12-2018

Developing Computer Codes for Analyzing Bridge Response Due to WIM Data Truck Loading

Kenneth Patrick Pasley
University of Arkansas, Fayetteville

Developing Computer Codes for Analyzing Bridge Response Due to WIM Data Truck Loading

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Civil Engineering

by

Kenneth Pasley
University of Arkansas
Bachelor of Science in Civil Engineering, 2016

December 2018
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

_____
Ernie Heymsfield, Ph.D.
Thesis Director

_____
Sarah Hernandez, Ph.D.
Committee Member

_____
Gary Prinz, Ph.D.
Committee Member

ABSTRACT

Despite actual truck demographics evolving over the past 30 years, the Arkansas legal loads for bridge posting have not been updated. Custom computer codes were developed.to facilitate comparison between the current Arkansas legal loads for bridge posting and the actual Arkansas truck traffic to ensure that actual reactions are enveloped by the state'. The program, WIMFluence, calculates shear and moment influence lines for a bridge and the resulting shear and moment reactions based on weigh-in-motion data. With supporting scripts, the reactions due to actual Arkansas truck traffic were compared to those due to the current Arkansas legal loads for bridge posting. The program and scripts enabled the researchers to determine if the Arkansas legal loads for bridge posting envelop the bridge response values due to Arkansas's actual truck traffic and to recommend revisions to said legal loads. This thesis details the methodology employed within WIMFluence and its supporting scripts.

Along with this thesis is a zipped folder, *WIMFluenceSourceAndManual.zip*, of supplementary materials. This folder contains the source code for WIMFluence and its supporting scripts, a manual for use of WIMFluence and the scripts, and copyright license information files.

ACKNOWLEDGEMENTS

I would like to first thank God for the strength that got me here. I thank my parents for their love and support through this venture. I thank my advisor for his patience and encouragement as well as his understanding when pieces of the puzzle didn't go the ways we wanted or expected. I thank my friends and peers for their support and camaraderie within and without the academic environment. I thank the Arkansas Department of Transportation for funding this project, this opportunity for me to learn.

TABLE OF CONTENTS

1. INTRODUCTION

The purpose of this thesis is to explain the necessity for and development of custom computer codes for the Arkansas Department of Transportation (ARDOT) project "Bridge Load Posting Based on Actual Arkansas Truck Traffic" (Heymsfield, Hernandez and Pasley 2018). The project will be henceforth referred to as TRC1701, its ARDOT project number. TRC1701 compares the ARDOT legal load trucks currently used for bridge load posting to Arkansas's actual truck traffic to determine if the legal loads envelop the bridge response effects of the actual truck traffic and if the legal loads need to be revised. This analysis process involves determining the load effects of approximately one million unique representative truck configurations on nine simple bridge configurations and 270 continuous bridge configurations. Consequently, a computer program was developed to reasonably perform the analyses. Chapter 2 expands on details of TRC1701, similar research, and the need for developing a custom computer code, WIMFluence, for this project. WIMFluence and its supporting scripts were developed by the thesis author.

WIMFluence computes the bridge response shear and moments for the 279 bridge configurations using influence lines. Moment influence lines at the internal bridge supports are developed using the Müller-Breslau principal. Subsequently, force and moment equilibria are used along with the internal-support influence lines to determine shear and moment influence lines everywhere along the bridge span. Chapter 3 explains the theoretical approach WIMFluence uses for calculating the influence lines.

WIMFluence is the custom program developed for TRC1701. It is the implementation of the analysis methods described in Chapter 3. The WIMFluence computer code is described in Chapter 4. TRC1701 uses ratios between response values from different sets of trucks to

compare actual truck traffic to legal loads. The creation and further manipulation and analysis of these ratios is handled by three Python scripts. The first develops response ratios, while the second and third consolidate response ratios in useful manners. These scripts are described in Chapter 5. The flow of data through WIMFluence and the three Python scripts is illustrated in Figure 1.1.

Bridge load rating and posting are processes involving multiple load types and factors. Bridge rating factors are calculated using the different loads, including legal loads, and factors. Safe posting loads are calculated by multiplying an adjusted rating factor by the legal load gross weight (AASHTO 2011). Since TRC1701 involves comparing the legal loads for posting to Arkansas's actual truck traffic, the program and scripts presented in this thesis do not account for effects other than individual vehicular live loads. The program and scripts compare only the responses from legal loads and actual Arkansas trucks.

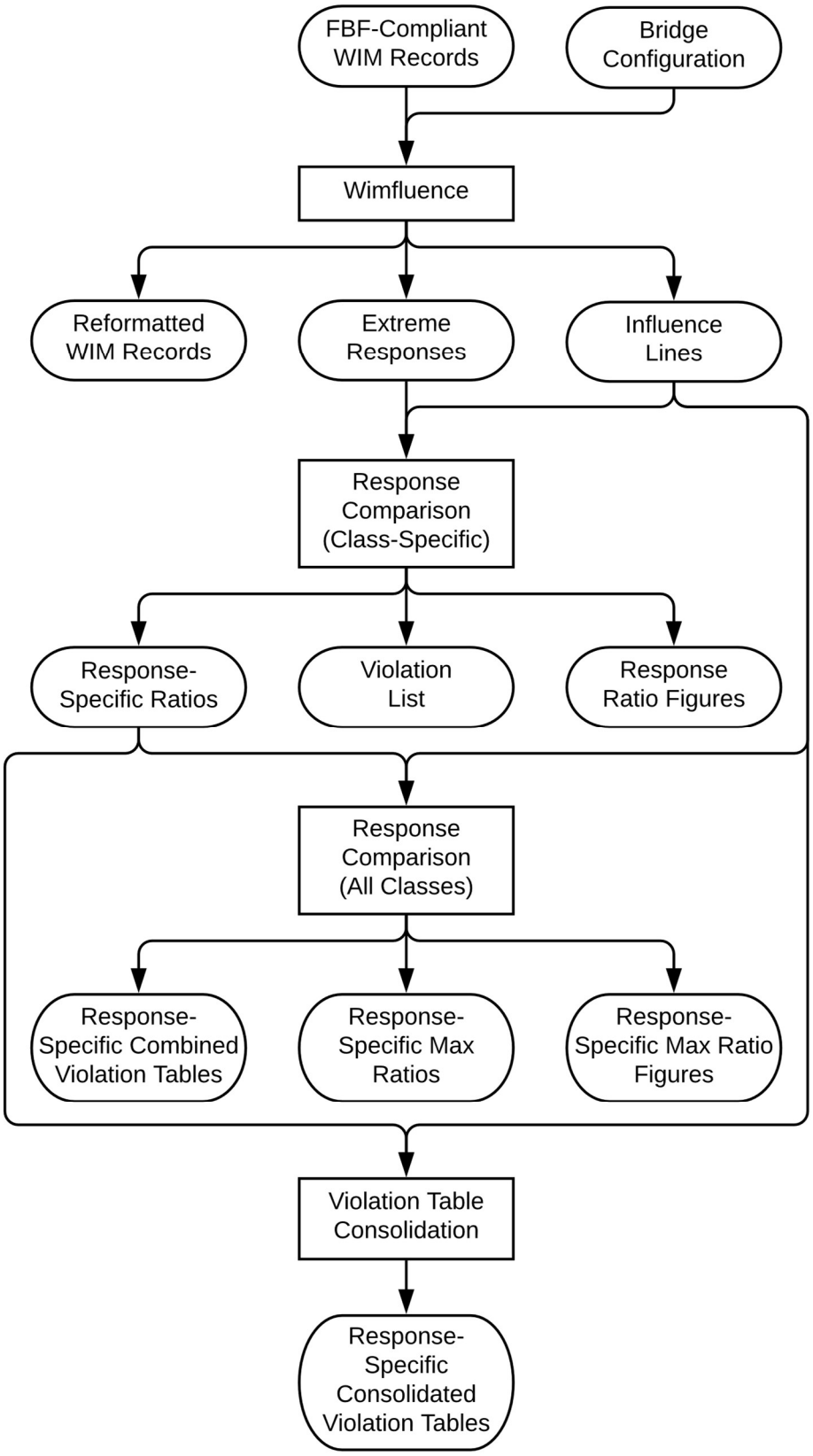

Figure 1.1. Development of response values, response ratios, and accompanying data

2. BACKGROUND

Bridge load posting establishes vehicle weight limits for individual bridges based on predefined hypothetical trucks. The process is a federal requirement for states to perform when bridges are insufficient for the "maximum unrestricted legal loads or State routine permit loads" (CFR 2011). To determine the bridge sufficiency, the same federal regulation requires each bridge to be rated based on its safe load-carrying capacity according to the Manual for Bridge Evaluation (AASHTO 2011). Load rating trucks are used to determine the bridge load posting values. These legal loads are designed to emulate and envelop the actual truck traffic within states.

AASHTO LEGAL LOADS

The American Association of State Highway and Transportation Officials (AASHTO) establishes the federal legal loads. AASHTO defines seven legal loads to be used for load posting (AASHTO 2011). Three typical AASHTO legal loads represent routine single and combination commercial vehicles. The three trucks are shown in Figure 2.1.

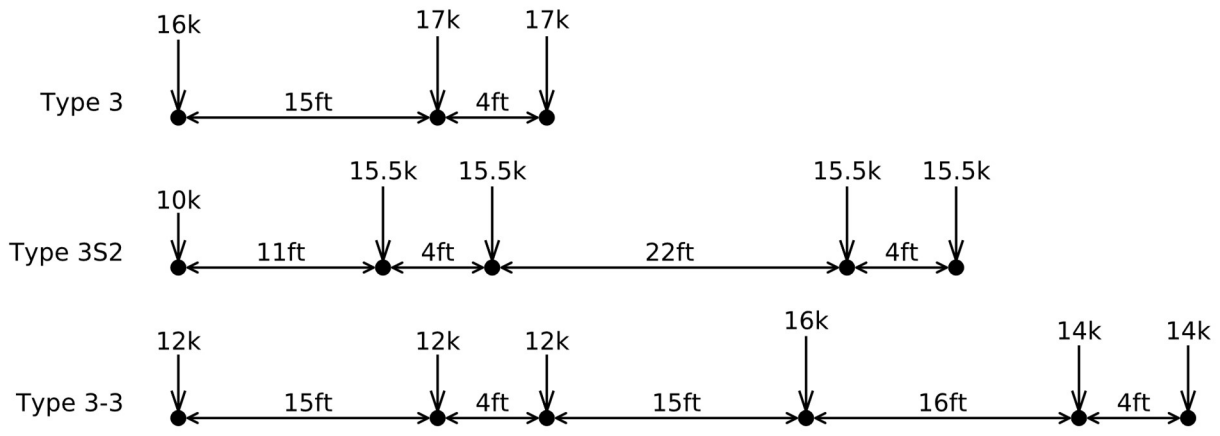

Figure 2.1. Typical AASHTO legal loads for posting — Type 3, Type 3S2, and Type 3-3 trucks

The other four trucks represent single-unit (SU) specialized hauling vehicles (SHVs). The represented SHVs are heavy short-wheelbase vehicles, such as concrete mixers and trucks. These four SU trucks are shown in Figure 2.2.

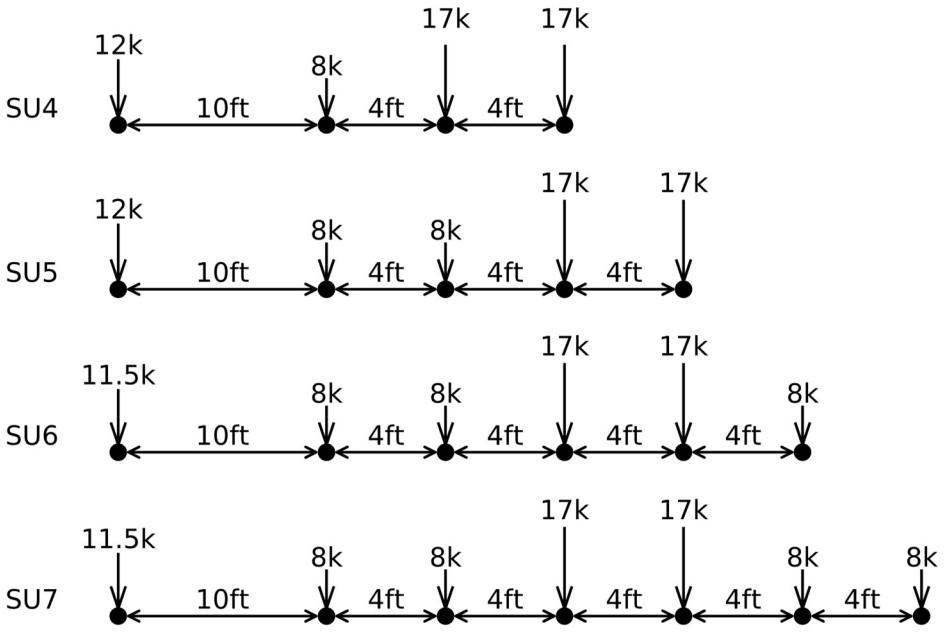

Figure 2.2. Single-unit AASHTO legal loads for posting — SU4, SU5, SU6, and SU7 trucks

Prior to the first edition of the AASHTO Manual for Bridge Evaluation, the four AASHTO SU legal loads did not exist. The SHVs represent a recent development in the trucking industry (Sivakumar, Moses, et al. 2007). With the introduction of the Federal Bridge Formula (FBF), some single unit trucks with short wheelbases had difficulty meeting legal weight guidelines. To comply with the FBF, the trucking industry introduced truck configurations with multiple closely spaced axles. While these configurations met legal requirements, they were still capable of inducing stresses exceeding those of the AASHTO typical legal loads shown in Figure 2.1. This issue was investigated in the National Cooperative Highway Research Program (NCHRP) Report 575 (Sivakumar, Moses, et al. 2007). The report presented a family of four SU truck

configurations with closely spaced rear axles to represent SHVs. A notional rating load is included in Report 575 that envelopes the bridge response from single-unit AASHTO loads shown in Figure 2.2. Therefore, the notional rating load serves as a prescreening load for the four SU legal loads. The seven bridge load posting trucks, three typical legal loads and four SHVs, are included in the Manual for Bridge Evaluation (AASHTO 2011).

ARDOT LEGAL LOADS AND ISSUES

The Arkansas Department of Transportation (ARDOT) establishes the state's legal loads. ARDOT uses three state-specific legal loads for bridge load posting. The Code 4, Code 9, and Code 5 trucks are shown in Figure 2.3.

