PREDICTING THE UNLOADING BEHAVIOR OF MONOTONICALLY AND LATERALLY LOADED SLENDER PILES USING THE FINITE ELEMENT METHOD AND P-Y CURVES

by

Peter George Hubbard

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Civil Engineering

Charlotte

2018

Approved by:

____
Dr. Miguel Pando

____
Dr. Youngjin Park

____
Dr. Harish Cherukuri

ABSTRACT

PETER GEORGE HUBBARD. Predicting the Unloading Behavior of Monotonically and Laterally Loaded Slender Piles Using Finite Element Method and P-Y Curves. (Under the direction of DR. MIGUEL PANDO)

The use of p-y curves is heavily engrained into geotechnical engineering practice. However, even with the prevalent use of p-y curves, these are only used to model the loading portion and not the unloading of a load test or loading demand in the field. A review of the literature shows that available software commonly used to model the problem of laterally loaded piles is used for loading and not unloading. Thus, there is a knowledge gap in the research for the unloading phase for laterally loaded piles. This thesis presents a method that has been developed to model the unloading behavior of a laterally loaded and unloaded single pile. A program has been developed using MATLAB that uses a 1D finite element model (FEM) for the pile and non-linear p-y curves to model the soil reaction in this soil-structure-interaction problem. The proposed method is based on a secant stiffness degradation model that models the soil and structure as a single system with a degrading stiffness. The method has been numerically implemented and validated with experimental data of a lateral load test involving loading and unloading.

DEDICATION

I would like to dedicate this work to my wife, Katie Hubbard. She has been a constant support throughout both my undergraduate and graduate careers at UNC Charlotte. I am grateful for her endless patience, encouragement, and uplifting spirit. Without her resilience and dedication, this work would not have been completed.

ACKNOWLEDGMENTS

The author would like to acknowledge and share his gratitude for UNC Charlotte's Energy Production and Infrastructure Center (EPIC), Dr. Miguel Pando, and his grandparents, Charles and Elizabeth Hubbard. He was awarded an EPIC Graduate Research Assistantship for the 2017-2018 academic year. The award provided funding and access to additional laboratories and office space, enabling him to complete the necessary work for this thesis. Dr. Miguel Pando has surpassed his role as a committee chair and advisor to this work. He has become a mentor who has aided in both the research and coursework involved in seeing this project come to completion, and his encouragement has been greatly appreciated. His grandparents, Charles and Elizabeth Hubbard, have financially supported his academic career and have given him the means to pursue his dream of graduate school.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1 Introduction

The objective of this study was to develop a model to predict the unloading behavior of laterally loaded piles using and expanding upon the p-y curve method. Although much research has been done to predict the behavior of piles subject to lateral loading during the loading phase, there is a knowledge gap on the behavior during unloading. This study was also to develop a computer program to implement the use of p-y curves for predicting pile behavior. Software referred to in this thesis as NinerPile was developed using the finite element method to approximately solve the governing differential equation while modeling the soil reaction as a series of p-y curves. An unloading model, referred to as the secant stiffness degradation method, was developed and implemented to model three lateral load tests published in the Federal Highway Report (FHWA-HRT-04-043) titled “A Laboratory and Field Study of Composite Piles for Bridge Substructures” released in March of 2006 (Pando et al. 2006). NinerPile allows for the user to have control over the pile material properties and locations of lateral loading. Flexural rigidity can be modeled as a function of moment, and lateral load can be applied at any location of depth along the pile. These abilities make NinerPile applicable for intricate load test configurations and composite piles. The method proposed and developed for this thesis has been shown to be able to be calibrated to effectively model complex cases, such as a composite pile under extreme lateral loading and unloading.

The unloading behavior of laterally loaded piles is important to assess the deterioration and long-term performance of many different types of infrastructure. This research was funded by the Energy Production Infrastructure Center (EPIC) with the application of offshore monopiles at the forefront. The lateral load tests analyzed herein took place in fluvial environments with sandy subsoils which is consistent with an offshore setting where load tests are much more difficult to perform. The permanent deflection of a pile after a monotonic lateral load is an important thing to monitor to predict the effects of intense short duration lateral loads that may influence energy infrastructure's efficiency and long-term performance.

1.2 Scope of research

The NinerPile software was developed using the equations derived in Chapter 3 of this thesis. First the program was developed to model the loading behavior of laterally loaded piles as currently available commercial software can do. This is validated and shown in Chapter 5 with comparisons between the famous LPILE software vs. NinerPile software for the lateral loading of three test piles. NinerPile was then expanded as a method was developed to model the lateral unloading behavior of the same piles.

Data from Pando et al (2006) were used in this study. Three composite piles; a prestressed reinforced concrete pile, a polyethylene pile and a fiber reinforced polymer pile were tested at the Route 351 bridge in Hamilton, Virginia. To investigate the potential for composite piles to be used in infrastructure in the state of Virginia, this was a necessary study because the previous Route 351 bridge had what was considered an unacceptable life span and ran into many challenges because of its repetitive loading and freeze-thaw conditions. Each pile published in the Federal Highway Report was used in this thesis to calibrate and test NinerPile.

Ensoft Inc.'s LPILE student version is the commercial software used for a comparative study in this thesis. NinerPile is only able to be verified in loading as there is no commercial software that uses p-y curves to predict the unloading of laterally loaded piles. NinerPile was shown to have a nearly identical loading calculation to LPILE while in addition being able to vary the flexural rigidity and loading location. The version of LPILE that was used to validate is LPILE Plus Student Version 2005 which is unable to model a change in flexural rigidity and is only able to be loaded at the pile head. The testing configurations require that the piles be loaded at various locations along their length which is typical for laterally loaded pile tests. The LPILE results, NinerPile results and field experimental results are compared in this thesis.

1.3 Organization of thesis

Chapter 1 provides an overview of the motivation and objectives of the research reported in this thesis. Chapter 2 is a literature review that focuses on p-y curves in geotechnical engineering practice, commercial software for predicting the loading and unloading behavior of laterally loaded piles, new research in the field of offshore pile design and the identified knowledge gaps. Chapter 3 describes the numerical methods of describing and solving the laterally loaded pile problem. This chapter includes a derivation of the governing differential equation and a derivation of the finite element method used to approximately solve the equation. Chapter 4 is a user manual for the NinerPile software developed for this thesis. It describes the process of how NinerPile functions as well as how a user is to interact with it. Chapter 5 is the results and verification of studies done using NinerPile to show its capabilities. This chapter shows the comparison between NinerPile and LPILE as well as comparisons of NinerPile, LPILE and experimental data collected in the Federal Highway research (Pando et. al, 2006). Chapter 6 reports the conclusions and findings of this study and makes recommendations on future work to expand and improve the NinerPile software.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

The objective of the literature review was to learn the state of the art of p-y curves in geotechnical engineering practice as applied to monotonic and low-frequency cyclic loading. An emphasis was placed on offshore monopile design for renewable energy infrastructure. Formulations for p-y curves for different soil types and loading environments are presented, and recent developments in this area are explored. Numerical methods to apply p-y curves for design of pile structures has also been studied. Commercial software that is currently used in industry is presented.

2.2 Loading of offshore wind turbine piles

The loading of offshore piles for wind turbines can be broken down into two categories: the operational loads and the environmental loads. The operational category refers to the inertial loading from the rotating turbine, while the environmental loading considers the wind and waves acting on the pile. The lateral loading from wind and waves makes up 80-90% of the overall lateral loading (Abadie, 2015). The wind loading contributes approximately 25% of the lateral loading for typical offshore turbine piles, while wave loading contributes 75% (Byrne and Houlsby, 2003). The typical dimensions and loading of an offshore wind turbine are shown in Figure 2.1.

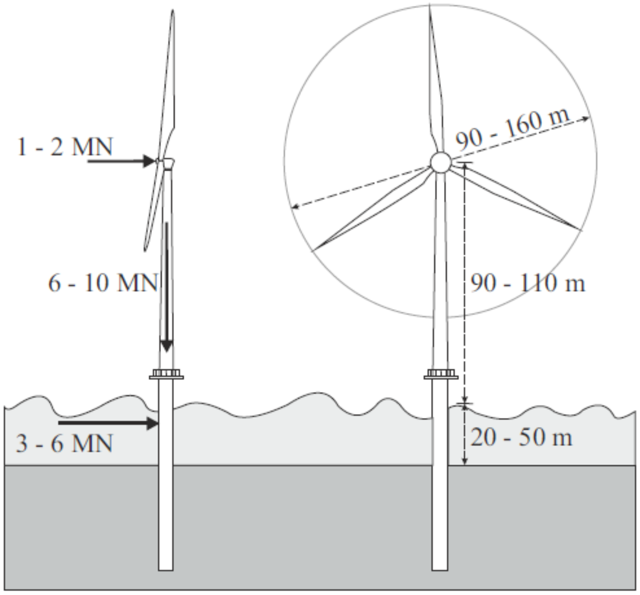

Figure 2.1. Typical dimensions and loading on an 3.5-5 MW offshore wind turbine (Byrne and Houlsby, 2015)

The lateral loads in Figure 2.1. vary in amplitude and frequency. A foundation must be designed for four limit states. The Ultimate Limit State (ULS), Accidental Limit State (ALS), the Serviceability Limit State (SLS), and the Fatigue Limit State (FLS) are the considered limit states for design (API, 2010). Each limit state has a recommended number of cycles and percent of maximum load to be used for the design. As recommended by Houslby et al. (2010), the ULS should be considered for one cycle, and 100% of the maximum load.

2.3 p-y Curves in the literature

Common p-y curves in the literature are discussed in this section. P-y curves are often categorized by the soil that they are specific to. This is a result of the method in which the p-y curve is derived. Reese and Cox (1974) reports that the p-y curve was originally conceptualized by McClelland and Focht (1956). The McClelland and Focht paper detail the results of lateral load tests on 24 in. diameter pipe piles in clay. Soil reaction deflection and stress strain relationships were plotted, and logarithmic trends were created. Hetenyi (1946) derived the differential equation that governs the laterally loaded pile problem. Using that derivation and the trends recorded from experimental data, p-y curves began to appear more frequently in the literature.

2.3.1 p-y Curves for Non-Plastic Soils

Table 2.1 shows the published variations of p-y curves for cohesionless soils that were reviewed for this thesis. P-y curves for cohesionless soil have been developed heavily in the last 40 years. The main driving factor has been the energy industry. Starting with Reese et al. (1974), the oil and gas industry funded research into engineering methods for designing offshore platforms. Murchison and O'Neill (1984) was work funded by the American Petroleum Institute (API) to simplify the Reese et al. (1974) procedure for computational purposes. The developed Murchison and O'Neill (1984) formulation is shown below:

$$p = AP_u \tanh(\frac{kz}{AP_u} y) \qquad \textit{(Eq. 2.1)}$$

Where:

y = pile deflection [L]

$A = 3.0 - \frac{0.8z}{d} \geq 0.9$ [Unitless]

k = depth stiffness factor [F/L^3]

z = depth [L]

P_u = ultimate soil resistance [F/L]

The ultimate soil resistance is computed using geotechnical theory for two types of failure. The first is a wedge type failure toward the top of the pile. The second is a flow-around type failure that occurs deeper in the subsurface. The P_u is taken as the lesser of the two calculations below:

$$P_u = \gamma' z\left[D\left(K_p - K_a\right) + zK_p \tan(\phi') \tan(\beta)\right]$$

And

$$P_u = \gamma' Dz\left(K_p^3 + 2K_0K_p^2 \tan(\phi') + \tan(\phi') - K_a\right)$$

Where:

γ' = effective unit weight [F/L^3]

D = diameter or width of the pile [L]

K_a = Rankine active earth pressure coefficient [Unitless]

K_p = Rankine passive earth pressure coefficient [Unitless]

K_0 = at-rest earth pressure coefficient [Unitless]

ϕ' = effective friction angle

$\beta = 45° + \phi'/2$

This formulation of p-y curve provides a simpler form to Reese et al. (1974) with undiscernible difference in results (Murchison and O'Neill, 1984). The API adopted this formulation for describing offshore pile behavior in dense sand. This p-y curve is used today for offshore infrastructure and is still published in the most recent API publications (API, 2011).

Research in offshore construction has shifted in this area in the last 20 years to focus on renewables. Offshore wind has been developed rapidly in the 21st century and has led to development in cyclic p-y curve formulations in cohesionless soils (Thieken et al.

2015). Wiemann et al. (2004), Lin et al. (2010), Heidari et al. (2014), Kirsch et al (2014), Thieken et al. (2015), Choi et al. (2015) and Liang et al. (2018) are all focused on cyclic loading of offshore piles with specific emphasis on the development of offshore wind energy.

Table 2.1 p-y curves in the literature for cohesionless soil

Author	**Year**	**Specialty**
Reese et al.	1974	submerged dense sand
Murchison and O'Neill	1984	submerged dense sand, simplified procedure
Yan and Byrne	1992	p-y curves in sand from lab testing
Wiemann et al.	2004	Modified stiffness for extreme loading in sand
Rollins et al.	2006	liquefied Sand
Lin et al.	2010	cyclic loading in sand
Heidari et al.	2014	cyclic loading in sand
Kirsch et al.	2014	large diameters subject to cyclic loading in sand
Thieken et al.	2015	offshore sand applications
Choi et al.	2015	p-y plasticity model for cyclic loading in sand
Liang et al.	2018	cyclic loading on offshore platforms

2.3.2 p-y Curves for Clays

Formulations for p-y curves in the literature for clays that were reviewed for this thesis are in Table 2.2. The subjects of the studies on clays tells an interesting story about the direction offshore construction technologies in clay. Instrumented pile test in clay that were conducted for Matlock et al. (1970) exhibited a geometry that indicated sensitivity. The p-y curves exhibited peaks at the maximum soil resistance and quickly dropped to the residual strength. Further resting by Reese et al. (1975) and Reese and Welch (1975) indicated that free water has a large effect on the geometry of a p-y curve in clay. Only one publication, (Rajashree and Sundaravadivelu, 1996), is widely accepted in industry for use in the cyclic loading case for clays. This is because the industry has heavily moved toward more complex constitutive models for modelling pile performance in clays. The complex mechanism of accumulated excess pore pressures with cyclic loading, sensitive behavior and changing OCR with depth has influenced to move away from p-y curves for use with foundations in clay (Andersen, 2015).

Table 2.2 p-y curves in the literature for cohesive soil

Author	**Year**	**Specialty**
Matlock et al.	1970	lacustrine clay
Reese et al.	1975	over-consolidated clay with free water
Reese and Welch	1975	over-consolidated clay without free water
O’Neill and Gazioglu	1984	includes pile diameter effects
Rajashree and Sundaravadivelu	1996	cyclic loading in soft clay

2.3.3 p-y Curves for special cases

A strength of the use of p-y curves is that they can be formulated based on region and site-specific testing. Formulations that have been published based on field testing in unique locations are shown in Table 2.3. This table demonstrates the versatility but also the oversimplification that is often associated with p-y curves. Mokwa et al. (2000) is an interesting study conducted at Virginia Polytechnical Institute on combining p-y curves for cohesive and non-cohesive soils to achieve a sort of weighted average for soils that exhibit characteristics of both. The accuracy of this method falls heavily on the selection of the two models selected and the characteristics of the soil, therefore it is difficult to calibrate. This leads to its limited usefulness in industry. Reese (1997) uses the Rock Quality Designation (RQD) to classify a weak rock for the use of the formulation published. For geologic conditions like that used in the publication (Northumberland Region, CA) the formulation is useful. However, use of RQD is highly empirical, leaving the usefulness of this p-y formulation up to the discerning engineer on a case-by-case basis. Formulations such as the Kramer (1988) publication on p-y curves for the Western Washington region are innovative because it is region specific. Simpson and Brown (2003) is similar, as it focuses on the American Southeast. The formulation of a p-y curve cannot be relied on as having the same geometric properties for a specific application unless verified to be true by the designing engineer. This makes special case p-y curves innovative but difficult to apply.

Table 2.3 p-y curves in the literature for special cases

Author	**Year**	**Specialty**
Mokwa et al.	2000	fine grain soils with both cohesive and frictional strength
Johnson et al.	2006	loess soil
Reese	1997	weak rock
Nyman	1980	strong rock, often called “vuggy limestone”
Kramer	1988	Western Washington region
Simpson and Brown	2003	piedmont residual soils

2.3.4 p-y Curves in Commercial Software

The two most popular commercial software that use p-y curves for design purposes are Ensoft Inc. LPILE and Bridge Software Institute FB-MultiPier. The method of numerical analysis and p-y curve formulations that are used by these programs are discussed in this section.

2.3.4.1 LPILE

LPILE is a famous computer program in civil engineering dating back to 1986. It uses a finite difference code to approximately solve the governing differential equation published by Hetenyi (1946). Reese and Meyer (1979) originally published the logic that is now implemented in LPILE. Figure 2.2 is borrowed from Reese and Meyer (1979) showing how the pile is discretized over its length.

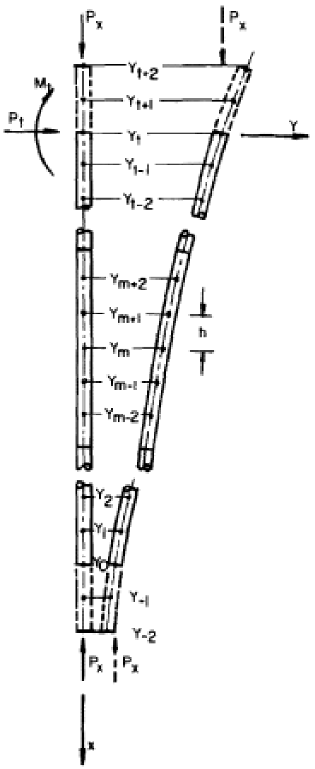

Figure 2.2. Pile discretization for finite difference code (Reese and Meyer, 1979)

As shown in the figure, plus and minus two mesh nodes must be identified for the computation to occur at a given point. Points “t” and “m” denote this in the figure. For this reason, imaginary points are created at the top and bottom of the pile to allow for computation to occur for the entire domain. At the original release of LPILE, the documentation recommended further work be done in the following areas:

- Instrumented tests on piles larger than 30 in. in diameter
- Instrumented tests in a variety of clay deposits
- Instrumented tests in desiccated and stiff soils
- In-Situ methods be developed to estimate p-y parameters without full-scale testing
- Laboratory methods be developed to estimate p-y parameters

Though these recommendations are old, they are paired with the initial development of LPILE. Revisiting original recommendations helps to put into perspective the usefulness and limitations of the early LPILE software, and investigate if these recommendations were met.

The development of LPILE has advanced since its initial development in release, however the main logic has stayed the same. In the 1980s many of the initial concerns of geotechnical community were addressed. The introduced the ability to have multiple soil layer, differing boundary conditions and distributed loading. As more p-y curves were published in the literature, many were added to the program. The Murchison and O'Neill (1984) p-y curve, also referred to as the API p-y curve for sand was added in 1993 (LPILE 2018). The ability to account for cracking and a decrease in flexural rigidity occurred in 1993 also. Many of the improvements from the mid-90s to the present have been compatibility with new operating systems and updating the p-y curve library to keep up with the industry. The p-y curves that are included in LPILE (2018) are shown in table 2.4.

Table 2.4 LPILE p-y curves

Name in LPILE software	**Reference in Literature**
Soft Clay	Matlock et al., 1970
Stiff Clay with Free Water	Reese et al., 1975
Modified Stiff Clay without Free Water	Reese and Welch, 1975
Sand	Reese et al., 1974
API Sand	Murchison and O’Neill, 1984
Liquified Sand	Rollins et al., 2006
Weak Rock	Reese, 1997
Strong Rock (Vuggy Limestone)	Nyman, 1980
Piedmont Residual	Simpson and Brown, 2003
Silt (cemented c-ϕ)	Evans and Duncan, 1982
Loess	Johnson et al., 2006
Elastic Subgrade	LPILE, 2018
User-input p-y curves	LPILE, 2018
API soft clay with J	API, 2011
Massive rock	Liang et al., 2009

All the LPILE p-y curves are for positive loading increments. For the cyclic loading case, LPILE computes the deflection at a node based on the principle of a backbone curve. A backbone curve is a trace of p versus y that follows the tops of the hysteretic loops. Figure 2.3 shows the backbone curve concept.

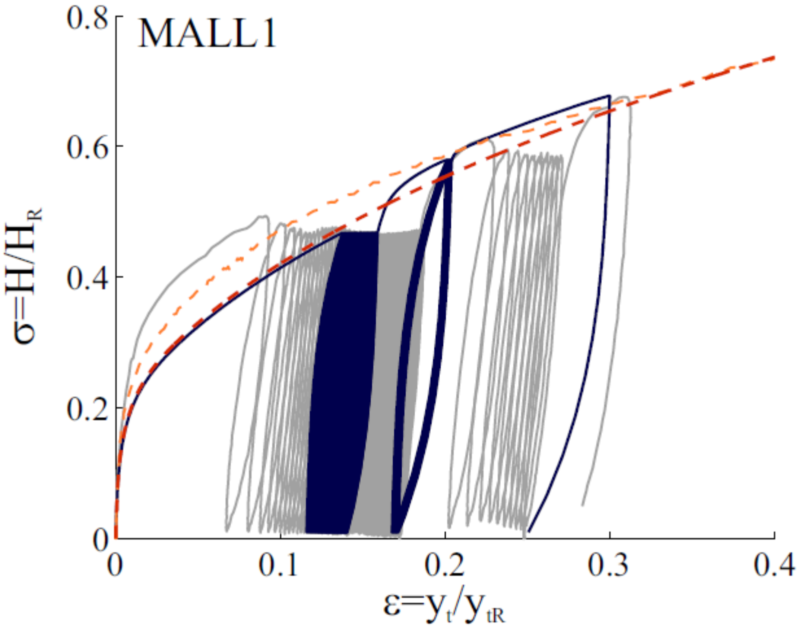

Figure 2.3. Demonstration of backbone curve concept (Abadie 2015)

Figure 2.3 is not associated with LPILE, and it used here to explain the backbone concept. Figure 2.3 shows normalized experimental p-y relationships for cyclic tests in sand. The dotted red and orange lines are backbone models. LPILE uses the same concept for each of its cyclic p-y models. Per the technical manual (LPILE 2018) cyclic models are based on the degradation of the depth stiffness factor and the ultimate soil resistance. Figure 2.4 is from the LPILE technical manual (LPILE 2018) and demonstrates a relationship between the initial soil modulus between cyclic and static loading.

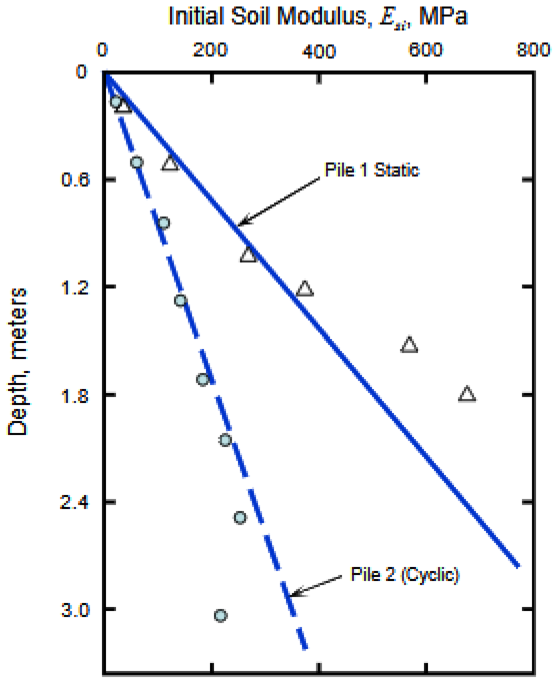

Figure 2.4. Initial soil modulus adjustment for cyclic loading

Each model other than the used specified input curves uses a degradation factor for either the depth stiffness factor of the ultimate soil resistance. For the p-y curve that this thesis uses for its studies, the ultimate soil resistance is degraded in the cyclic case by adjusting the factor *A* (see eq. 2.1).

The point of this explanation is to show that the backbone curves are adjusted in LPILE. This only allows for the prediction of deflection along the backbone curve. For any other location, such as the path during the unloading process, LPILE does not have the capability to predict.

2.3.4.2 FB-MultiPier

FB-MultiPier is a non-linear finite element analysis program mainly focused on the design of the pile foundation of bridge structures. It is developed by Bridge Software Institute (BSI). FB-Multipier can analyze pile groups in a variety of loading scenarios. The focus of this review of FB-MultiPier and its documentation is on the programs ability to model the lateral loading of single piles. This is a small facet of what FB-MulitPier does and is not an analysis of all of the program's capabilities. FB-MultiPier models piles using a non-linear finite element code that discretizes the pile into 10 elements as the default. The soil can be modelled using the p-y curve formulations in Table 2.5.

Table 2.5 FB-MultiPier p-y curves

Name in FB-MultiPier software	Reference in Literature
Stiff clay below water table	Reese et al., 1975
Florida Limestone	McVay and Niraula, 2004
Limestone	McVay and Niraula, 2004
Sand	API, 2011
Clay	API, 2011

Similarly to LPILE, the p-y curves are modified for cyclic loading by a multiplying factor on the ultimate soil resistance or stiffness depth factor. FB-MultiPier has fewer p-y options as compared to LPILE.

The advantage of FB-MultiPier is that pile group effects can be analyzed. A pile cap can be defined to connect to multiple piles with user-specifier material and geometric properties. The boundary conditions can be specified at the pile cap-pile interface. This means that the behavior of the pile cap changes the boundary conditions on the pile. For example, if the interface between the pile and the cap have a moment connection, and the pile cap rotates, the fixity of the pile head rotates as well. This allows for a complex interaction between the pile or pile group and the structure to be modelled. (FB-MultiPier, 2018).

2.5 Knowledge Gaps Identified

As discussed in section 2.2, the ULS for the design of offshore wind turbine foundations is a single cycle of loading at the maximum amplitude considered for design. In this case, repeated cyclic parameters do not apply. It is the opinion of the author that the ULS should be considered for both loading and unloading, due to the permanent deflections that can be observed at high lateral loading (Pando et al., 2006). The current literature does not provide a method for predicting the deflection throughout the unloading process and then the ultimate permanent displacement. This thesis introduces a method that has been developed to describe the behavior of a pile through the loading and unloading process for the case of a single load and unload cycle. Once developed, the method needs a program with which it can be implemented and tested. The development of NinerPile was undertaken to fill these knowledge gaps.

CHAPTER 3: SOLVING THE LATERALLY LOADED PILE PROBLEM

3.1 Introduction

Laterally loaded piles can be analyzed much in the same way as transversely loaded beams. The pile problem consists of an externally applied shear and/or moment to the head of the pile and a soil reaction from the ground surface to the toe of the pile. This is a soil structure interaction problem. The soil reaction along the length of the pile is not constant and is dependent on many fundamental factors including passive pressure, active pressure, pure shear, plugging (in the case of a pipe pile) and other complex factors. Individual elements of soil reaction can be lumped in a more simplistic manner into an overall reaction term that can be referred to as the soil resultant which opposes the direction of pile deflection at a certain depth. This chapter describes the procedure for numerically determining the pile deflections, slope, shear, moment and soil reaction at depths along a laterally loaded pile. Important relationships and definitions are displayed in Table 3.1 and will be used in the derivation of the strong form of the differential equation used to solve the laterally loaded pile problem.

Table 3.1 Definitions and formulas for the laterally loaded pile problem

Name of variable	**Formula or representative character**	**Units**
Distance along the pile from the pile head	x	[L]
Deflection of the pile	y	[L]
Slope of the pile	$\varphi = \frac{dy}{dx}$	[Unitless]
Curvature	$\kappa = \frac{d^2y}{dx^2}$	[Rad/L]
Bending moment	$M = E_pI_p\frac{d^2y}{dx^2} = E_pI_p\kappa$	[F-L]
Shear force	$V = E_pI_p\frac{d^3y}{dx^3}$	[F]
Soil reaction	$p = E_pI_p\frac{d^3y}{dx^3} = E_sy(x)$	[F/L]

3.2 Governing Differential Equation

The laterally loaded pile problem can be described using a differential equation originally derived for transversely loaded beams by Hetenyi (1946). The differential equation used in this thesis has been slightly modified to include terms of Young's modulus, moment of inertia (second moment of area), the 4th derivative of deflection with respect to length along the pile and the secant stiffness of the soil. The differential equation is derived as follows using Figure 3.1 as a free-body diagram.

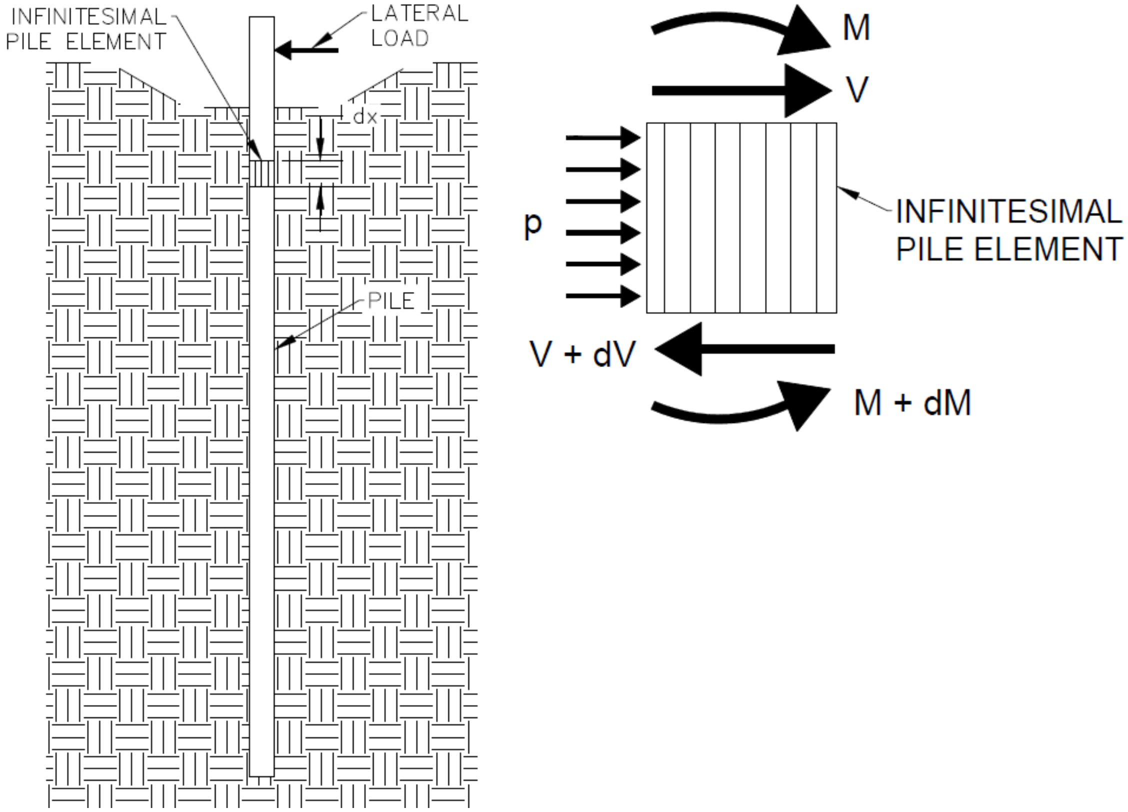

Figure 3.1. Free-body diagram of infinitesimal pile element

The infinitesimal element shown in Figure 3.1 illustrates what forces must be considered to achieve an equilibrium state. By summing the moments on the element the following is obtained:

$$\sum M = (M + dM) - M - Vdx - (pdx)\frac{dx}{2} = 0 \qquad \textit{(Eq 3.1)}$$

Eq. 3.1 is then differentiated twice with respect to x, and quadratic terms are ignored. This produces:

$$\frac{d^2M}{dx^2} - \frac{dV}{dx} = 0 \qquad \textit{(Eq. 3.2)}$$

The bending moment can also be assessed by integrating the normal stresses, $\sigma(z)$, on a cross-section of the pile area. Where z is the distance from the neutral axis, and A is the cross-sectional area. This is shown as:

$$M = \int_A \sigma(z)z \, dA \qquad \textit{(Eq. 3.3)}$$

Since it is reasonable to assume that planar cross-section remains planar during bending (constant rotation at a given x), we can write the displacement in the x direction (u), as a function of x and y.

$$u(x,y) = \frac{dy}{dx}z \qquad \textit{(Eq. 3.4)}$$

Using the definition of the pile slope from Table 3.1, we obtain:

$$u(x,y) = \varphi z \qquad \textit{(Eq. 3.5)}$$

The strain in the x direction (ε) can be defined as:

$$\varepsilon(z) = \frac{du}{dx} \qquad \textit{(Eq 3.6)}$$

Substituting Eq. 3.4 into Eq. 3.6 and using the definition of curvature from Table 3.1 produces:

$$\varepsilon(z) = \frac{d^2y}{dx^2}z = kz \qquad \textit{(Eq. 3.7)}$$

Using Hooke's law for the normal stress in the pile section creates:

$$\sigma(z) = E_p\varepsilon(z) \qquad \textit{(Eq 3.8)}$$

Substituting Eq. 3.7 into Eq. 3.8 yields:

$$\sigma(z) = E_pkz \qquad \textit{(Eq 3.9)}$$

Now Eq. 3.9 can be substituted into Eq. 3.3:

$$M = \int_A \left(E_pkz\right)z \, dA \qquad \textit{(Eq. 3.10)}$$

Rearranging, using the definitions from Table 3.1 for second moment of area and curvature produces the expression for moment:

$$M = E_p k \int_A \; z^2 dA = E_p I_p k = E_p I_p \frac{d^2 y}{dx^2} \qquad \textit{(Eq. 3.11)}$$

Eq. 3.11 is then substituted into Eq. 3.2 to achieve the almost completed ordinary differential equation:

$$E_p I_p \frac{d^4 y}{dx^4} - \frac{dV}{dx} = 0 \qquad \textit{(Eq. 3.12)}$$

The soil reaction along the pile can be expressed in a useful way by summing the forces on the element in Figure 3.1 in the horizontal direction:

$$\sum F_H = p(x)dx - dV = 0 \quad or$$

$$\frac{dV}{dx} = p(x) \qquad \textit{(Eq. 3.13)}$$

Substituting Eq. 3.13 into Eq. 3.12 produces an often-recognized version of the governing differential equation:

$$E_p I_p \frac{d^4 y}{dx^4} - p(x) = 0 \qquad \textit{(Eq. 3.14)}$$

To further prepare the governing differential equation for numerical analysis the definition of $p(x)$ from Table 3.1 is used, where E_s is the equivalent, or secant, stiffness of the soil represented as a spring.

$$E_p I_p \frac{d^4 y}{dx^4} - E_s y(x) = 0 \qquad \textit{(Eq. 3.15)}$$

This form of the governing differential equation is referred to as the strong form. The strong form will be further manipulated in section 3.3.2 for use in a numerical approach to analyze laterally loaded piles. It should be noted that this ODE assumes homogenous, isotropic, linear elastic material with the same flexural rigidity in tension and compression (Pando et al. 2006).

3.3 Using p-y curves to model the soil reaction

As discussed in Chapter 2, p-y curves have been used in geotechnical engineering practice to describe the reaction of a soil mass on a laterally loaded pile. The curves take on the form of a non-linear spring. Their form has been derived using a semi-empirical method to describe test piles' behavior under lateral loading. The simplified method to describe the soil reaction is preferred over a mechanistic approach because of the complexity of the soil-structure interaction. For example, a theoretical section of a round pile is shown in Figure 3.2. If the pile is deflecting to the left, the same soil body may be exerting passive pressure, active pressure and simple shear at different locations on the same pile section.

