created to complement the example in the previous section. The track grade function defined in Figure 3 on page 7 will be used. This function must be copied and pasted into a file on its own. A screenshot of the resulting input file is shown in Figure 61.

```

1 # 1. track grade
2 Function_
3 0.0, 0.0; 5280.0, 1.0; 21120.0, 0.0
4 21120.0, 0.0; 26400.0, -0.5; 29040.0, -1.0; 47520.0, 0.0
5 47520.0, 0.0; 79200.0, 2.5
6 Function
7

```

Figure 61. Example text for function input file

As was described in the “Track” section beginning on page 3, the sequence of characters `Function_` and `_Function` indicates the start and end of a function definition, respectively. Each line defines an interval, and within an interval, two-dimensional points are separated by a semicolon (;). The first line in Figure 61 beginning with a # indicates a comment line that is ignored by the input file processor.

Using the steps put forth in the previous section (“Running TPS Function Visualizer”) and plotting the resulting data in the `function.csv` should result in a plot similar to the plot shown in Figure 62. For illustration purposes, the interpolation points for the first, second, and third intervals have been plotted in blue, orange, and green, respectively. The yellow dots represent the points inputted by the user.

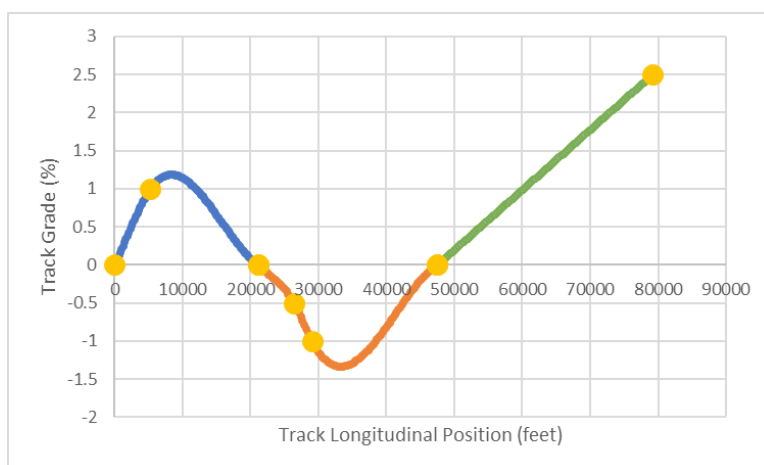


Figure 62. Plot of results in function.csv file

Appendix B. Mass Flow Rate Equation

Equation 161 is used to calculate the mass flow rate through an orifice (Abdol-Hamid 1986). $\dot{m}_{y,x}$ is the mass flow rate into component y due to component x , P_y is the pressure in component y , P_x is the pressure in component x , A is the orifice area between component y and component x , Θ is the gas temperature, and R_g is the specific gas constant. The specific gas constant R_g has units of J/(kg*K). In TPS, the gas is always assumed to be air, and therefore, the value of R_g is equal to 287.0 J/(kg*K).

$$\dot{m}_{y,x} = 0.6 A P_x \sqrt{\frac{\left(\left(\frac{P_y}{P_x}\right)^2 - 1\right) \left|1 - \frac{P_y}{P_x}\right|}{R_g \Theta \left(1 - \frac{P_y}{P_x}\right)}} \quad (161)$$

If the pressure in component x is greater than the pressure in component y , then $\dot{m}_{y,x}$ will be positive. Conversely, if the pressure in component x is less than the pressure in component y , then $\dot{m}_{y,x}$ will be negative.

The total mass flow rate to component x is given by the symbol \dot{m}_x . For example, if there are two components named y_1 and y_2 that are connected to component x in parallel, then the total mass flow rate to component x is the sum of the individual flow rates from each of the components y_1 and y_2 , as shown in Equation 162.

$$\dot{m}_x = \dot{m}_{x,y_1} + \dot{m}_{x,y_2} \quad (162)$$

Appendix C. Brake Shoe Temperature Modeling

TPS does not currently model brake shoe temperature. Low et al (1977) puts forth Equation 163 for modeling brake shoe temperature. T represents the brake shoe temperature in degrees Fahrenheit. The integration time step is represented by Δt . The wheel diameter in inches is represented by d . V represents the speed of the car in miles per hour. F_{NS} is the normal force on each brake shoe in units of pounds. F is the coefficient of friction. C_{FL} is a constant which is equal to 1.0 for plain shoes or 0.7 for flanged shoes. K_3 is a constant which is equal to 214,500.0 for plain shoes or 175,000.0 for flanged shoes.