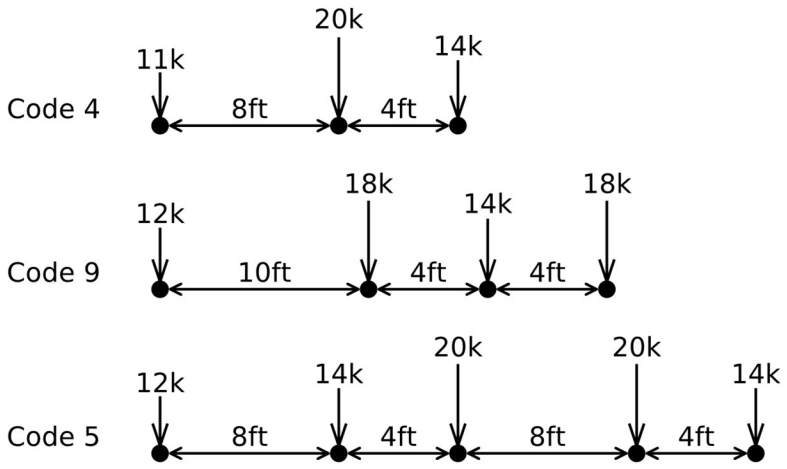

Figure 2.3. ARDOT legal loads for posting — Code 4, Code 9, and Code 5 trucks

The ARDOT legal loads were developed in the 1980s (Linz 2016). Despite actual truck demographics evolving over the past 30 years and the AASHTO legal loads being updated, the ARDOT legal loads for bridge load posting have not been adapted. Therefore, the current ARDOT legal loads may no longer represent Arkansas's actual truck traffic. TRC1701, the project for which WIMFluence and its accompanying scripts were developed, investigated this

possibility (Heymsfield, Hernandez and Pasley 2018). Through the custom program WIMFluence, bridge response due to actual Arkansas truck traffic as well as ARDOT and AASHTO load posting trucks were calculated. Through the accompanying scripts, the truck sets were compared to determine the adequacy of the currently used ARDOT legal loads. The ARDOT load posting trucks were found to be inadequate in representing the Arkansas truck traffic. A class of Arkansas's truck traffic was also found to frequently (75%) exceed the FBF maximum weight. The program and scripts prepared for this thesis enabled the research team to evaluate current bridge load posting vehicles and propose new load posting loads to ARDOT.

TRC1701 AND SIMILAR RESEARCH

TRC1701 – Bridge Load Posting Based on Actual Arkansas Truck Traffic

TRC1701 began with access to 346 million available WIM records over 11 years from 2005 to 2015. These records were filtered according to NCHRP Report 683 (Sivakumar, Ghosn and Moses 2011) to remove "bad and or unreliable data." The remaining records were consolidated into a set of 1.11 million representative unique configurations. The initial record set included vehicles that were not trucks, and therefore were removed from the study set. The truck configurations failing the FBF were removed using a lead-in script to WIMFluence. The filtering and consolidation methods provided 1.09 million unique truck configurations for bridge response evaluation.

NCHRP Report 575

NCHRP Report 575 presents a notable instance of similar research to TRC1701 (Sivakumar, Moses, et al. 2007). The study considered both simple and continuous span bridges of up to four spans. Initial span lengths were from 10ft to 100ft in 5ft increments then to 200ft in 10ft

increments. The continuous span ratios are shown in Table 2.1. TRC1701 considered simple and continuous span bridges of up to six spans with end span lengths from 20ft to 100ft in 10ft increments.

Table 2.1. Ratios of span lengths to the length of the initial span

Study	Number of Spans	Span Ratios
NCHRP Report 575	2	1.0 – 1.0
	3	1.0 – 1.25 – 1.0
	4	1.0 – 1.25 – 1.25 – 1.0
TRC1701 n: 1.0, 1.1, 1.2, 1.3, 1.4, 1.5	2	1.0 – n
	3	1.0 – n – 1.0
	4	1.0 – n – n – 1.0
	5	1.0 – n – n – n – 1.0
	6	1.0 – n – n – n – n – 1.0

NCHRP Report 575 is primarily concerned with specialized hauling vehicles. The NCHRP study began with access to 19.4 million weigh-in-motion (WIM) truck records. Through automated filtering and manual screening, the study reduced the number of truck records to 108. These 108 trucks formed the basis for the family of candidate legal loads that later became the AASHTO SU legal loads. To limit the study to SHVs, WIM trucks with wheelbases greater than 35 feet were removed. The trucks were then filtered by the FBF. Since the study was concerned with legal trucks not being represented in load rating, trucks failing the FBF were removed. The remaining trucks were compared to the three typical AASHTO legal loads in Figure 2.1, considering bridge response on 11 simple span bridges from 10ft to 200ft. Trucks that produced a load effect exceeding the maximum of the three AASHTO legal loads were identified as problematic trucks for the future development of new legal loads.

Review of Load Rating and Posting Procedures and Requirements

The Joint Transportation Research Program (JTRP) carried out a study to review the bridge load rating and posting procedures and requirements used by the Indiana Department of

Transportation (INDOT) and recommend ways to modify them to satisfy 23 CFR §650.313 (Bowman and Chou 2014; CFR 2011). This was accomplished by summarizing and comparing the load rating and posting procedures used in other states and evaluating procedures on a select group of bridges.

Prior to the study, INDOT used the AASHTO H-20 truck for its bridge load rating and posting procedures. Through a questionnaire and survey, the researchers found that most of the states responding to the survey used the AASHTO typical legal loads or similar state variations for their load rating and posting procedures. Of the 41 responding states, 25 states used the AASHTO typical legal loads or similar state variations. Of those 25 states, 14 included the AASHTO SHVs in their load rating and posting procedures. State-specific legal loads were used by 10 states. Six states used other legal loads such as the AASHTO H-20 and HS-20.

The researchers evaluated different load rating and posting procedures and different legal loads using four different bridges. The H-20 truck was found to not always encompass all of the AASHTO loads. The researchers determined that the H-20 should not be used for load rating and posting procedures. The study recommended that the AASHTO typical legal loads or the AASHTO SHVs be used for load rating and that posting loads be determined by the AASHTO Manual for Bridge Evaluation, 2nd Edition (Bowman and Chou 2014; AASHTO 2011).

Statistical Analysis of Weigh-in-Motion data for Bridge Design in Vermont

The University of Vermont Transportation Research Center (UVM TRC) carried out a study to confirm that the AASHTO specified design loadings were consistent with field data (Hernandez 2014). The study focused on the AASHTO bridge design loading, or the HL93 legal load for bridge design.

The UVM TRC study is fairly similar to TRC1701. Both studies sought to perform an automated analysis of WIM records and bridge response to determine extreme load cases. The Vermont Agency of Transportation provided 37 million WIM records across 12 WIM stations during 12 consecutive years. An algorithm was developed to determine the maximum stress demands (shear and moment) in simple span bridges. The spans ranged from 5m to 60m. All 37 million records were analyzed for comparison with the AASHTO HL-93 live load. In contrast, ARDOT provided 346 million WIM records considering 61 WIM stations over 11 consecutive years for TRC1701. These 346 million WIM records were condensed to one million relevant and unique truck records. The one million records were analyzed by WIMFluence considering a set of simple and continuous bridges to find extreme response distributions across the length of each considered bridge configuration. The difference in source WIM station distribution suggests the records provided for TRC1701 better encompass the state's truck traffic than those provide for the UVM TRC study

From there, the studies diverge. TRC1701 went on to determine response ratios between the ARDOT WIM response extreme values and the extreme values produced by the legal loads in question. TRC1701 found that the ARDOT legal loads needed to be revised to meet the bridge response requirements of both AASHTO and the current actual Arkansas truck traffic. The UVM TRC study went on to perform reliability analyses on various models for use comparing WIM data to the AASHTO HL-93 live load. Lognormal mixture models produced bridge failure probabilities varying across five orders of magnitude. However, the probabilities did not exceed the AASHTO target threshold. The extreme value theory proved capable of overestimating probabilities of bridge failure when modeling live load effects.

Developing Representative Michigan Truck Configurations for Bridge Load Rating

Another similar study was recently completed in Michigan at Wayne State University (Eamon and Siavashi 2018). As in Arkansas, there was a possibility that Michigan's legal loads for bridge load posting no longer represented Michigan's actual truck traffic. The Federal Highway Act establishing the FBF allowed preexisting legal loads to be retained even if they did not satisfy the new requirements. This resulted in the Michigan Department of Transportation (MDOT) having 28 legal loads for posting, including the grandfathered configurations. However, these grandfathered loads only represent a limited number of truck configurations. This resulted in a significant difference between Michigan's actual truck traffic and the set of legal loads intended to represent that traffic.

Like with TRC1701, Eamon and Siavashi sought to use WIM data to develop an accurate set of legal loads. From the state's 41 WIM stations, 20 stations were selected as representative sites. WIM data was gathered for 34 months from February 2014 to January 2017, except for April and May of 2014. 101.4 million records from the 20 selected sites were filtered down to 89.5 million legal and extended permit trucks.

These trucks were used to determine extreme response values for a set of bridges. Single span bridges and continuous spans of two equal spans lengths were analyzed. Span lengths ranged from 20 ft to 200 ft in 20 ft increments. Maximum positive moment, negative moment, and shear were considered. For continuous bridges, center support shears and negative moment values were reported. TRC1701 considered the maximum response values at multiple points along each bridge span. Eamon and Siavashi considered multiple presence vehicles while TRC1701 did not.

Eamon and Siavashi projected the load effects out to five years. Ratings were determined using the load factor rating (LFR) method as well as the load and resistance factor rating (LRFD) method. Adjustments to the Michigan load rating protocol were presented depending on the rating method used.

ECONOMIC IMPACTS

Bridge posting decisions are not only safety decisions but also economic ones (Fitzsimmons, Mulinazzi and Schrock 2014). A bridge closed or posted due to structural deficiencies can immediately burden citizens, industries, and governments with economic hardships. In addition to cost, the repair or replacement of a structurally deficient bridge creates hardships during bridge closure or restriction. Detours cause strain to citizens' personal lives, industries' efficient transportation of goods, and governments' official business.

Arkansas had 1441 posted bridges in 2016 (Linz 2016). To minimize excess financial burdens and other hardships on any entities affected by bridge closure while ensuring safety, the ARDOT legal loads for bridge posting must accurately represent Arkansas's actual truck traffic. The thesis author sought to achieve this by creating a tool capable of analyzing the bridge load response of a million Arkansas unique truck configurations considering any bridge configuration. This task was accomplished through WIMFluence and its accompanying scripts developed by this thesis author.

3. INFLUENCE LINE DEVELOPMENT

Influence lines are essential components of bridge design and analysis. An influence line shows for a set beam location the variation in response due to a unit load as it moves along the length of the beam (Michalos and Wilson 1965). This is invaluable for determining bridge load response when considering the moving nature of vehicle loads. The response value at a location is determined with a simple sum of multiplications regardless of the position or configuration of a vehicle moving across a bridge. This chapter details formulating the influence line methodology using the Müller-Breslau principle for inclusion in WIMFluence. In this thesis, the load response location is called an analysis point. The point that the unit load is applied at is called an ordinate location.

MÜLLER-BRESLAU PRINCIPLE

According to the Müller-Breslau principle, a response influence line will take a scaled form of the deflected beam shape when a unit deflection or rotation in the beam is introduced at the analysis point (Michalos and Wilson 1965). To determine an influence line, resistance to the response at the analysis point is removed and a relative unit deflection is applied for the shear influence line or a unit rotation is applied for the moment influence line. To remove shear resistance, a fictitious roller guide is applied. This allows vertical deflection at the analysis point while maintaining resistance to rotation. To remove moment resistance, a fictitious hinge is applied. The hinge allows rotation at the analysis point with zero relative displacement. The resulting deflected shape provides the influence line for a unit point load. Within WIMFluence, the Müller-Breslau principal is only used to find moment influence lines at the internal bridge supports of a continuous span bridge. These influence lines at the interior supports are used to find both shear and moment influence lines for all analysis points within the spans using force

and moment equilibrium at the analysis point. Therefore, creation of influence lines by the Müller-Breslau principle will only be explained for moments at internal supports.

The entire bridge is first treated as a single simple supported beam. At the first internal support, a hinge and unit rotation are introduced. The resulting vertical deflections are found at every internal support and ordered within a column vector. This is repeated for each internal support in turn with the resultant deflections stored in corresponding column vectors. Equations (3.1)-(3.3) provide a generalized method of determining deflections resulting from unit rotations. Figure 3.1 depicts the simple span layout. In these equations, internal support hinge locations serve as analysis points whereas locations of calculated deflections are ordinate locations. The formulation of equation (3.3) can be found in Appendix A.

$$L_O = \begin{cases} L_T - x_{AP} & x \leq x_{AP} \\ x_{AP} & x > x_{AP} \end{cases} \tag{3.1}$$

$$\xi = \begin{cases} x & x \leq x_{AP} \\ L_T - x & x > x_{AP} \end{cases} \tag{3.2}$$

$$h = \frac{L_O}{L_T} * \xi \tag{3.3}$$

$$\{h\}_j = \begin{bmatrix} h_1 \\ \vdots \\ h_n \end{bmatrix}_j$$

- n: number of internal supports
- x: ordinate location; location of deflection h
- ξ: distance from x to the bridge end on the same side of the analysis point
- x_{AP}: location of the support acting as the analysis point
- L_T: total length of the bridge

- L_O: length of the bridge on the opposite side of the analysis point from the load point
- h: deflection at the load point given a hinge and unit rotation at the analysis point
- $\{h\}_j$: column vector of deflections at every load point given the single analysis point at support j

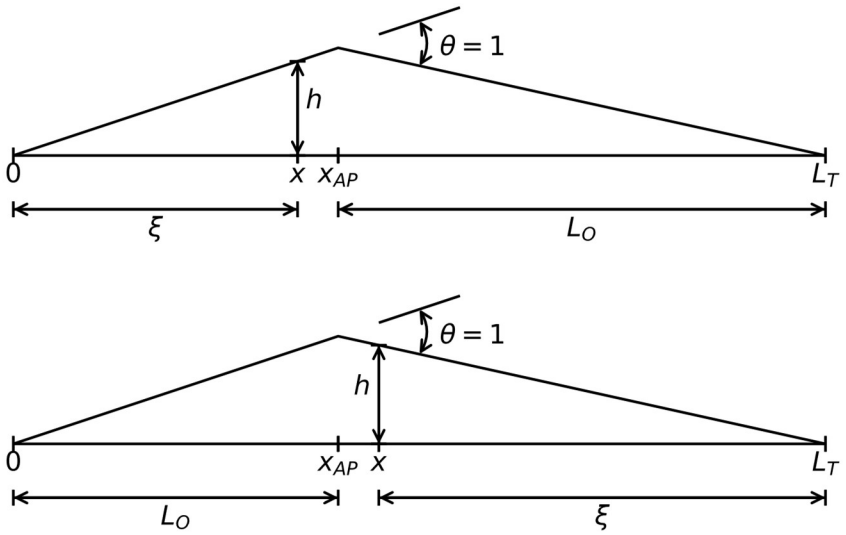

Figure 3.1. Variable orientation for deflection, h, calculation when $x \leq x_{AP}$ (top) and $x > x_{AP}$ (bottom)

CORRECTION FOR CONTINUOUS SPAN BRIDGES

If the bridges the Müller-Breslau principle is being used for were simple spans, the deflections provided by the previous equations would be the moment influence lines. Since the bridges are multi-span continuous bridges, the previously calculated deflection values must be corrected. Accounting for the vertical deflection being constrained to zero at the internal supports provides the means to correct the deflections everywhere along the beam length.