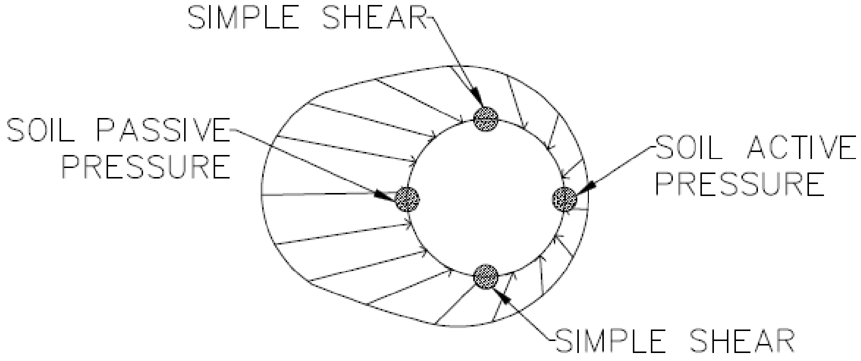

Figure 3.2. Soil reaction complexity

To simplify modeling the laterally loaded pile problem and make it possible in practice to efficiently predict pile behavior, p-y curves were developed. Figure 3.3 shows a comparison of the numerical approach taken (bottom) with the reality condition (top). The stiffness of the springs and their non-linearity is a matter of soil type and properties, depth, pile cross-section and loading type.

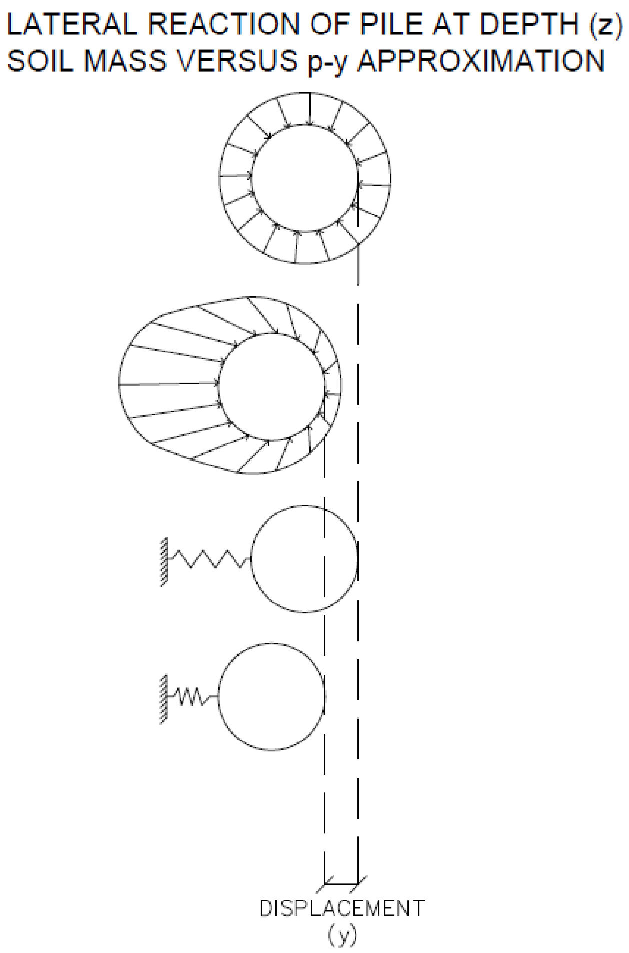
LATERAL REACTION OF PILE AT DEPTH (z)
SOIL MASS VERSUS p-y APPROXIMATION
DISPLACEMENT
(y)

Figure 3.3. Compression of p-y springs

For this thesis, the Murchison and O’Neill (1984) p-y curve is used. It is recommended for sandy soils and are used by the American Petroleum Institute (API) to model the subsurface for offshore structural design. Because the application of NinerPile in this thesis is to model fluvial sand deposits below the phreatic surface, the Murchison and O’Neil (1984) formulation was chosen. The p-y curve takes on the mathematical form shown in chapter 2:

$$p = AP_u \tanh(\frac{kz}{AP_u} y)$$

Where:

$y =$ pile deflection [L]

$A = 3.0 - \frac{0.8z}{d} \geq 0.9$ [Unitless]

$k =$ depth stiffness factor [F/L^3]

$z =$ depth [L]

$P_u =$ ultimate soil resistance [F/L]

The ultimate soil resistance is computed using geotechnical theory for two types of failure. The first is a wedge type failure toward the top of the pile. The second is a flow-around type failure that occurs deeper in the subsurface. The P_u is taken as the lesser of the two calculations below:

$$P_u = \gamma' z\left[D\left(K_p - K_a\right) + zK_p \tan(\phi') \tan(\beta)\right]$$

And

$$P_u = \gamma' Dz\left(K_p^3 + 2K_0 K_p^2 \tan(\phi') + \tan(\phi') - K_a\right)$$

Where:

$\gamma' =$ effective unit weight [F/L^3]

$D =$ diameter or width of the pile [L]

$K_a =$ Rankine active earth pressure coefficient [Unitless]

$K_p =$ Rankine passive earth pressure coefficient [Unitless]

$K_0 =$ at-rest earth pressure coefficient [Unitless]

$\phi' =$ effective friction angle

$\beta = 45° + \phi'/2$

Figure 3.4 shows the general geometry of the Murchison and O'Neill (1984) p-y curve. The curve is hyperbolic. The initial slope, $E_{py-\ max}$ is the multiplier of deflection from within the hyperbolic tangent in the p-y curve definition. This is equal to the depth stiffness factor multiplied by the depth (kz). After a lateral load has been applied to the pile, and a deflection is experienced, the soil resistance follows the path of a specific p-y curve. The soil can then be modelled as an equivalent linear spring that intersects the p-y curve at the correct deflection and soil resistance. This is referred to as the secant stiffness and is denoted on the figure as E_s. This notation is consistent with the industry standard Reese and Van Impe (2011).

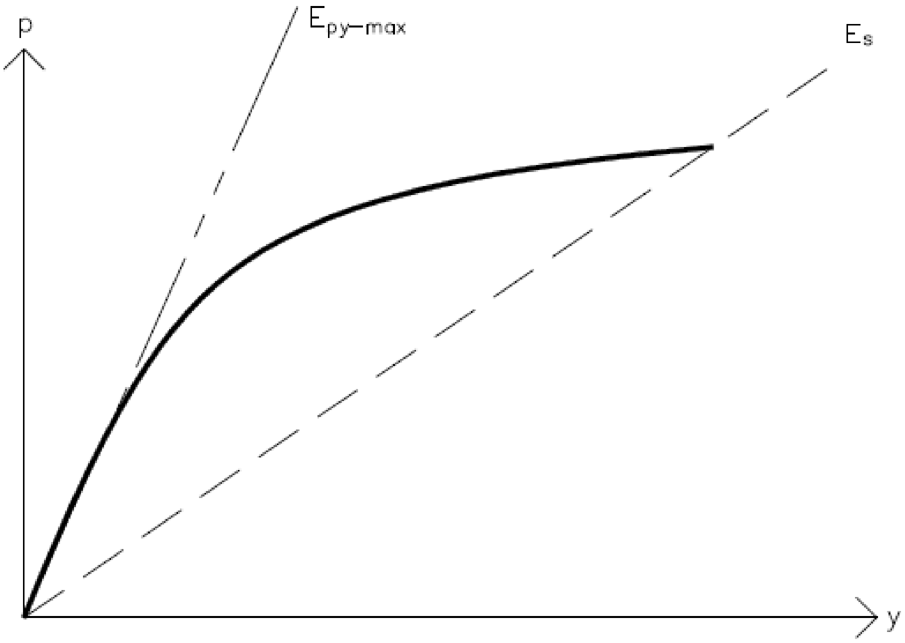

Figure 3.4. Geometry of a Murchison and O'Neill (1984) p-y curve

3.4 Numerical analysis approach

The first step to creating an approximate solution to the governing differential equation is to discretize the pile. The pile can be modelled as a 1D line of elements that connect at nodes. The soil reaction can then be calculated at the nodal points using an iterative procedure between an approximate solution to the governing differential equation, and the p-y relationship defined by the p-y curve being used. Figure 3.5 illustrates the discretization of a pile experiencing bending, with the trend of growing p-y curves describing the soil reaction vs. deflection relationship with depth.

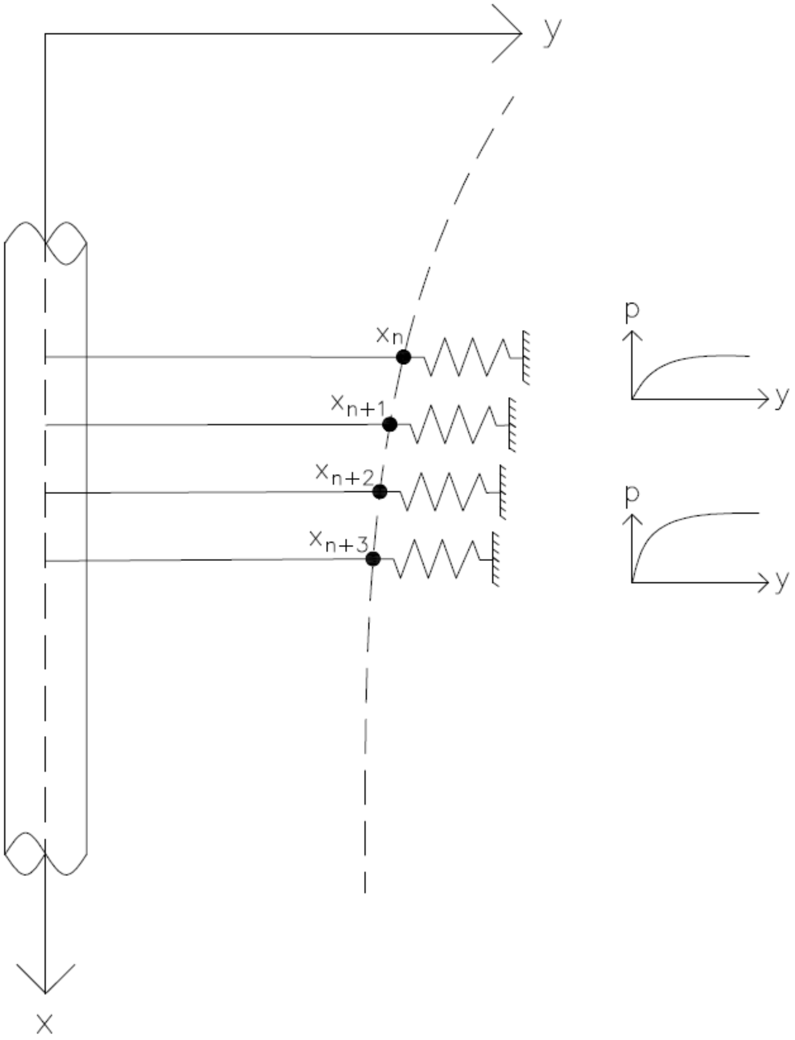

Figure 3.5. Pile discretization

3.4.1 P-Y curve method of analysis for soil reaction

The most important parameter in solving the laterally loaded pile problem is being able to accurately predict the soil resultant as a function of deflection with depth along the pile.

3.4.2 Finite element method of analysis for pile behavior

The notation and derivations in this section are consistent with Hughes (2000). The strong form displayed in Eq. 3.15 can be used in the Galerkin finite element method to numerically solve for y and its derivatives with depth along the pile. Subscript notation is used in this section to denote differentiation with respect to the subscript. First, the weak form of the governing differential equation needs to be derived.

3.4.2.1 Derivation of the weak form of the laterally loaded pile problem

The strong form is once again presented:

$$E_p I_p \frac{d^4 y}{dx^4} - E_s y(x) = 0 \qquad \textit{(Eq. 3.15)}$$

Using more concise notation for derivation of the weak form:

$$E_p I_p y_{,xxxx} - E_s y = 0 \qquad \textit{(Eq. 3.15)}$$

To find an approximate solution to the laterally loaded pile problem, the approximate solution must satisfy the following boundary conditions:

$$E_p I_p y_{,xxx} = V \; and \; E_p I_p y_{,xx} = -M \; at \; x = 0 \qquad \textit{(BC 1)}$$

$$E_p I_p y_{,xxx} = 0 \; and \; E_p I_p y_{,xx} = 0 \; at \; x = L \qquad \textit{(BC 2)}$$

Consider a function $y(x)$ that satisfies these boundary conditions but is not necessarily a solution to Eq. 3.15. The ODE would not then be equal to 0, rather to some residual $R(x)$.

$$E_p I_p \frac{d^4 y}{dx^4} - E_s y(x) = R(x) \qquad \textit{(Eq. 3.16)}$$

The residual can be minimized by multiplying by a weighting function (ω) that satisfies all Dirichlet boundary conditions imposed on the problem and integrating over the domain (Ω). It can be written:

$$\int_{\Omega} E_p I_p y_{,xxxx}\omega + \int_{\Omega} E_s y\omega = 0 \qquad \textit{(Eq. 3.17)}$$

To reduce the smoothness required by the function $y(x)$, the order can be reduced by integrating each term containing y by parts. This leads to:

$$\int_{\Omega} E_p I_p y_{,xx}\omega_{,xx} + \int_{\Omega} E_s y\omega = E_p I_p y_{,xxx}(0)\omega(0) - E_p I_p y_{,xxx}(L)\omega(L) + E_p I_p y_{,xx}(0)\omega_{,x}(0) - E_p I_p y_{,xx}(L)\omega_{,x}(L) \qquad \textit{(Eq. 3.18)}$$

Using the boundary conditions previously defined:

$$\int_{\Omega} E_p I_p y_{,xx}\omega_{,xx} + \int_{\Omega} E_s y\omega = V(0)\omega(0) - M(0)\omega_{,x}(0) \qquad \textit{(Eq. 3.19)}$$

Eq. 3.18 can then we rewritten using bilinear and linear operator notation. The flexural rigidity and secant stiffness is assumed to be included.

$$a\left(y_{,x}, \omega_{,x}\right) + (y, \omega) = V(0)\omega(0) - M(0)\omega_{,x}(0) \qquad \textit{(Eq. 3.20)}$$

This is the weak form of the governing differential equation. The next step is to identify shape functions, interpolation functions, and to write the variational form of the problem.

3.4.2.2 Derivation of the shape functions for the laterally loaded pile problem

The shape functions must satisfy that both the primary variable, y, and its first derivative, $y_{,x}$, be continuous and the second derivative, $y_{,xx}$, be at least piece-wise continuous on the domain. Figure 3.6 shows the discretization of an example domain of length "z" with the e^th^ and N^th^ elements labeled. The figure also shows the e^th^ element with nodes e_1 and e_2. The degrees of freedom associated with each node are listed as well. A shape function is associated with each degree of freedom. The shape function must have a unit value at its associated node on the order of its associated degree of freedom. For example, the shape function associated with θ_1^e must have a derivative equal to 1 at node e1. To keep continuity down to the third derivative, which is ideal for a laterally loaded pile because shear is of interest with depth, a cubic is used as the shape function form for the primary variable. The shape function is expressed as a function of a relative x coordinate.

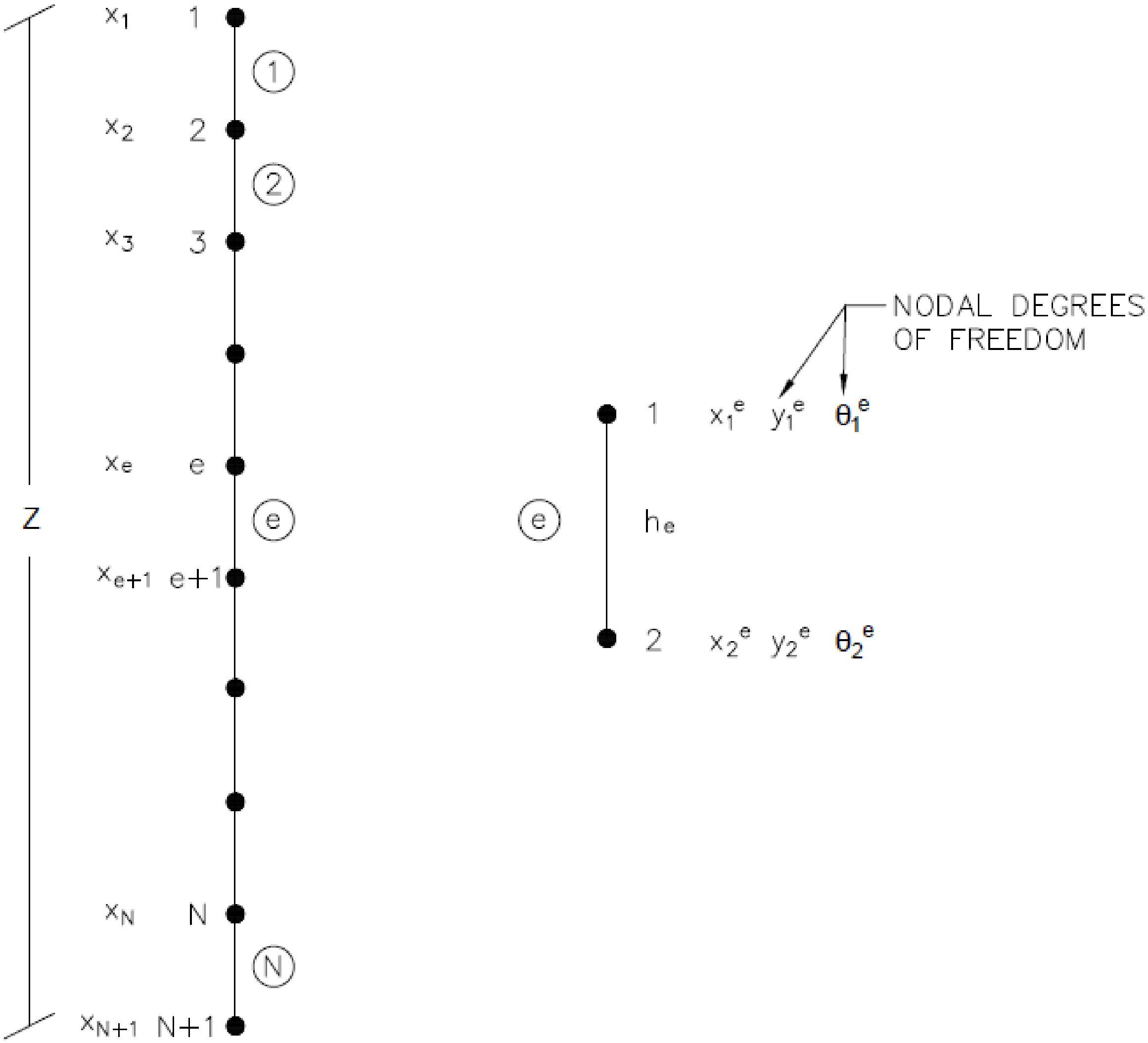

Figure 3.6. Discretization of the pile problem

To begin the derivation, a cubic polynomial is defined for the primary variable.

$$y^e = \alpha + \beta x + \gamma x^2 + \delta x^3$$

$$y^e = \alpha + \beta(x - x_1^e) + \gamma(x - x_1^e)^2 + \delta(x - x_1^e)^3 \qquad \textit{(Eq. 3.21)}$$

The length of the element can be defined as:

$$h_e = x_1^e - x_2^e \qquad \textit{(Eq. 3.22)}$$

When $x = x_1^e$, y^e can be written as:

$$y_1^e = \alpha \qquad \textit{(Eq. 3.23)}$$

and

$$y_2^e = \alpha + \beta h_e + \gamma {h_e}^2 + \delta {h_e}^3 \qquad \textit{(Eq. 3.24)}$$

The derivative can also be written as:

$$\theta_1^e = \beta \qquad \textit{(Eq. 3.25)}$$

and

$$\theta_2^e = \beta + 2\gamma h_e + 3\delta {h_e}^2 \qquad \textit{(Eq. 3.26)}$$

Eqs. 3.23, 3.24, 3.25 and 3.26 can be used to describe the coefficients α, β, γ and δ in terms of degrees of freedom. Through substitution, the following equations are reached:

$$\alpha = y_1^e \qquad \textit{(Eq. 3.27)}$$

$$\beta = \theta_1^e \qquad \textit{(Eq. 3.28)}$$

$$\gamma = \frac{1}{{h_e}^2}[-3y_1^e - 2h_e\theta_1^e + 3y_2^e - h_e\theta_2^e\ ] \qquad \textit{(Eq. 3.29)}$$

$$\delta = \frac{1}{{h_e}^3}[2y_1^e + h_e\theta_1^e - 2y_2^e + h_e\theta_2^e] \qquad \textit{(Eq. 3.30)}$$

By substituting Eq. 3.27, 3.28, 3.29 and 3.30 into Eq. 3.21, it can be written:

$$y^e = y_1^e\left[1 - \frac{3}{{h_e}^2}(x - x_1^e)^2 + \frac{2}{{h_e}^3}(x - x_1^e)^3\right]$$

$$+\theta_1^e\left[(x - x_1^e) - \frac{2}{h_e}(x - x_1^e)^2 + \frac{1}{{h_e}^2}(x - x_1^e)^3\right]$$

$$+y_2^e\left[\frac{3}{{h_e}^2}(x-x_1^e)^2-\frac{2}{{h_e}^3}(x-x_1^e)^3\right]$$

$$+\theta_2^e\left[-\frac{1}{h_e}(x-x_1^e)^2+\frac{1}{{h_e}^2}(x-x_1^e)^3\right] \qquad \textit{(Eq. 3.31)}$$

This is the form of the approximate solution to the laterally loaded pile problem. The coefficients to each degree of freedom in Eq. 3.31 are called shape functions. They can each be written as:

$$N_1^e = 1-\frac{3}{{h_e}^2}(x-x_1^e)^2+\frac{2}{{h_e}^3}(x-x_1^e)^3 \qquad \textit{(Eq. 3.32)}$$

$$N_2^e = (x-x_1^e)-\frac{2}{h_e}(x-x_1^e)^2+\frac{1}{{h_e}^2}(x-x_1^e)^3 \qquad \textit{(Eq. 3.33)}$$

$$N_3^e = \frac{3}{{h_e}^2}(x-x_1^e)^2-\frac{2}{{h_e}^3}(x-x_1^e)^3 \qquad \textit{(Eq. 3.34)}$$

$$N_4^e = -\frac{1}{h_e}(x-x_1^e)^2+\frac{1}{{h_e}^2}(x-x_1^e)^3 \qquad \textit{(Eq. 3.35)}$$

The approximate solution then can be written:

$$y^e = y_1^eN_1^e+\theta_1^eN_2^e+y_2^eN_3^e+\theta_2^eN_4^e \qquad \textit{(Eq. 3.36)}$$

For computational efficiency, it is convenient to map the shape functions to a master element. A master element allows the necessary integration of Eq. 3.20, which will be further developed, to be a repetitive process using the same shape functions. If the shape function mapping was not changed, each element would have unique functions to define and evaluate. The domain in the x coordinate space will be mapped to the ξ coordinate space. The domain of the ξ space is bi-unit, spanning from -1 to1. This process is illustrated in Figure 3.7.

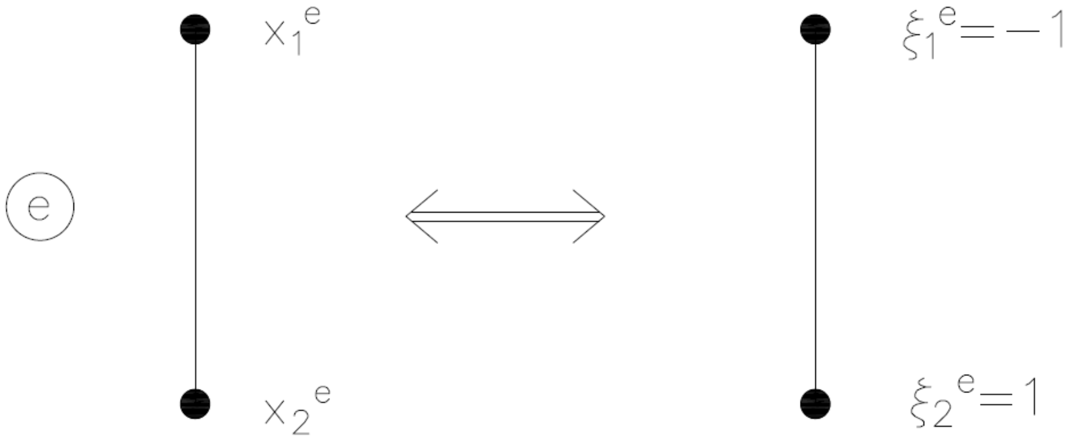

Figure 3.7. Mapping to master element domain

The above figure shows that for the mapping to occur, x_1^e must become equal to -1, and x_2^e must become equal to 1. The following mapping function follows this rule.

$$x = \left(\frac{h_e}{2}\right)\xi + \left(x_1^e + \frac{h_e}{2}\right) \qquad \textit{(Eq. 3.37)}$$

Eq. 3.37 can be rearranged as:

$$\frac{x - x_1^e}{h_e} = \frac{1}{2}(1+\xi) \qquad \textit{(Eq. 3.38)}$$

By substituting Eq. 3.38 into expressions for the shape functions, a new set of shape functions can be written.

$$N_1 = 1 - \frac{3}{4}(1+\xi)^2 + \frac{1}{4}(1+\xi)^3 \qquad \textit{(Eq. 3.39)}$$

$$N_2 = \frac{1}{8}h_e(1+\xi)(\xi-1)^2 \qquad \textit{(Eq. 3.40)}$$

$$N_3 = \frac{3}{4}(1+\xi)^2 - \frac{1}{4}(1+\xi)^3 \qquad \textit{(Eq. 3.41)}$$

$$N_4 = \frac{1}{8}h_e(\xi-1)(1+\xi)^2 \qquad \textit{(Eq. 3.42)}$$

The shape functions mapped to the master element can be seen in Figure 3.8 for the case of a unit length element. Notice the properties that are satisfied by these shape functions in relation to their corresponding degree of freedom. N_1 corresponds to the primary variable at $\xi = -1$. As shown, it has a unit value in the primary variable. N_2 corresponds to the derivative of the primary variable at $\xi = -1$. As shown, it has a unit slope at $\xi = -1$.

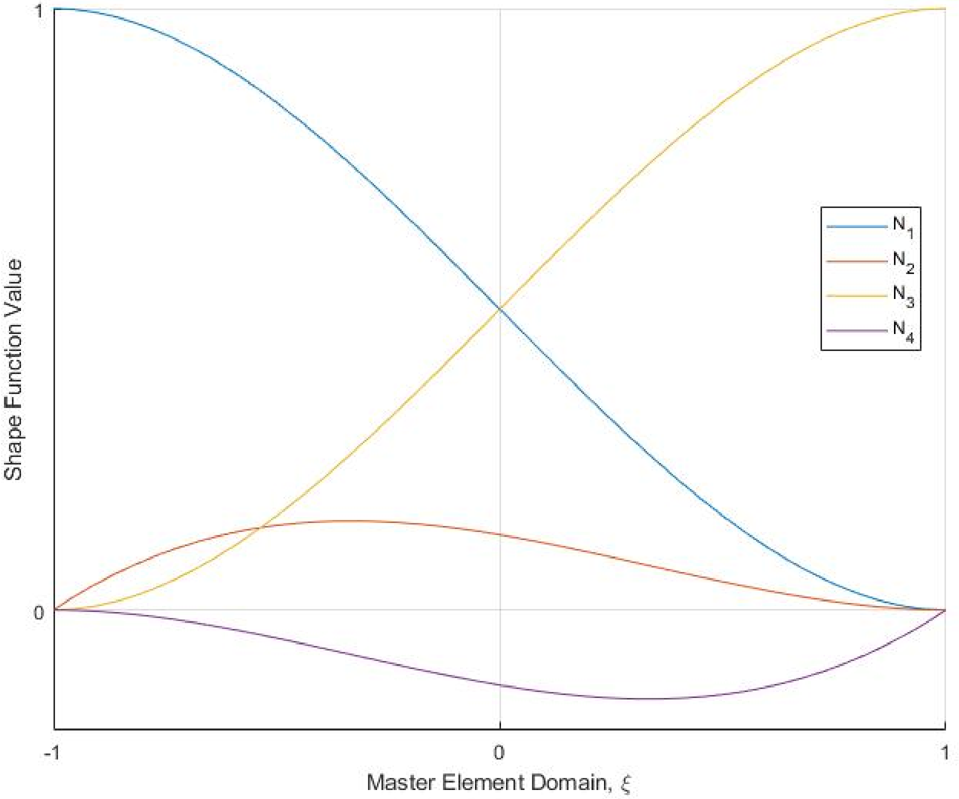

Figure 3.8. Shape functions for modeling the laterally loaded pile problem

3.4.2.3 Derivation of the matrix formulation of the laterally loaded pile problem

So far, the approximate solution has been written in element, or local, notation. The approximate solution over the entire domain of the problem (the length of the pile) is the summation of the local approximate solution components. Since there are 4 degrees of freedom on each element, this can be expressed as:

$$y^h = \sum_{A=1}^{4n_{el}} d_A N_A \qquad \textit{(Eq. 3.43)}$$

Where n_{el} is the number of elements in the problem, and A is an index corresponding to the degree of freedom number. The weighting function discussed in section 3.3.2.1 will take on the same form, with the same shape functions.

$$\omega^h = \sum_{B=1}^{4n_{el}} c_B N_B \qquad \textit{(Eq. 3.44)}$$

Substituting Eqs. 3.43 and 3.44 into the Eq. 3.20 and pulling out the summations, the problem can be written as:

$$\sum_{A=1}^{4n_{el}} \sum_{B=1}^{4n_{el}} d_A c_B \left[a\left(N_{A_{,x}}, N_{B_{,x}}\right) + (N_A, N_B) \right]$$

$$= \sum_{B=1}^{4n_{el}} c_B \left[V(0) N_B(0) - M(0) N_{B_{,x}}(0) \right] \qquad \textit{(Eq. 3.45)}$$

For a unique solution to exist, c_B must equal 1. It is arbitrary as shown in Eq. 3.45. The equation can be rewritten, assuming summation:

$$d_A \left[a\left(N_{A_{,x}}, N_{B_{,x}}\right) + (N_A, N_B) \right] = V(0) N_B(0) - M(0) N_{B_{,x}}(0) \qquad \textit{(Eq. 3.46)}$$

The left side of the equation inside the brackets is the stiffness matrix, K. The right side is the force vector, F.

$$K_{AB} = a\left(N_{A_{,x}}, N_{B_{,x}}\right) + (N_A, N_B) \qquad \textit{(Eq. 3.47)}$$

$$F_B = V(0) N_B(0) - M(0) N_{B_{,x}}(0) \qquad \textit{(Eq. 3.48)}$$

The above definitions allow for the problem to be written as a matrix system.

$$[K]\{d\} = \{F\} \qquad \textit{(Eq. 3.49)}$$

The stiffness matrix and force vectors are formulated locally, and then assembled into the global formulation. The next chapter discusses the implementation of this finite element method in NinerPile and explains the assembly and solving process of the global set of matrices.

CHAPTER 4: NINERPILE SOFTWARE

4.1 Introduction

This chapter describes the software created for this thesis. The structure of the software, its user interaction, and execution are detailed herein. Mathematical derivations that go beyond the basic understanding of the program are included as appendices. The program, to be referred to as the code in the following chapter, has been created using MATLAB^{TM} programming language and interface. The code is broken down into a wrapper file, a driver function, and 16 additional functions used by the driver to complete the analysis. The program outputs 6 data files and 3 plot windows to report the results. Each of the functions and outputs are discussed in this chapter. The complete code for the final version of NinerPile reported in this thesis is included as an appendix.

4.2 Wrapper file

The wrapper file is the user interface with the code. It is used in place of a graphical user interface (GUI) for utility and simplicity of programming. The use of a wrapper file gives the user greater control of input data and parameters to be used in the analysis. The wrapper file is the only file that needs to be explicitly opened and altered. It is also the only file that needs to be executed as it in turn, calls the driver to complete the analysis. The file name format used for the wrapper is "NinerPile_wrapper_*alterable_name*.m." This naming format allows a user to make multiple copies of the wrapper file for different analyses. The wrapper file is broken down into the following sections: pile geometric and material properties, p-y curve inputs, load time history, mesh-specific input data, output parameters, units, and driver activation.

The pile geometric and material properties section allows for the input of Young's modulus, moment of inertia about the bending axis (second moment of area), pile length, and pile width/diameter. Young's modulus and moment of inertia can be input as functions of moment, allowing for a pile with varying material and cross-section to be analyzed. Each of these parameters are combined into one input EI (flexural rigidity). The EI parameter is stores in a data structure with other pile-specific parameters. The structure variable is "Pile." Using a data structure allows for multiple values to be transferred to the driver under a single variable name. If the EI parameter is input as a function of moment, three parameters need to be identified by the user. First, the initial EI value. Second, the maximum or "cracking" moment that causes a drop in EI. Third, a function of moment that defines the EI value above the cracking moment.

The p-y curve input section is where the soil properties in accordance with Murchison and O'Neill (1984) are entered to determine the shape of the hyperbolic p-y curves as well as a few other values needed for execution. The distance from pile head to soil surface, the effective friction angle, the effective unit weight, the average depth stiffness factor, the cyclic parameter, and the tolerance for convergence are saved in the data structure "PY." Also, in this section, the unloading model is selected by the user. The unloading is selected by inputting either a 1 or 2 in the specified field. If a 1 is entered, the secant stiffness degradation model is selected. If a 2 is entered, the linear unloading model is selected.

In the load time history section of the wrapper file, the user identifies the external loading and is saved in the data structure as "Loading." First, a loading type is identified. In this field, a 1 may be entered for increasing load steps only. If a 2 is entered, incremental ascending and descending load steps will be calculated by the program. Next, the user inputs the external shear and moment applied to the head of the pile. These values are entered as vectors. Using the MATLAB$^{\text{TM}}$ language, the vector is defined using the following syntax "*0:load increment:maximum load*." This format allows for the program to use the same increment specified by the user to calculate a descending load loop if so chosen. For this thesis, the program has been verified using only the external shear. For the verification examples, the applied moment has been taken as 0. Finally, the user specifies the loading node. This is useful if the lateral load is applied below the head of the pile. External loads can be applied at any node of the pile mesh.

For the mesh-specific input data, 2 parameters are specified and saved in the data structure as "femesh." First, the number of elements to be used for the calculation in the finite element method for modeling the pile. The second is the element bias. This parameter allows for the mesh to be refined at the top of the pile where loading is greater and become sparser towards the bottom of the pile. This parameter is active in the program, though for the verification examples is taken as 1 (no bias). This section identifies a strong suit of the program as the number of elements needed is exceptionally low. For example, often only 10 elements are needed for an accurate computation of a pile over 10 feet long. In comparison, finite difference codes may use in excess of 500 elements.

The output parameters section allows for the identification of the node and load steps to be used in the output plots and are saved in the data structure as "plot." The node selected will then be used to create 2 plot windows. One of the windows is known as the progression window. In this window, displacement, slope, moment, and soil reaction are plotted vs. load step number (not load magnitude). In the displacement path window, the displacement of the node is plotted for each load step vs. load step magnitude. The load steps input into the wrapper file are then used to create whole-pile plots of displacement, slope, soil reaction, bending moment, and shear force vs. pile depth in the same window for the specific steps identified.

The next section of the wrapper file is for user reference only. In the units section, the user can enter force, length, and time units that correspond to all inputs in the wrapper file. This is to keep dimensions consistent across the program.

The wrapper file then activates the driver, passing all the input data into the driver function using the 5 data structures “Pile,” “PY,” “Loading,” “femesh,” and “plot.”

4.3 Driver function

The driver function is the brain of the code. It takes all the user inputs outlined in section 4.2 and uses them to conduct calculations necessary for the pile analysis. The driver function is named “pilefem_driver.m.” Due to the complexity of the driver function, each task will not be explicitly discussed. However, the main processes of each section will be outlined. The driver function is broken down into the following sections: initializing inputs and preallocating variables, main algorithm, and output.