$$T = T + \left(4.831928 \times 10^{-7} \left(\frac{C_{FL}}{d} \right) \left(\frac{V^2 (F_{NS} F)^{1.727273}}{T} \right)^{1.43} - d \frac{T^{1.5}}{K_3} \right) \Delta t \quad (163)$$

Equation 163 put forth by Low et al (1977) appears to be a differential equation derived from empirical data. It would be more desirable to model the brake shoe temperature using a physical model with sound physical underpinnings. The rest of this appendix puts forth a brief physical analysis of the approach that may be taken to create a more robust physical model of brake pad temperatures which may be incorporated into TPS in the future. Equation 164 puts forth the potential governing differential equation which is a physical heat transfer model. Similar models have been used to model rail temperatures, as can be seen in US Patent Application Publication US 2007/0265780 A1 to Kesler et al (2007).

$$\rho c V \frac{dT}{dt} = \dot{E}_{absorbed} - \dot{E}_{emitted} \quad (164)$$

In Equation 164, ρ is the density of the brake shoe, c is the specific heat of the brake shoe, V is the volume of the brake shoe, and T is the temperature of the brake shoe.

$\dot{E}_{absorbed}$ represents the rate of total energy absorbed by the brake shoe. Similarly, $\dot{E}_{emitted}$ represents the rate of total energy emitted by the brake shoe. Each side has units of power ($\text{kg} \cdot \text{m}^2 / \text{s}^3$).

The rate of total energy absorbed by the brake shoe could be represented by Equation 165 which also has units of power.

$$\dot{E}_{absorbed} = p A_{contact} \omega \delta \mu \quad (165)$$

In Equation 165, p is the pressure between the brake shoe and wheel, $A_{contact}$ is the contact area between the brake shoe and wheel, ω is the angular rotation of the wheel, δ is the radial distance from the wheel's axis of rotation to the center of the brake shoe, and μ is a friction factor.

The rate of total energy emitted by the brake shoe could be represented by Equation 166.

$$\dot{E}_{emitted} = h_{conv} A_{surface} (T - T_{\infty}) \quad (166)$$

In Equation 166, h_{conv} is a convection term, $A_{surface}$ is the surface area of the brake shoe, T is the temperature of the brake shoe, and T_{∞} is the ambient air temperature. A suitable convection term h_{conv} can likely be found in the existing fluid mechanics technical literature and should be a function of the velocity of the car.

References

- Abdol-Hamid, K. S. *Analysis and Simulation of the Pneumatic Braking System of Freight Trains*. PhD Thesis, Department of Mechanical Engineering, University of New Hampshire. 1986.
- Afshari, Ali, Stefano Specchia, Ahmed A. Shabana, Nelson Caldwell. *A Train Air Brake Force Model: Car Control Unit and Numerical Results*. Journal of Rail and Rapid Transit. Proc IMechE PartF: J Rail and Rapid Transit, 227(I), 38-55. 2012.
- Garg, V. K., R.V. Dukkipati. *Dynamics of Railway Vehicle Systems*. Academic Press Canada. 1984.
- Kesler, Kevin and Yu-Jiang Zhang. *System and Method For Predicting Future Rail Temperature*. US Patent Application Publication US 2007/0265780 A1. November 15, 2007.
- Kirk, Walter B. *Fluid Pressure Brake Control Apparatus With Accelerated Release After Service Application*. US Patent 3,175,869. March 30, 1965.
- Low, E. M., V. K. Garg. *Train Operations Simulator*. Report Number R-269. August 1977.
- Panebianco, Fredrick. *Quick Service Function For Railway Freight Brake Control Valve Device*. US Patent 5,118,166. June 2, 1992.
- Specchia, Stefano, Ali Afshari, Ahmed A. Shabana, Nelson Caldwell. *A Train Air Brake Force Model: Locomotive Automatic Brake Valve and Brake Pipe Flow Formulations*. Journal of Rail and Rapid Transit. Proc IMechE PartF: J Rail and Rapid Transit, 227(I), 29-37. 2012.
- Wilson, Richard L. and Daniel G. Scott. *Quick Service Valve Device*. US Patent 3,716,276. February 13, 1973.
- Zienkiewicz, O. C., K. Morgan. *Finite Elements and Approximation*. Dover Publications. 2006.