The bridge is again treated as a single simple supported beam. With no other modifications to the bridge, a unit point load is applied at the first internal support or analysis point. The resulting deflection at every internal support is calculated with equation (3.4):

$$\delta = \frac{L_O * \xi * (L_T^2 - L_O^2 - \xi^2)}{6 * L_T} \tag{3.4}$$

- δ: deflection at x
- δ_{AP}: deflection at the analysis point (when $x = x_{AP}$)

Figure 3.2 depicts the variable orientations allowing equation (3.4) to work as a function of x being greater than or less than x_{AP}. The modulus of elasticity and area moment of inertia are treated as constants and are therefore omitted from (3.4) since they cancel in the following derivation.

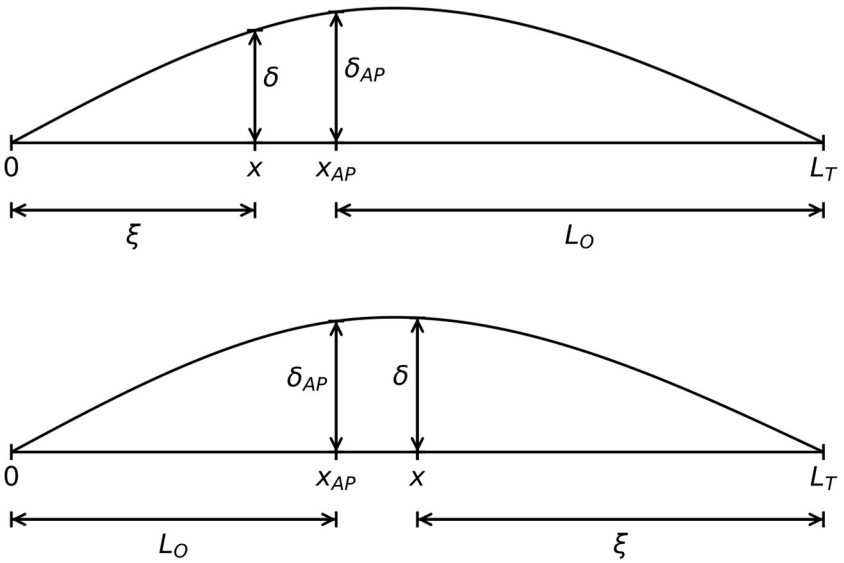

Figure 3.2. Variable orientation for deflection, δ, calculation when $x \leq x_{AP}$ (top) and $x > x_{AP}$ (bottom)

The ratio between each deflection and the deflection occurring at the loaded support is found with equation (3.5). These values are ordered in the first column of a matrix. Each subsequent internal support is treated as the analysis point in turn. A unit load is applied, internal support deflections are found, and resultant deflection ratios to loaded support deflection are stored in corresponding columns of the matrix.

$$\Delta = \frac{\delta}{\delta_{AP}} \tag{3.5}$$

$$[\Delta] = \begin{bmatrix} \Delta_{11} & \cdots & \Delta_{1n} \\ \vdots & \ddots & \vdots \\ \Delta_{n1} & \cdots & \Delta_{nn} \end{bmatrix}$$

- Δ: ratio of deflections between a load point and the support serving as the analysis point
- $[\Delta]$: $n \times n$ matrix of deflection ratios
- Δ_{ij}: deflection at i due to a unit load at j

The deflection correction factors are solved using systems of linear equations formed from the deflection ratios and the uncorrected deflections. To constrain the deflections in each vector $\{h\}_j$ to zero, a corresponding correction factor vector, $\{f\}_j$, is found such that

$$[\Delta]\{f\}_j + \{h\}_j = \{0\}, \tag{3.6}$$

where h is defined in equation (3.3) and j is the analysis point support number. Rearranging Equation (3.6) results in Equation (3.7), which is solved for $\{f\}_j$.

$$[\Delta]\{f\}_j = -\{h\}_j \tag{3.7}$$

- $\{f\}_j$: column vector of correction factors for each support when the analysis point is at support j

Within WIMFluence, equation (3.7) is solved using Gaussian elimination by matrix row operations on an augmented matrix formed with $[\Delta]$ and $-\{h\}_j$. Elements below the diagonal of $[\Delta]$ are set to 0 using row multiplication and row addition. Elements above the diagonal of $[\Delta]$ are modified likewise. The diagonal elements of $[\Delta]$ are set to 1 with row multiplication. The resulting column vector is the solution to $\{f\}_j$. This process is performed at each internal support to solve for the corresponding correction factor vector.

MOMENT INFLUENCE LINES AT SUPPORTS

The true moment influence lines can be determined for each support once the correction factors are known. Vectors are established to store the influence factors for each internal support. Zeroes are stored as the first element in each vector. These represent the moment influence factor at the first external bridge support, or bridge abutment. For each ordinate location, the resulting moment value at each internal support is found with equation (3.8):

$$M_j = h + \{\Delta\}\{f\}_j = h + [\Delta_1 \quad \cdots \quad \Delta_n]\begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}_j \tag{3.8}$$

- M_j: moment at support j due to unit load at ordinate location
- $\{\Delta\}$: row vector of deflection ratios between the load point and every support location

The influence line development process includes modeling the bridge as a simple span with a hinge and unit rotation at the internal support serving as the analysis point. The resulting deflection, h, is found at the ordinate location by equation (3.3). The bridge is again modeled as a simple span. A unit load is introduced at each internal support in turn. For each loaded support the deflection at the ordinate location, δ, and the deflection at the loaded internal support, δ_{AP}, are calculated. The deflection ratio, Δ, is found between δ and δ_{AP}. The deflection ratios are contained within the row vector, $\{\Delta\}$. The solution of equation (3.8) then gives the moment at internal support j when a unit load is placed at the ordinate location. The solution is stored in the next element of the internal support's moment influence line vector. This process is repeated for each internal support considering the same ordinate location. When the moments at all internal supports have been found, the next ordinate location is used to solve equation (3.8) for each internal support. After completing each ordinate location, zeroes are stored as the final

elements in each vector to represent the moment influence factor at the bridge end abutment support.

MOMENT INFLUENCE LINES WITHIN SPANS

Moment and shear influence lines within the spans are determined by force and moment equilibrium (Hibbeler 2011). For each beam ordinate location, the resulting moments at each end of the span containing the analysis point are used to calculate the moment and shear values at the analysis point. For WIMFluence, this requires the moment to be known at every bridge support, including end supports. Since the moments at the end supports of the bridges are assumed zero, moment influence line vectors composed entirely of zeroes are constructed. These vectors allow WIMFluence to handle the end spans of continuous span bridges and the single spans of single span bridges with the same computer code protocol used to handle the internal bridge spans.

For each ordinate location, the moment ordinate and shear ordinates are calculated for every analysis point. If the analysis point in question is at a support, the moment influence factor is pulled from the already developed support influence line. If the analysis point is within a span, a span-specific coordinate system is developed. The left beam support of the span serves as the origin. The analysis point location is converted to the corresponding location. If the ordinate location is not within the same span as the analysis point, the moment influence factor is calculated with the first part of equation (3.12). If the ordinate location is within the same span as the analysis point, the moment influence factor is calculated with either the second or third part of (3.12) depending on which side of the analysis point the ordinate location is on. Figure 3.3 details the different variable cases for internal moment calculation. The left support and right

support moment values are referenced from the support moment influence lines previously developed.

$$x_{AP,S} = x_{AP} - x_L \tag{3.9}$$

$$a = x - x_L \tag{3.10}$$

$$b = x_R - x \tag{3.11}$$

$$M = \begin{cases} \left(\frac{M_R - M_L}{L_S}\right) * x_{AP,S} + M_L & x \le x_L \text{ or } x_R \le x \\ \left(\frac{a}{L_S} - \frac{M_R - M_L}{L_S}\right)\left(L_S - x_{AP,S}\right) + M_R & x_L < x < x_{AP} \\ \left(\frac{b}{L_S} + \frac{M_R - M_L}{L_S}\right) * x_{AP,S} + M_L & x_{AP} \le x < x_R \end{cases} \tag{3.12}$$

- x: ordinate location
- $x_{AP,S}$: location of the analysis point within the containing span
- x_L: location of bridge support to the left of the analysis point
- x_R: location of bridge support to the right of the analysis point
- L_S: length of the span containing the analysis point
- M_L: moment at the support to the left of the analysis point
- M_R: moment at the support to the right of the analysis point
- M: moment at the analysis point

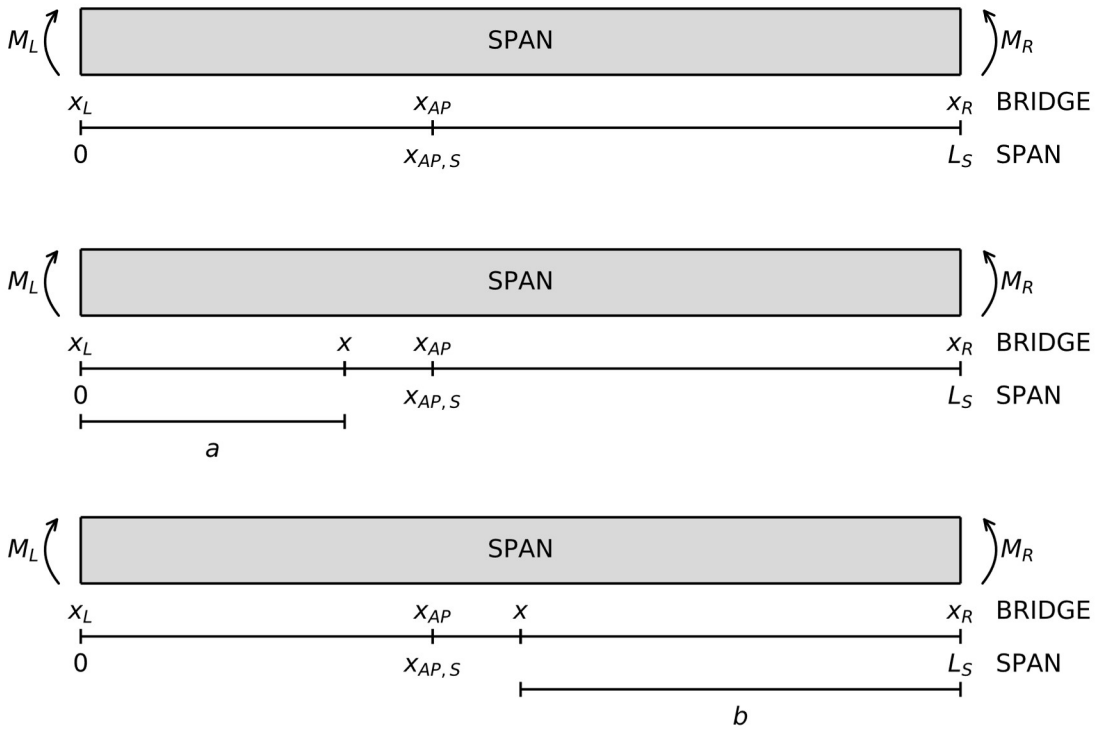

Figure 3.3. Variable orientation for internal span responses when $x \leq x_L$ or $x_R \leq x$ (top), $x_L < x \leq x_{AP}$ (middle), and $x_{AP} \leq x < x_R$ (bottom)

SHEAR INFLUENCE LINES AT SUPPORTS

Shear influence lines contain a discontinuity at the analysis point. In the context of the Müller-Breslau principle, the discontinuity is represented by a relative unit deflection introduced to produce the deflection curve that is the influence line. Since the Müller-Breslau principle is not directly used by WIMFluence to find shear influence lines, the discontinuity must be accounted for in the shear influence factor calculations. To capture the discontinuity, two instances of the shear influence line are developed simultaneously. One represents the shear produced on the immediate left side of the analysis point while the other represents the shear produced on the immediate right side of the analysis point. Separating these two values ensures

that every shear value is developed. This prevents a negative shear value of a larger magnitude than the positive shear value on the other side from being neglected.

If the analysis point is on a support, including an end support, the analysis point is treated as within the left span for calculating the left shear and within the right span for calculating the right shear. Two exceptions apply when the analysis point is on the external supports since there is no span on the outer side of the analysis point. If the analysis point is at the first (left) external support, the left shear value is zero. If the analysis point is at the second (right) external support, the right shear value is zero. Equations (3.15) and (3.16) give the left shear and right shear values, respectively. Figure 3.4 depicts the variable layouts for support shear calculation.

$$a = \begin{cases} 0 & x \leq x_L \text{ or } x_R < x \\ x - x_L & x_L < x \leq x_R \end{cases} \tag{3.13}$$

$$b = \begin{cases} 0 & x < x_L \text{ or } x_R \leq x \\ x_R - x & x_L \leq x < x_R \end{cases} \tag{3.14}$$

$$V_L = \frac{M_R - M_L}{L_S} - \frac{a}{L_S} \tag{3.15}$$

$$V_R = \frac{b}{L_S} + \frac{M_R - M_L}{L_S} \tag{3.16}$$

- x: ordinate location
- x_L: location of the left support of the span considered to contain the analysis point
- x_R: location of the right support of the span considered to contain the analysis point
- V_L: shear on the left of the analysis point
- V_R: shear on the right of the analysis point
- M_L: moment at left support of the span considered to contain the analysis point
- M_R: moment at right support of the span considered to contain the analysis point
- L_S: length of the span considered to contain the analysis point

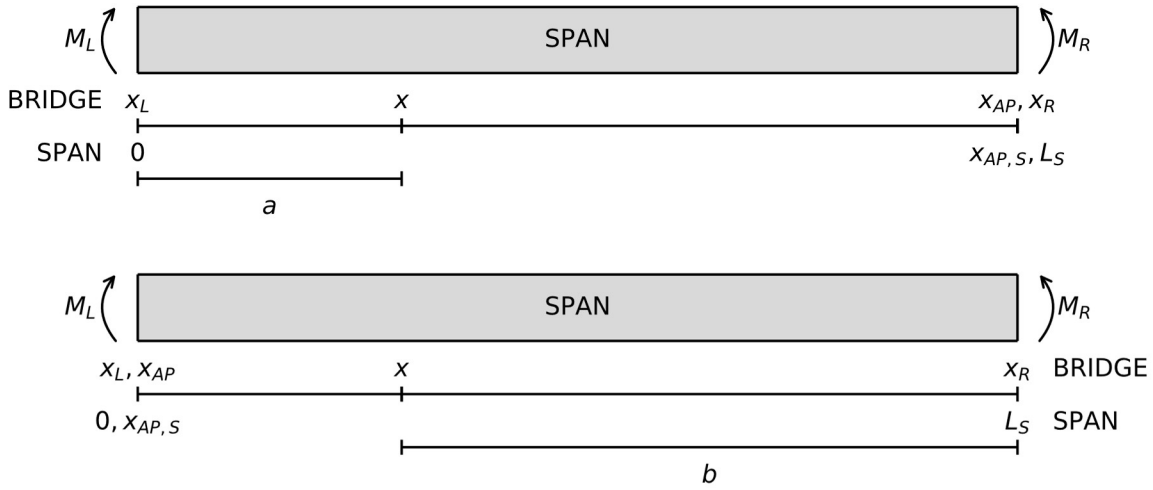

Figure 3.4. Variable orientation for calculation of left shear (top) and right shear (bottom) when the analysis point is at a support