4.3.1 Initializing inputs and pre-allocating variables

In the initializing inputs and pre-allocating variables section, the main goal is to sort and organize the user input data into useful arrays to be used in further computations, as well as identify the size of variables that are yet to be determined and preallocate their memory. The first step in this process is determining the coordinates along the pile that each node will be located at. This is done using a function called “getx.m.” This function uses the data structures “femesh” and “Pile” to discretize the pile length in 1 dimension.

Using these coordinates, pile properties and p-y curve properties are extrapolated. The function “getEI.m” determines the nodal values of flexural rigidity using the user input constant values or functions. The function “getPu.m” determines the ultimate soil capacity at each node using the maximum of wedge failure and flow failure based on geotechnical theory (Meyer and Reese 1979). The function “getEunl.m” determines the linear soil stiffness at each node to be used for the linear unloading model. This value is equal to the maximum p-y curve stiffness (E_{py-max}).

The driver then sorts the external load inputs entered by the user. If the user chose to use the second load option, this section adds the descending load steps to both the shear and moment vectors. The external shear and moment values are then combined into a 2-column matrix with the first column corresponding to the shear, and the second corresponding to the moment. The variables “displacement,” “slope,”” moment,”” shear,” and “soil” are then defined as 0 matrices that have the same number of columns as load steps, and the same number of rows as there are nodes.

4.3.2 Main algorithm

The main algorithm is comprised of an indexed loop that does an iteration for each load step, a logical test to determine if the load increment is positive or negative, and then either a loading or unloading convergence loop. The main algorithm chooses which loop to use based upon the relationship between the current load and the last, and then calculates the outputs for that load step. The outputs “displacement,” “slope,” “moment,” shear,” and “soil” are values at each node of displacement, slope, internal moment, internal shear, and soil reaction. With each iteration of the indexed load loop, the outputs are built into matrices with each column corresponding to a load step, and each row corresponding to a node.

4.3.2.1 Loading loop

The loading loop is a "while" loop that runs until a defined convergence is reached. The following discussion refers to the finite element method that is coded into this program. The derivations of this method and further discussion about its implementation can be found in the appendices.

4.3.2.1.1 Finite element matrix formulation

The finite element matrix formulation is comprised of 3 parts: the stiffness matrix, the force vector, and finally a vector containing the displacements and rotations at each node. Each part of this formulation is discussed in detail below.

4.3.2.1.1.1 Stiffness matrix

The first step of the loading logic is to determine a preliminary stiffness matrix, "K." The stiffness matrix is calculated using the function "getK.m." This function uses the Galerkin Finite Element Method to solve the boundary value problem outlined in chapter 3. The definition of the element stiffness matrix for the pile problem implemented in NinerPile is shown below.

$$k_{ab}^{e} = \int_{-1}^{1} [(EIN_a'' N_b'' (\frac{2}{h_e})^3 \; + E_s \, N_a N_b (\frac{h_e}{2})] \, d\xi \qquad \textit{(Eq 3.1)}$$

k_{ab}^{e} = Stiffness matrix value corresponding to location (a, b) in element matrix "e"

ξ = The coordinate system mapping associated with the master element from -1 to 1

E = Young's modulus mapped to the master element

I = Moment of inertia mapped to the master element

N_a = Shape function "a" mapped on the master element

h_e = The length of element "e"

E_s = The secant stiffness as a function on the master element mapping

Gaussian four-point quadrature is used to evaluate the integral for each combination of the 4 shape functions on each element. The function results in a 4x4 stiffness matrix for each element. All values of output parameters (i.e. displacement, slope) are unknown at every node. Due to this, all values of element stiffness are assembled into the global stiffness matrix. The global stiffness matrix created by the "getK.m" function is a symmetrical matrix with an upper bandwidth of 3.

4.3.2.1.1.2 Force vector

Another advantage of using the finite element method to solve the laterally loaded pile problem is the assembly of the force vector. The force vector is simply a column vector with the same number of values as there are nodes. The first value in the force vector is the externally applied shear, and the second is the externally applied moment. Due to this simplicity, an external function is not used, and the global force vector is assembled directly in the driver.

4.3.2.1.1.3 Solving for the displacement and rotation vector

After the stiffness matrix and force vector have been determined, solving for the displacements and rotation at the nodes is straight forward. In matrix notation, the operation is shown in equation 3.2.

$$\{Y\} = [K]^{-1}\{F\} \qquad \textit{(Eq 3.2)}$$

$\{Y\}$ = The displacements and rotations at each node assembled as a vector

$[K]^{-1}$ = The inverse of the global stiffness matrix

$\{F\}$ = The global force vector

This procedure is completed in the function “gety.m.” At this point, an iteration of the FEM has been completed to solve the pile problem. However, the secant stiffness is a preliminary guess during the first iteration. The results from the FEM must coincide with the soil secant stiffness defined by the p-y curves.

4.3.2.1.2 Convergence using p-y curves

After the FEM has been used to calculate the displacements and rotation using a certain secant stiffness of the soil, the driver then calculates a new set of secant stiffnesses using the previously calculated displacements. This process is done by the function “py.m.” The function uses the node-specific values of ultimate soil resistance (Pu), location along the pile and last computed displacement. Parameters that are also used, but do not vary from node to node are the depth stiffness factor (k), cyclic parameter (A) and the number of elements in the problem. The function then iterates through every node determining the p-y curve specific to that location. The previous value of displacement is then used to output a soil resultant value (p). The displacement and soil resistance values are then used to determine the secant stiffness value. This new secant stiffness value is then used in the FEM problem to adjust the stiffness vector. This process continues until the sum of the square of the new displacement vector and the sum of the square of the last displacement vector have a difference less than the user input tolerance.

4.3.2.2 Setup for unloading loop

After convergence but before the loading iterative loop is completed, the last values of secant stiffness, displacement and soil reaction are saved as special variables that are needed for the unloading logic. These values define the starting point of the unloading in respect to the p-y curve. The variables are aptly named “Estart,” “ystart,” and “pstart.”

4.3.2.3 Unloading loop

The unloading loop is also a “while” loop that runs until a defined convergence is reached. When the loop is engaged, a logical test is used to determine what unloading model was selected by the user.

4.3.2.3.1 Secant stiffness degradation model

If the secant stiffness degradation model is selected, the first part of the unloading loop is that the increment of stiffness degradation is determined for each node. The last positive loading secant stiffness “Estart” is used to accomplish this. “Estart” is divided by the number of evenly spaced unloading increments previously organized in the descending load loop vector (see section 4.3.1). This allows for the secant stiffness values to descend at the same rate as the external loading. Each time the load is then reduced in the unloading loop, the secant stiffness is reduced by the same relative amount. The program was optimized by weighting the distribution of the secant stiffness to the unloading steps. The recommended weighting that is programmed into NinerPile is an initial stiffness drop of 2 times the “even” increment. The increment is then reduced by 90% for each additional step. When the load is reduced by half of the peak, the stiffness is reduced evenly by the uniform stiffness increment determined in the beginning of this step.

This unloading method is preferred for several reasons. One reason is that it is an explicit method. There is no convergence. Calculation is thus very fast. The newly adjusted secant stiffness value that is calculated for each load step is used in the same FEM procedure discussed in section 4.3.2.1.1. Figure 4.1 demonstrates a graphical representation of this method for an unloading process with 5 increments. The secant stiffness begins from the value calculated in the loading loop, and then is evenly degraded as the load is diminished. This process is separate for each node, meaning stiffness increments vary.

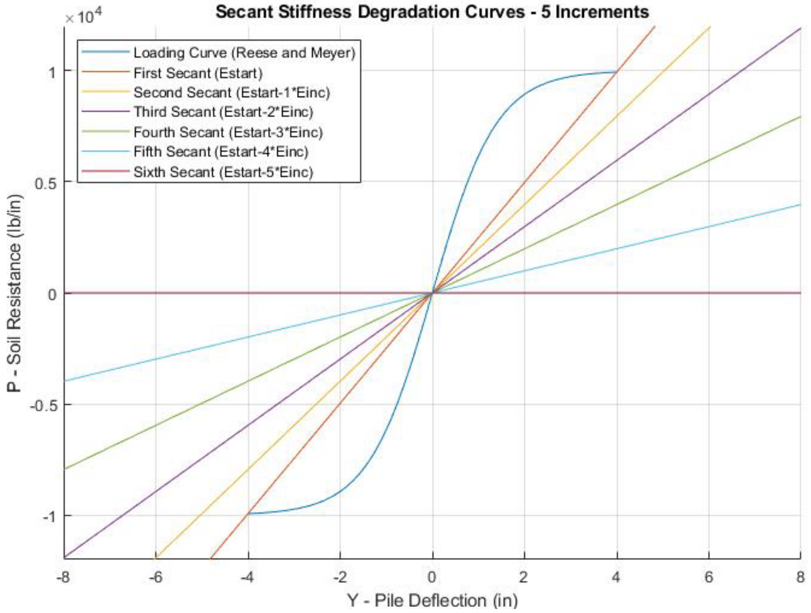

Figure 4.1. Graphical demonstration of the secant stiffness degradation method

4.3.2.3.2 Linear $E_{py\text{-}max}$ p-y unloading model

Part of this study was testing a method that uses a linear model to unload from the maximum deflection point on a given p-y curve. If the linear model is used, a process similar to the loading phase is used to agree the secant stiffness from the FEM with the secant stiffness determined by a model in the p-y plane. The linear unloading model has computational challenges, and understanding these challenges is important to understanding the computational method implemented in NinerPile.

The development of the hyperbolic shape in the Murichison and O’Neill (1984) p-y curve happens at loads relatively close to the maximum soil resistance value. This shape is only developed with significance in the case of a very highly loaded pile, nearing failure.

This geometric challenge causes no substantial deflection to be accumulated under moderate and low loading and unloading. Figure 4.2 demonstrates this.

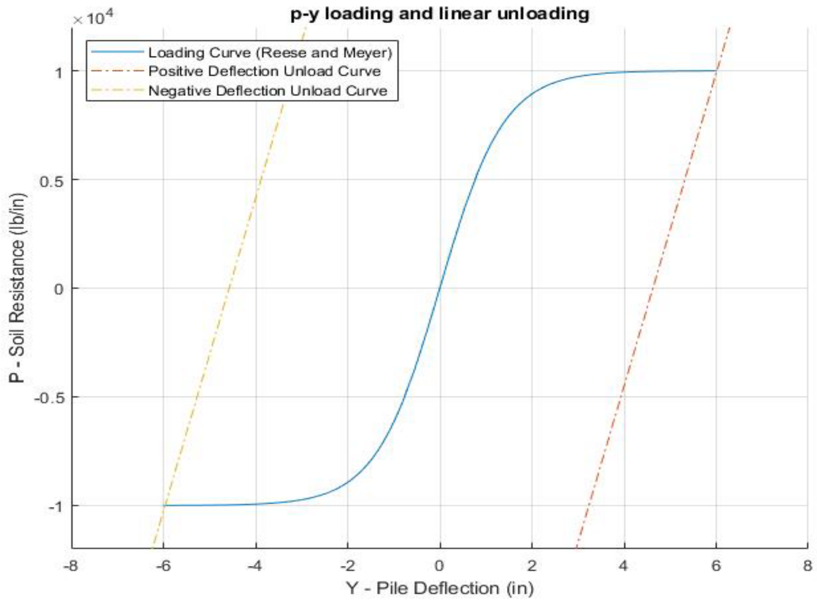

Figure 4.2. Linear unloading curves for a pile at failure

The above figure shows the fully developed shape of the hyperbolic p-y curve. For this case, up until 7,500 lb/in of soil resistance, a linear unload curve would for all practical purposes trace the loading curve back to the origin.

Another challenge is iterative in nature and has to do with the convergence to the unloading curve. The p-y curves have a higher $E_{py\text{-}max}$ at greater depths and with greater depth stiffness factors. In many instances, the slope is so high that the line nears vertical. In this case, a small difference in displacement yields a major difference in soil pressure.

Finally, due to the fact that the hyperbolic p-y curve is odd (reflects about the origin) using a linear unloading model requires that in actuality two lines are implemented. These lines must be bounded between horizontal y-axis and the maximum value of the p-y curve.

This is computationally challenging and can often result in erroneous values outside of the desired range.

With these difficulties in mind, a strategy was developed to converge the displacements calculated by the FEM with the displacements along a linear p-y unloading curve. First, it is checked by the program that the loading is symmetrical. Meaning that for each positive load step, there is an unloading step with an equal external load. The soil reaction calculated at the positive loading step is used as a preliminary guess for the soil reaction at the unloading step of equal magnitude. This guessed soil reaction value is used with the unloading model to calculate a guessed deflection. Both guesses are then used to guess a secant stiffness. From this point, the FEM discussed in section 4.3.2.1.1 is used to solve the pile problem initially. Much like the loading loop, the calculated deflections are then checked against the linear unloading model and a new secant stiffness is calculated. This process continues until convergence is achieved. To solve the piecewise function issue, the unloading curve is defined based on the point of the p-y curve used in the last positive loading step. This allows for the line to be defined from the point of intersection with the p-y curve. Therefore, the piecewise problem is avoided. This process is completed for each node, so each unloading curve is unique. The linear unloading curve is defined by the following function.

$$p = E_{py-max} \times y + (-E_{py-max} \times y_{start} + p_{start}) \qquad \textit{(Eq 3.3)}$$

p = Soil reaction

E_{py-max} = Maximum soil stiffness

y = Deflection at the node

y_{start} = Deflection value at the intersection with the hyperbolic p-y curve

p_{start} = Soil reaction value at the intersection with the hyperbolic p-y curve

Once convergence has been achieved for each unloading step, the piles values of displacement, slope, moment, shear and soil reaction are combined into a matrix storing the values of each load step and each node. The driver function then moves on to the output stage.

4.3.3 Output of results

In this portion of the driver function, the results determined by the program are output in several formats. One of the formats is graphically, and another is in tabulated raw data that can be post-processed by the user. The graphical output is displayed in 3 plot windows: whole pile plots, progression plots, and a displacement plot which is a type of progression plot. This program takes the single or multiple load steps identified by the user in the wrapper file that are desired for whole pile plots and plots displacement, slope, moment, shear and soil reaction vs. length along the pile in the same window for the desired load steps. The progression plots are produced using the function "plotprog.m." The progression plots display a single, user identified node that corresponds to a depth. The progression plots display displacement, slope, moment and soil reaction vs. the load step number in the same window. The final plot window is the displacement path at a user specified node. The displacement path plots the pile displacement vs. the lateral load magnitude. This plot shows the piles displacement during loading, and its rebounding behavior during unloading.

There are 6 MATLAB^{TM} data files that are created by the program: "coordinates.mat," "displacement.mat," "slope.mat," "moment.mat," "shear.mat" and "soil.mat." Each of the data files tabulates all of the results determined by the program in matrix form. The matrices are organized so that all the columns are load steps and the rows are nodes. This allows for users to quickly identify the results at each node and corresponding load step. Once the program creates the files, it then saves them to the file path folder. The user can pull the saved data and enter it into another platform to perform additional analyses. This gives the user the power to form additional plots that the program may not be specified to create. Some of the plots created for this thesis were done so by using output data and Excel. Tables 4.1 and 4.2 tabulate the input and output variables used by NinerPile.

Table 4.1 Ninerpile Inputs

Input Variable	**Symbol in Program**
length (in)	Pile.L
width (in)	Pile.D
flexural regidity (kip-in^2)	Pile.EI, Pile.mcrack and Pile.EIcrack
distance from pile head to soil surface (in)	PY.Depth
effective friction angle (°)	PY.phi
effective unit weight (lb/ft^3)	PY.gamma
depth stiffness factor (lb/in^3)	PY.k
tolerance of convergence	PY.tol
number of elements	femesh.nel
node for lateral load to be applied	Loading.node

Table 4.2 Ninerpile Outputs

Output Variable Vectors	**Symbol in Program**
depth coordinates (in)	coordinates
displacement of pile (in)	displacement
slope of pile (rad)	slope
internal shear force (kip)	shear
internal moment (kip-in)	moment
soil reaction (lb/in)	soil

CHAPTER 5: VERIFICATION AND VALIDATION

5.1 Introduction

This chapter will use field test data in the Federal Highway Report (FHWA-HRT-04-043) published in March 2006 titled “A Laboratory and Field Study of Composite Piles for Bridge Substructures” authored by Pando et al. (2006). This report was compiled to study the applicability of using composite piles in the construction of bridge substructures in the state of Virginia. Specifically, the field tests located at the Route 351 bridge will be used in this thesis. The Route 351 bridge is located in Hampton, Virginia and underwent a bridge replacement project beginning in 2001. The original bridge used reinforced concrete piles embedded into the bottom of the Hampton River and was constructed in the early 1940’s. The bridge experienced excessive deterioration over its lifetime, requiring a replacement be built. The original route 351 bridge is sown in Figure 5.1.

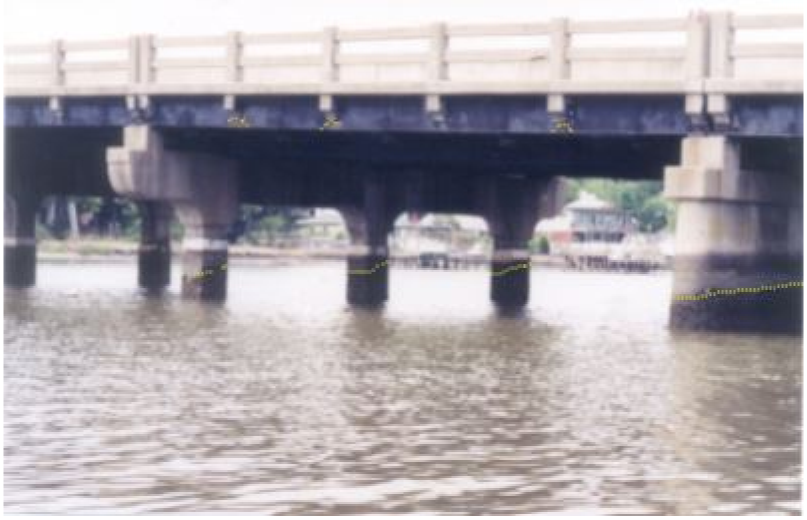

Figure 5.1. Deterioration of the Route 351 bridge

It was proposed that composite piles be evaluated for their ability to be used in bridge structures to mitigate the effect of deterioration due to freeze-thaw and water salinity over time. Piles underwent axial and lateral load tests at the bridge project site.

The uppermost layer of soil is approximately 3 feet thick and consists of silty fine sand fill. Underneath the fill is loose to medium dense silty fine sand which is 39.5 feet thick. Beneath the sand layer is a stiff sandy clay layer that is 8.5 feet thick. Medium dense to dense silty and clayey sand was found beneath the clay layer to an ultimate depth of 100 feet which was the end of the test borings. The site was evaluated at its northern and southern extents using 2 hollow stem auger borings, 4 cone penetrometer tests, and 1 dilatometer probe. The results are shown in Figures 5.2 and 5.3.

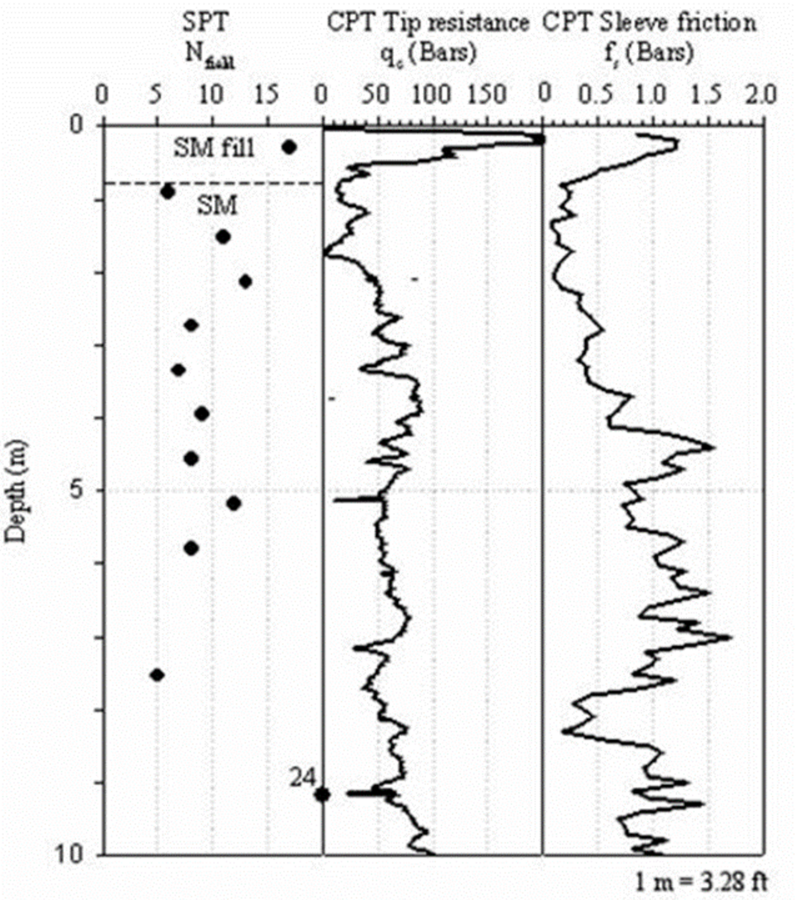

Figure 5.2. Route 351 bridge site northern end in-situ soil test results

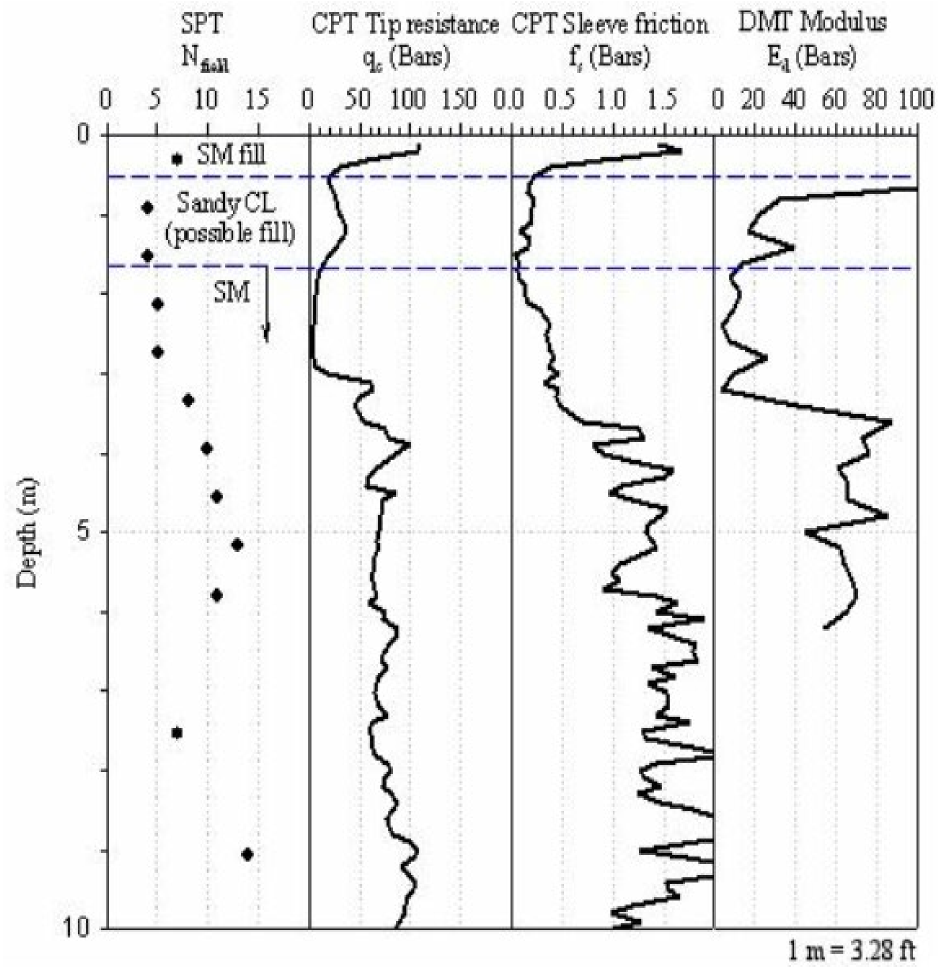

Figure 5.3. Route 351 bridge site southern end in-situ test results

The types of piles that were tested at the Route 351 bridge site and are used in this thesis are a prestressed concrete pile, a composite pile composed of polyethylene with steel reinforcing bars and a fiber reinforced polymer tube filled with concrete and steel reinforcement. Repeated loading and unloading are a potential concern at the project site and may have played a role in the previous bridge deterioration. Modeling the loading and unloading behavior of the piles is important for deterioration and longevity concerns.

The lateral load tests were performed in accordance with ASTM D3966. A calibrated load jack, calibrated load cells, dial gauges and LVDTs were used in the testing configuration. Displacements near the pile head were monitored with increasing static load. Each pile was instrumented with 16 sister bar strain gauges at 6 different depths. Each strain gauge used a #12M Grade 420 steel bar which is also a #4 Grade 60 steel bar. Each steel bar was 0.9 meters (2.9 feet) long mounted with 4 foil strain gauges. An inclinometer casing was also installed on each test pile for better resolution of the deflected shapes. Deformed shapes of the piles at different lateral load magnitudes were obtained from the inclinometer results along the installed casings. The results are shown in Figure 5.4. The results are displayed in a way that is the predominant concern in this thesis. The applied load is plotted against the pile deflection at the original ground surface. This plot nicely demonstrates the unloading behavior of the piles as the load is reduced several times during the test. This data is used for the study of the developed NinerPile software in this section.

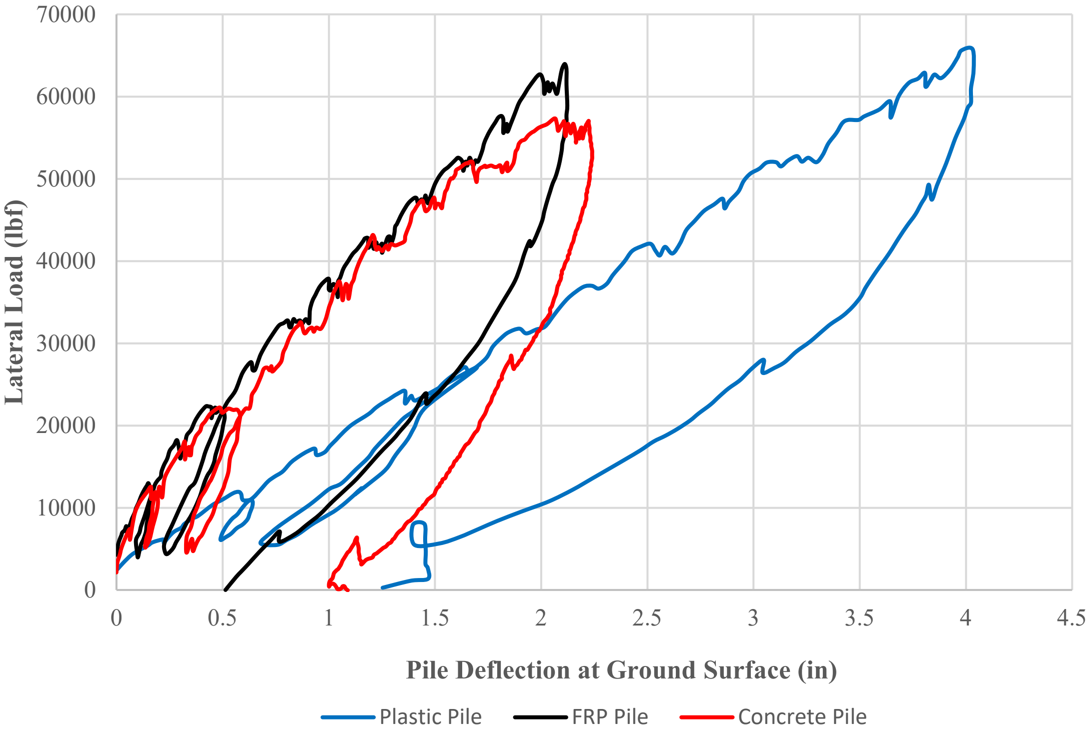

Figure 5.4. Route 351 bridge lateral load test results (Pando et al. 2006)

These pile tests are used for the verification of the loading phase of NinerPile in comparison with the industry standard LPILE. Each load step was modelled using both software, and the results have been compared. A single positive load step for each pile is compared in this section for results of slope, shear, moment and soil reaction along the pile’s full depth as calculated by both LPILE and NinerPile. Six load steps for each pile type are used to compare the deflected shape results with the output of both LPILE and NinerPile. In addition, the lateral load vs. deflection at the ground surface for the full loading and unloading cycle for each pile has been modelled by NinerPile and compared to the experimental results.

LPILE requires that the lateral load be modeled at the head of the pile. This requires that the LPILE model not consider the section above the load. This can be seen in the plots to follow that show the LPILE model only extending to the load location. Also, the LPILE version used in the thesis (LPILE Plus Student Edition 2005) does not have the capability to model a change in flexural rigidity as a function of internal moment. As recommended by Pando et al. (2006) the piles were modeled in NinerPile with varying flexural rigidity (EI) with internal moment. For this modeling strategy, a constant EI value is used until a “cracking” moment is reached, at which point the EI begins to reduce in accordance to some function. NinerPile and LPILE are compared directly for slope, shear, moment and soil reaction at a load below that needed to begin to alter the EI in NinerPile.

5.2 Prestressed concrete pile

5.2.1 Introduction

The prestressed concrete pile was tested at the southern end of the site. The testing configuration consisted of an excavated pit for the testing equipment to be installed. The dimensions are shown in Figure 5.5. The input parameters for both NinerPile and LPILE are displayed in Tables 5.1 and 5.2 respectively.

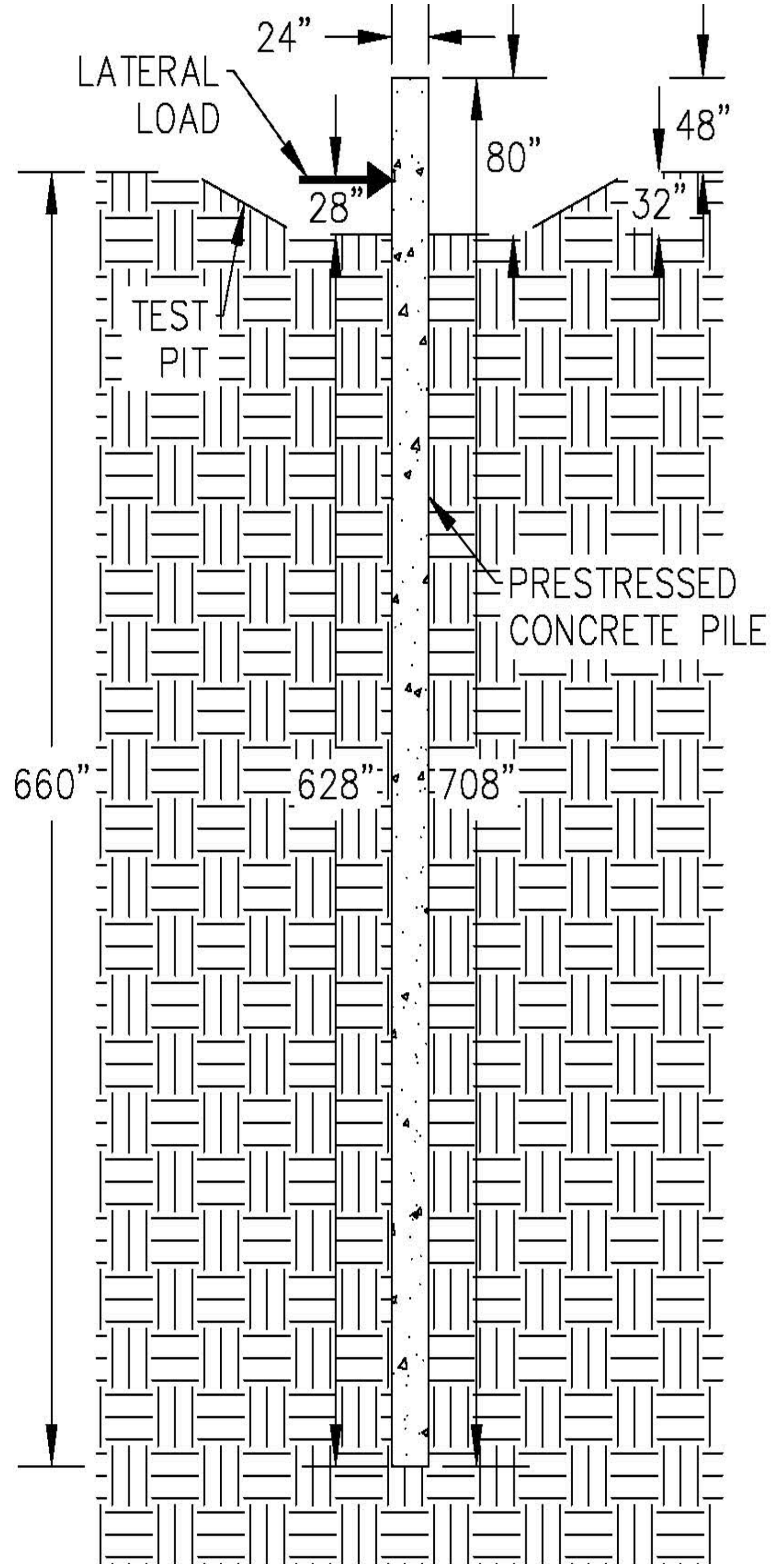

Figure 5.5 Testing and modeling configuration for the concrete pile

Table 5.1 NinerPile input parameters for the concrete pile

Parameter	Symbol in Program	Value
length (in)	Pile.L	708
width (in)	Pile.D	24
flexural regidity (kip-in^2)	Pile.EI, Pile.mcrack and Pile.EIcrack	see Figure 5.6
distance from pile head to soil surface (in)	PY.Depth	80
effective friction angle (°)	PY.phi	35
effective unit weight (lb/ft^3)	PY.gamma	66.84
depth stiffness factor (lb/in^3)	PY.k	11
tolerance of convergence	PY.tol	0.01
number of elements	femesh.nel	708
node for lateral load to be applied	Loading.node	48

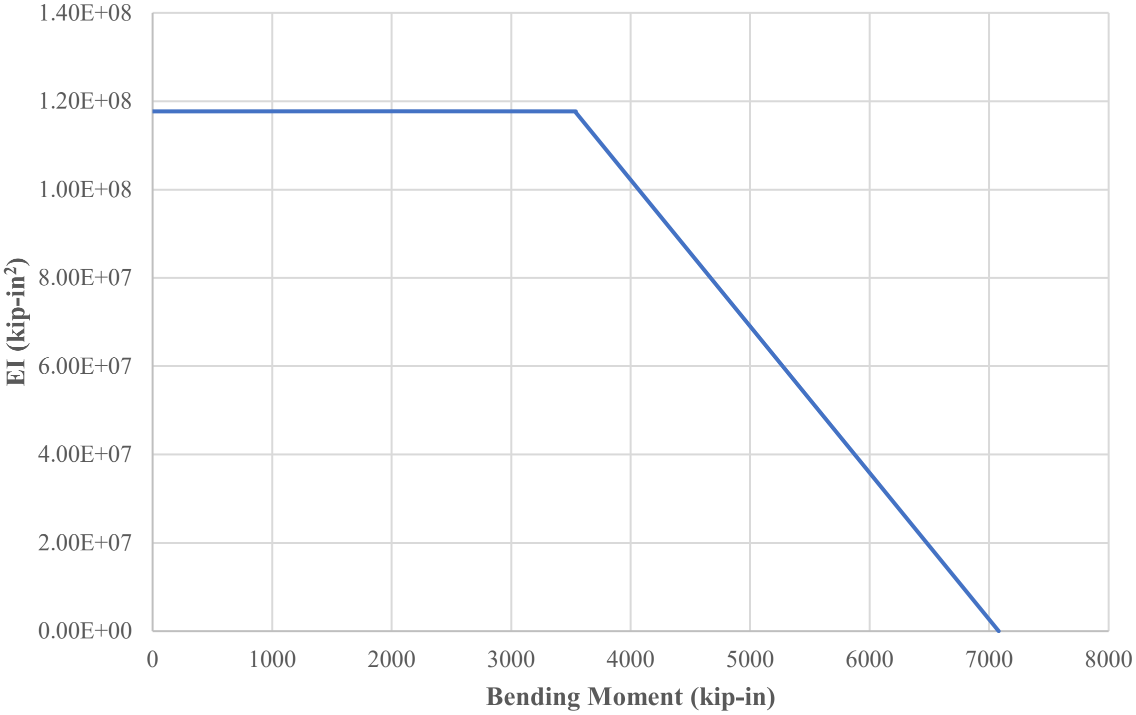

Figure 5.6. Concrete pile flexural rigidity vs. moment for NinerPile

Table 5.2 NinerPile input parameters for the concrete pile

Parameter	Value
length (in)	658
width (in)	24
flexural regidity (kip-in^2)	1.177 x 10^8
distance from pile head to soil surface (in)	28
effective friction angle (°)	35
effective unit weight (lb/ft^3)	66.84
depth stiffness factor (lb/in^3)	11
Soil Model	API Sand (O'Neill)

Note: Measurements are adjusted because LPILE does not model section above lateral load

5.2.2 NinerPile and LPILE detailed results comparison at 31,742 lb lateral loading

The following plots are a comparison between results generated by both LPILE and NinerPile for the prestressed concrete pile. Shear, moment, slope, soil resistance and lateral deflection are plotted vs. depth from the head of the pile at 31,742 pounds of lateral loading. The NinerPile results are all in close agreement with the LPILE results as well as the experimental data, which serves as verification of the NinerPile code.