SHEAR INFLUENCE LINES WITHIN SPANS

The equations used for shear within the spans are dependent on the ordinate location relative to the analysis point. Therefore, the equations to calculate shear when the analysis point is at a support are not reused when the analysis point is truly within a span. Equations (3.19) and (3.20) give the left shear and right shear respectively as well as the ordinate location dictating the equations' uses. Figure 3.3 is applicable here.

$$a = \{x - x_L \quad x_L < x \leq x_{AP} \tag{3.17}$$

$$b = \{x_R - x \quad x_{AP} \leq x < x_R \tag{3.18}$$

$$V_L = \begin{cases} \frac{M_R - M_L}{L_S} & x \leq x_L \text{ or } x_R \leq x \\ \frac{M_R - M_L}{L_S} - \frac{a}{L_S} & x_L < x \leq x_{AP} \\ \frac{b}{L_S} - \frac{M_R - M_L}{L_S} & x_{AP} < x < x_R \end{cases} \tag{3.19}$$

$$V_R = \begin{cases} \frac{M_R - M_L}{L_S} & x \leq x_L \text{ or } x_R \leq x \\ \frac{M_R - M_L}{L_S} - \frac{a}{L_S} & x_L < x < x_{AP} \\ \frac{b}{L_S} - \frac{M_R - M_L}{L_S} & x_{AP} \leq x < x_R \end{cases} \tag{3.20}$$

- x_L: location of the support to the left of the analysis point
- x_R: location of the support to the right of the analysis point
- M_L: moment at left support of the span containing the analysis point
- M_R: moment at right support of the span containing the analysis point
- L_S: length of the span containing the analysis point

After the moment influence factor and shear influence factors have been found for the individual ordinate location-analysis point pair, WIMFluence moves to the next analysis point. After the influence factors for all analysis points have been found, WIMFluence moves to the next ordinate location. This process is repeated until every ordinate location-analysis point pair has been considered. At this point, valid influence lines for moment, left shear, and right shear are stored in memory.

INFLUENCE LINE GENERALIZATION

One of the program constraints introduced by the research project required that values for different bridge configurations be comparable. This constraint is satisfied by writing the influence line values as dimensionless terms. Each moment influence value is divided by the length of the first bridge span and the unit point load, as shown in equation (3.21). Since the shear influence lines are inherently independent of the actual span lengths, they remain unchanged. WIMFluence outputs the resulting influence lines to a file. These dimensionless parameters develop influence lines that are length independent and only a function of interior span to end span ratios. For example, the influence lines produced for a two-span 20ft-40ft bridge are valid for a two-span 30m-60m bridge. However, span length dependence is reintroduced when calculating the actual moment response values due to trucks since the truck is length dependent.

$$M' = \frac{M}{L_1 * P} \tag{3.21}$$

- M': nondimensionalized moment value
- L_1: the length of the first span of the bridge
- P: unit point load

4. EXTREME RESPONSE DEVELOPMENT – WIMFLUENCE

WIMFluence is the major component of the computer codes developed for evaluating WIM data trucks for bridge load posting. WIMFluence determines extreme load response values along a bridge due to a set of individual trucks moving across a bridge. To satisfy this task, WIMFluence develops influence lines for multiple analysis points along a bridge. The program is written in C++11. For TRC1701, it is compiled with the GNU GCC compiler. This chapter details the inner workings of WIMFluence as shown in Figure 4.1.

WIMFluence can be considered as a main program with three distinct major operational sections. The main program handles user interaction, output file initializations, operation of the major sections, and file closings. The first major operational section determines influence line correction factors. The second major operational section determines influence lines. These two sections are constructed in accordance with the formulation described in Chapter 3. The third major operational section determines extreme response values for each analysis point.

The WIMFluence code is divided into six major files. The main file contains the main function of the program. Four of the files are function definition files dedicated to output creation, user input, displayed information, and calculation implementation. These four files each have an associated header file with function declarations. The sixth major file is a header file with data struct definitions.

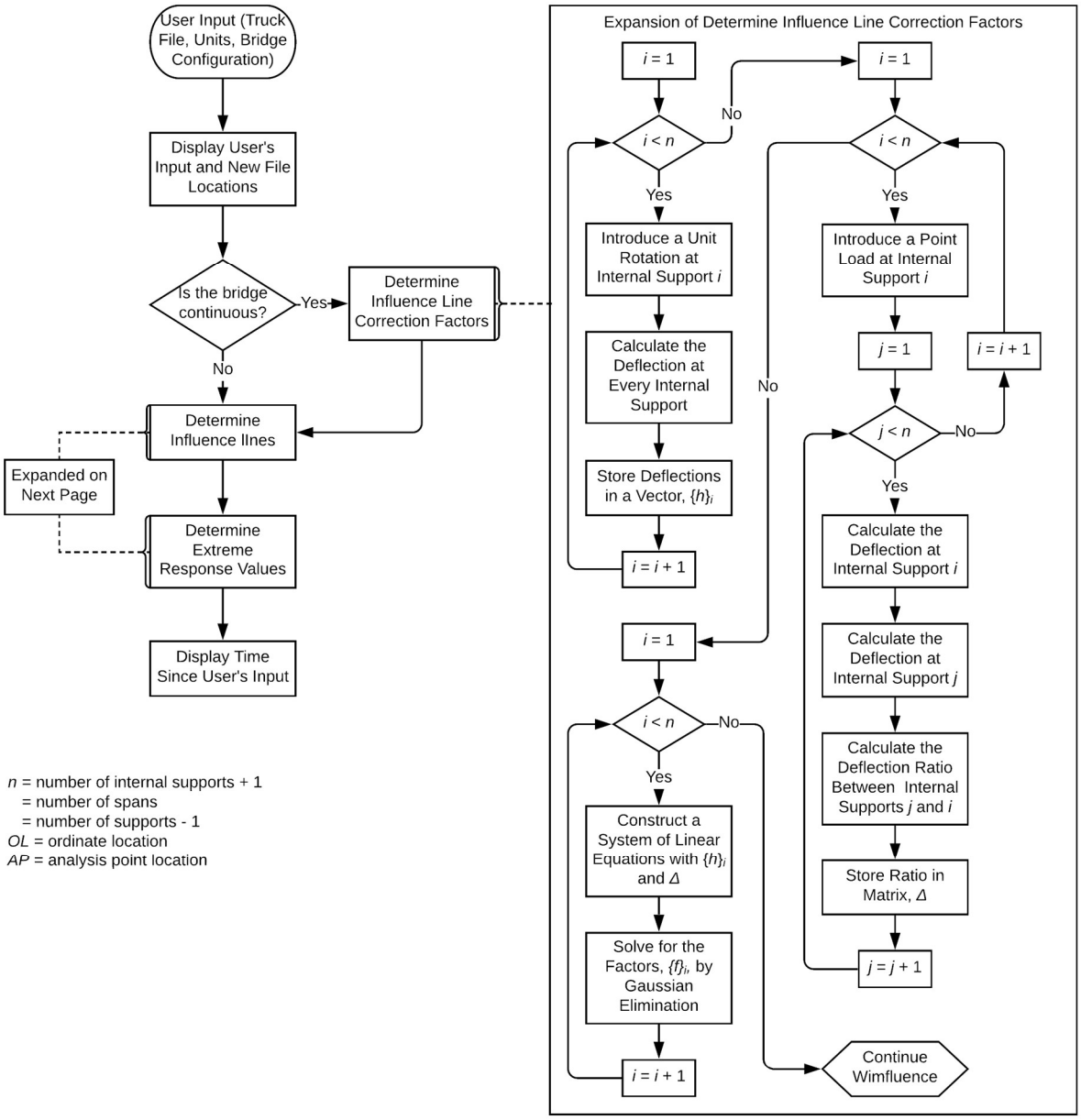

Figure 4.1. WIMFluence Flowchart

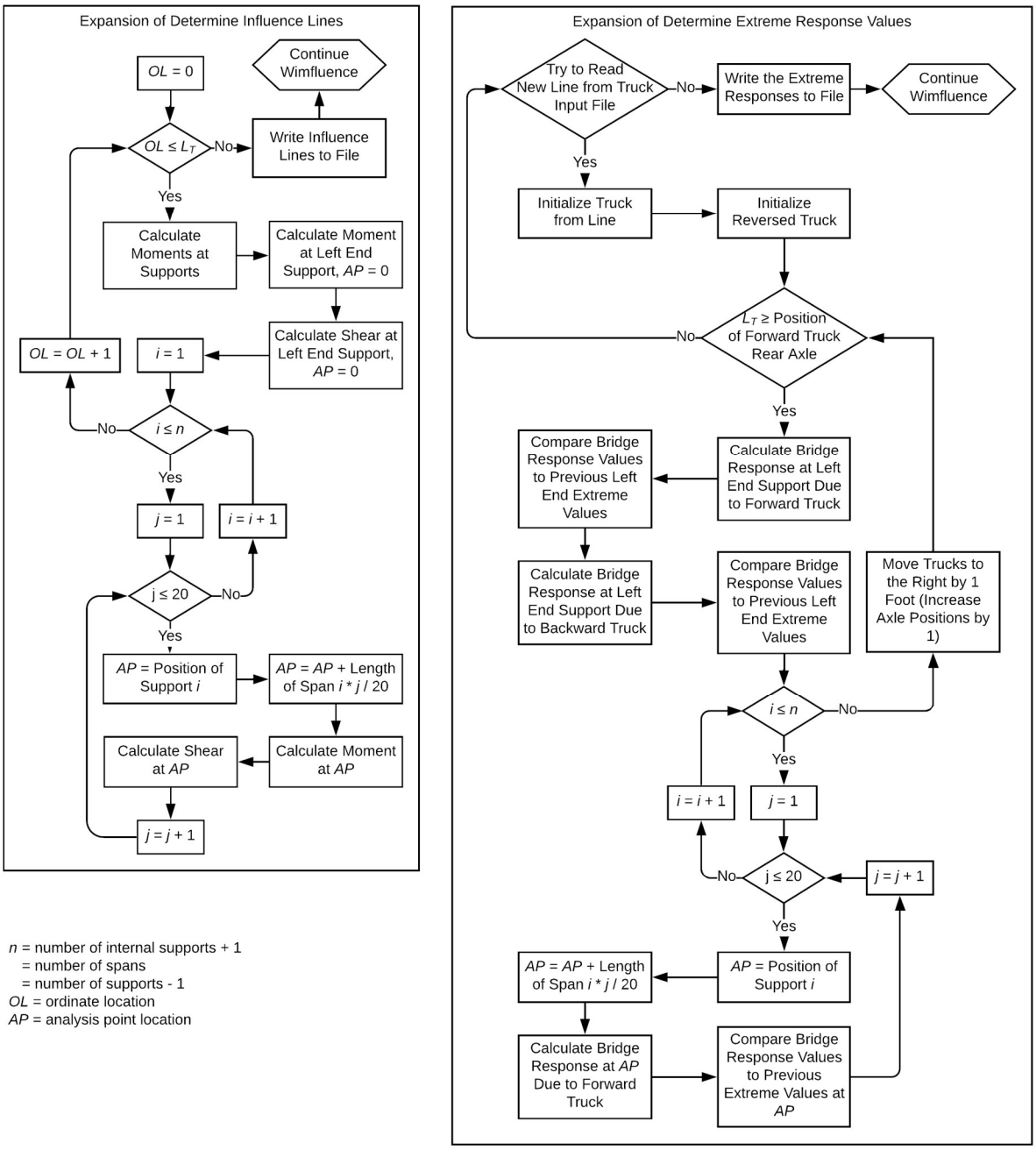

Figure 4.1 (Cont.). WIMFluence Flowchart

WIMFLUENCE DATA STRUCTS

WIMFluence defines four data structs for use throughout the code. The structs are custom groupings of related data under a single name. The four structs are *axle*, *truck*, *extreme_set*, and *bridge*.

The *axle* data struct represents the axles of a truck. It is composed of two floating point numbers, or floats, containing an axle's position and weight. The *axle* is mainly used as a component of the *truck* data struct. The *axle* struct is detailed in Table 4.1.

Table 4.1. Composition of *axle* data struct

axle	
type	name
float	position
float	weight

The *truck* data struct represents a truck read from the input truck file. Two integers contain the truck number from the input file and the number of axles the truck has. The truck number is the position of the truck within the input file (e.g. the first truck has a truck number of "1"). A dynamically allocated array, or vector, contains the axles of the represented truck. Each element of the container is an instance of the *axle* struct. A character contains the direction the truck is facing. Trucks are initialized with the character 'f' to represent a forward-facing truck. When WIMFluence reverses a truck, the direction character is changed to 'b' to represent a backwards-facing truck. The data struct is detailed in Table 4.2.

Table 4.2. Composition of *truck* data struct

truck		
type	name	default
int	number	
int	num_axles	
vector<axle>	axles	
char	direction	'f'

The *extreme_set* data struct contains the extreme response values for a single analysis point and the corresponding trucks producing the response values. It is detailed in Table 4.3. Six floats are the six extreme response values at an analysis point: the maximum positive and maximum negative of the moment, left shear, and right shear. Each float is initialized as either the maximum or minimum possible float. For example, the maximum negative moment is initialized as the maximum float value. This ensures that any greater response value in the desired direction will be appropriately captured. Six instances of the *truck* struct are created to contain the truck configurations for the six extreme response values. Each instance contains all the information necessary to recreate the loading conditions producing the extreme response.

Table 4.3. Composition of *extreme_set* data struct

extreme_set		
type	name	default
float	max_shear_left	–MAX
float	min_shear_left	MAX
float	max_shear_right	–MAX
float	min_shear_right	MAX
float	max_moment	–MAX
float	min_moment	MAX
truck	max_shear_left_truck	
truck	min_shear_left_truck	
truck	max_shear_right_truck	
truck	min_shear_right_truck	
truck	max_moment_truck	
truck	min_moment_truck	

The *bridge* data struct identifies the bridge that WIMFluence is determining influence lines and response values for. Table 4.4 details the *bridge* composition. An integer contains the number of bridge spans. Two floats contain the length of the first span and the length of the entire bridge. Two vectors of floats contain the span lengths and the internal support positions. Three multidimensional vectors of floats contain the support deflections produced by equation (3.3), the support deflection ratios produced by equation (3.5), and the correction factors calculated by solving equation (3.7).