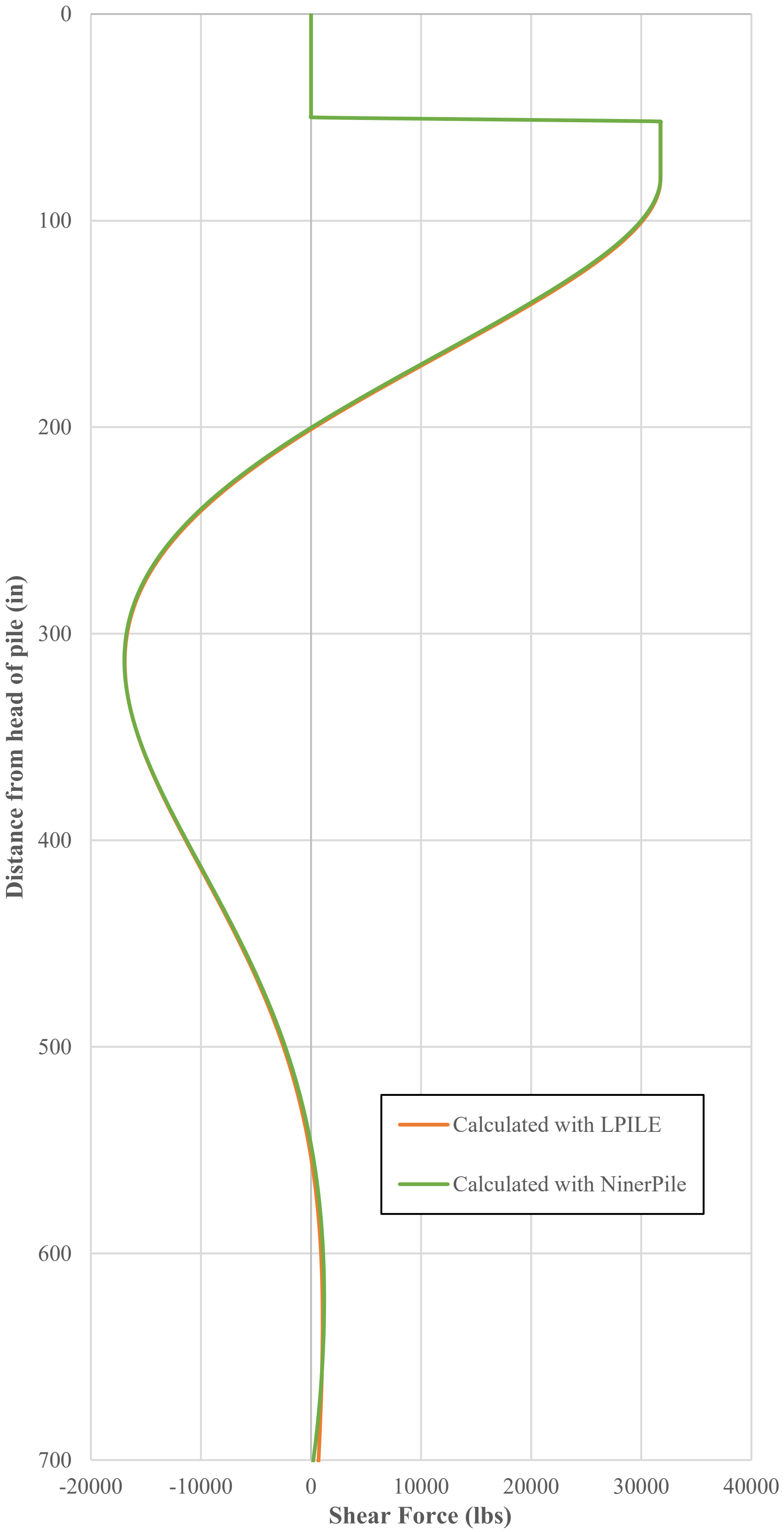

Figure 5.7. NinerPile vs. LPILE shear verification at 31,742 lb. lateral loading

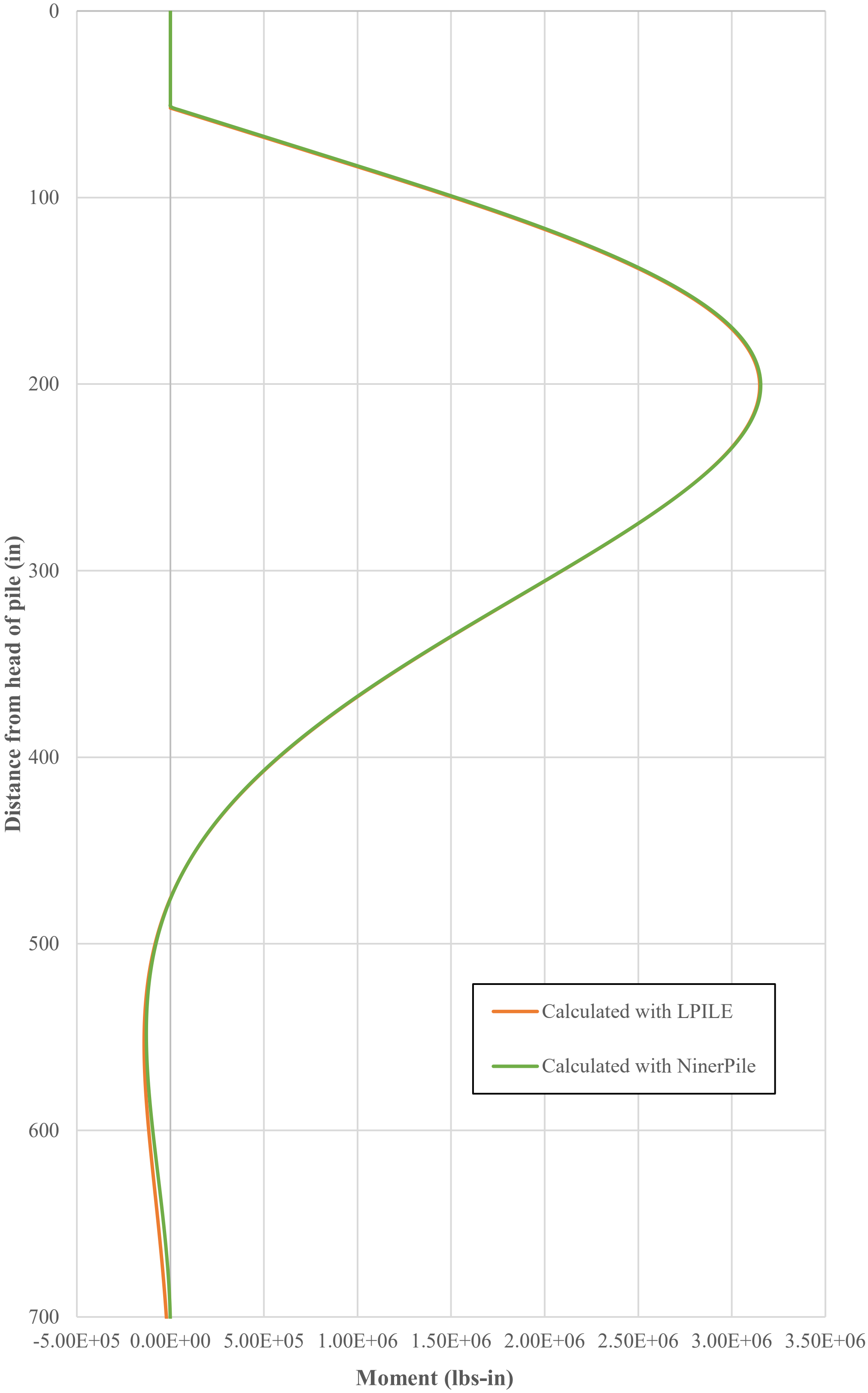

Figure 5.8. NinerPile vs. LPILE moment verification at 31,742 lb. lateral loading

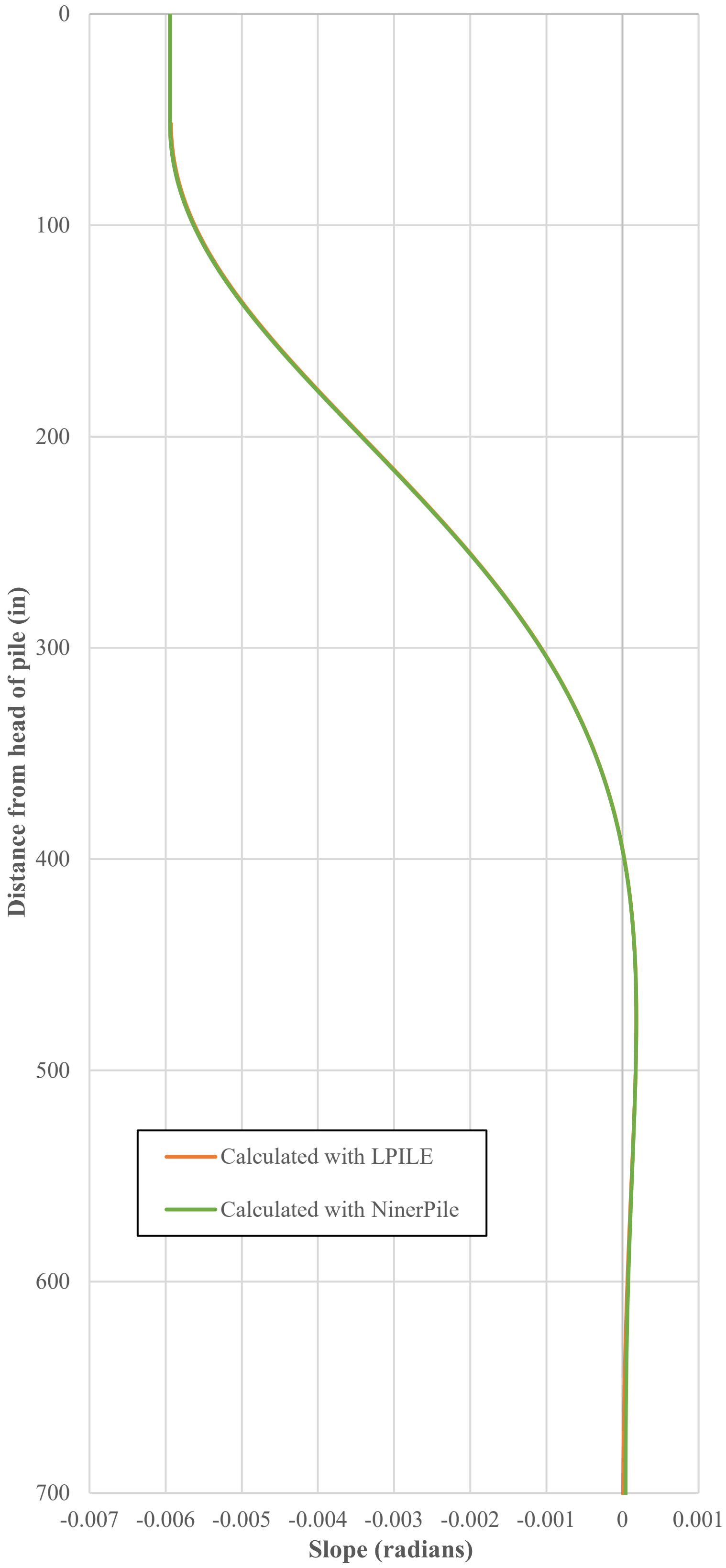

Figure 5.9. NinerPile vs. LPILE slope verification at 31,742 lb. lateral loading

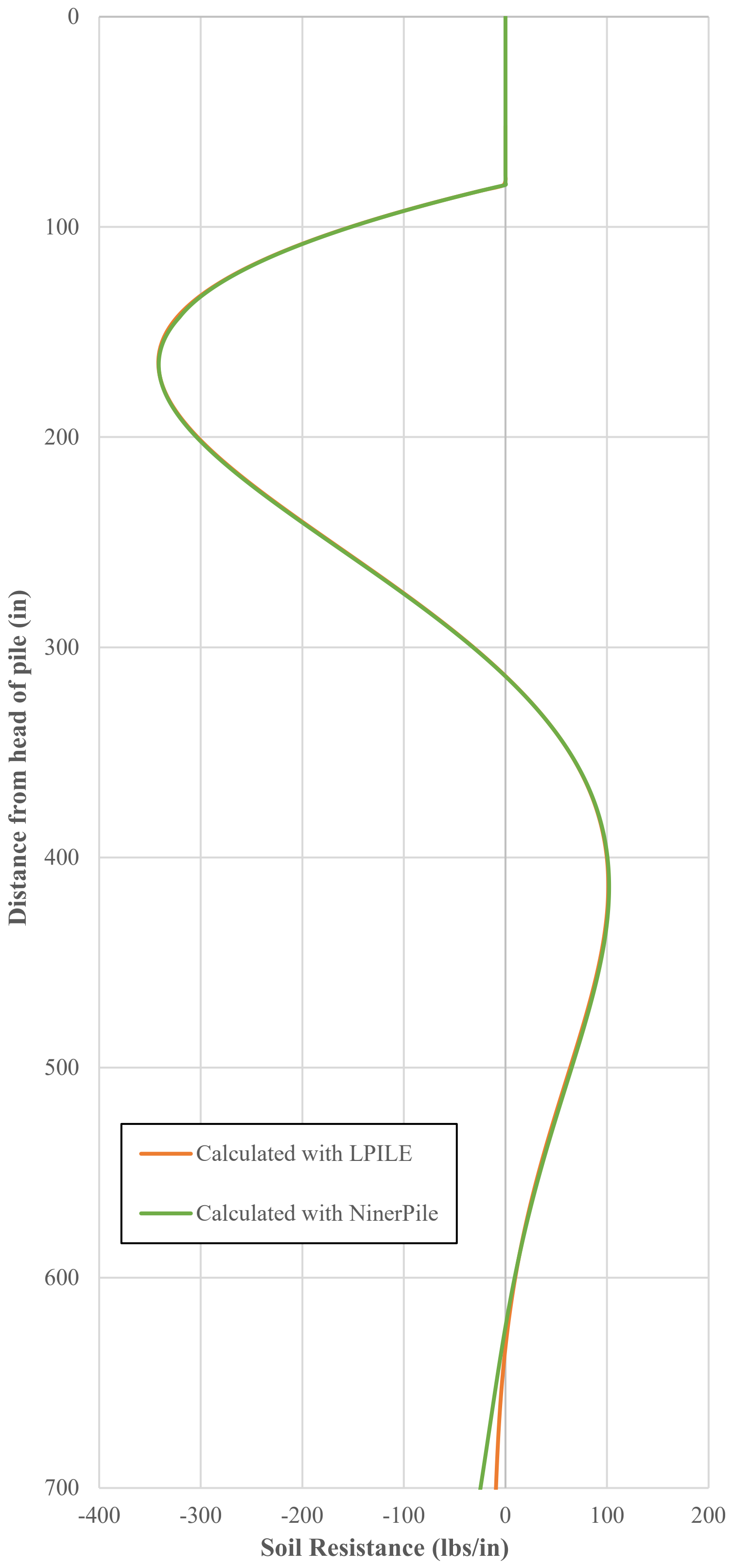

Figure 5.10. NinerPile vs. LPILE soil resistance verification at 31,742 lb. lateral loading

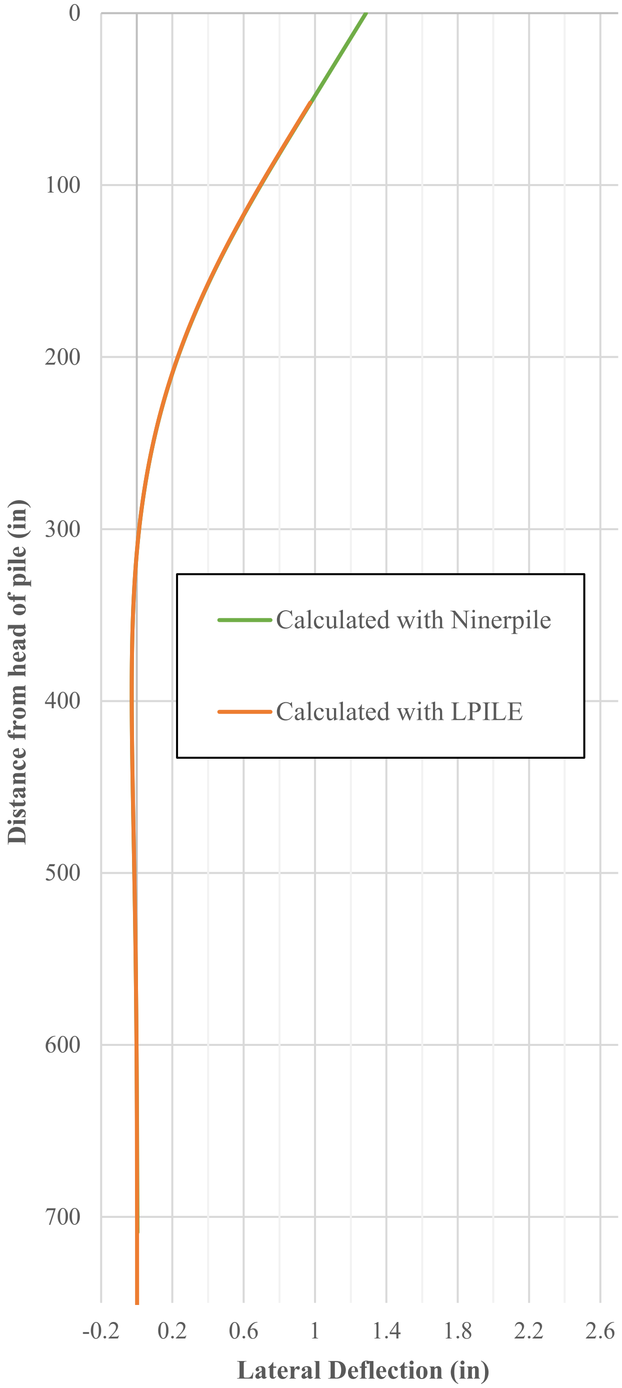

Figure 5.11. NinerPile vs. LPILE lateral deflection verification at 31,742 lb. lateral loading

5.2.3 Test results, NinerPile and LPILE deflection comparison at various loads

For each load step that a full set of deflection data was published in Pando et al. (2006) NinerPile and LPILE were used to model the deflections. The results are presented in this section.

5.2.3.1 Deflection plots for various positive load steps

The following deflection plots are a comparison between results generated by LPILE, NinerPile and experimental data for the prestressed concrete pile. Lateral deflection is plotted vs. depth from the head of the pile at the following lateral loads: 11,510 lbs., 21,873 lbs., 31,742 lbs., 41,161 lbs., 51,254 lbs. and 61,887 lbs. The NinerPile results are in close agreement with both LPILE results and experimental data.

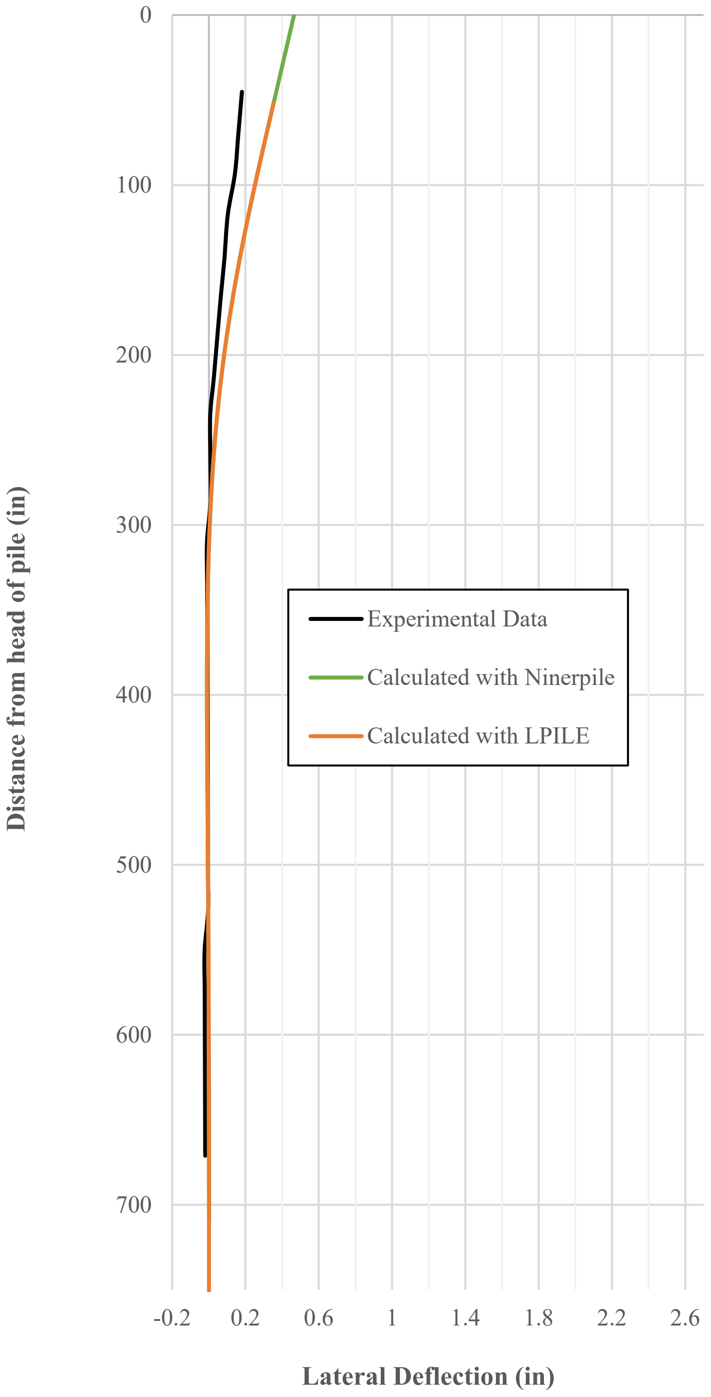

Figure 5.12. Concrete pile lateral deflection at 11,510 lb. lateral loading

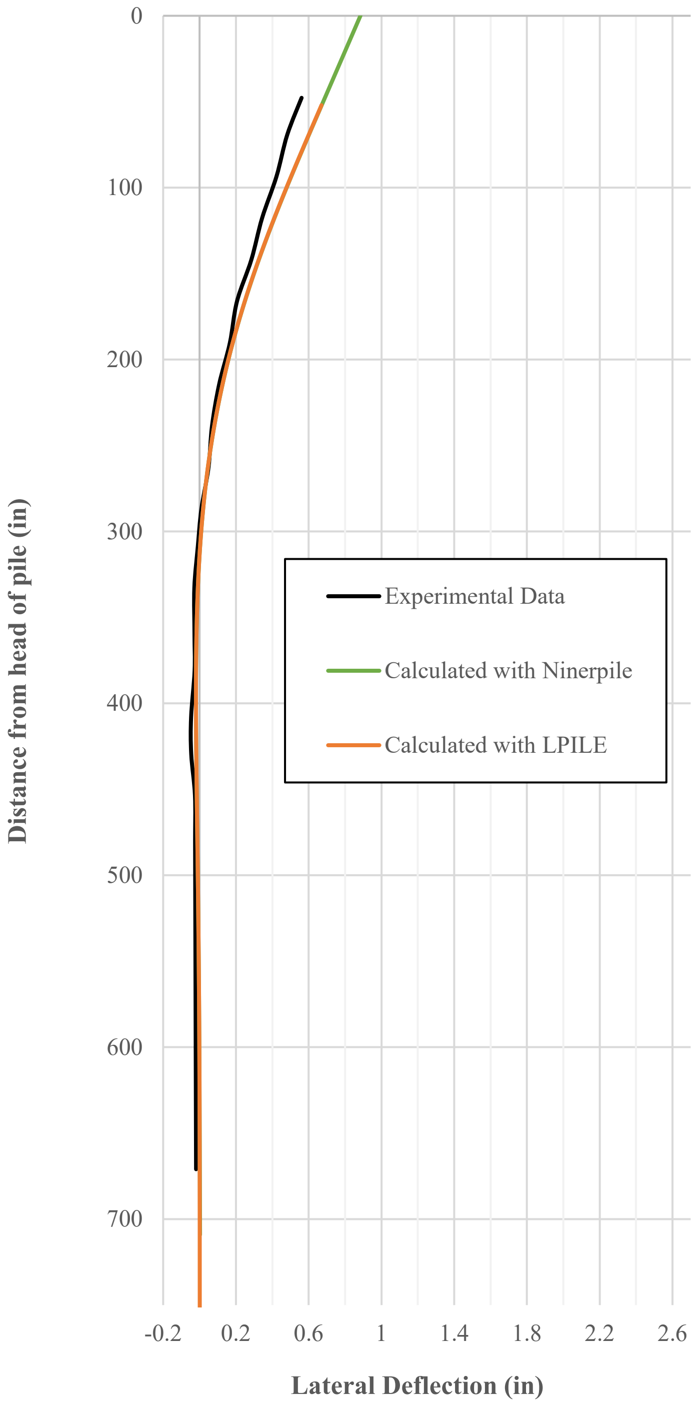

Figure 5.13. Concrete pile lateral deflection at 21,873 lb. lateral loading

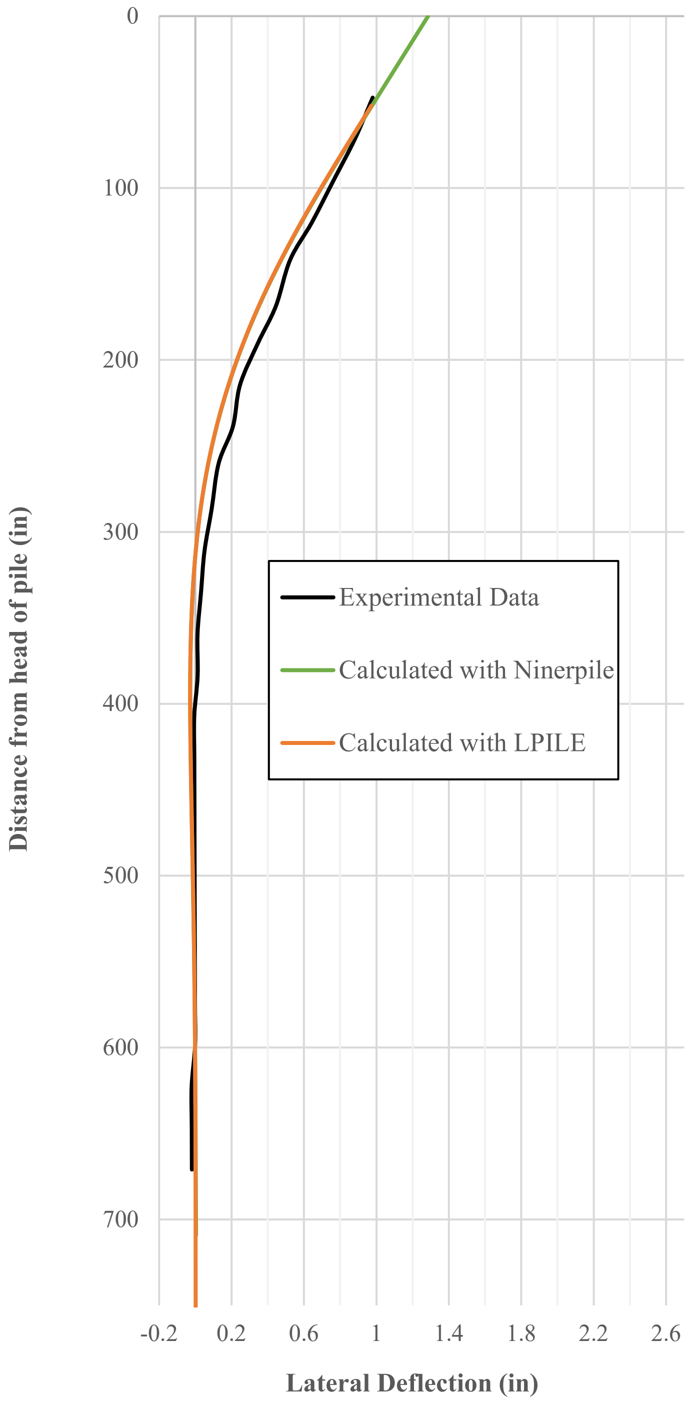

Figure 5.14. Concrete pile lateral deflection at 31,742 lb. lateral loading

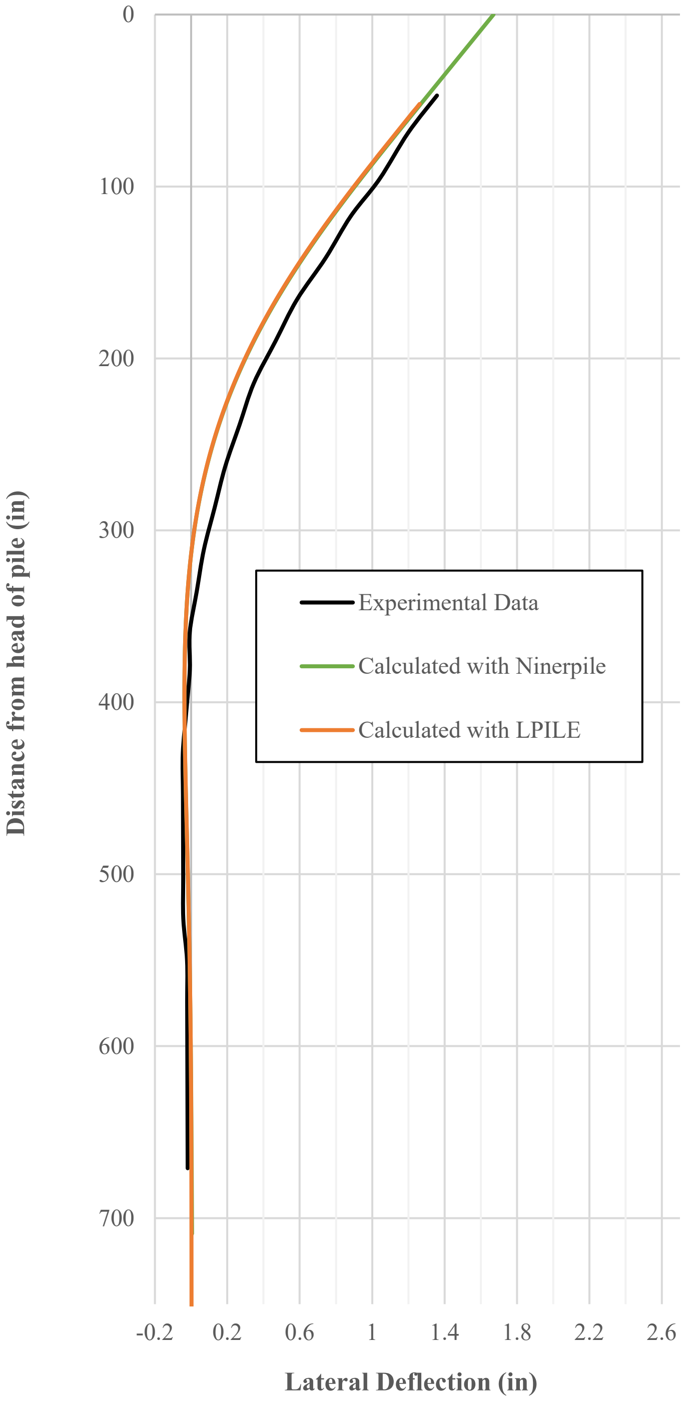

Figure 5.15. Concrete pile lateral deflection at 41,161 lb. lateral loading

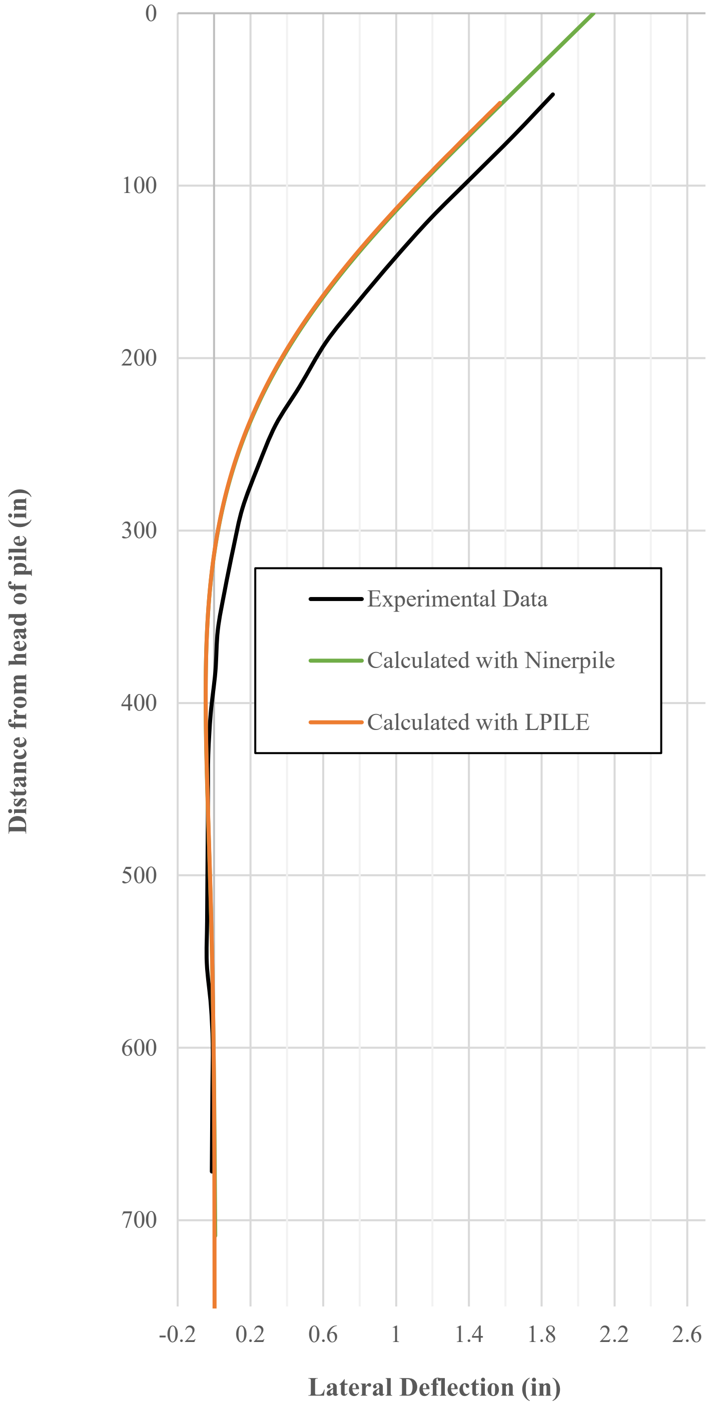

Figure 5.16. Concrete pile lateral deflection at 51,254 lb. lateral loading

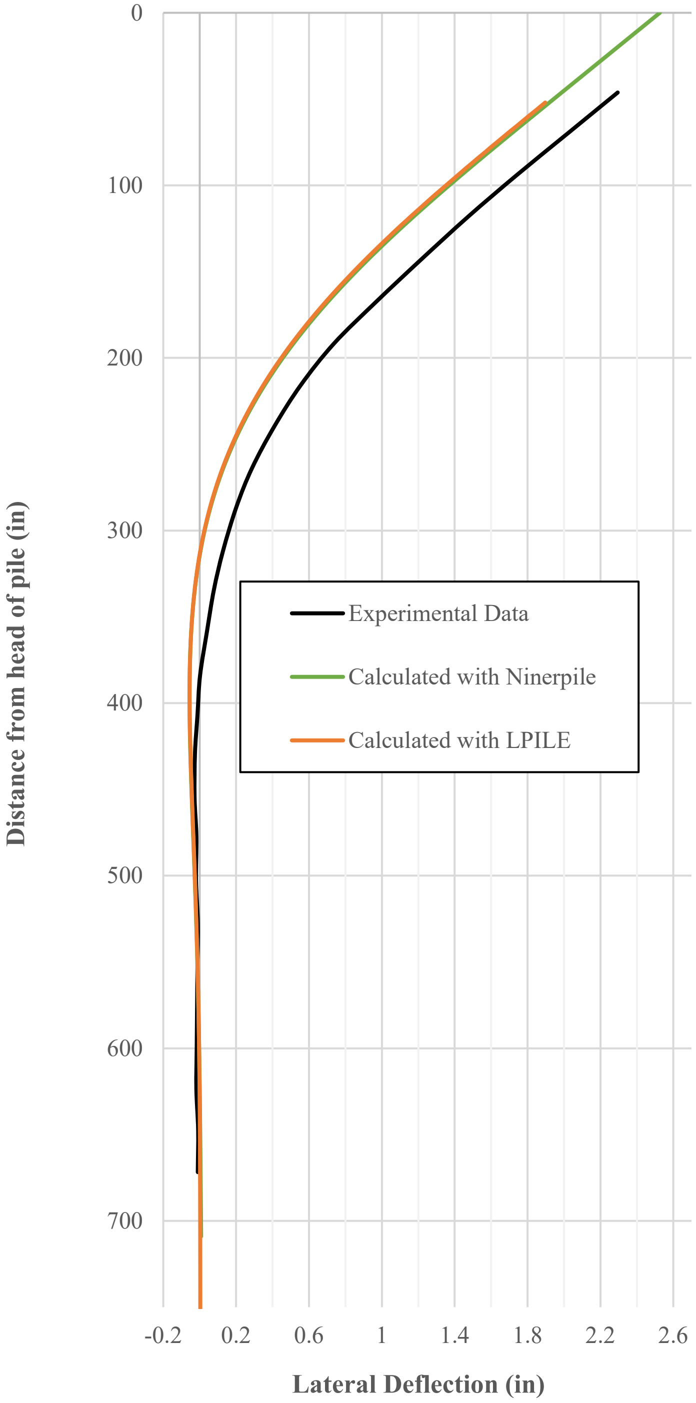

Figure 5.17. Concrete pile lateral deflection at 61,887 lb. lateral loading

5.2.3.2 Slope at pile head vs. positive load step

Both the LPILE and NinerPile models predicted the experimental slope at the pile head well. Since NinerPile considers the change in flexural rigidity with internal moment, the slope is more closely predicted at higher loads with NinerPile. Figure 5.18 displays this data.