A map of float vectors contains the moment values at each support used for calculating influence lines. The map uses floats as keys representing each possible ordinate location. Each vector is associated with a single ordinate location key. The vector indices are the indices of the internal supports. The values within the vectors are the moments at the corresponding internal support due to the loaded ordinate location. The use of a map allows the location keys to be non-integers values. While the final version of WIMFluence only uses integer values as possible ordinate locations, earlier versions did not. Using a map for the ordinate locations was necessary at an earlier time, and there was never sufficient reason to change it later.

Three multidimensional maps contain the influence lines for every analysis point. The keys of the outer maps are the floats representing analysis point locations. Since analysis point locations can be non-integer values, maps are needed here instead of vectors. Associated with each analysis point key is a map containing its corresponding influence line. The keys of the inner maps are floats representing ordinate locations. The associated values are the influence factors. As with the map containing support moments, the internal maps here were necessary when ordinate locations were allowed to be non-integer values.

Table 4.4. Composition of *bridge* data struct

bridge		
type	name	default
int	spans	0
float	length_span_one	0
float	length_total	0
vector<float>	span_lengths	
vector<float>	support_positions	
vector< vector<float> >	support_simple_set	
vector< vector<float> >	support_delta_set	
vector< vector<float> >	support_factors	
map<float, vector<float> >	support_influence	
map< float, map<float, float> >	shear_influence_left	
map< float, map<float, float> >	shear_influence_right	
map< float, map<float, float> >	moment_influence	

MAIN BODY

While WIMFluence does have distinct major operational sections, many aspects of the code do not specifically fit into any of these major operational sections. Therefore, they are considered to simply be parts of the program's main body. The main body of the program handles user interaction, output file initializations, operation of the major sections, and the file closings. Some parts of the code fit into similar categories within the main body but are executed at different times. Description of the code here follows the order of execution instead of a categorical organization. The order of execution is designed in part according to logistical constraints, in part to aide code readability, and in part based on arbitrary decisions.

Variable Declarations

The first section is a set of variable declarations. File variables are declared for the truck input file name, the truck input file, the influence line output file, the reformatted truck output file, and the extreme response output file. An instance of the *bridge* struct is declared to contain

the bridge intended for analysis. A string variable is declared to contain each line of the truck input file as it is read.

User Input

Input files for WIMFluence are to be inside the folder "input" within the same directory as WIMFluence. Output files are written inside the folder "output" within the same directory as WIMFluence. In case these folders do not exist, WIMFluence attempts to create them. If either folder is created, the user is notified. If the folders already exist, the program proceeds to request a truck input file. A request is displayed for the truck input file name and the program waits for the user's input. After a name is given, WIMFluence attempts to open the file. If the file is not found, the request is repeated until a valid input name is provided. At the successful file opening, the reading of the file is advanced by one line. This prevents an assumed header line from being used as a truck data line and causing a crash. As the input file can be a text file with any file extension, the file extension must be included when the user inputs the name. However, this extension is not required for later operations, so it is removed from within the name variable. An example for the user input process is shown in Figure 4.2.

WIMFluence is designed to use feet and kips as dimensional units. However, initial WIM files were provided with decimeters and tenths of a metric ton as dimensional units. Since the files containing the legal loads use feet and kips, WIMFluence needs to know which unit system the truck set in question is using. The program asks a yes or no question to determine if unit conversions are required. For TRC1701, the final set of WIM files used feet and kips. However, the feature to convert the dimensional units if necessary is available.

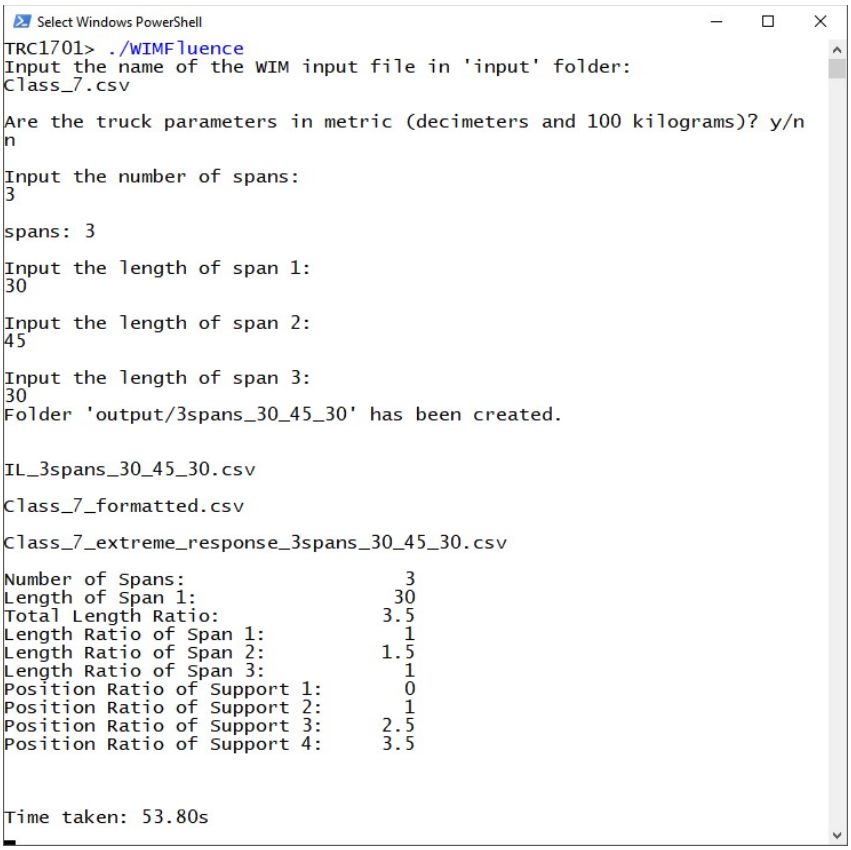

Figure 4.2. User input and WIMFluence operation example.

Next, the user inputs the bridge configuration. A request is displayed for the number of spans and the span lengths. All applicable members of the *bridge* struct are updated for each user response.

Output File Initialization

The output file names are created automatically based on the truck set and bridge configuration. Therefore, the files are initialized after both have been provided from the user. Output files are organized in bridge-specific output folder subdirectories. Of the output files, the influence line output file is initialized first. Therefore, its initialization function also handles the directory creation. A directory name is constructed from the bridge configuration. WIMFluence attempts to create a directory with the constructed name. If the directory is created, the user is notified. If the directory already exists, WIMFluence proceeds to initialize the output files. For

the influence line file, a name including the bridge configuration is generated. The corresponding file is created and opened. For the reformatted truck file, a name including the original truck set name is generated. A file is created using this name and opened. Since the extreme responses are dependent on both the truck set and the bridge configuration, a name is generated accordingly. The file is then created and opened.

Each of the output files has information that is provided at the beginning of the file. The bridge configuration is included in the top lines of the extreme response file. The necessary column headers are written to the reformatted truck file and the extreme response file. The header for the influence line file is created later.

Major Operational Sections

WIMFluence's runtime provided useful information for developing the program. The runtime information helped determine the level of analysis detail that was practical and what alterations were needed to satisfy the goals of TRC1701 within the project's time constraints. The runtime of WIMFluence's major operational sections is determined using a timer. Timing begins after all user interaction is completed to prevent skewed runtime results due to a variation in user interaction.

At this point in the WIMFluence calculation process, the three major WIMFluence operational sections of WIMFluence are executed. If the bridge is a multi-span continuous bridge, the influence line correction factors are determined. Influence lines are constructed for every analysis point along the bridge. These are written to the output file and the file is closed. A map of *extreme_set* data structs is initialized to contain all extreme response data. The map's

keys are floats representing analysis point locations. The extreme responses produced by the truck set are determined and written to the file.

Program Conclusion

After the WIMFluence bridge response calculations are complete, all files still open are closed. WIMFluence displays the runtime of the major operational sections. The program waits for any key to be pressed to allow the user time to review any displayed information if desired. When a key is pressed, the program terminates. The runtime of a single instance of WIMFluence is dependent on the bridge length, number of bridge spans, and number of trucks. During TRC1701, runtimes ranged from less than one second to over two days due to different input combinations.

INFLUENCE LINE CORRECTION FACTORS

Correction factors for the influence lines must be found for multi-span continuous bridges. As described in Chapter 3, the calculations for correction factors are accomplished by restricting deflections at supports to be zero and solving sets of linear equations. These systems are comprised of vectors of deflection values, a matrix of deflection ratios, and vectors of correction factors. The calculation process is divided into three sections. The construction of the deflection value vectors is carried out first. The matrix of deflection ratios is constructed next. Finally, the systems of linear equations are formed and solved for the correction factors.

Construction of the Simple Sets

Within WIMFluence, the vectors of deflection values are referred to as the simple sets. The name is a reference to their calculation by assuming the entire bridge to be a single simple span with unit rotations. A for-loop is used to loop through all internal supports and apply a hinge and

unit rotation at the internal support. A nested for-loop is used to calculate the resulting deflection at each internal support using Equation (3.3). Each deflection is pushed onto the end of a vector dedicated to the hinged internal support. After all the deflections have been calculated for the hinged support, the deflection vector is pushed onto the vector of simple-set vectors within the *bridge* struct.

Construction of the Delta Set

The matrix of deflection ratios is referred to as the delta set within WIMFluence. The deflections and ratios are calculated using Equations (3.4) and (3.5). Similar to the simple sets, the delta set calculation is comprised of an inner for-loop nested within an outer for-loop. The outer loop determines which support location receives a unit load to induce deflections. The inner loop determines the support for which the deflection and ratio are found. The two deflections and the ratio are calculated in the inner loop. Each ratio is pushed onto a vector dedicated to the loaded internal support. At the inner loop completion, the ratio vector is pushed onto the vector of delta set vectors inside the *bridge* struct.

Calculation of Correction Factors

To solve the systems of equations, the simple set values must be multiplied by -1. A nested for-loop walks through the *bridge* struct's simple set values and sets each to its negative. This allows Equation (3.7) to be constructed. For each internal support, an augmented matrix is constructed and solved by Gaussian elimination. A multidimensional vector of vectors is initialized as a copy of the *bridge* struct's delta set. The simple set for the internal support in question is pushed onto the end of this new vector to form the augmented matrix.

Row operations are used to convert the augmented matrix into reduced row echelon form. Functions are defined for both row multiplication and row addition. A nested for-loop walks through the matrix to change elements below the diagonal into zeroes through row multiplication and row addition. A second nested for-loop changes the elements above the diagonal into zeros. A final for-loop uses row multiplication to change elements along the diagonal into ones. The last vector is the solution for the correction factors. This vector is pushed onto the correction factor vector of vectors within the *bridge* struct.

INFLUENCE LINES

Developing influence lines is performed according to Chapter 3. A for-loop within a nested for-loop walks through all combinations of axle load point ordinate locations and influence line analysis points to calculate influence factors. The outermost loop determines the ordinate location. The ordinate location is incremented by one foot after each outermost loop iteration. The middle loop determines which bridge span the analysis point is within. The innermost loop determines the analysis point within the span. The analysis point is incremented by one twentieth of the length of the containing span after each innermost loop iteration.

At the beginning of the outer loop, the moments due to a unit load at the ordinate location are calculated for each support using Equation (3.8). The influence line values for the left end support are calculated. Since the left end analysis point is at a support, the moment is set equal to the corresponding value in the *bridge* struct's support moment influence lines. At the left end support, left shear is immediately assumed to be zero and the right shear is calculated with Equation (3.16).

In the innermost for-loop, the analysis point is calculated by multiplying the span length by one twentieth of the loop counter and then adding the calculated value to the left span support position. If the analysis point is at a support, the moment is pulled from the support moment influence lines in the *bridge* struct. If the analysis point is not at a support, the moment is calculated using Equation (3.12). If the analysis point is at a support, the left shear and right shear are calculated by Equations (3.15) and (3.16), respectively. If the analysis point is at the right end of the bridge, a zero right shear is assumed. If the analysis point is within a span, the left shear and right shear are calculated by Equations (3.19) and (3.20), respectively. These equations are implemented with a series of conditional statements to determine which sub-functions to use. Figure 4.3 shows the organization for the moment calculation function. Figure 4.4 shows the organization for the shear calculation function. After the moment values are calculated, they are made dimensionless quantities through Equation (3.21).

The dimensionless influence lines are written to the influence line output file. The bridge info is written in the file header. The influence line column headers are written below that. A nested for-loop walks through every combination of analysis points and ordinate locations. For each combination, the analysis point, the ratio of the analysis point to the length of the first span, the ordinate location, and the three responses are written to a new line. The output file is closed after all influence lines have been recorded.