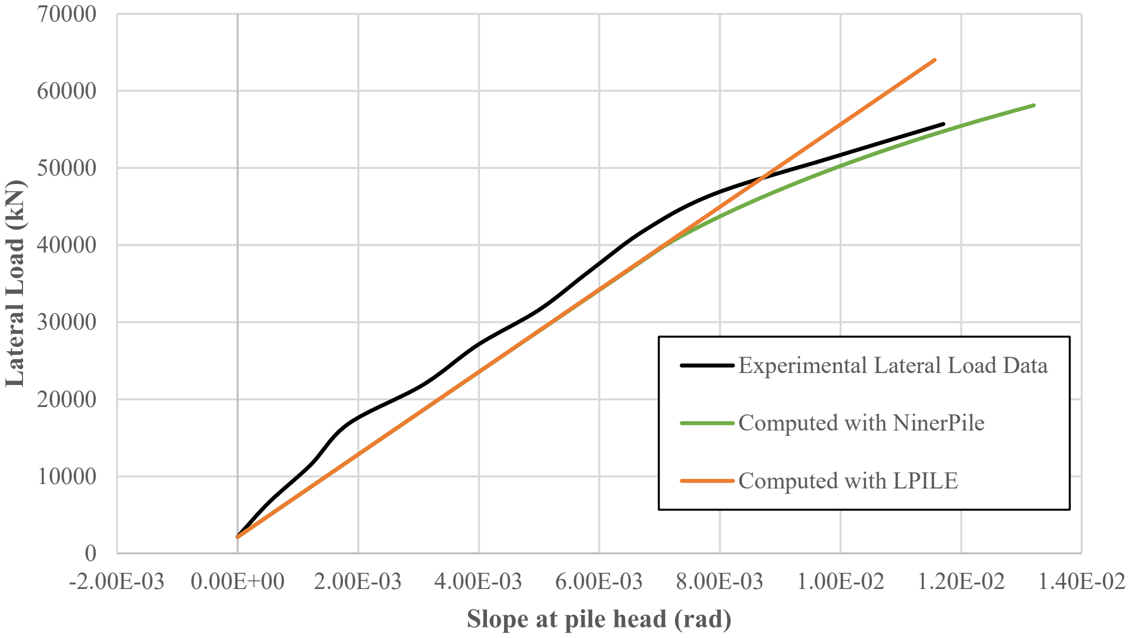

Figure 5.18. Slope at pile head vs. lateral load for concrete pile

5.2.3.3 Pile head displacement vs. load step for loading and unloading

NinerPile has been developed with an emphasis on proposing models for predicting the unloaded deflection. In this section, the secant stiffness degradation model is used. The initial stiffness increment for unloading used is 2, and the stiffness rate is degraded by 90% for each load step. These parameters are discussed in Chapter 4. Due to the definition of secant stiffness, the deflected soil spring must always return to the zero position under zero loading. For this reason, the analysis is only conducted until the unloading becomes linear for a short period of time. It is the opinion of the author that when the unloading path of the pile head becomes linear it should be interpolated to the deflection axis for a sustained deflection prediction. Figure 5.19 is drawn until the unloading path achieves the discussed shape.

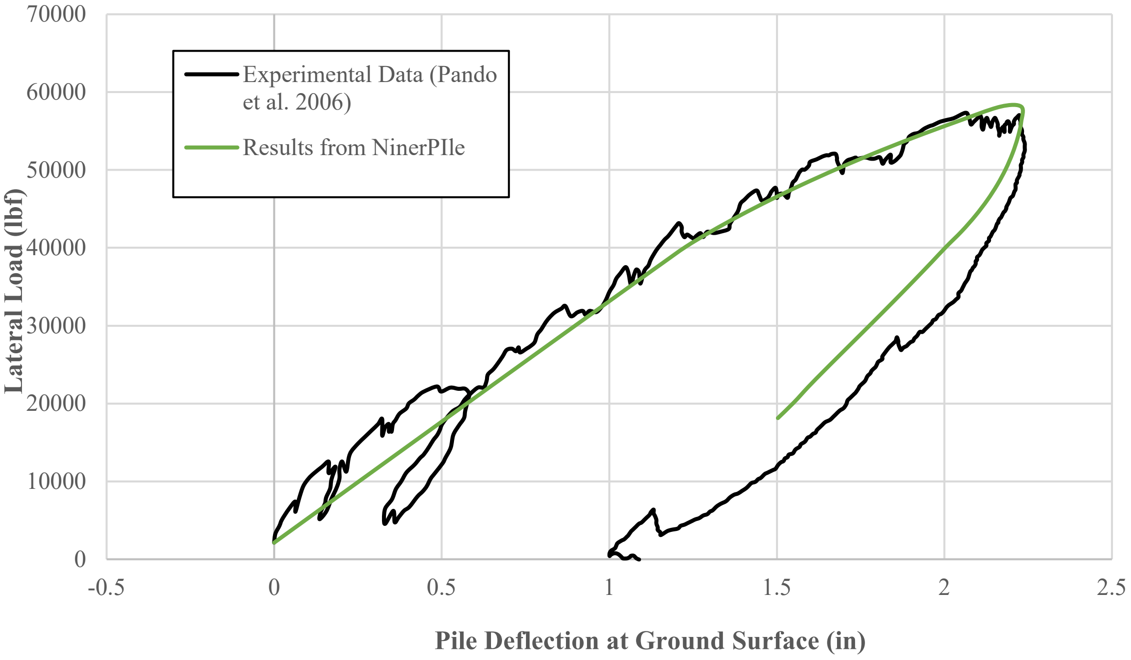

Figure 5.19. NinerPile prediction and experimental data for the concrete pile deflection

5.2.2.4 Spaghetti output plot from NinerPile

The below plot is the output from NinerPile displaying the displacement, slope, soil reaction, bending moment and shear force with depth along the pile for each step of the analysis.

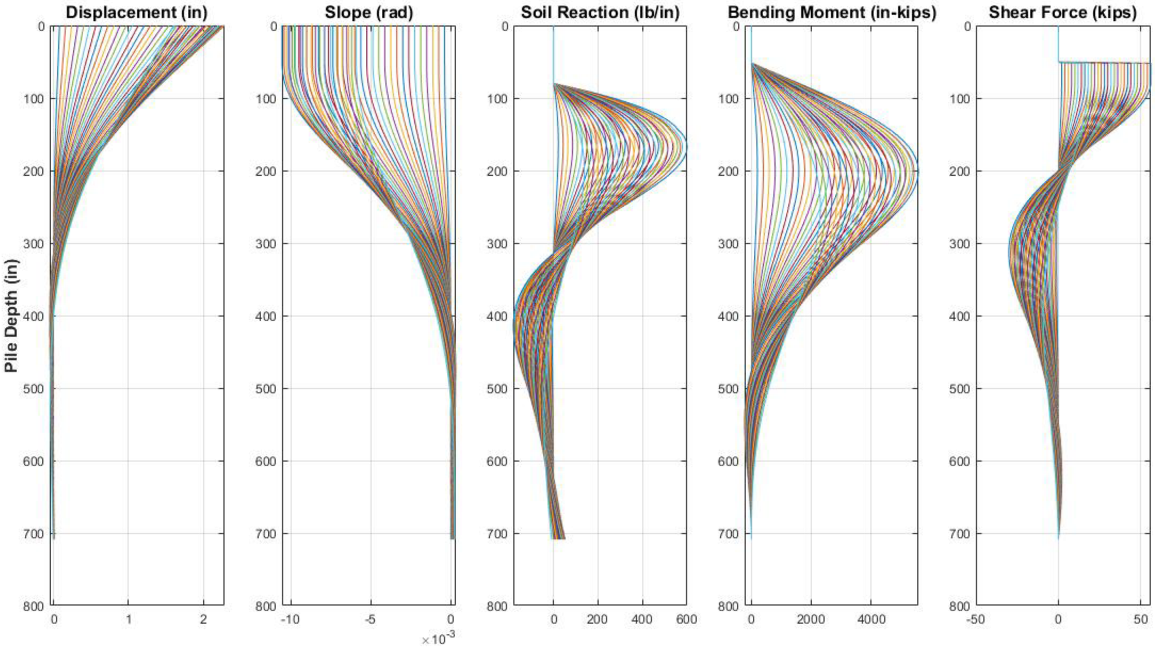

Figure 5.20. Spaghetti plot of NinerPile output for concrete pile

5.3 Polyethylene Composite (Plastic) pile

5.3.1 Introduction

The plastic pile was tested in the center of the site. The dimensions are shown in Figure 5.21. The input parameters for both NinerPile and LPILE are displayed in Tables 5.3 and 5.4 respectively. According to Pando et al (2006) the plastic pile had a constant EI over the range of moments experienced during testing. The EI is plotted vs. moment for consistency in Figure 5.22.

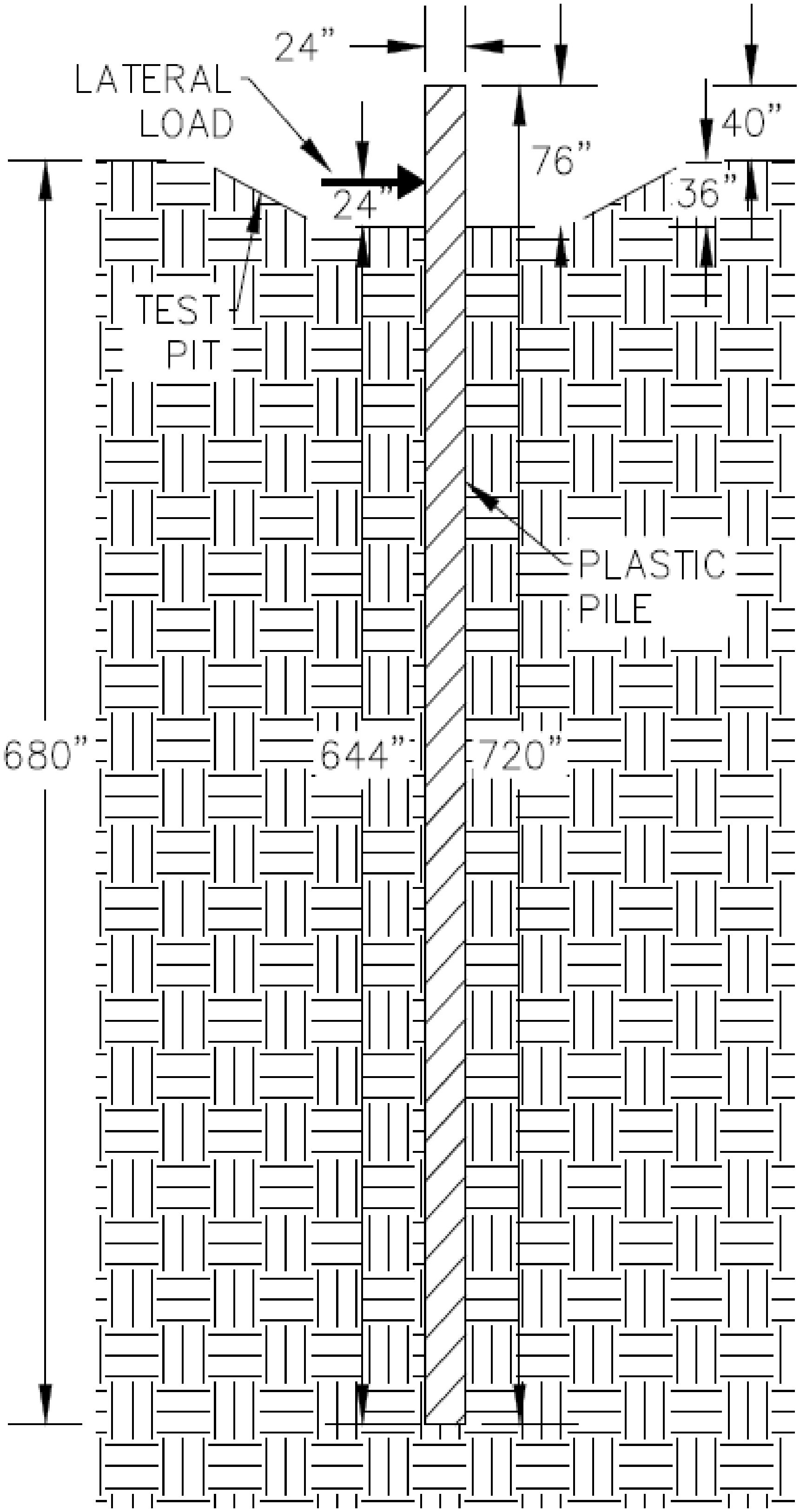

Figure 5.21. Testing and modeling configuration for the plastic pile

Table 5.3. NinerPile input parameters for the plastic pile

Parameter	Symbol in Program	Value
length (in)	Pile.L	720
width (in)	Pile.D	24
flexural regidity (kip-in^2)	Pile.EI, Pile.mcrack and Pile.EIcrack	see Figure 5.22
distance from pile head to soil surface (in)	PY.Depth	76
effective friction angle (°)	PY.phi	33
effective unit weight (lb/ft^3)	PY.gamma	66.84
depth stiffness factor (lb/in^3)	PY.k	14
tolerance of convergence	PY.tol	0.01
number of elements	femesh.nel	720
node for lateral load to be applied	Loading.node	48

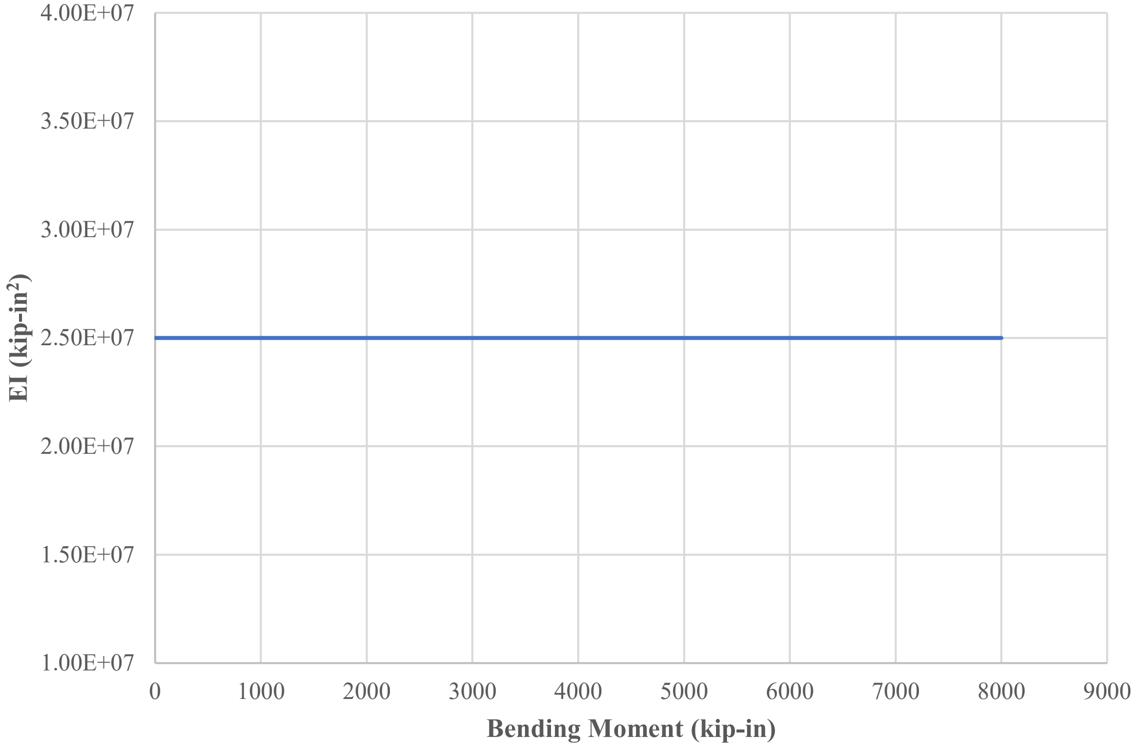

Figure 5.22. Plastic pile flexural rigidity vs. moment for NinerPile

Table 5.4. NinerPile input parameters for the plastic pile

Parameter	Value
length (in)	668
width (in)	24
flexural regidity (kip-in^2)	2.50 x 10^7
distance from pile head to soil surface (in)	24
effective friction angle (°)	33
effective unit weight (lb/ft^3)	66.84
depth stiffness factor (lb/in^3)	14
Soil Model	API Sand (O'Neill)

Note: Measurements are adjusted because LPILE does not model section above lateral load

5.3.2 NinerPile and LPILE detailed results comparison at 31,494 lb lateral loading

The following plots are a comparison between results generated by both LPILE and NinerPile for the plastic pile. Shear, moment, slope, soil resistance and lateral deflection are plotted vs. depth from the head of the pile at 31,494 pounds of lateral loading. The NinerPile results are all in close agreement with the LPILE results as well as the experimental data, which serves as verification of the NinerPile code.

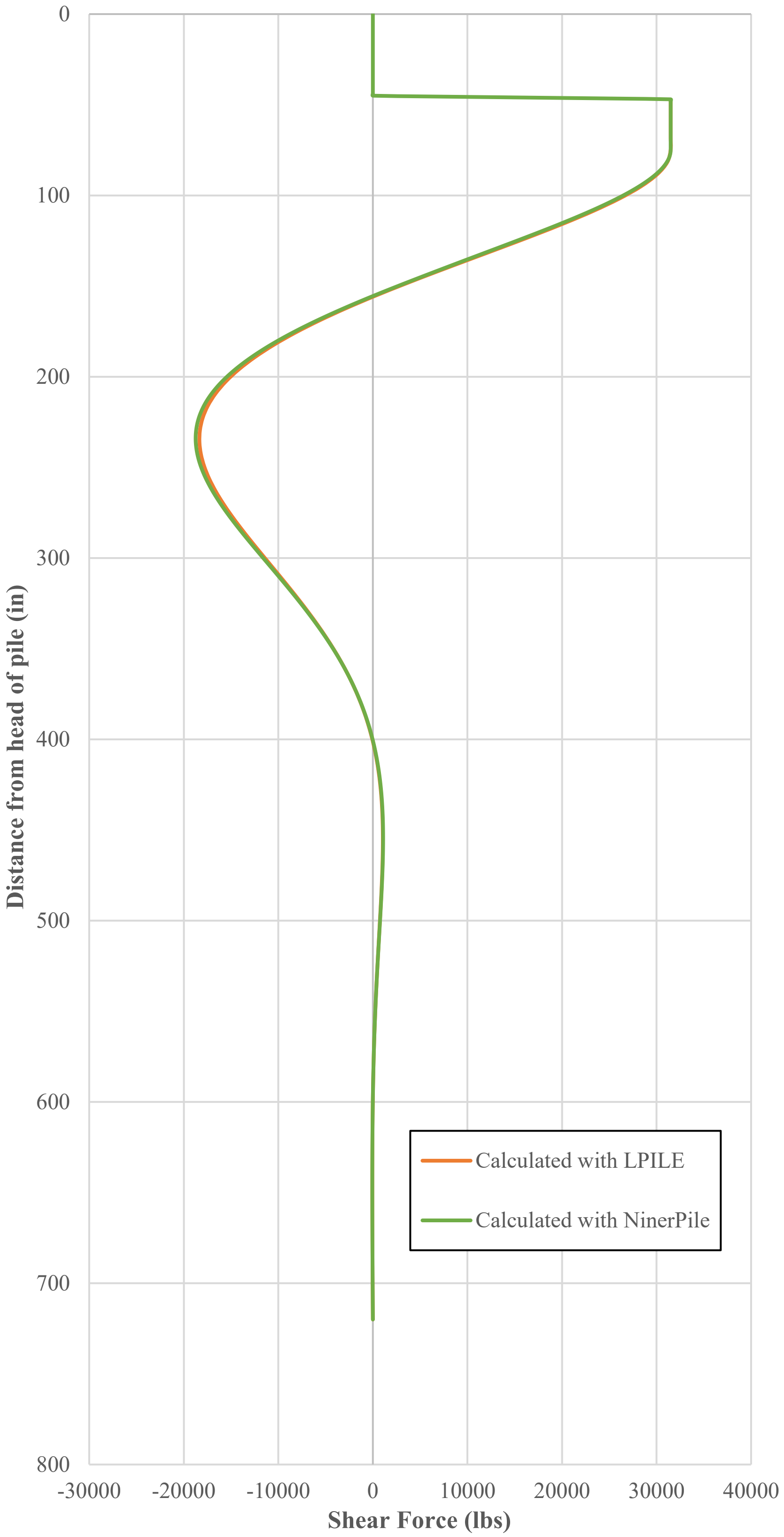

Figure 5.23. NinerPile vs. LPILE shear verification at 31,494 lb. lateral loading

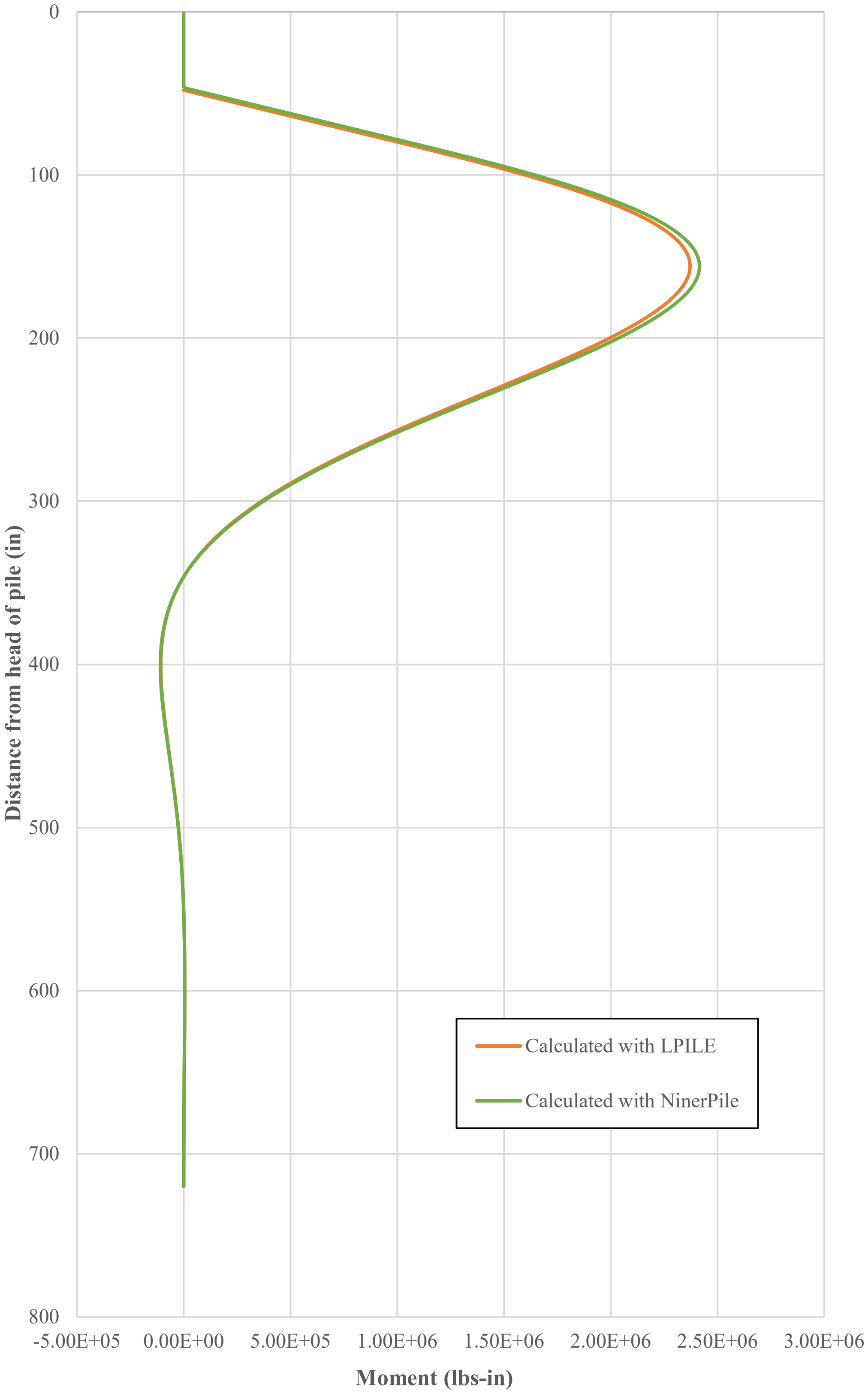

Figure 5.24. NinerPile vs. LPILE moment verification at 31,494 lb. lateral loading

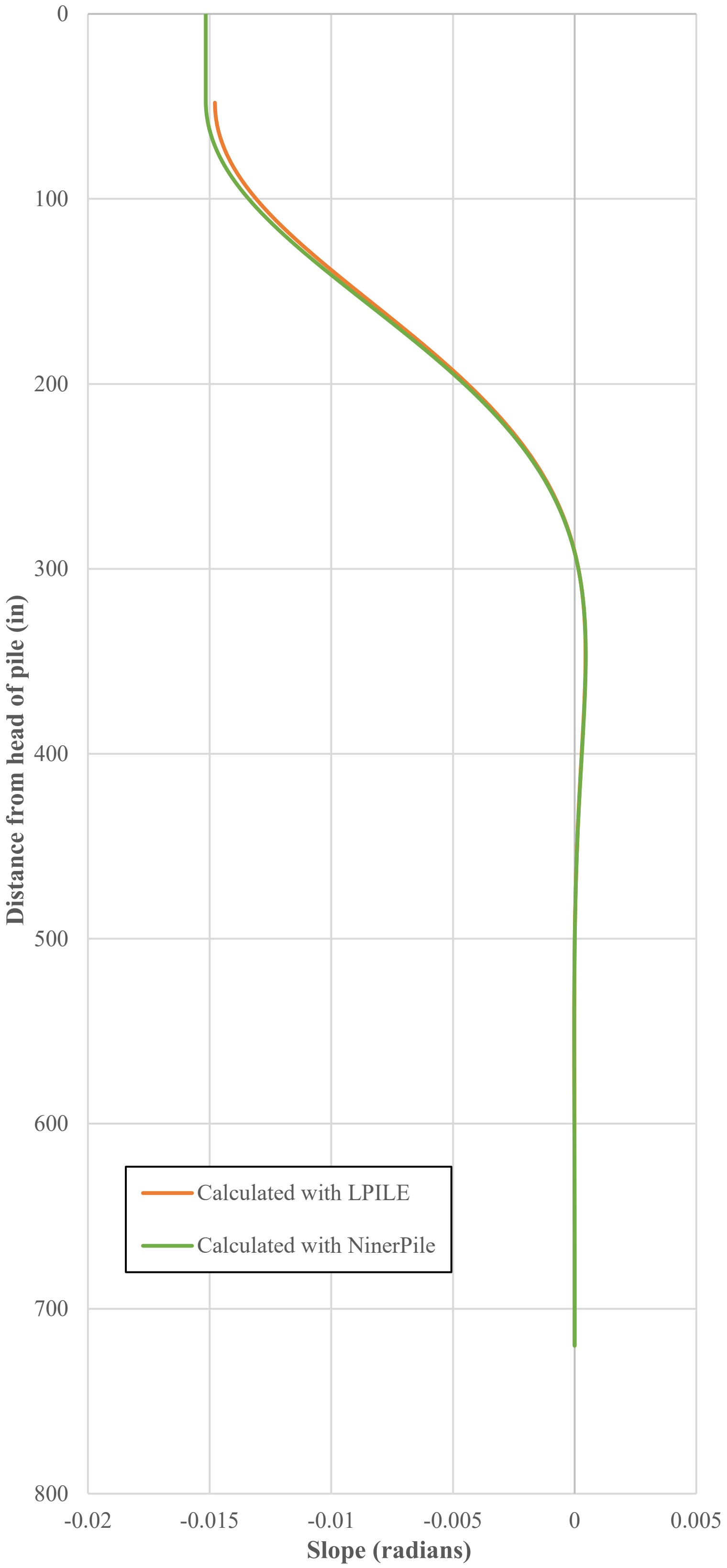

Figure 5.25. NinerPile vs. LPILE slope verification at 31,494 lb. lateral loading

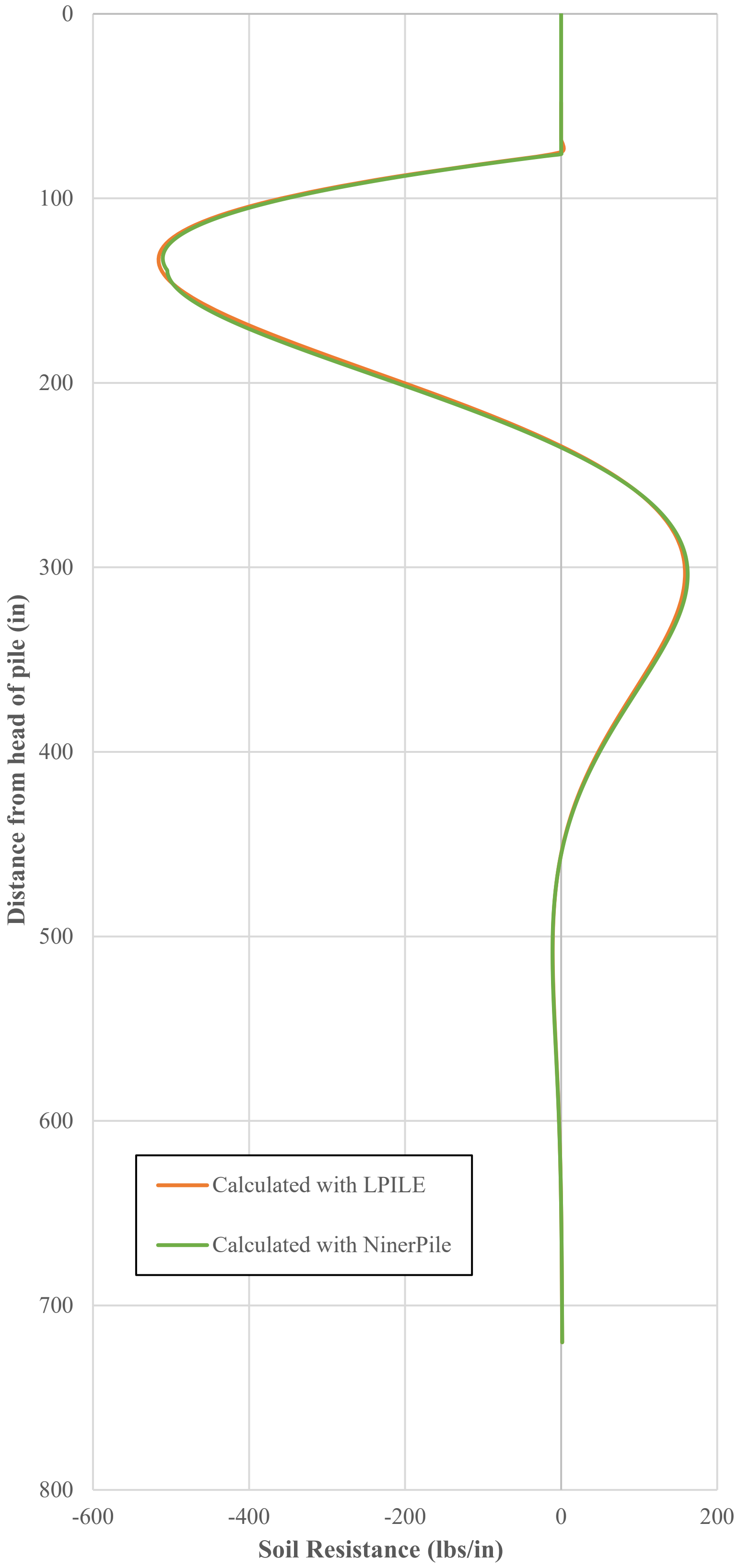

Figure 5.26. NinerPile vs. LPILE soil resistance verification at 31,494 lb. lateral loading

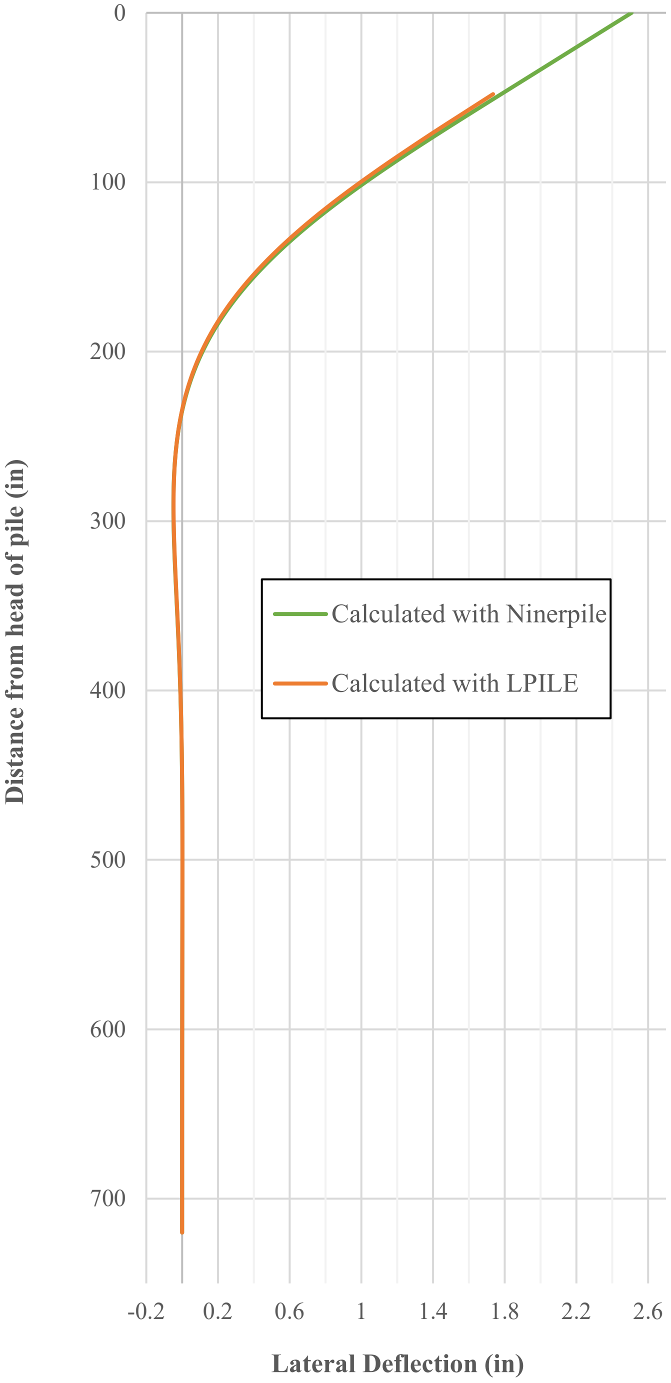

Figure 5.27. NinerPile vs. LPILE lateral deflection verification at 31,494 lb. lateral loading

5.3.3 Test results, NinerPile and LPILE deflection comparison at various loads

For each load step that a full set of deflection data was published in Pando et al. (2006) NinerPile and LPILE were used to model the deflections. The results are presented in this section.

5.3.3.1 Deflection plots for various positive load steps

The following deflection plots are a comparison between results generated by LPILE, NinerPile and experimental data for the plastic pile. Lateral deflection is plotted vs. depth from the head of the pile at the following lateral loads: 10,813 lbs., 23,020 lbs., 31,494 lbs., 41,161 lbs., 51,884 lbs. and 61,887 lbs. The NinerPile results are in close agreement with both LPILE results and experimental data.

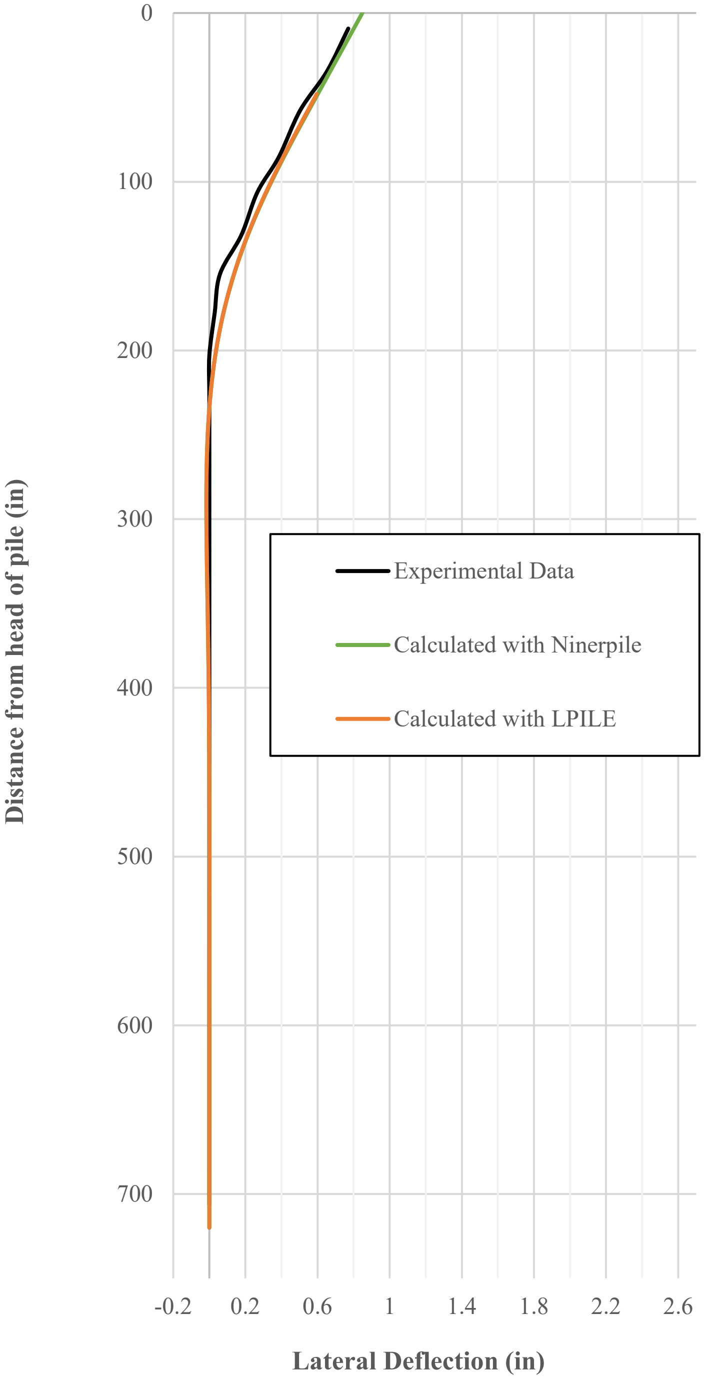

Figure 5.28. Plastic pile lateral deflection at 10,813 lb. lateral loading

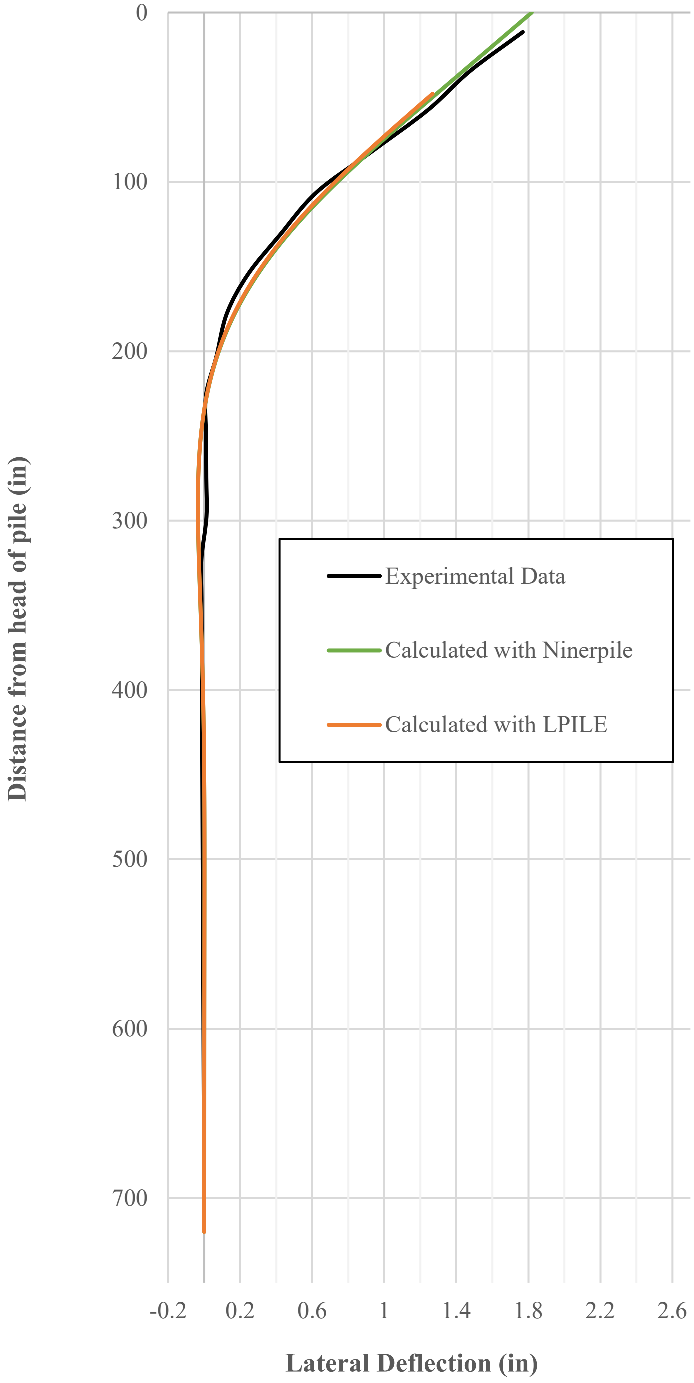

Figure 5.29. Plastic pile lateral deflection at 23,020 lb. lateral loading

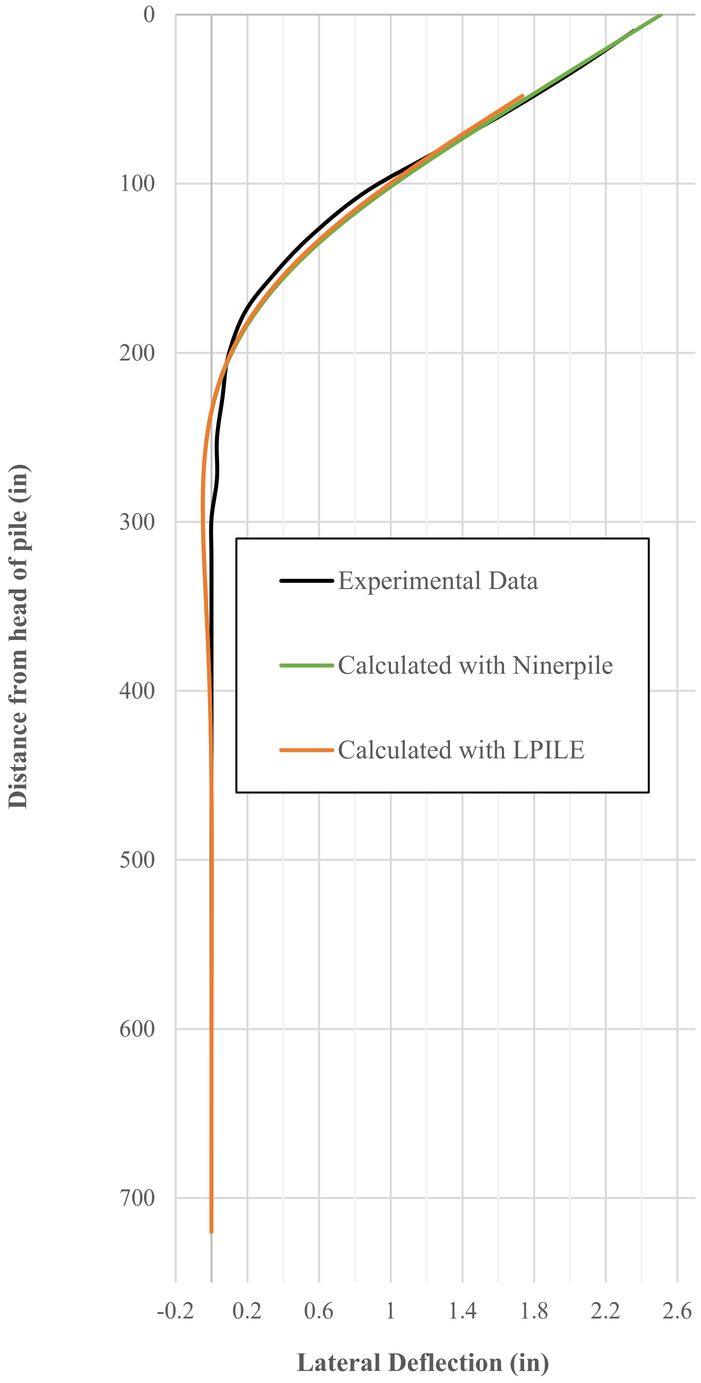

Figure 5.30. Plastic pile lateral deflection at 31,494 lb. lateral loading

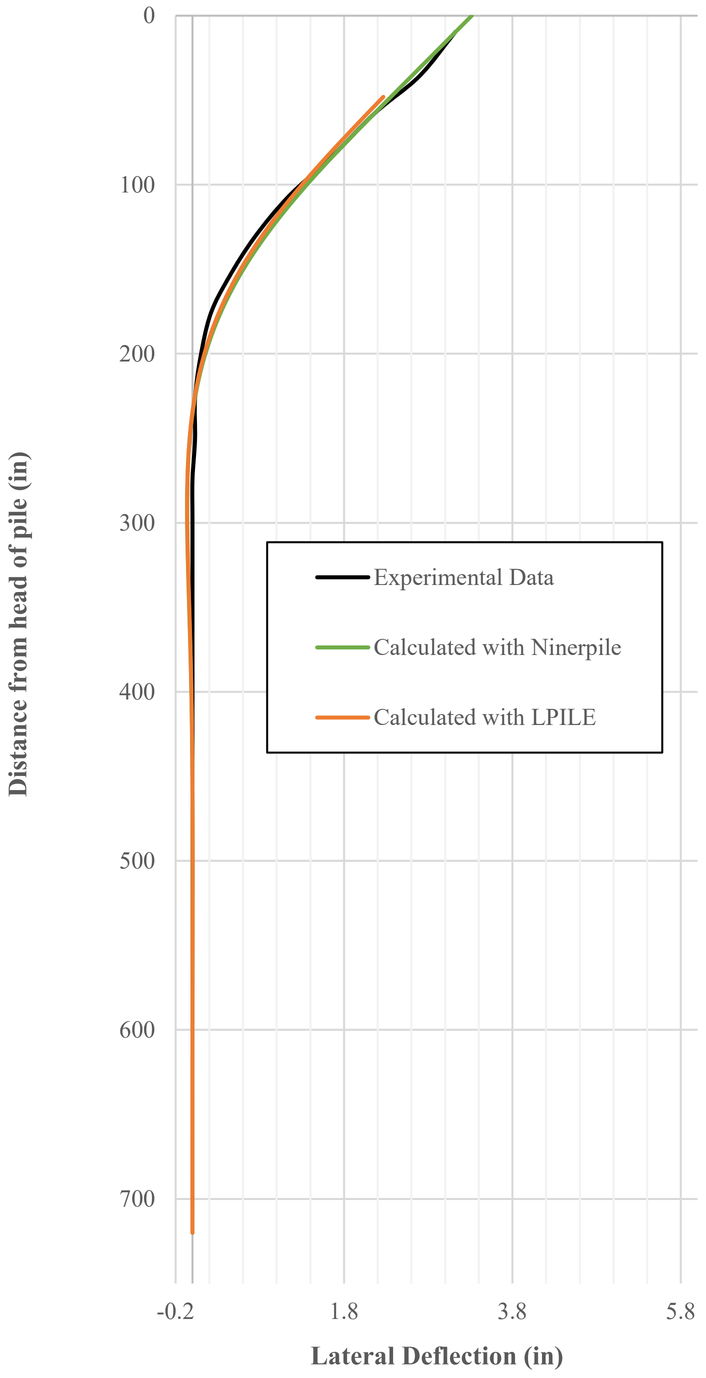

Figure 5.31. Plastic pile lateral deflection at 41,161 lb. lateral loading

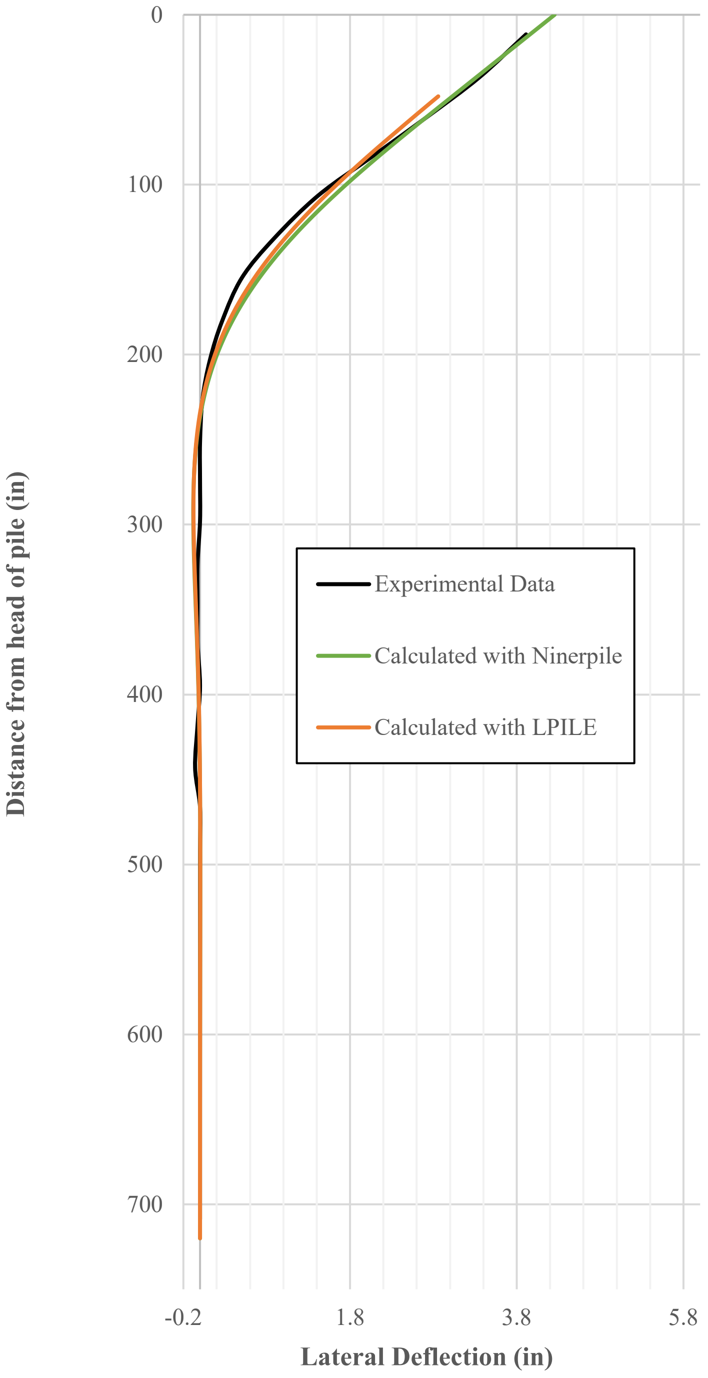

Figure 5.32. Plastic pile lateral deflection at 51,884 lb. lateral loading

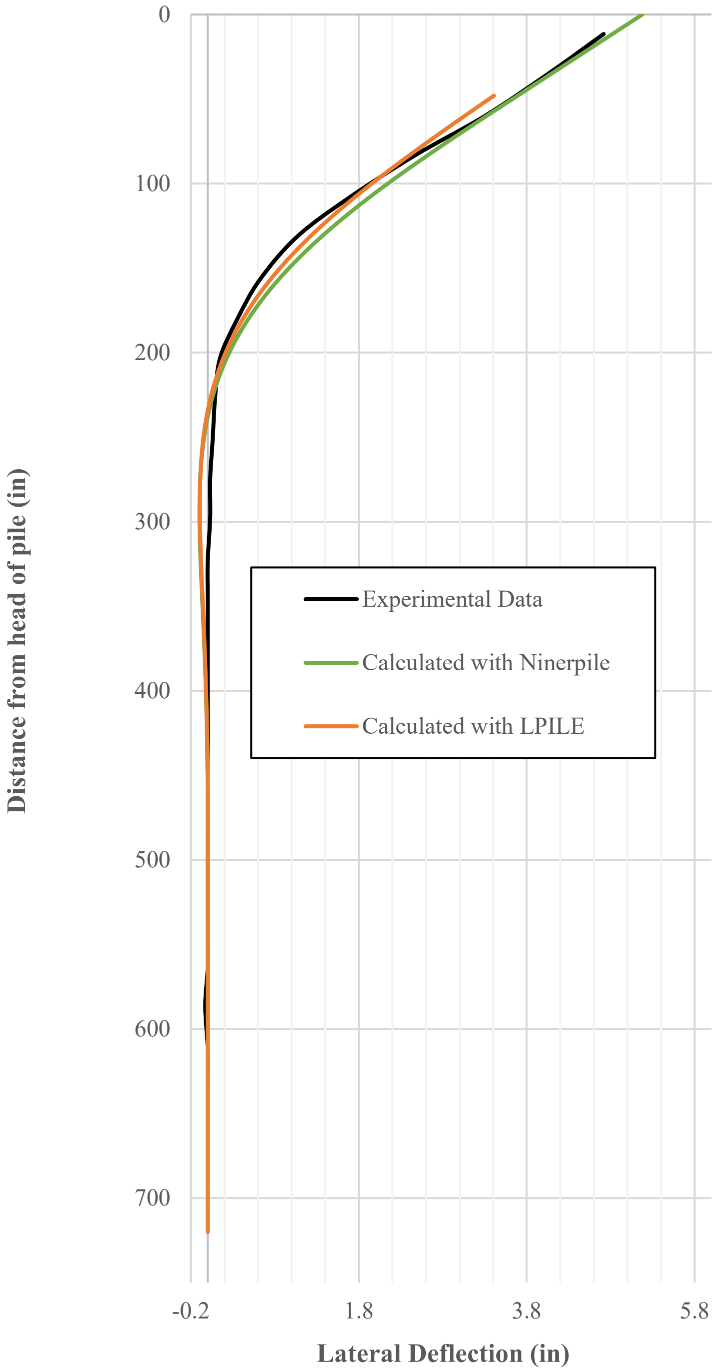

Figure 5.33. Plastic pile lateral deflection at 61,887 lb. lateral loading

5.3.3.2 Slope at pile head vs. positive load step

Both the LPILE and NinerPile models predicted the experimental slope at the pile head well. The comparison is shown in Figure 5.34.

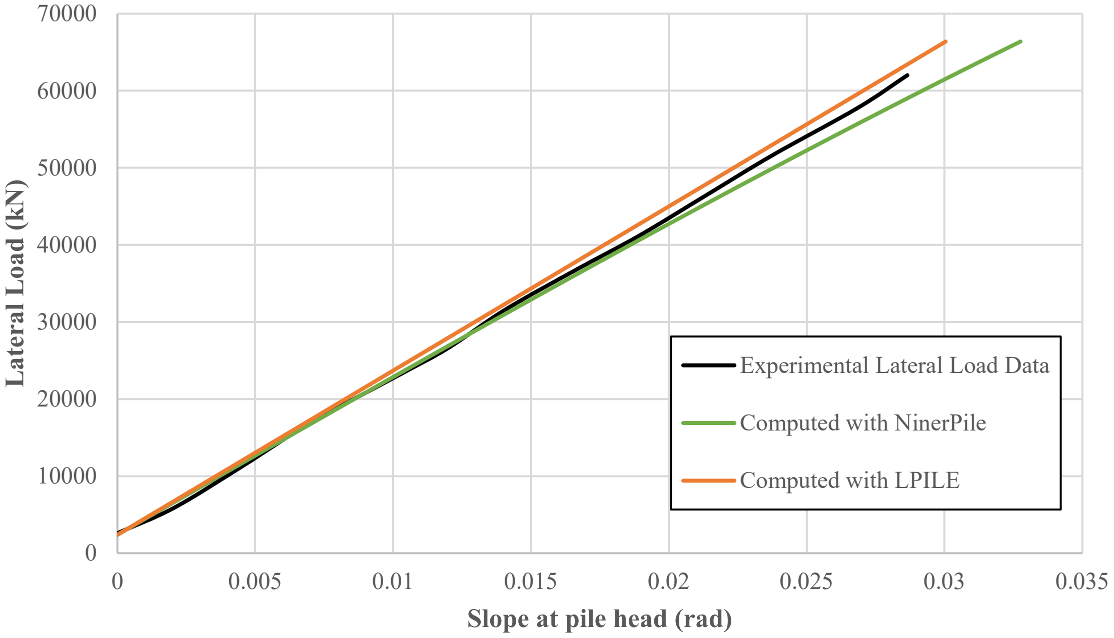

Figure 5.34. Slope at pile head vs. lateral load for plastic pile

5.3.2.3 Pile head displacement vs. load step for loading and unloading

In this section, the secant stiffness degradation model is used to model unloading in NinerPile. The initial stiffness increment for unloading used is 2, and the stiffness rate is degraded by 90% for each load step. This process is repeated until 1/2 of the unloading has been completed. The rest of the unloading is competed with a uniform degradation rate. For this pile, it was determined that this method best captures the shape and trajectory of the unloading loop.

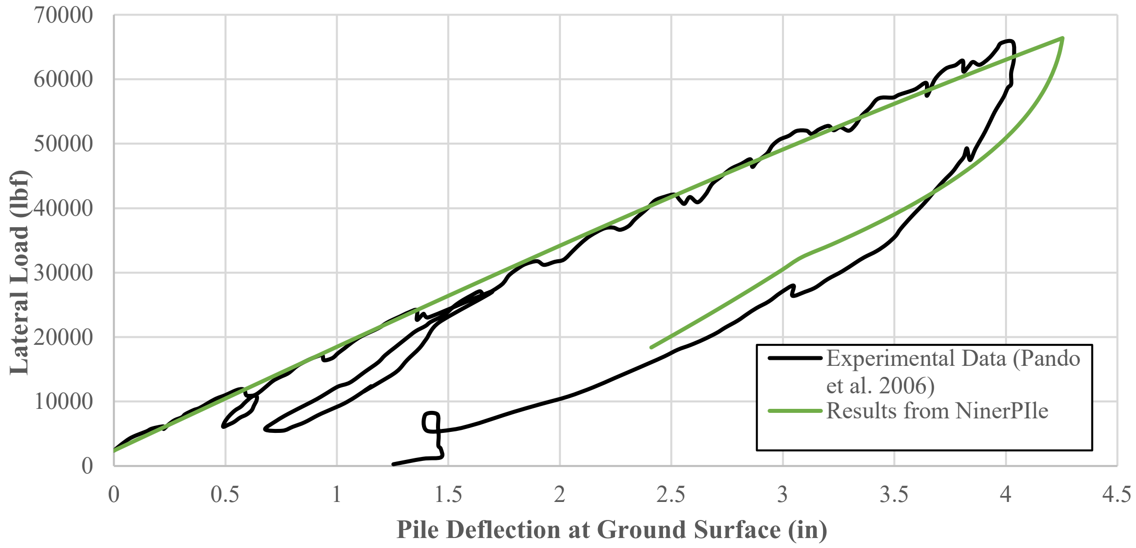

Figure 5.35. NinerPile prediction and experimental data for the plastic pile deflection

5.3.3.4 Spaghetti output plot from NinerPile

The below plot is the output from NinerPile displaying the displacement, slope, soil reaction, bending moment and shear force with depth along the pile for each step of the analysis.

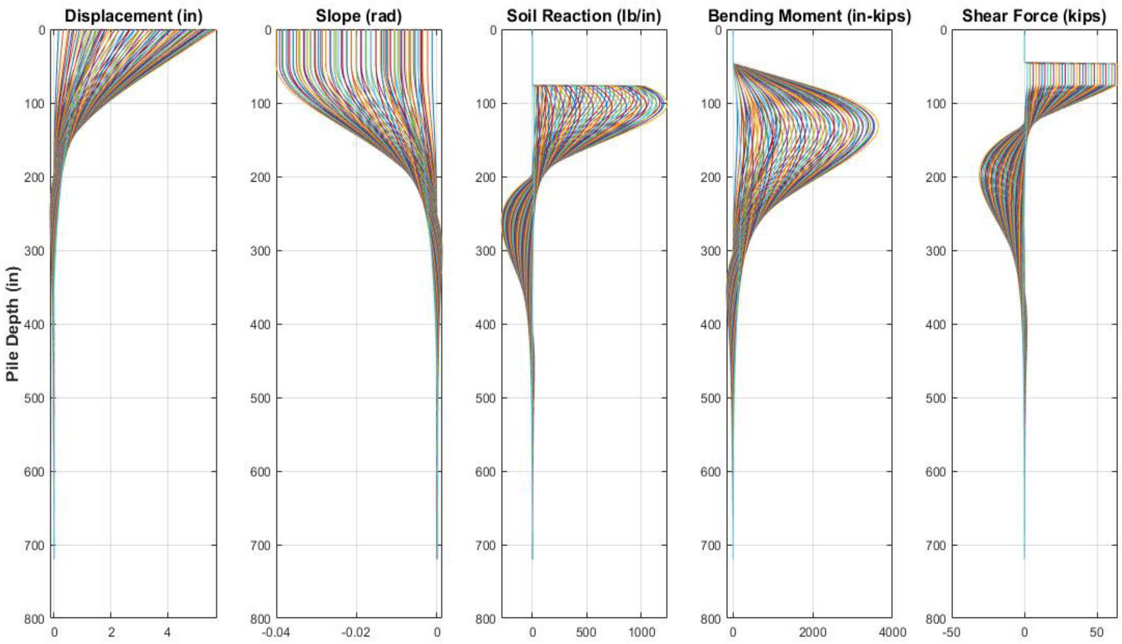

Figure 5.36. Spaghetti plot of NinerPile output for plastic pile

5.4 Fiber reinforced polymer pile

5.4.1 Introduction

The FRP pile was tested at the northern end of the site. The testing configuration consisted of an excavated pit for the testing equipment to be installed. The dimensions are shown in Figure 5.37. The input parameters for both NinerPile and LPILE are displayed in Tables 5.4 and 5.5 respectively.

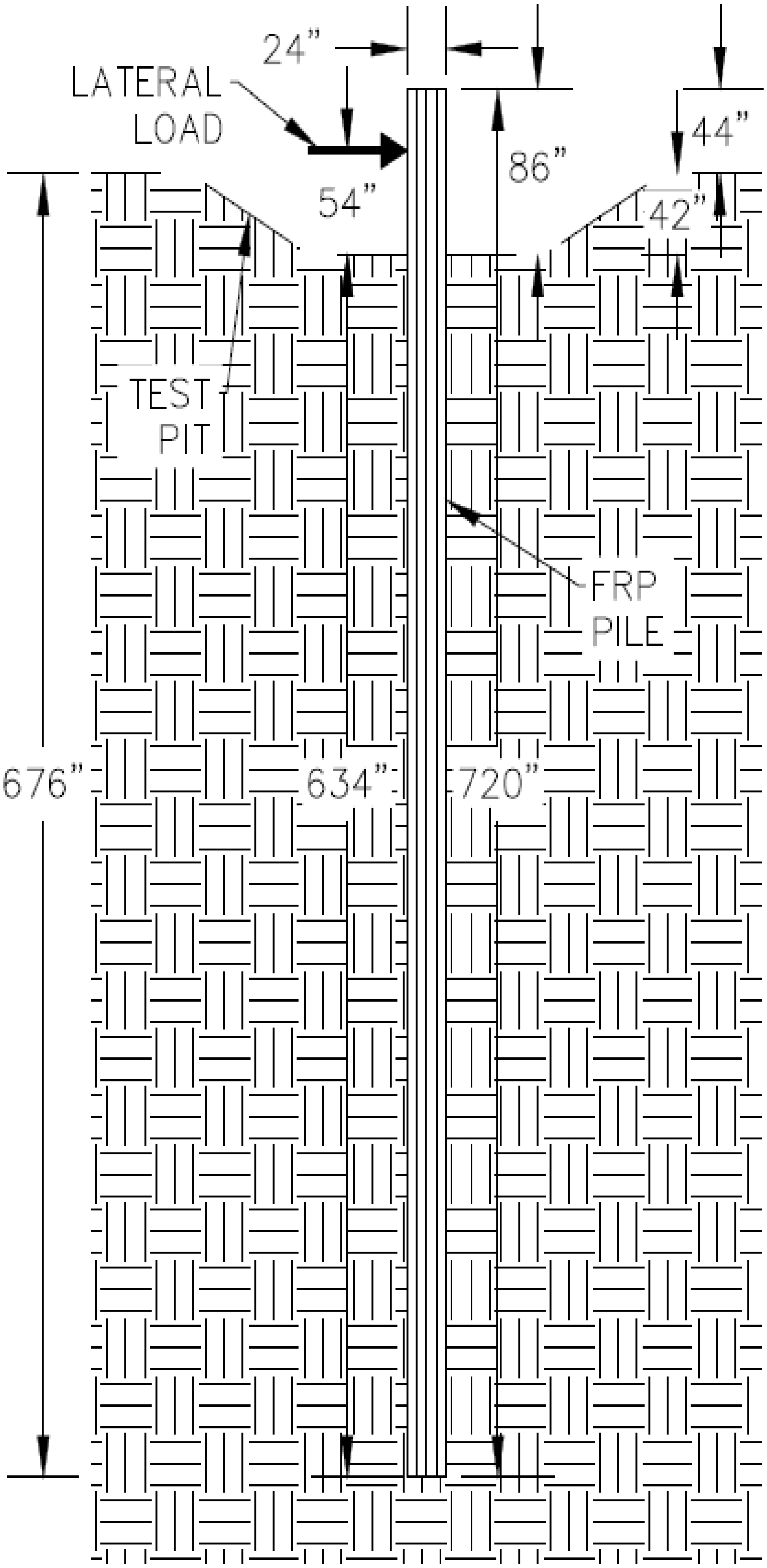

Figure 5.37. Testing and modeling configuration for the FRP pile

Table 5.5. NinerPile input parameters for the FRP pile

Parameter	Symbol in Program	Value
length (in)	Pile.L	720
width (in)	Pile.D	24
flexural regidity (kip-in^2)	Pile.EI, Pile.mcrack and Pile.EIcrack	see Figure 5.38
distance from pile head to soil surface (in)	PY.Depth	86
effective friction angle (°)	PY.phi	35
effective unit weight (lb/ft^3)	PY.gamma	66.84
depth stiffness factor (lb/in^3)	PY.k	40
tolerance of convergence	PY.tol	0.01
number of elements	femesh.nel	720
node for lateral load to be applied	Loading.node	31

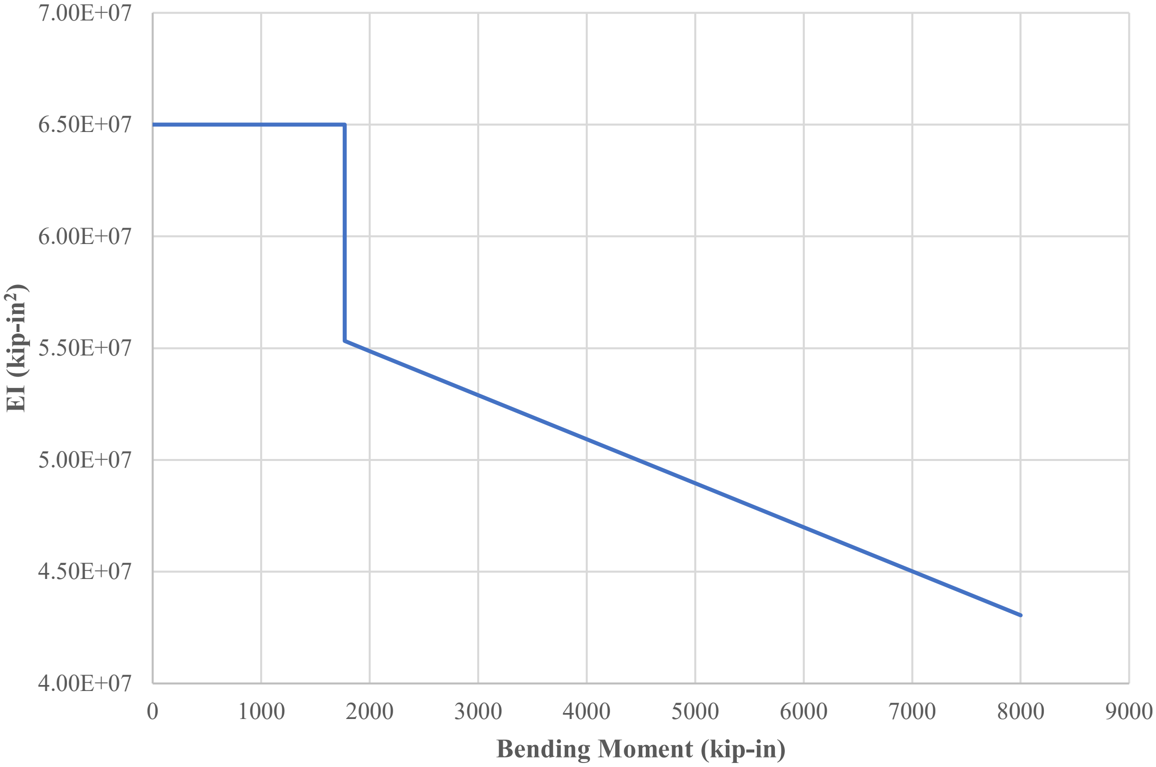

Figure 5.38. FRP pile flexural rigidity vs. moment for NinerPile

Table 5.6. NinerPile input parameters for the FRP pile

Parameter	**Value**
length (in)	688
width (in)	24
flexural regidity (kip-in^2)	6.50 x 10^7
distance from pile head to soil surface (in)	54
effective friction angle (°)	35
effective unit weight (lb/ft^3)	66.84
depth stiffness factor (lb/in^3)	40
Soil Model	API Sand (O'Neill)

Note: Measurements are adjusted because LPILE does not model section above lateral load

5.4.2 NinerPile and LPILE detailed results comparison at 32,551 lb lateral loading

The following plots are a comparison between results generated by both LPILE and NinerPile for the fiber reinforced polymer pile. Shear, moment, slope, soil resistance and lateral deflection are plotted vs. depth from the head of the pile at 32,551 pounds of lateral loading. The NinerPile results are all in close agreement with the LPILE results as well as the experimental data, which serves as verification of the NinerPile code.