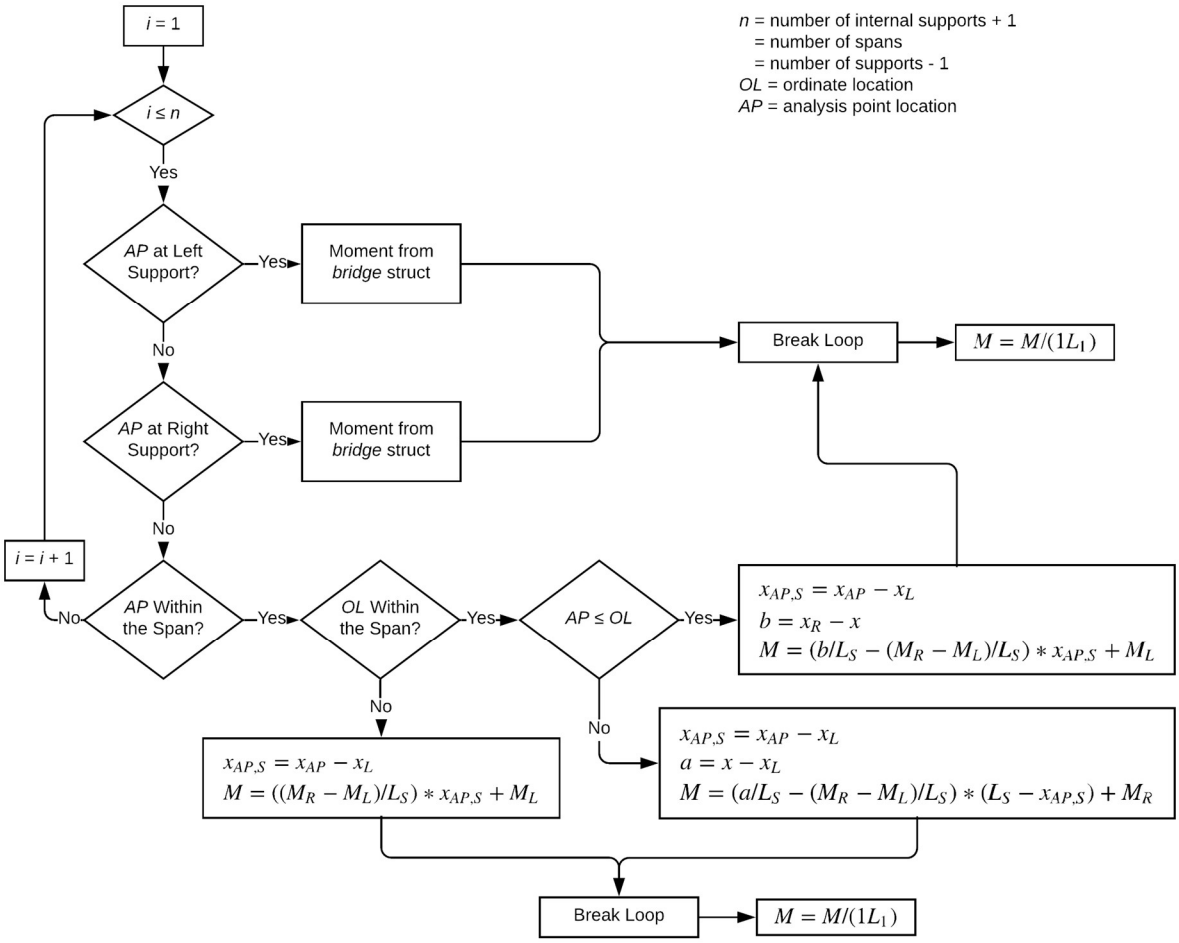

Figure 4.3. WIMFluence function for influence line moment calculations

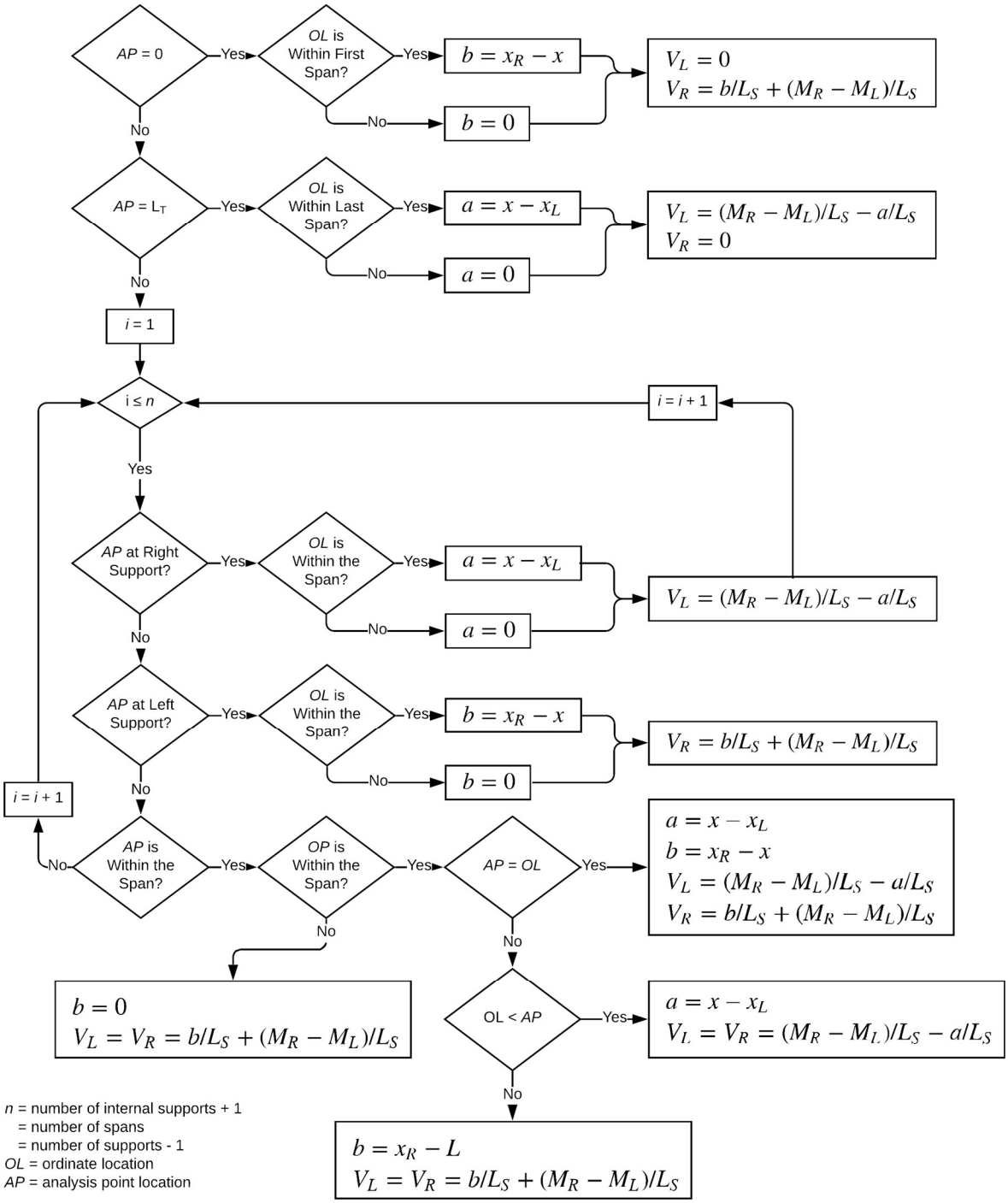

Figure 4.4. WIMFluence function for influence line shear calculations

EXTREME RESPONSE DETERMINATION

The final major operational section calculates the extreme response values. Extreme responses and their corresponding truck configurations are found for every analysis point within the bridge. A map of *extreme_set* data structs is created. The map's keys are floats representing the locations of analysis points.

A counter is initialized to track the truck number as the trucks are read from the input file. For each line of truck data in the file, a new truck is effectively driven across the bridge to determine the response values it produces and the extreme response values of the entire truck set. The counter is incremented such that the first truck has a truck number of 1. A *truck* data struct is initialized with the information in the input file line. The first value in the line is the number of truck axles. A default *axle* struct is initialized and pushed onto the *truck*'s *axle* vector. The second value in the input line is the first axle weight. This weight and a position of zero are stored in the *axle*. Subsequent values in the input file are alternating axle spacings and axle weights. These are used to construct new instances of the *axle* struct. Each new instance is pushed onto the *axle* vector until the truck is complete. If the truck set uses metric units, the units are converted to feet and kips. Axle positions are approximated to force axles to align with ordinate locations.

After a new truck is constructed based on the axle weights and ordinate locations, it is written to the reformatted truck output file. Each file line is an axle of a truck instead of an entire truck. A for-loop walks through all the truck's axles and writes the truck number, number of axles, axle weight, and axle position to the file.

A second truck is initialized as a reversed version of the first truck. The two trucks are moved across the bridge simultaneously. Only a single truck at a time is used to load the bridge. The trucks are placed such that the front axle of the forward-facing truck and the rear axle of the backward-facing truck coincide at the initial external support. For every analysis point, the resulting moment, left shear, and right shear from the forward-facing truck are calculated. Each response is compared to the previous extreme value within the analysis point's *extreme_set* struct. If the new response is greater, the value and corresponding truck configuration is updated. This process is repeated for the reversed truck version. After responses have been calculated for every analysis point, each truck is moved to the right along the bridge by one foot. Responses and comparisons are calculated for every analysis point again. This process is continued until there are no axles positioned on the bridge.

When there are no axles positioned on the bridge, analysis for the truck is complete. The program moves on to the next truck in the sequence. A new truck is initialized from the next line of the truck input file and this new truck is used for the response analysis. This process continues until all trucks within the input file have been analyzed.

The extreme responses for the entire bridge are written to the extreme response output file. A new line of data is written for each extreme response and analysis point combination. The truck number, truck direction, first axle position, analysis point, moment, left shear, and right shear are each written in turn. Since each extreme response value receives its own data line, only one of the three response columns (moment, left shear, right shear) has a numerical value per line. The other two response values have "NaN" written to them. After extreme responses have been written to the file, the extreme response calculation section is complete. WIMFluence returns to the main body portion to execute program conclusion pieces.

CODE RECOMMENDATIONS

WIMFluence can potentially greatly benefit from parallelization. For small sets of trucks such as the legal load sets, the runtime is a fraction of a second. However, for larger truck sets, WIMFluence can take hours or days to complete. The influence line calculations account for an insignificant portion of the runtime. Conversely, the calculation of extreme response values accounts for most of the runtime. WIMFluence can greatly benefit from the parallelization of the extreme response calculations. However, the potential gain was irrelevant to TRC1701. For the study, WIMFluence was executed at the Arkansas High Performance Computer Center (AHPCC) at the University of Arkansas. This allowed many instances of WIMFluence to be executed simultaneously rather than in series. The lack of parallelization allowed instances of WIMFluence to require only a single computer core, which allowed WIMFluence to have shorter wait times in the AHPCC queue. Consequently, in such a setting, the benefits of parallelization were insignificant, but for a single user instance of WIMFluence, the potential performance increase could be worth updating the program for parallelization.

5. EXTREME RESPONSE ANALYSIS – PYTHON SCRIPTS

WIMFluence is accompanied by three Python scripts for analysis of extreme response values. The first script develops extreme response ratios between each truck set and a legal load set in question. The second and third scripts consolidate the ratios produced by different truck sets. The second script consolidates the maximum problematic ratios for each individual bridge configuration and for all bridge configurations together. The third script consolidates all problematic ratios for all bridge configurations together. Unlike with WIMFluence, these scripts are designed solely for TRC1701. Therefore, the scripts would need to be modified for other situations. Each script assumes that WIMFluence and every previous script has already been run and that the files produced by them have not been modified in any manner.

Matplotlib, NumPy, and pandas are the main Python modules in the scripts. Matplotlib is a graphics library (Hunter 2007) used for plotting data. NumPy is a scientific computing library (Oliphant 2006). pandas is a library providing data structures and data analysis tools (McKinney 2010). Its DataFrame data structure is comparable to a relational database. pandas provides methods for manipulating DataFrames in similar manners to the capabilities provided by Structured Query Language (SQL) for relational databases.

Ratios are chosen as the comparison metric for being dimensionless and treating one value as a baseline. Having a baseline allows the specific comparison of actual trucks to legal loads. Some alternative metrics, such as percent differences, do not treat one value as a baseline. They instead compare the two values to each other through means such as the average. Simple differences of values allow for a baseline but do not provide a dimensionless metric. Ratios are therefore deemed the best metric for these comparisons.

RESPONSE RATIO DEVELOPMENT

The first Python script develops extreme response ratios at each analysis point for every truck set and bridge configuration combination. For each combination, this script produces a response ratio file for each of the three response types (positive moment, negative moment, and shear) and a figure depicting the three response ratio sets. These four files are saved in their corresponding bridge-specific output subdirectory. A list of violation Booleans is saved in the main output directory. These Booleans indicate the presence of response ratios greater than 1 for their corresponding truck set and bridge configuration combinations. The response ratio development process is shown in Figure 5.1.

A hardcoded string determines which truck set is considered as the baseline to compare other truck sets against. This string is the name of the truck set file, without the file extension, as it was when processed by WIMFluence. This string had values of “ArDOT" and “AASHTO” for comparisons to the ARDOT and AASHTO legal loads during TRC1701. The output file for violation Booleans is initialized to be written to as the script processes each truck set and bridge configuration combination.

A for-loop walks through the subdirectories in the output directory. If a subdirectory does not contain extreme response files produced by WIMFluence, the script moves to the next subdirectory. If extreme response files are found, the script reconstructs the bridge information from the influence line file name. The script then finds all truck sets present in the subdirectory other than the baseline set. If no other truck sets are present, the script proceeds to the next subdirectory. If other truck sets are present, the script compares them to the baseline set.

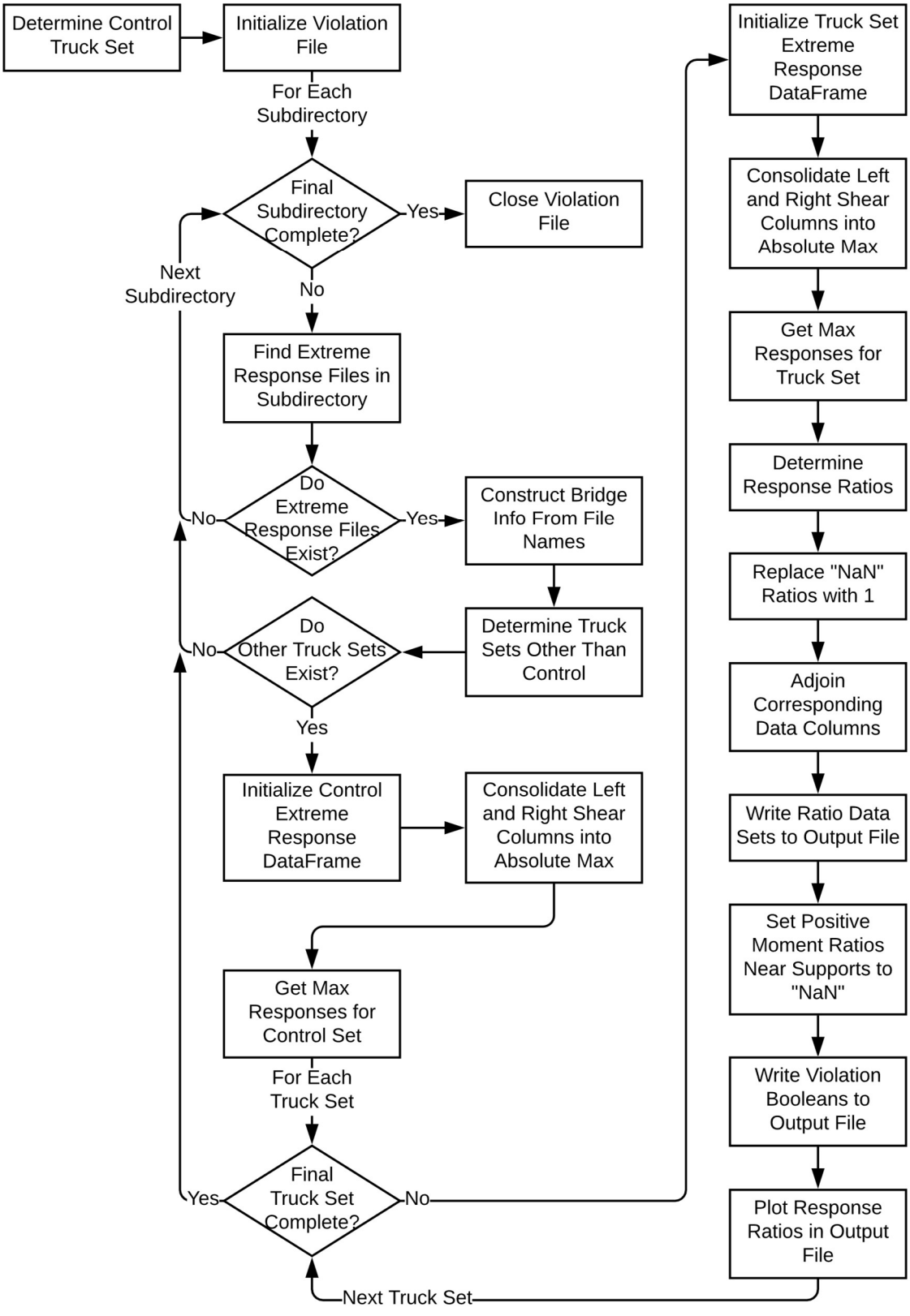

Figure 5.1. Script for initial response comparison

To compare extreme response sets, a DataFrame is initialized from the baseline set's extreme response file. WIMFluence treats left shear and right shear as different response types. This script consolidates the two shear types into a single column in the DataFrame. The shear column contains the maximums between the absolute values of the left shear and right shear values. The DataFrame is then split into three response-specific DataFrames containing the maximum values at every analysis point for each response type (positive moment, negative moment, and shear). Analysis point locations are the indices of the three response-specific DataFrames.