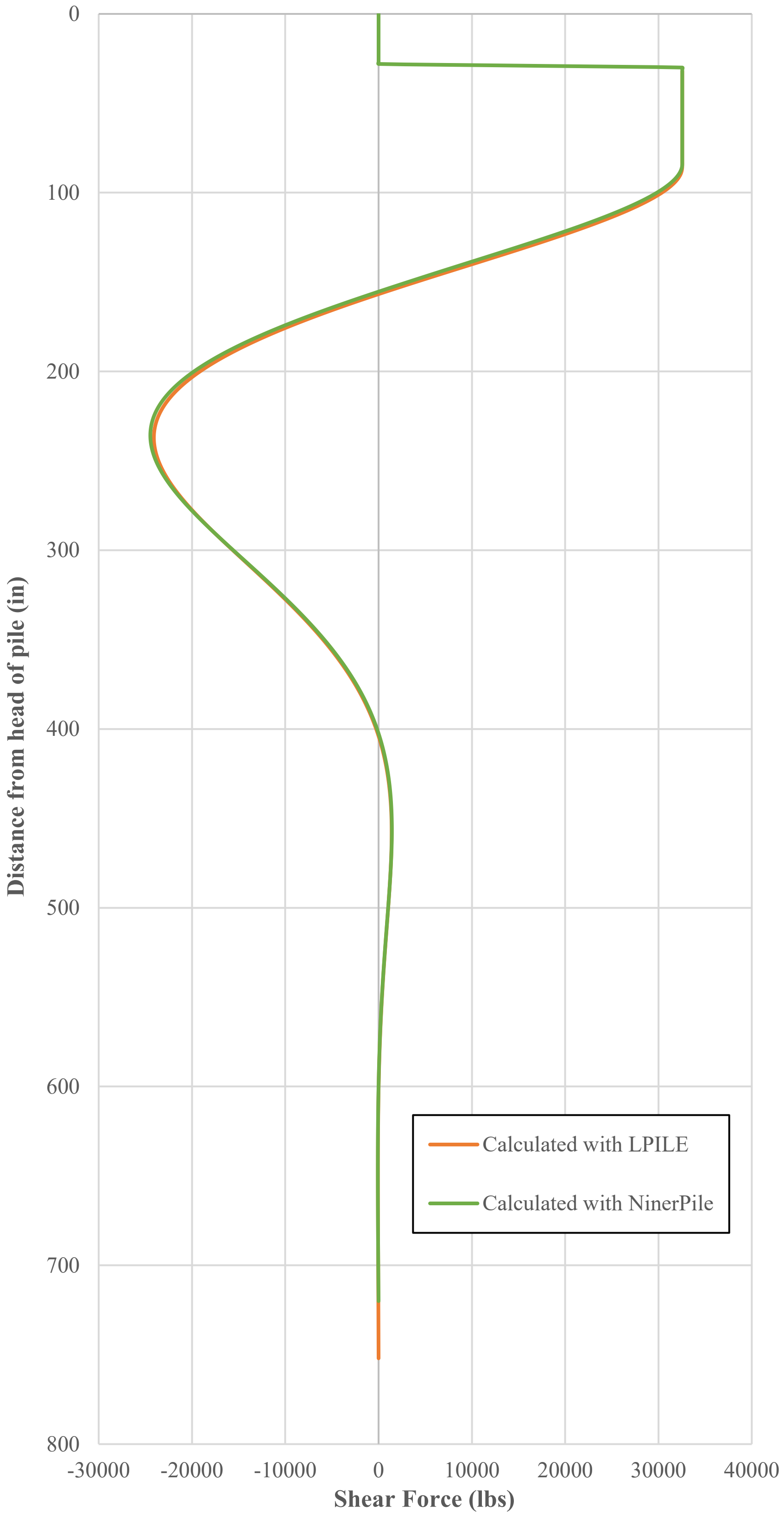

Figure 5.39. NinerPile vs. LPILE shear verification at 32,551 lb. lateral loading

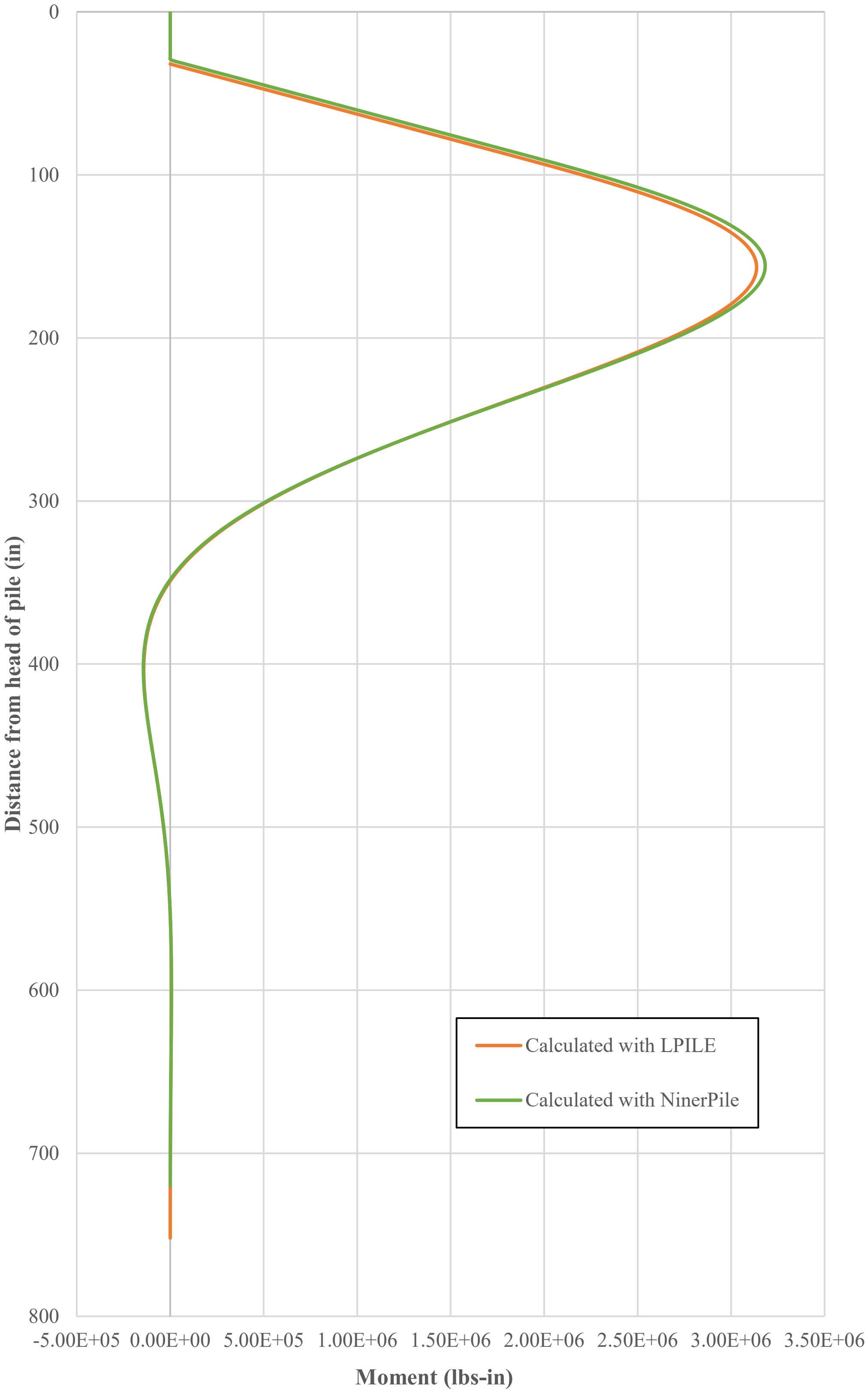

Figure 5.40. NinerPile vs. LPILE moment verification at 32,551 lb. lateral loading

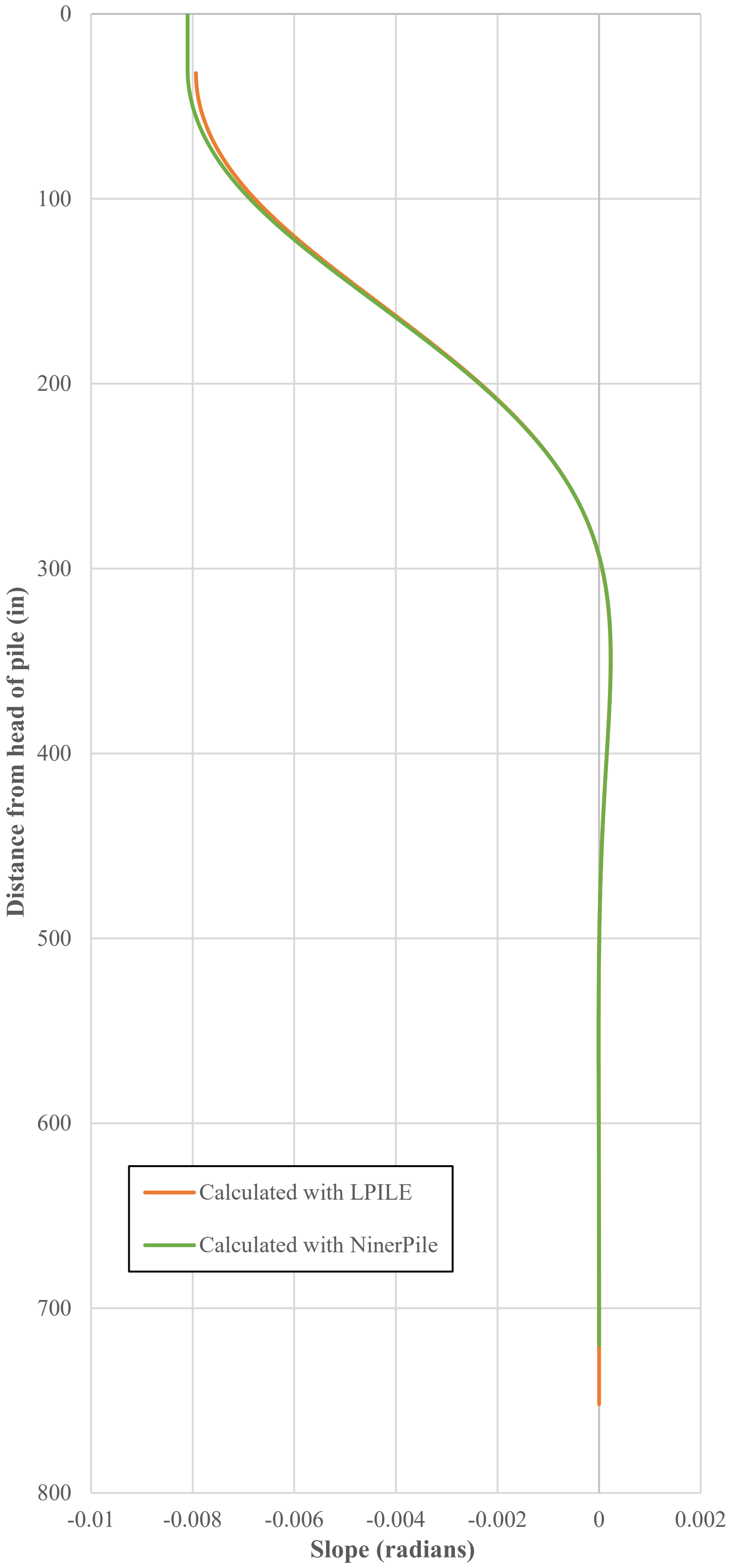

Figure 5.41. NinerPile vs. LPILE slope verification at 32,551 lb. lateral loading

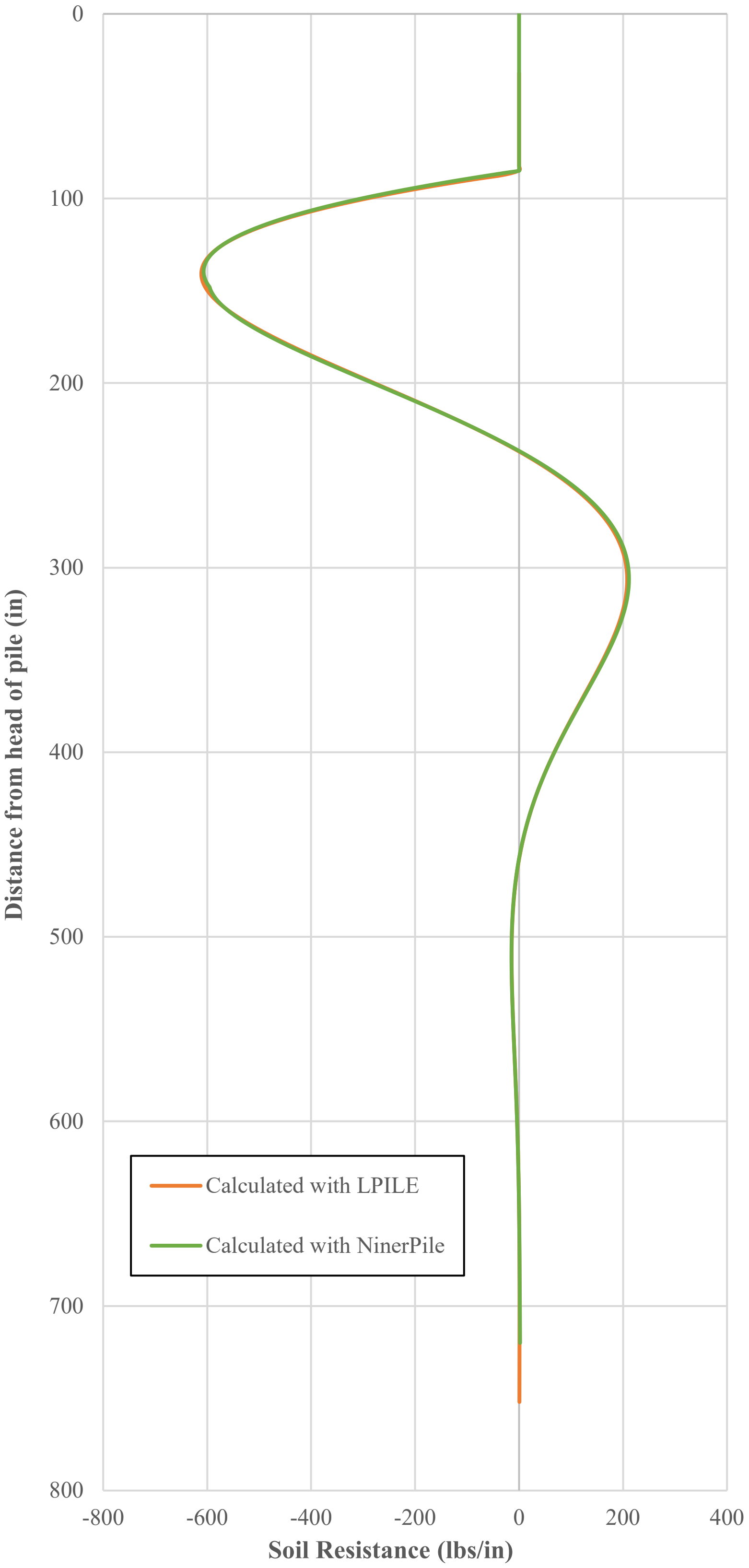

Figure 5.42. NinerPile vs. LPILE soil resistance at 32,551 lb. lateral loading

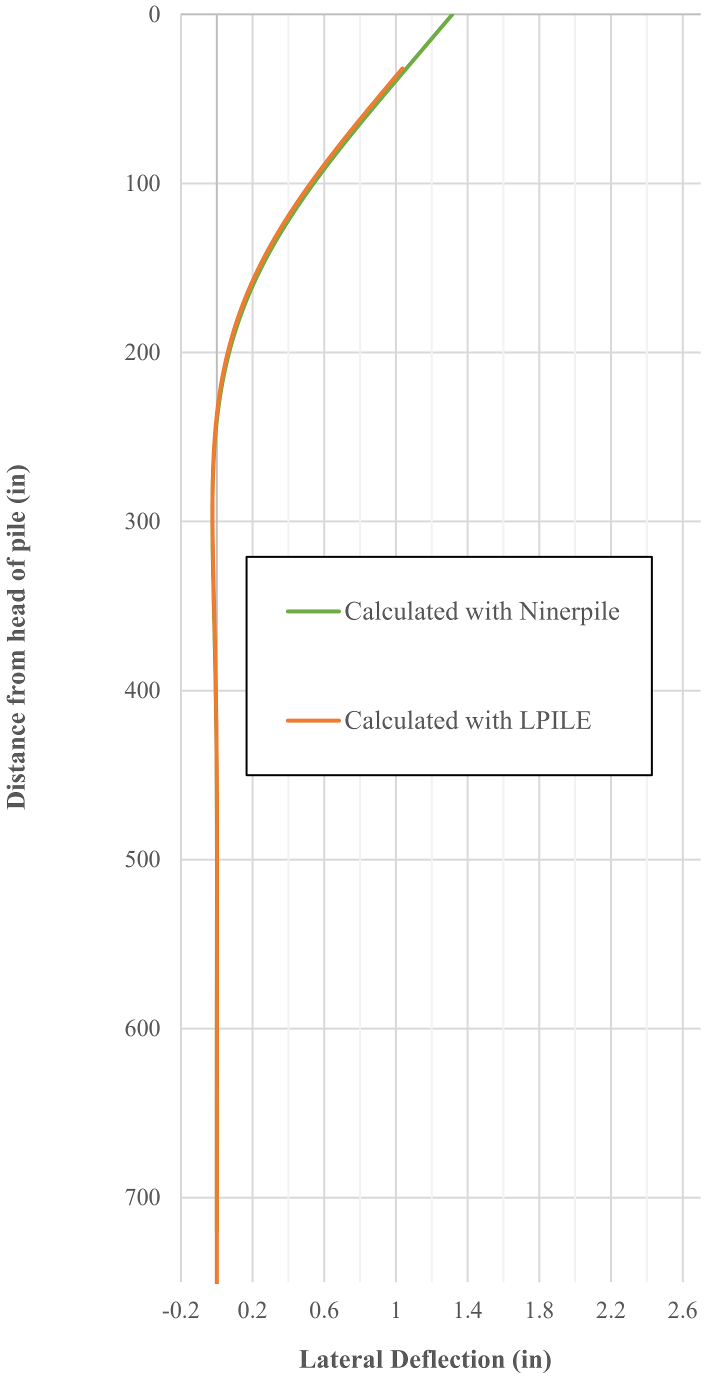

Figure 5.43. NinerPile vs. LPILE lateral deflection verification at 32,551 lb. lateral loading

5.4.3 Test results, NinerPile and LPILE deflection comparison at various loads

For each load step that a full set of deflection data was published in Pando et al. (2006) NinerPile and LPILE were used to model the deflections. The results are presented in this section.

5.4.3.1 Deflection plots for various positive load steps

The following deflection plots are a comparison between results generated by LPILE, NinerPile and experimental data for the fiber reinforced polymer pile. Lateral deflection is plotted vs. depth from the head of the pile at the following lateral loads: 11,600 lbs., 21,581 lbs., 32,551 lbs., 41,835 lbs., 51,726 lbs. and 60,808 lbs. The NinerPile results are in close agreement with both LPILE results and experimental data.