A for-loop compares the extreme responses from each truck set to those of the baseline set. A DataFrame is initialized from the extreme response file of the truck set in question. As with the baseline set, the left and right shear are consolidated and the DataFrame is split into three response-specific DataFrames for the maximum response values. Analysis point locations are the indices.

Three new response-specific DataFrames are declared for the response ratios between the truck set in question and the baseline set. In each of these DataFrames, the analysis point locations serve as indices. Using the analysis point locations as indices in the response ratio DataFrames and the maximum response DataFrames allows calculations and data movement without needing to otherwise specify which DataFrame record corresponds to which.

A new column is created in the ratio DataFrames containing nondimensionalized analysis point locations. The integer part of these values refers to the bridge support preceding the analysis point. For example, a 0 refers to the first or left-most external support and a 1 refers to the first internal support. The fractional part is a percentage indicating how far into the containing bridge span the analysis point is. Therefore a 1.4 indicates 40% into the second span.

Response ratio columns are added to the ratio DataFrames by dividing the truck set response column by the baseline set response column. A zero divided by zero produces a "NaN" value. These values are replaced with 1s to indicate the corresponding response values are equal. Data corresponding to the ratios are joined to the ratio DataFrames. These include response values and the truck information needed to recreate the response values. The ratio DataFrames are then saved as comma-separated values (CSV) files in the corresponding bridge subdirectory of the output folder.

Maximum positive moment values produced by a set of trucks are smaller near bridge supports than near bridge span centers. If the value produced by the baseline set is sufficiently smaller than the antecedent value, the resulting ratio will be misleadingly problematic despite the corresponding response values being insignificant. To prevent such misleading ratios from skewing the analysis within TRC1701, positive moment ratio values near bridge supports are replaced with "NaN" values. The replaced values are those at a support and the three analysis points on each side of the support.

If any remaining ratio value across the three response types is greater than one, the current truck set bridge configuration combination has an issue. The presence of such a violating ratio is stored as a Boolean. This Boolean, the maximum ratios of each response type, the data corresponding to the ratios, the bridge configuration, and the truck set in question are written to the violation output file.

The three response type ratio sets are plotted in a single figure. This figure is saved in the bridge-specific subdirectory of the output folder. The script then compares the next truck set in the subdirectory to the baseline set. After comparing all truck sets in the subdirectory, the script

proceeds to the next subdirectory to compare its extreme response sets. After all subdirectories of the output folder are processed, the violation output file is closed, and the script is finished.

MAXIMUM PROBLEMATIC RATIOS ACROSS ALL TRUCK CLASSES

A second Python script consolidates all comparison ratios into sets of maximum problematic ratios across all bridge configurations. For every bridge configuration, the script creates a file containing the maximum ratios for each response type at every analysis point and a figure plotting these maximum ratios. These two files are saved in their corresponding bridge-specific subdirectory of the output folder. A table of maximum problematic ratios at each analysis point for every bridge configuration is saved in the output folder. This table is visually reminiscent of bridge spans with problematic ratios appearing at their corresponding positions along each bridge. This script is limited to consolidating response ratios produced by truck sets following the naming scheme in TRC1701. The truck sets being compared to the baseline set must contain "Class_" within their filenames. This ensures the script only consolidates ratio files produced by the first script. The consolidation process of this second script is shown in Figure 5.2.

As with the first script, a hardcoded string determines which truck set is considered the baseline to compare other truck sets against. During TRC1701, values of "ArDOT" and "AASHTO" allowed for comparison to the ARDOT and AASHTO legal loads, respectively.

A DataFrame is declared for each response type to contain maximum problematic ratios for all bridge configurations. The three DataFrames contain the script's main output after all bridge configurations have been processed. The indices of these DataFrames are the dimensionless analysis point locations for a six-span bridge. This allows data from any bridge-specific DataFrame, for bridges up to six spans, to be joined to the all-bridge DataFrames.

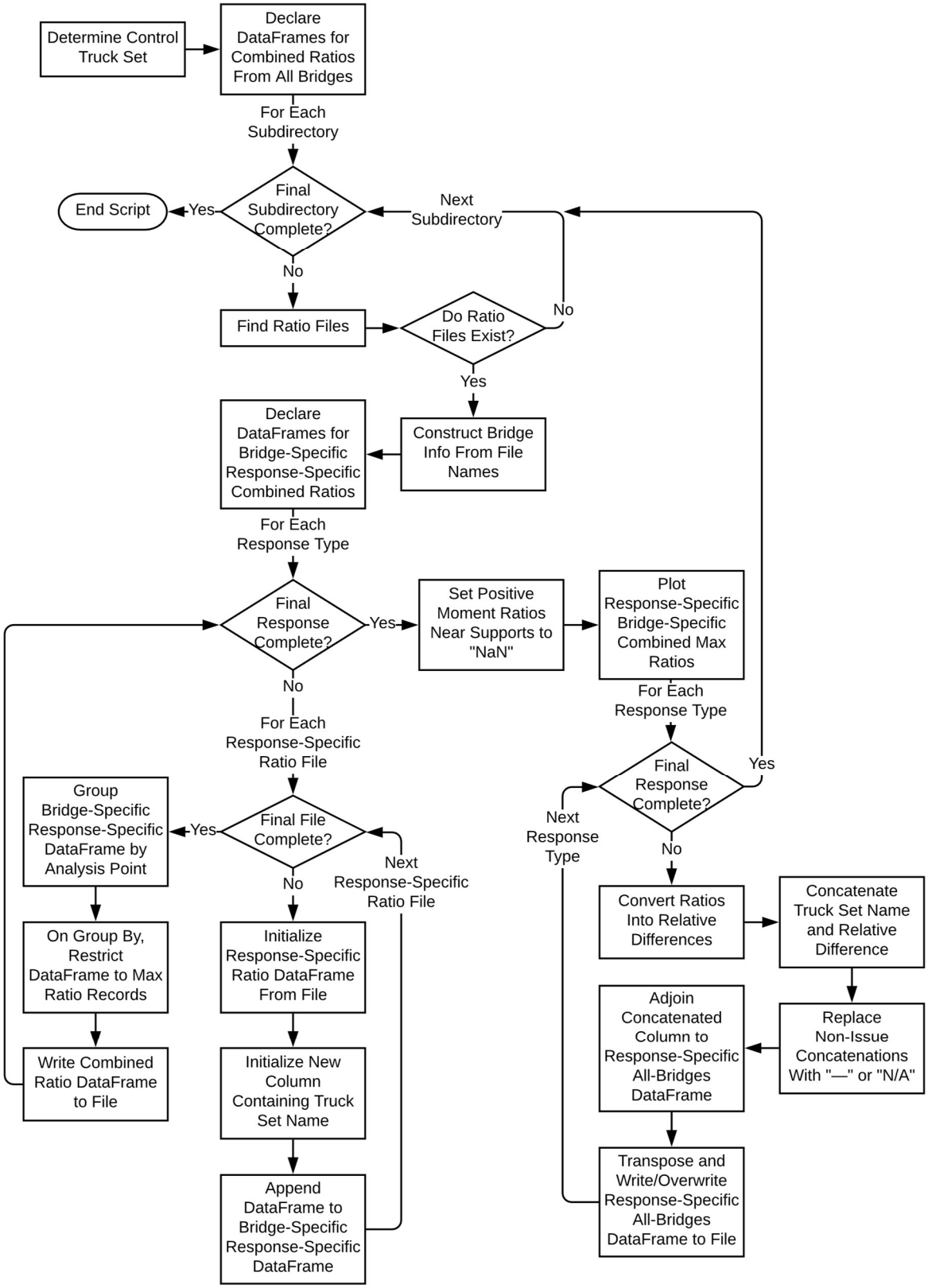

Figure 5.2. Script to combine ratios produced by different truck sets

A for-loop walks through the subdirectories in the output directory. For each response type, a list enumerates the response ratio files produced by the first script within the current subdirectory. If a subdirectory does not contain response ratio files, this script moves to the next subdirectory. If response ratio files are found, the script reconstructs the bridge information from the influence line file name. Blank DataFrames are declared for each response type to combine bridge-specific ratios in.

For each response type, the response ratios are consolidated in the bridge-specific DataFrame. A for-loop opens each ratio file in the corresponding ratio file list as a DataFrame. Column names specific to truck sets are replaced with generic names. A new column is initialized containing the name of the truck set producing the current set of ratios. The set-specific ratio DataFrame is appended to the bridge-specific DataFrame. The for-loop then opens the next ratio file as a DataFrame, modifies the DataFrame, and appends the DataFrame to the bridge-specific DataFrame. This continues until all ratio sets in the response-specific list have been appended.

The bridge-specific ratio DataFrame is restricted to records containing the maximum ratio for each analysis point. Grouping the DataFrame by the analysis point column allows the maximum ratio to be determined for each analysis point. Records containing the maximum ratios are retained and other records are removed. The DataFrame is written as a CSV file to the corresponding bridge-specific subdirectory of the output folder. This process from opening listed response ratio files as DataFrames to writing the DataFrame containing maximum ratios is performed for positive moment ratios, negative moment ratios, and shear ratios in turn.

As in the first script, positive moment ratios near supports are replaced with “NaN” values. This prevents the ratios from appearing as problematic despite insignificant corresponding

response values. The maximum ratios for each response type are plotted to figures. The response-specific figures are saved in the bridge-specific subdirectory of the output folder.

The main output of this second script gives the corresponding relative difference for maximum problematic response ratios. For each response type, the response ratios in the bridge-specific DataFrames are converted to relative differences using Equation (5.1). To retain the source of problematic ratios, the corresponding truck set name is concatenated with the relative difference value. Concatenations with a non-problematic corresponding response ratio are replaced with an em dash. In the positive moment bridge-specific DataFrame, concatenations corresponding to an analysis point near a support are replaced with "N/A" to indicate instances where misleading response ratios are possible.

$$d = (r - 1) * 100\% \tag{5.1}$$

- d: relative difference between response values
- r: response ratio

The column of concatenations in the bridge-specific DataFrame is joined to the all-bridge DataFrame as a new column named after the corresponding bridge configuration. The all-bridge DataFrame is transposed and written as a CSV to the output folder, overwriting previous versions of the file if necessary. After the relative difference calculations, concatenations, and writing to a file are complete for all three response types, the script proceeds to the next subdirectory to process the next bridge configuration. Transposing the all-bridge DataFrames causes bridge configuration names to serve as indices and dimensionless analysis point locations to serve as column names. Writing the all-bridge DataFrames to files after each bridge is processed allows the output files to contain partial data in the case of a crash. After all bridge configurations have been processed, the second script is complete.

ALL PROBLEMATIC RATIOS ACROSS ALL TRUCK CLASSES

A third Python script consolidates the response ratios from all compared truck sets into sets of all problematic response ratios across all bridges. For each response type, this script outputs a table of all problematic response ratios and their corresponding data. Like the second script, this script is limited to truck sets following the naming scheme used in TRC1701. "Class_" must be contained within the filenames. The development of these tables is shown in Figure 5.3.

As in the previous two scripts, a hardcoded string determines which truck set is considered the baseline to compare other truck sets against. During TRC1701, string values of "ArDOT" and "AASHTO" allowed for comparison to the ARDOT and AASHTO legal loads, respectively.

For each response type, a blank DataFrame is declared to contain all problematic response ratios and corresponding data for all bridge configurations. These three DataFrames contain the script's output tables after all bridge configurations have been processed. A blank dictionary is declared to contain truck set DataFrames. Storing truck set DataFrames in memory allows each truck set to be opened a single time. This decreases the runtime and increases the memory requirements when compared to opening each truck set every time it is needed (once for every bridge configuration). This tradeoff was deemed preferable during TRC1701.

A for-loop walks through the subdirectories in the output directory. For each response type, a list enumerates the response ratio files produced by the first script within the current subdirectory. A fourth list enumerates the formatted truck files produced by WIMFluence. If a subdirectory does not contain response ratio files, the script moves to the next subdirectory. If response ratio files are found, the script reconstructs the bridge information from the influence

line file name. A blank DataFrame is declared for each response type to combine bridge-specific response ratio data in.

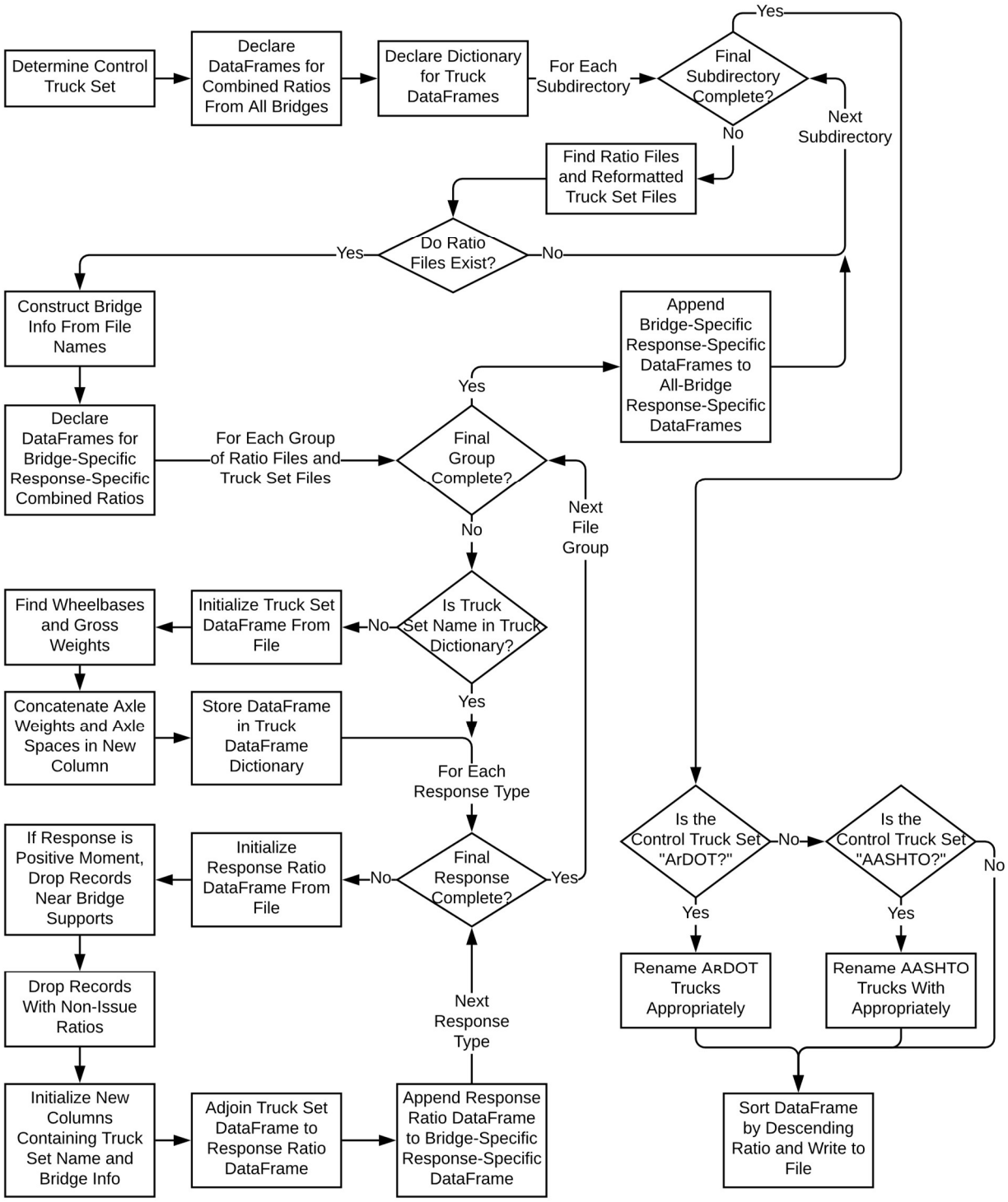

Figure 5.3. Script to combine ratios produced by different truck sets in a concise table

The four lists are zipped together such that elements in the same position in each list correspond to the same original truck set. A for-loop walks through each tuple of the aggregate sequence to initialize truck set DataFrames and consolidate problematic response ratio data into the bridge-specific DataFrames. In this loop, the script first determines if the truck set DataFrame corresponding to the current formatted truck file is in the truck set dictionary.

If the truck set DataFrame is already stored, the script proceeds to consolidate problematic response ratio data. If the truck set DataFrame is not already stored, it is generated. The formatted truck file is opened as a DataFrame. The gross weight for each truck is found by summing the axle weight column when grouping the DataFrame by the truck number. In the formatted truck files, axle positions are measured relative to the frontmost axle with the front axle having a position of zero. The rearmost axle has the largest negative, and therefore minimum, relative position value. The wheelbase is determined by taking the negative minimum of the relative axle position column when grouping the DataFrame by the truck number.

In the output table, truck configurations are detailed by their axle spaces and axle weights in a single column. Axle spaces are determined from axle positions when grouping the DataFrame by the truck number. The spaces are concatenated and stored in a pandas Series, a labeled array. Axle weights are contained in the truck set DataFrame. Axle weights are concatenated for each truck configuration and stored in a Series. Axle spaces and axle weights are concatenated together along with their respective unit symbols in a Series. These concatenations allow each truck configuration to be entirely described in a single element of a Series or DataFrame. The concatenated truck configurations, gross weights, wheelbases, and truck numbers are combined in a new truck set DataFrame. This DataFrame is stored in the truck set dictionary. The DataFrame's dictionary key is the name of its contained truck set.

After checking for the truck set DataFrame and constructing it if necessary, the script consolidates the problematic response ratio data for the current truck set. The consolidation is performed for each response type in turn. A DataFrame is initialized from the current response ratio file. Truck set-specific column names are replaced with generic names to allow appending to other DataFrames. Columns unnecessary for the final output table are dropped. These columns are the first axle position columns and truck direction columns. If the current response type is positive moment, records for analysis points near bridge supports are dropped. This prevents misleading response ratios born of small response values from appearing in the output table. Records with non-problematic ratios, or ratios less than or equal to 1, are dropped to limit output data to problematic ratios. Two new columns are created for the truck set name and the bridge configuration description. The corresponding truck set DataFrame from the dictionary is joined to the problematic ratio DataFrame on the truck numbers. The combined DataFrame is appended to the bridge-specific DataFrame. This consolidation is performed for positive moment ratios, negative moment ratios, and shear ratios in turn.

After every response type has been processed, the script proceeds to the next tuple of response ratio files and truck set file. After every tuple from the current subdirectory is processed, the bridge-specific DataFrames are appended to their corresponding all-bridge DataFrames and the script proceeds to the next subdirectory. Once all subdirectories are processed, all consolidation is complete. The all-bridge DataFrames contain all problematic ratios from all truck sets across all bridge configurations.

In the all-bridge DataFrames, columns containing control set truck numbers are renamed to describe named trucks. Control set truck numbers are replaced with their corresponding name if the control set is either an ARDOT or AASHTO legal load set. If the control set is an ARDOT

legal load set, truck numbers 1, 2, and 3 are replaced with “Code_4”, “Code_9”, and “Code_5”, respectively. If the control set is an AASHTO legal load set, truck numbers 1-7 are replaced with “Type_3”, “Type_3S2”, “Type_3-3”, “Type_SU4”, “Type_SU5”, “Type_SU6”, and “Type_SU7”, respectively. The three all-bridge DataFrames are sorted by response ratio values in descending order and written to file in the output folder. This completes the third script, and all response and response ratio data are created.

SCRIPT DESIGN CONSIDERATIONS AND RECOMMENDATIONS

The three Python scripts detailed here were developed expressly for use in TRC1701. They should be reworked to be compatible with other data sets and alternate naming schemes. Parallelization within each script should be considered. However, parallelization may be unnecessary without as large numbers of truck sets (11) and bridge configurations (279) as were analyzed in TRC1701. Since the three scripts are run in immediate succession, they can be combined as a single script. While this would simplify operation, it may be more difficult to find any potential computer code glitches.

6. SOURCE CODE AND MANUAL ACCESS

WIMFluence and its supporting scripts are open source. The source code, user manual, and copyright information can be found in a zipped folder, WIMFluenceSourceAndManual.zip, on ProQuest along with this thesis. The folder contains a subdirectory for the WIMFluence program source code and another for the supporting scripts source code. The user manual and copyright files are found in the main folder. The files can also be found at https://github.com/kppasley/WIMFluenceArchive. This repository is intended to maintain the WIMFluence files as they were at the conclusion of TRC1701. For future versions of WIMFluence, see the repository at https://github.com/kppasley/WIMFluence. Future versions of WIMFluence are not guaranteed to function the same as the version presented in this thesis. Each version can be cloned or forked from GitHub for use and modification elsewhere or for contribution to future development of WIMFluence.

7. CONCLUSION

Developing WIMFluence and three Python scripts was necessary to satisfy the TRC1701 project objectives. These computer codes developed for the project enabled the project investigators to compare Arkansas's actual truck traffic from WIM data to the ARDOT and AASHTO legal load sets. This comparison required the response analysis of approximately one million unique truck configurations across 279 different bridge configurations. The considered bridge configurations included single span bridges to six-span continuous bridges.

During TRC1701, several classes of Arkansas's actual trucks could have been removed from the study due to never producing extreme response values greater than those produced by other classes. This was unknown prior to the analyses and removal of those classes in future studies is not advised due to potentially different results.

Through WIMFluence and its corresponding scripts, recommendations were developed for revising the ARDOT legal loads. ARDOT was recommended to include the AASHTO Type 3, Type 3S2, Type 3-3, Type SU6, and Type SU7 legal loads and increase the ARDOT Code 9 and Code 5 axle loads by 10%. The ARDOT Code 4 was determined to be unwarranted due to never producing extreme response values greater than the Code 9 and Code 5 extreme response values in any of the study cases (Heymsfield, Hernandez and Pasley 2018).

WIMFluence calculates the bridge responses due to a set of truck configurations en masse. This permitted Heymsfield, Hernandez, and Pasley (2018) to consider the effects of all of Arkansas's truck traffic. The subsequent Python scripts compare the responses from the actual traffic to the responses from the ARDOT and AASHTO legal loads and summarize the results in organized fashions. This allowed Heymsfield, Hernandez, and Pasley (2018) to determine if the

ARDOT legal loads enveloped the responses of Arkansas's actual truck traffic and how to update the ARDOT legal loads.

DESIGN RECOMMENDATIONS AND FUTURE REVISIONS

WIMFluence can greatly benefit from parallelization when responses are needed for a single bridge configuration. During TRC1701, multiple WIMFluence runs were made concurrently and therefore negated the need for parallelization. WIMFluence being a command line interface program allowed for simple operation of multiple program instances on high performance computers. The addition of a graphical user interface (GUI) will allow it to be easily operated in single-instance cases and lower the program's learning curve.

Theoretically, a learning application could be incorporated into WIMFluence to filter trucks producing non-extreme response values. However, the benefit may not be substantial enough to warrant the inclusion. This should be tested in future research or separate development.

The Python scripts were written solely for TRC1701 and have various parts hardcoded for that setting. They should be modified to be compatible with other use cases. The three scripts can also be combined as a single script. As with WIMFluence, the Python scripts can benefit from parallelization and GUIs. Due to short runtimes, parallelization was unnecessary during TRC1701. However, parallelization would still be useful. GUIs can allow a user to more easily track the progress of a script during runtime. GUIs can also simplify operation if the scripts are modified to be compatible with use cases other than TRC1701. The addition of GUIs in the Python scripts opens possibilities for immediately interactive final outputs. These could include graphing data in various forms, visualizing problematic trucks, visualizing problematic trucks superimposed on corresponding bridge configurations, and more. With these updates,

WIMFluence and the scripts can become significantly more useful in future research projects and other scenarios.

Bibliography

American Association of State Highway and Transportation Officials (AASHTO). 2011. *Manual for Bridge Evaluation (2nd Edition).* Washington, DC.

Bowman, Mark D, and Raymond Chou. 2014. *Review of Load Rating and Posting Procedures and Requirements.* West Lafayette, IN: Purdue University.

Code of Federal Regulations (CFR). 2011. ""Inspection Procedures," Title 23 - Highways, Part 650 - Bridges, Structures, and Hydraulics, §650.313 - Inspection Procedures." Washington, DC.

Eamon, Christopher D, and Sasan Siavashi. 2018. "Developing Representative Michigan Truck Configurations for Bridge Load Rating." Detroit, MI.

Fitzsimmons, Eric J, Thomas E Mulinazzi, and Steven D Schrock. 2014. "Economic Impact of Closing Structurally Deficient or Functionally Obsolete Bridges on Very Low-Volume Roads." *Transportation Research Record: Journal of the Transportation Research Board.* Washington, DC: Transportation Research Board.

Hernandez, Eric M. 2014. *Statistical Analysis of Weigh-in-Motion Data for Bridge Design in Vermont.* Burlington, VT: UVM Transportation Center.

Heymsfield, Ernie, Sarah Hernandez, and Kenneth Pasley. 2018. "Bridge Load Posting Based on Actual Arkansas Truck Traffic." Final Report, Fayetteville, AR.

Hibbeler, Russell C. 2011. *Statics and Mechanics of Materials 3rd Edition.* Upper Saddle River, NJ: Prentice Hall.

Hunter, John D. 2007. "Matplotlib: A 2D Graphics Environment." *Computing in Science & Engineering* (Institute of Electrical and Electronics Engineers) 90-95.

Linz, Stewart, interview by Ernie Heymsfield. 2016. (November 3).

McKinney, Wes. 2010. "Data Structures for Statistical Computing in Python." *Proceedings of the 9th Python in Science Conference.* Austin, TX. 51-56.

Michalos, James P, and Edward N Wilson. 1965. *Structural Mechanics and Analysis.* New York: Macmillan Publishing.

Oliphant, Travis E. 2006. *Gude to NumPy.* USA: Trelgol Publishing.

Sivakumar, Bala, Fred Moses, Gongkang Fu, and Michel Ghosn. 2007. *Legal Truck Loads and AASHTO Legal Loads for Posting.* NCHRP Report 575, Washington, DC: Transportation Research Board.

Sivakumar, Bala, Michel Ghosn, and Fred Moses. 2011. *Protocols for Collecting and Using Traffic Data in Bridge Design.* NCHRP Report 683, Washington, DC: Transportation Research Board.

Suparp, Suniti, and Panuwat Joyklad. 2013. "A Study on Simple Beam Bridge Responses Due to Thai Truck Loads." *Procedia - Social and Behavioral Sciences* (Elsevier Ltd.) 239-249.

APPENDIX A. FORMULATION OF EQUATION (3.3)

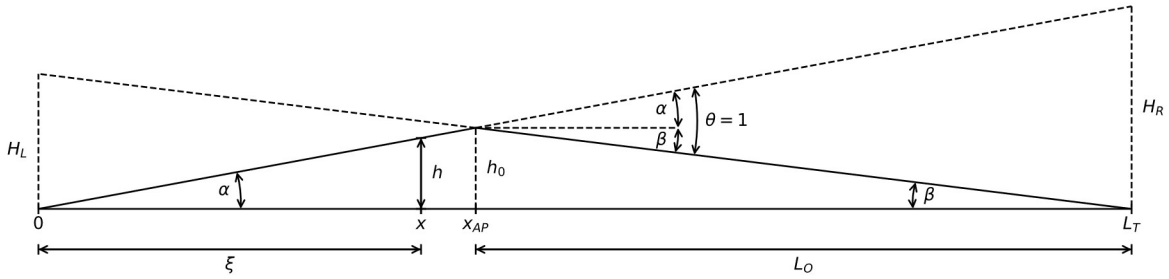

Figure A.2. Variable orientation for derivation of h in equation (3.3).

$$\alpha + \beta = 1 \tag{A.1}$$

Substituting the inverse tangent functions for α and β:

$$\arctan\left(\frac{h_0}{x_{Ap}}\right) + \arctan\left(\frac{h_0}{L_O}\right) = 1 \tag{A.2}$$

Using the small angle approximations, $\arctan(\theta) = \theta$, for α and β:

$$\frac{h_0}{x_{Ap}} + \frac{h_0}{L_O} = 1 \tag{A.3}$$

$$h_0 = \frac{x_{AP} * L_O}{L_O + x_{AP}} \tag{A.4}$$

Substituting L_T for $L_O + x_{AP}$:

$$h_0 = \frac{x_{AP} * L_O}{L_T} \tag{A.5}$$

Using similar triangles to solve for H_R:

$$\frac{h_0}{x_{AP}} = \frac{H_R}{L_T} \tag{A.6}$$

$$H_R = \frac{h_0 * L_T}{x_{AP}} \tag{A.7}$$

Substituting $\frac{x_{AP} * L_O}{L_T}$ for h_0:

$$H_R = \frac{x_{AP} * L_0}{L_T} * \frac{L_T}{x_{AP}} = L_O \tag{A.8}$$

Solving for h:

$$h = \frac{L_O}{L_T} * \xi \tag{3.3}$$