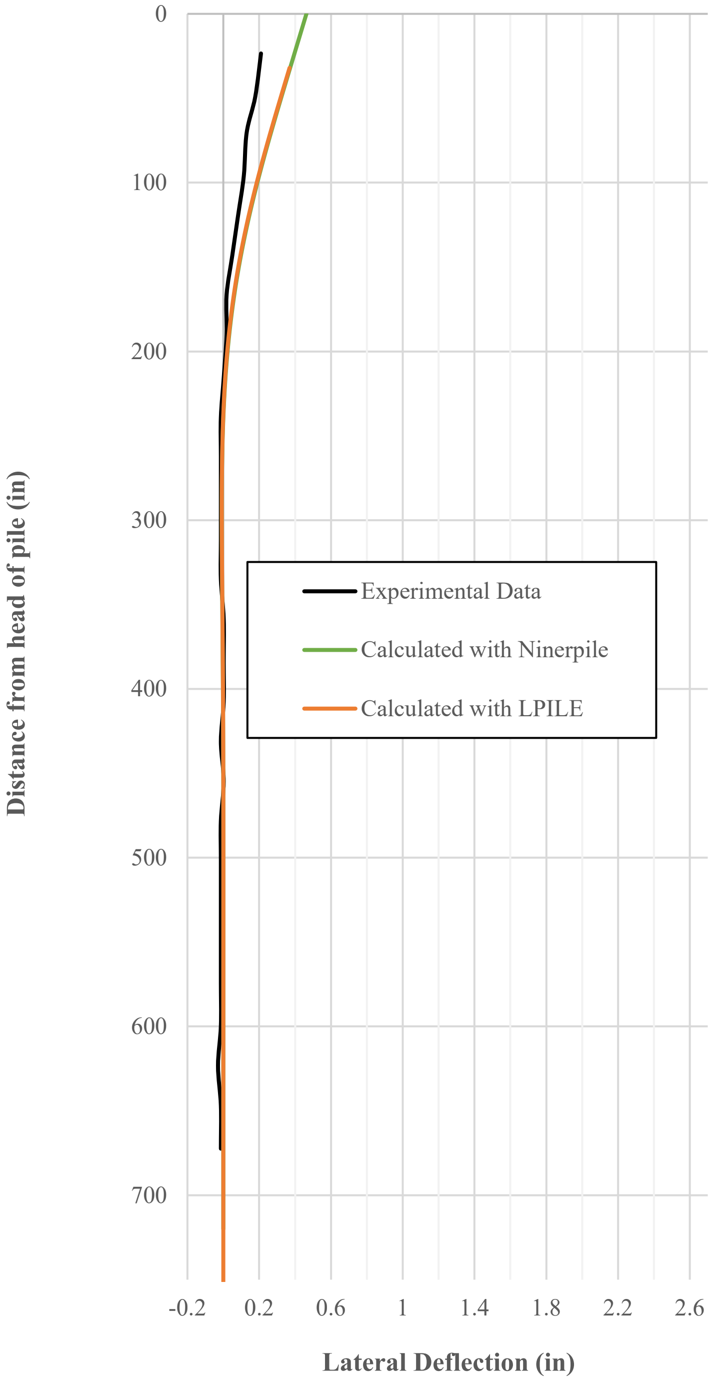

Figure 5.44. FRP pile lateral deflection at 11,600 lb. lateral loading

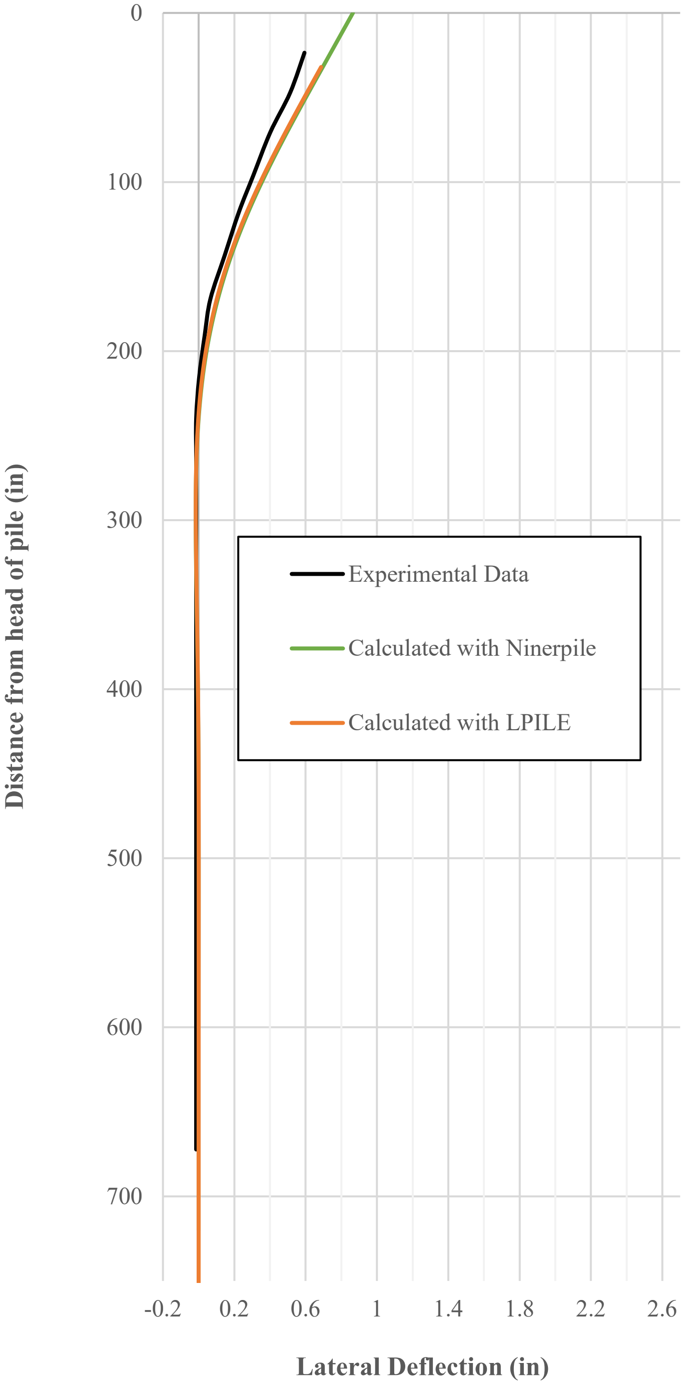

Figure 5.45. FRP pile lateral deflection at 21,581 lb. lateral loading

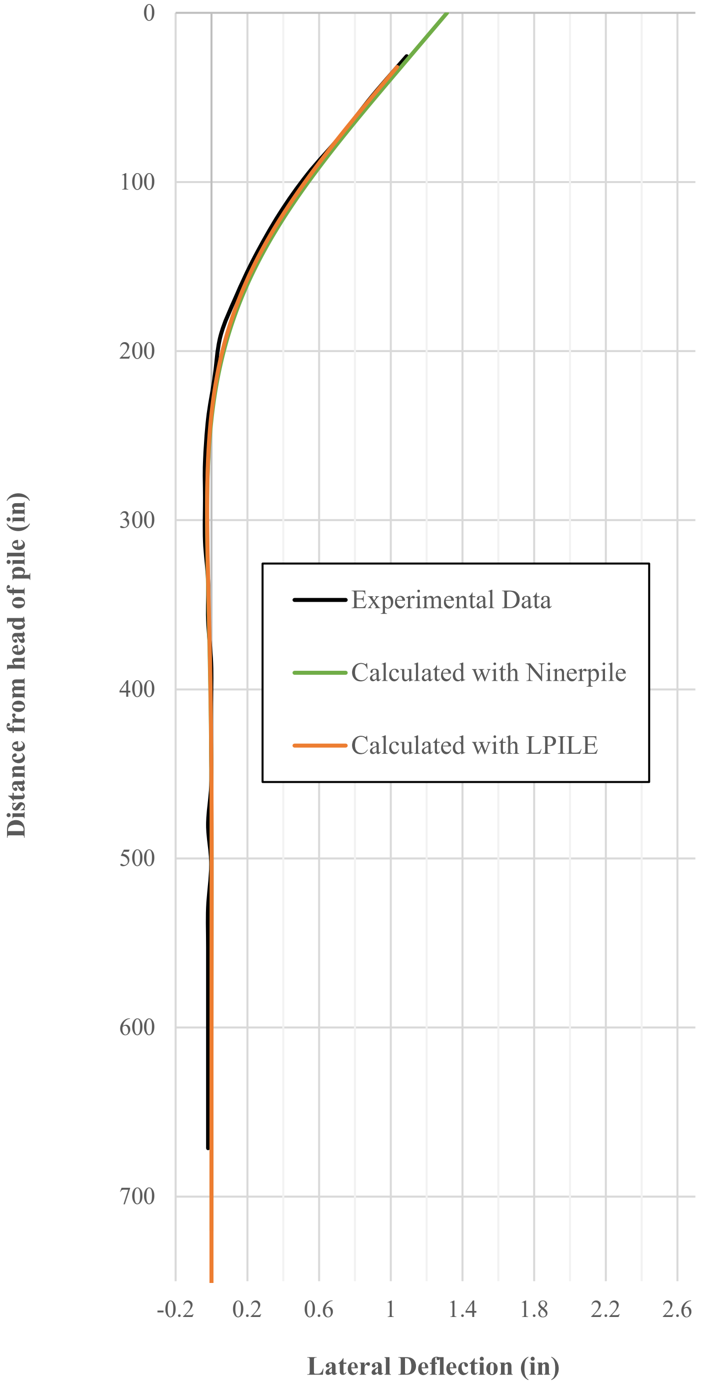

Figure 5.46. FRP pile lateral deflection at 32,551 lb. lateral loading

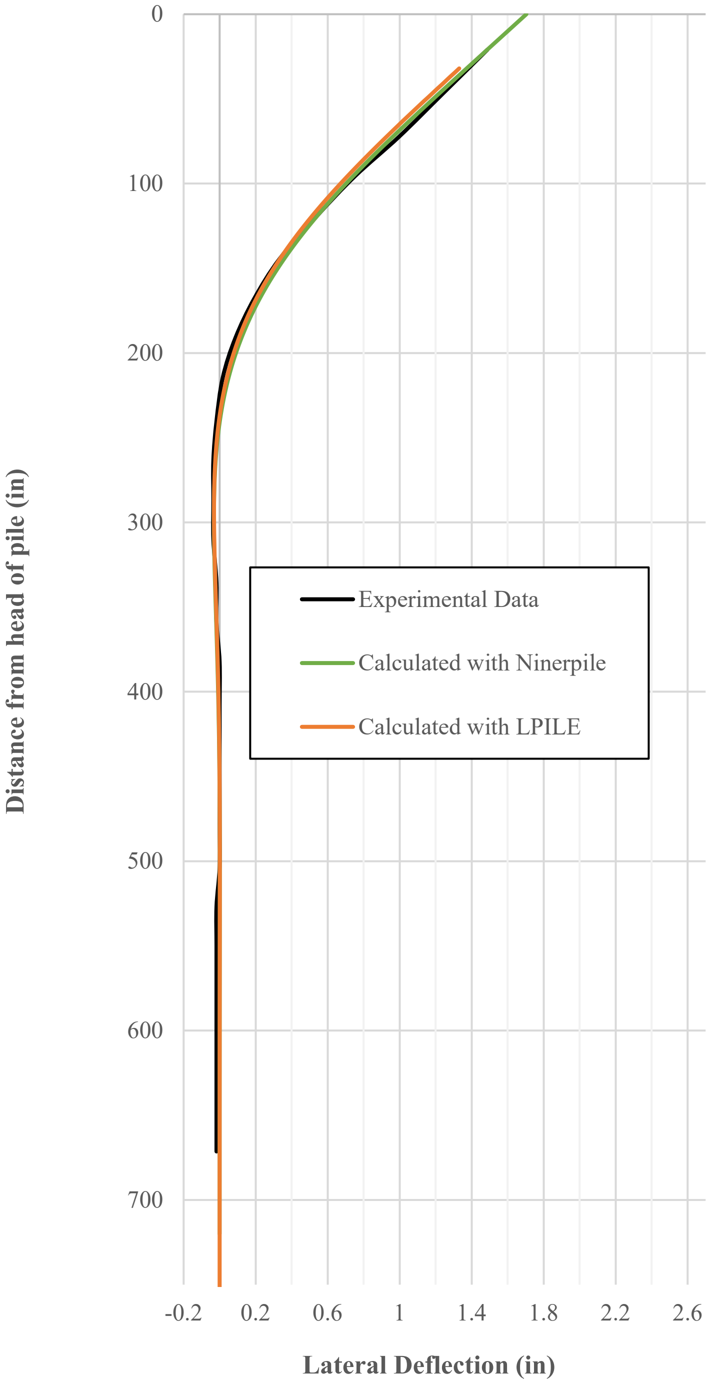

Figure 5.47. FRP pile lateral deflection at 41,835 lb. lateral loading

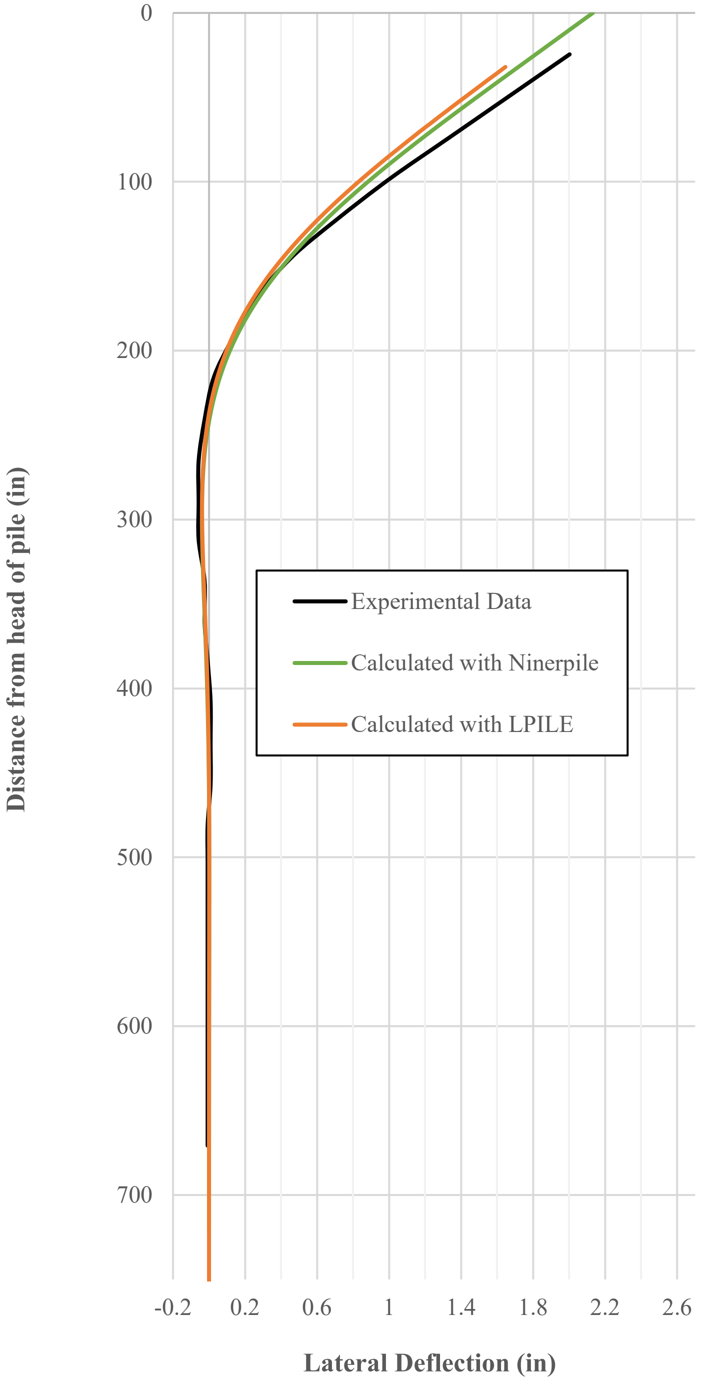

Figure 5.48. FRP pile lateral deflection at 51,726 lb. lateral loading

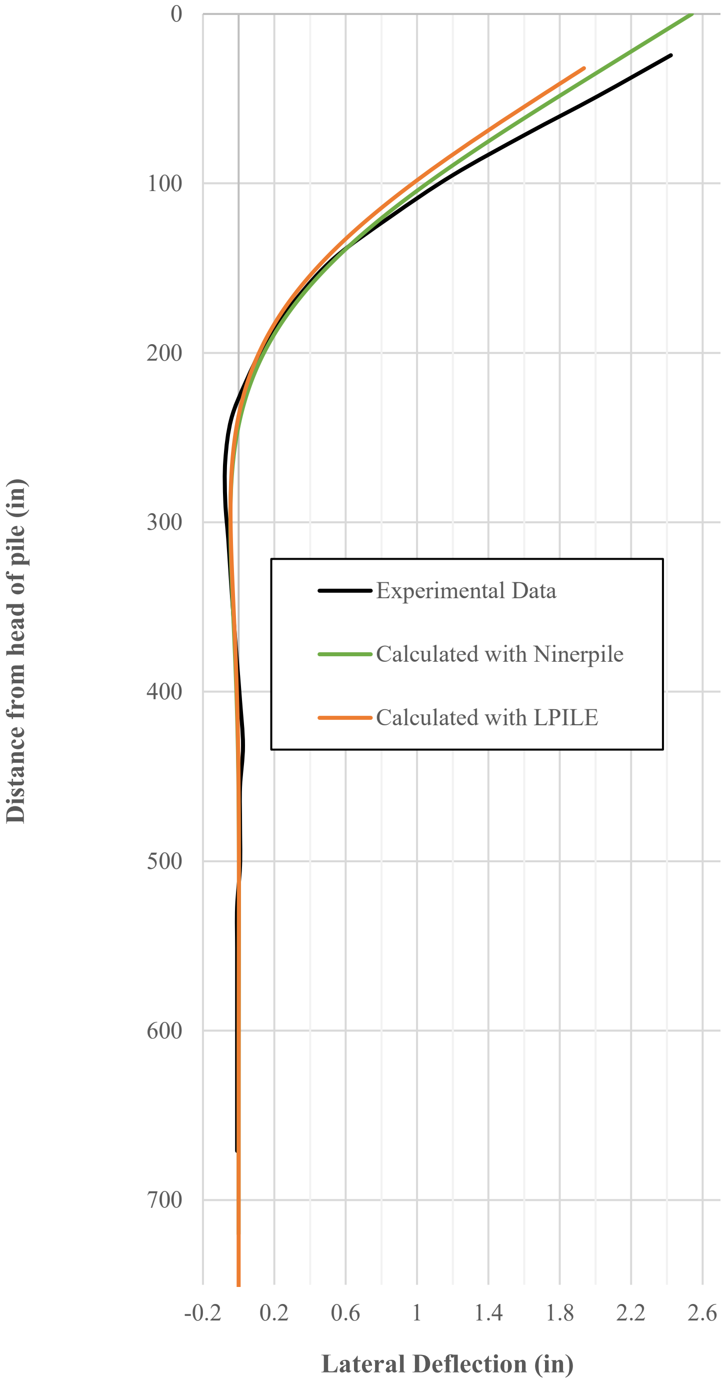

Figure 5.49. FRP pile lateral deflection at 60,808 lb. lateral loading

5.4.3.2 Slope at pile head vs. positive load step

Both the LPILE and NinerPile models predicted the experimental slope at the pile head well. Since NinerPile considers the change in flexural rigidity with internal moment, the slope is more closely predicted at higher loads with NinerPile. Figure 5.50 displays this data.

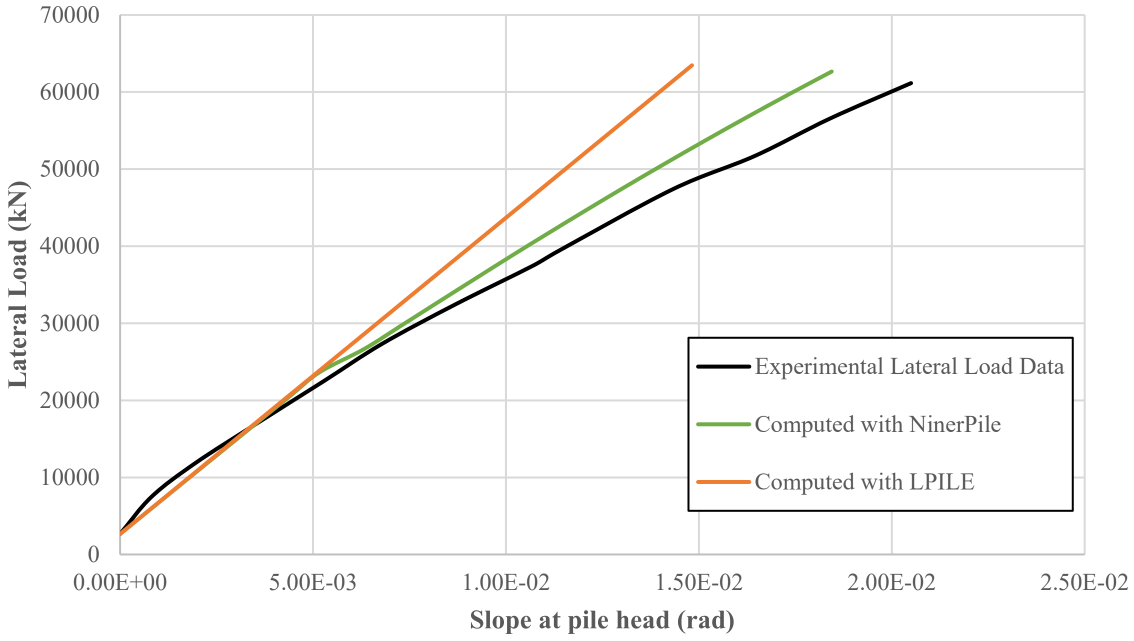

Figure 5.50. Slope at pile head vs. lateral load for FRP pile

5.4.3.3 Pile head displacement vs. load step for loading and unloading

The initial stiffness increment for unloading used is 2 and the stiffness rate is degraded by 90% for each load step. These parameters are discussed in Chapter 4 and the previous sections in this chapter. Similarly, to the plastic pile, the incremented degradation was used until ½ of the unloading was completed, and then a uniform degradation was used. This combination produced an unloading loop that most represents the experimental data.

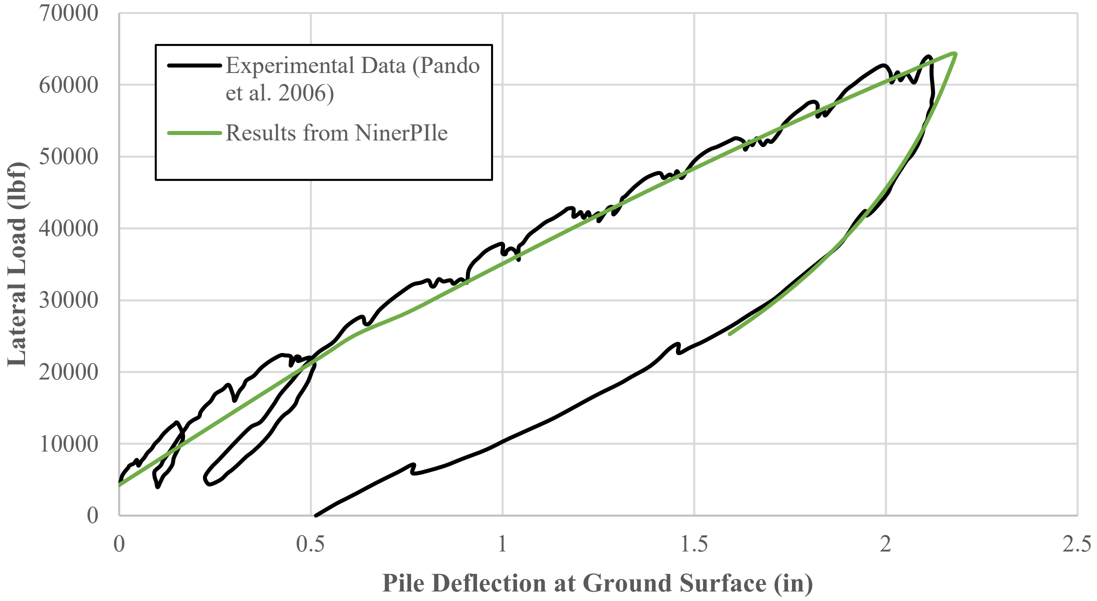

Figure 5.51. NinerPile prediction and experimental data for the FRP pile deflection

5.4.3.4 Spaghetti output plot from NinerPile

The below plot is the output from NinerPile displaying the displacement, slope, soil reaction, bending moment and shear force with depth along the pile for each step of the analysis.

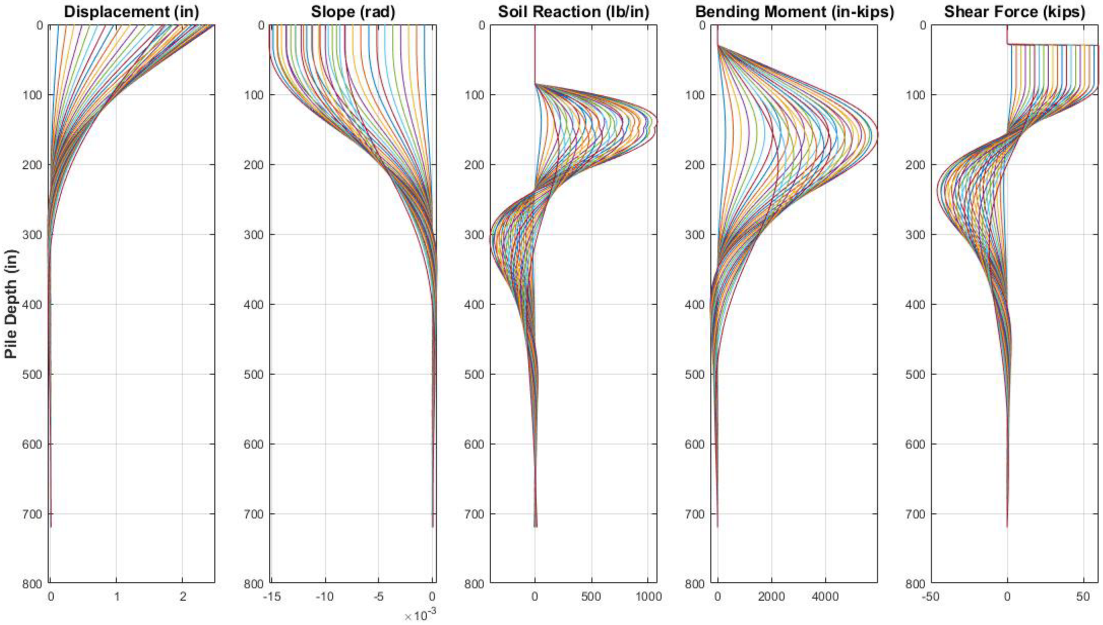

Figure 5.52. Spaghetti plot of NinerPile output for FRP pile

CHAPTER 6: CONCLUSIONS AND RECOMMENDATIONS

6.1 Summary

This thesis reports the work undertaken to develop the program NinerPile. NinerPile's purpose was to address the knowledge gap of modeling the behavior of laterally loaded piles subject to loading and unloading. The main objectives of this study are as follows:

- Derive the numerical methods necessary to implement a solution to the laterally loaded pile problem using computer code.
- Evaluate the state of the art of p-y curves and implement them to model soil behavior in NinerPile.
- Validate NinerPile's capability to model loading behavior with the commercially available software LPILE.
- Extend the current abilities of commercially available software by developing a method for modeling the unloading behavior of a laterally loaded pile.
- Use NinerPile to model 3 lateral load tests that were conducted by the Federal Highway Association (Pando et al. 2006).
- Make recommendations on how to improve the unloading model developed and expand the NinerPile software.

The following are findings and accomplishments of this thesis:

- NinerPile has been developed and is able to match the capability of commercially available software in modeling the loading phase.
- The secant stiffness degradation model has been created and proposed and has been shown that it is able to capture the beginning of the unloading behavior of laterally loaded piles. Using secant stiffness has been shown as a possible route to model

the stiffness behavior of piles subject to lateral loading and unloading by incrementing the secant stiffness step size.

- A path forward has been clearly defined to effectively model the permanent pile deflection sustained after lateral loading and unloading.

6.2 Conclusions

The method of modeling the unloading behavior of laterally loaded piles developed for this thesis is an explicit method. As the secant stiffness is degraded its value becomes fixed. This means that there is no iterating between the finite element code and the p-y curve formulation as is done for loading. Therefore, the method cannot be described by a so-called unloading p-y curve. Instead, the method can be implemented, and the results can be plotted in the p-y plane for visualization. Note that this visualization is an output of the method, not a defining input. If the results of the concrete verification are plotted to create equivalent p-y curves, the shapes shown in Figure 6.1. are created for various depths.

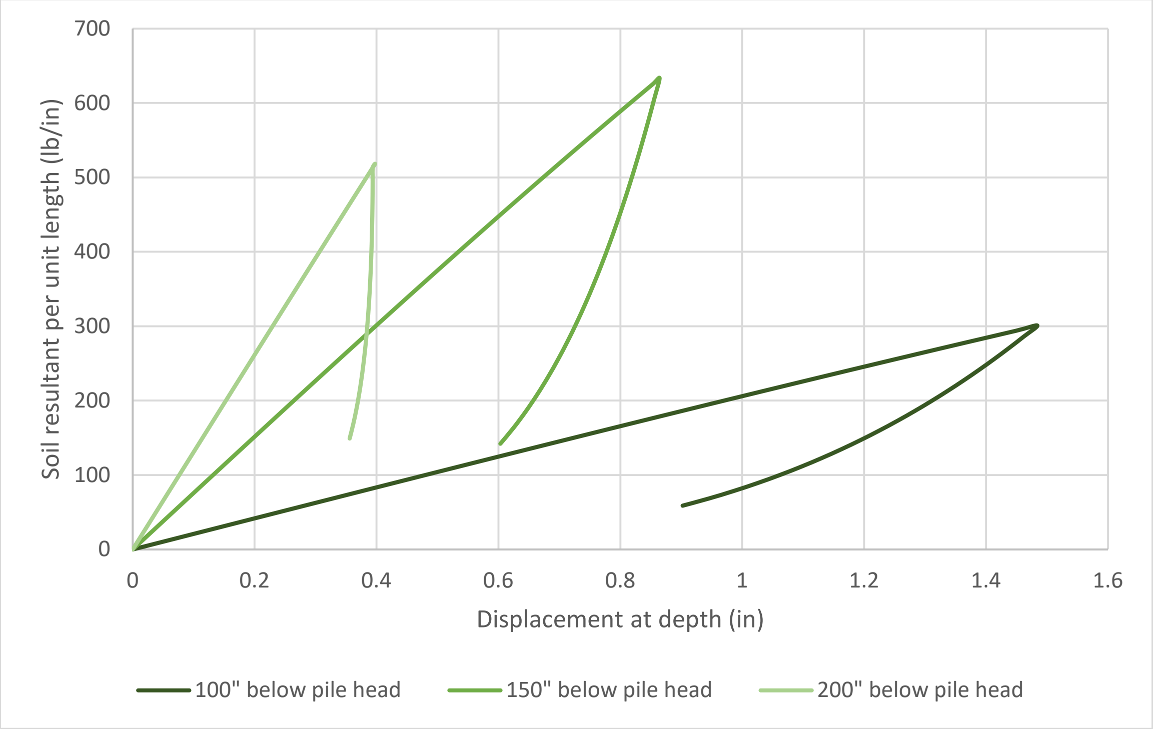

Figure 6.1. Secant stiffness degradation model equivalent p-y curves

Figure 6.1 is from the prestressed concrete pile validation used in chapter 5. As shown, the equivalent p-y curves do not extend to the displacement axis, demonstrating a limitation of this method.

A method has been developed to begin to model the behavior of unloading phase of laterally loaded piles. The method has shown the ability of be applied to different pile types that exhibit completely different unloading behavior.

6.3 Recommendations for future work

The following are recommendations for future work in this area of study:

- Adding more loading phase p-y formulations to NinerPile.
- Explore modeling soil reaction using plasticity theory.
- Investigation of the actual distribution of soil reaction versus depth for the completely unloaded case. It may be non-zero as the numerical methods imply.
- A method to extend the unloading model to zero lateral load using an interpolative method is suggested, as the method developed for this thesis has been shown to predict lateral deflection down to about half of the positive loading value.
- A reloading parameter be programmed into NinerPile that allows a non-linear increase in secant stiffness to capture the unloading and reloading shown in the Federal Highway Report (Pando et al. 2006) lateral load test data.
- A graphic user interface (GUI) is recommended to be developed for easier user interaction with NinerPile.

REFERENCES

Abadie, C. (2015). "Cyclic Lateral Loading of Monopile Foundations in Cohesionless Soils." Doctoral dissertation, University of Oxford, England.

Andersen, K. H. (2015). “Frontiers in Offshore Geotechnics III.” *Cyclic soil parameters for offshore foundation design*, Taylor & Francis Group, London, 3–88.

API (2000). American Petroleum Institute. *Recommended Practice for Planning, Design and Constructing Fixed Offshore Platform,* API Recommended Practice 2A-WSD (RP2A-WSD) 21st edition, Dallas, TX.

API (2011). American Petroleum Institute. *Recommended Practice for Planning, Design and Constructing Fixed Offshore Platform,* API Recommended Practice 2GEO/ISO 19901-4, Geotechnical and Foundation Design Considerations, 1st Edition.

Arroyo, M., & Abadias, D. (2015). Alternative p-y curve formulations for offshore wind turbines in clays. *Third International Symposium on Frontiers in Offshore Geotechnics* (pp. 693-699). Leiden: CRC Press. doi:10.1201/b18442-105

Byrne, B. W. and Houlsby, G. T. (2003), “Foundations for offshore wind turbines”, *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* 361(1813), 2909–2930.

Byrne, B. W. and Houlsby, G. T. (2015), ‘Helical piles: an innovative foundation design option for offshore wind turbines’, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 373(20140081).

Choi, J. I., Kim, M. M., and Brandenberg, S. J. (2015). “Cyclic p-y Plasticity Model Applied to Pile Foundations in Sand.” *Journal of Geotechnical and Geoenvironmental Engineering*, 141(5).

DNV (2005). Det Norske Veritas. *Guidelines on Design and Operation of Wave Energy Converters,* Marine Energy Challenge, Carbon trust.

Duncan, J. M., and Chang, C. Y. (1970). "Nonlinear Analysis of Stress and Strain in Soils", *Journal of the Soil Mechanics and Foundations Division,* ASCE, 96(SM5), 1629-1653.

EMEC (2009). European Marine Energy Center Limited. *Guidelines for Design Basis of Marine Energy Conversion Systems,* London, UK.

“FB-MultiPier.” (2018). program documentation, Bridge Software Institute.

Gazioglu, S. M., and O'Neill, M. W. Evaluation of p-y Relationships in Cohesive Soils. Proc., Analysis and Design of Pile Foundations, ASCE National Convention, San Francisco, Calif., 1984, pp. 192-213.

Hanssen, S. B. (2015). “Small strain overlay to the API p-y curves for sand.” *Third International Symposium on Frontiers in Offshore Geotechnics* (pp. 557-562). Leiden: CRC Press. doi:10.1201/b18442-71

Hetenyi, M. (1946) Beams on Elastic Foundations. University of Michigan Press, Ann Arbor.

Heidari, M., Naggar, H. E., Jahanandish, M., and Ghahramani, A. (2014). “Generalized cyclic p–y curve modeling for analysis of laterally loaded piles.” *Soil Dynamics and Earthquake Engineering*, 63, 138–149.

Hughes, T. J. R. (2000). *The finite element method: linear static and dynamic finite element analysis*. Dover Publications, Mineola, NY.

Johnson, R.M., Parsons, R. L., Dapp S., And Brown D. (2006). Soil Characterization And P-Y Curve Development for Loess. Report. Kansas Department of Transportation. Topeka, Kansas.

Kirsch, F., Richter, T., Coronel, M. (2014). “Geotechnische Aspekte bei der Gründungsbemessung von Offshore-Windenergieanlagen auf Monopfählen mit sehr großen Durchmessern”. Stahlbau Spezial 2014 – Erneuerbare Energien, pp. 61-67 (in German).

Kramer, S. L. (1988). “Development of P-Y Curves for Analysis of Laterally Loaded Piles in Western Washington” (Tech. No. WA-RD 153.1) *Seattle, WA: Washington State Department of Transportation*.

Le blanc, C. (2009). "Design of Offshore Wind Turbine Support Structures." Doctoral dissertation, Aalborg University, Denmark.

Liang, F., Chen, H., and Jia, Y. (2018). “Quasi-static p - y hysteresis loop for cyclic lateral response of pile foundations in offshore platforms.” *Ocean Engineering*, 148, 62–74.

Lin, S.-S., Lai, C.-H., Chen, C.-H., and Ueng, T.-S. (2010). “Derivation of Cyclic p-y Curves From Instrumented Dynamic Lateral Load Tests.” *Journal of Mechanics*, 26(02), 123–133.

Little, R. L., and Briaud, J. L. (1988). "Full Scale Cyclic Lateral Load Tests on Six Single Piles in Sand." *Miscellaneous Paper GL-88-27, Geotechnical Div.* Texas A&M Univ., College Station, Tex.

Long, J., and Vanneste, G. (1994). "Effects of Cyclic Lateral Loads on Piles in Sand." *J. Geotechnical Engineering,* 120(1), 225-244.

"LPILE." (n.d.). *Ensoft Inc*, <https://www.ensoftinc.com/products/lpile/> (Oct. 18, 2018).

Matlock, H. (1970). Correlation for Design of Laterally Loaded Piles in Soft Clay. *Offshore Technology Conference*. doi:10.4043/1204-MS

McClelland, B., and Focht, J. A. (1956). "Soil Modulus for Laterally Loaded Piles." *Journal of the Soil Mechanics and Foundations Division*, 82(4), 1–22.

McVay, M. C., and Niraula, L. (2004). "Development of P-Y Curves for Large Diameter Piles/Drilled Shafts in Limestone for FBPIER. Development of P-Y Curves for Large Diameter Piles/Drilled Shafts in Limestone for FBPIER", rep.

Meyer, J. R. (1984). "Analysis and design of pile foundations." *Proceedings of a symposium sponsored by the ASCE Geotechnical Engineering Division,* San Francisco, California, October 1-5, 1984.

Mokwa, R. L., Duncan, J. M., and Helmers, M. J. (2000). "Development of p-y Curves for Partly Saturated Silts and Clays." *New Technological and Design Developments in Deep Foundations*.

Murchison, J. M., and O'Neill, M. W. (1984). "Analysis and Design of Pile Foundations." *Evaluation of p-y Relationships in Cohesionless Soils*, San Francisco, CA, 174–191.

Nyman, K.J. (1980) "Fields load tests of instrumented drilled shafts in coral limestone". MS Thesis, University of Texas, Austin.

Peng, J. R., Clarke, B. G., and Rouinia, M. (2006). "A Device to Cyclic Lateral Loaded Model Piles." *Geotechnical Testing Journal (GTJ),* ASTM, 29(4), 1-7.

Rajashree, S., & Sundaravadivelu, R. (1996). Degradation model for one-way cyclic lateral load on piles in soft clay. *Computers and Geotechnics, 19*(4), 289-300.

Randolph, M., and Gourvenec, S. (2011). "Offshore Geotechnical Engineering", Spon Press, Abingdon.

Reese, L. C. "Analysis of Laterally Loaded Piles in Weak Rock," Journal of Geotechnical and Geoenvironmental Engineering, ASCE, Vol. 123, No. 11, November 1997, pp. 1010-1017.

Reese, L. C., Cox, W.R., and Koop, F. D. (1974). "Analysis of Laterally Loaded Piles in Sand." *Offshore Technology Conference,* Paper OTC 2080, Houston, TX, 473-483.

Reese, L. C., Cox, W.R., and Koop, F. D. (1975). "Field Testing and Analysis of Laterally Loaded Piles in Stiff Clay." Proceedings, Offshore Technology Conference, Paper 2312, May, 1975, Vol. II, pp. 672-690.

Reese, L. C., and Meyer, B. J. (1979). "Analysis of Single Piles Under Lateral Loading" (Tech. Research Report 244-1) *Austin, TX: Center for Highway Research*, The University of Texas at Austin.

Reese, L. C., and Impe, W. V. (2011). *Single piles and pile groups under lateral loading*. CRC Press/Balkema, Leiden.

Reese, L. C., and Wang, S.-T. (2018). "LPILE." program documentation, Ensoft Inc.

Reese, L. C., and Welch, R. C. (1975). "Lateral Loading of Deep Foundations in Stiff Clay." Journal of the Geotechnical Engineering Division, Proceedings, American Society of Civil Engineers, Paper No. 11456, Vol. 101, No. GT7, February, 1975, pp. 633-649.

Rollins, K. M., Hales, L. J., Ashford, S. A., and Camp, I. W. M. (2005). "P-Y Curves for Large Diameter Shafts in Liquefied Sand from Blast Liquefaction Tests." *Seismic Performance and Simulation of Pile Foundations in Liquefied and Laterally Spreading Ground*.

Schneider, J. A., and Senders, M. (2010). "Foundation Design: A Comparison of Oil and Gas Platforms with Offshore Wind Turbines " *Marine Technology Society Journal* 44(1), 32-5l.

Simpson, M., and Brown, D. A. (2003). "Development of P-Y Curves for Piedmont Residual Soils". Highway Research Center, Auburn University. rep.

Thieken, K., Achmus, M., & Lemke, K. (2015). Evaluation of a new p-y approach for piles in sand with arbitrary dimensions . In *Third International Symposium on Frontiers in Offshore Geotechnics* (pp. 741-746). Leiden: CRC Press. doi:10.1201/b18442-102

Wiemann J. Bemessungsverfahren (2007) "für horizontal belastete Pfähle, Untersuchungen zur Anwendbarkeit der p–y Methode." Doctoral dissertation, University of Duisburg-Essen, Germany.

Yan, L., and Byrne, P. M. (1992). "Lateral pile response to monotonic pile head loading." *Canadian Geotechnical Journal*, 29, 955–970.

Yang, M., Luo, R., and Li, W. (2017). "Numerical study on accumulated deformation of laterally loaded monopiles used by offshore wind turbine." *Bulletin of Engineering Geology and the Environment*, 77(3), 911–921.

APPENDIX I: NINERPILE WRAPPER FILE

```
% wrapper file for input deck
% this is the file that the user modifies.
clc; clear all; close all;
%Pile Geometric and Material Properties
Pile.EI = @(x);     % Flexural Rigidity (kip-in^2)
Pile.L =;                 % Length of the Pile (in)
Pile.D=;                    % Dimension parallel to bending axis (in)
Pile.mcrack=;           %Cracking moment
Pile.EIcrack=@(m);
%function that relates EI (kip-in^2) to moment (kip-in)
%P-Y Curve Inputs
PY.Depth=;        %Distance from pile head to soil surface (in);
PY.phi=;           %Effective friction angle (degrees)
PY.gamma=;        %Effective unit weight (lb/ft^3)
PY.k=;            %Average depth stiffness factor (lb/in^3)
PY.tol=;    %tolerance of p-y iteration convergence
PY.UnlModel=1; %1 is secant stiffness degradation model, 2 is linear with the slope of
E_pymax
% Load Time History
%%% Enter Load Steps Here
%%%% Make sure that Shear and Moment have the same number of inputs!
Loading.type=1; %keep as 1 for explicit loading inputs;
Loading.stepsize=2000;
Loading.Shear =[... %% Pounds %%
0
;
   ];
Loading.Moment=[... %% inch-pounds %% the format is minval:step:maxval
    zeros(length(Loading.Shear),1)
    ];
%%%Specify the node at which loading should be applied (1 if at top of pile)
Loading.node=;
%mesh-specific input data
femesh.nel = ;       % no. of elements (recommended 1 per inch)
femesh.bias =;       % bias of the elements
%output parameters
plot.nd=; %node number of progression plot
%specify load step for displacement, moment, slope and soil reaction plots
plot.step=;
%Units
units.force = 'lb';
units.length = 'in';
units.time = 's';
driver=; %This line if for prototyping, leave as 1 for normal use
%pass the above user input data to the driver function which does the rest.
if driver==2
pilefem_driver_proto(Pile,femesh,PY,Loading,plot);
else
pilefem_driver_multistep(Pile,femesh,PY,Loading,plot);
en
```

APPENDIX II: NINERPILE DRIVER FUNCTION

```
function pilefem_driver_multistep(Pile,femesh,PY,Loading,plot)
%% the main driver function for the one-dimensional finite element code.
% Pile = a structure array that contains Pile geometry data
% femesh = a structure array that contains mesh-related data for the Pile
% units = a structure array holding the units used
% xc = x coordinates of the nodes
% EIc, is flexureal regidity
%% Initializing inputs and preallocating variables
femesh.nen=2;
nnp = femesh.nel+1;    % Total number of nodes
neq = 2*(nnp);         % No. of equations, size of K
[xc,he] = getx(femesh.nen,femesh.nel,nnp,femesh.bias,Pile.L); % Determines coordinates of
nodes
EIc = getEI(xc,Pile.EI); %Determines young's modulus at nodes
EIc=EIc*1000; %convert to pounds
EIo=EIc;
A=getA(xc,PY.Depth,Pile.D,femesh.nel);
Pu=getPu(xc,PY.phi,PY.gamma,Pile.D,PY.Depth); %Determines ultimate soil capacity at nodes
Es=ones(femesh.nel+1,1)*2000; % Initializes the initial secant stiffness for the p-y
curves
Eslast=zeros(femesh.nel+1,1);
F=zeros(neq); % initializes F vector
yn=ones(neq/2);
Eunl=getEunl(xc,PY.k,Pu,femesh.nel);
adj=1;
iter=0;
Fmax=0;
count=0;
%%% Sorting load inputs
if size(Loading.Shear,2)>1 % If the vector is entered with the wrong dimensions
    Loading.Shear=transpose(Loading.Shear); % fix the dimensions
end
Pileload(:,1)=Loading.Shear; % initialize variable to be used
Pileload(:,2)=Loading.Moment;   % initialize variable to be used
s=size(Pileload); %s is used for indexing
% preallocation of output variables
displacement=zeros(femesh.nel+1,s(1)); % displacement is preallocated for the size of
nodes x loading steps
slope=zeros(femesh.nel+1,s(1)-1); % slope is preallocated for the size of nodes x (load
steps - 1)
moment=zeros(femesh.nel+1,s(1)-1);%""
shear=zeros(femesh.nel+1,s(1)-1);%""
soil=zeros(femesh.nel+1,s(1)-1);%""
%% Begin Main Algorithm
for jj=2:s(1) %% Overall Loop for Load History
count=0; % count can be used for counting iterations for debugging
dif=1; % initializing dif, which is used for convergence
tol=double(PY.tol); % initializing tol from the tolerance in the wrapper

%% Unloading logical test
if Pileload(jj,1)<=Pileload(jj-1,1) % if the load step is less than...
    % the last, we need to use the unloading algorithm
    sr=0.92;
    if PY.UnlModel==1
    if iter==0
    numinc=Fmax/Loading.stepsize;
    Einc=(Estart./(numinc))%*2;
    Eincsave=(Estart./(numinc));
   %elseif iter < (numinc/3)
      % Einc=Einc.*sr;
        %sr=sr-.05;
    else
        Einc=Eincsave;
    end
    for ii=1:femesh.nel+1
        if iter==0
    Es(ii)=Estart(ii);
    Eslast(ii)=Es(ii);
        else
    Es(ii)=Eslast(ii)-Einc(ii);
    Eslast(ii)=Es(ii);
        end
    end
    iter=iter+1;
    else
    plast=soil(:,jj-adj);
    adj=adj+2;
    ydguess=guessyd(pstart,plast,Eunl,ystart,femesh.nel);
    Es=guessEs(plast, ydguess,femesh.nel);
    end
%% Begin unloading loop
while dif>tol % loop for convergence of soil reaction with p-y curves
K = getK(femesh.nel,neq,EIc,he,Es); % forming the stiffness matrix..
% based on the last secant stiffness calculated in the loop
F(Loading.node*2-1)=Pileload(jj,1); % forming the force vector from applied loads
F(Loading.node*2)=Pileload(jj,2);% forming the force vector from applied loads
y=gety(K,F); % solving for the displacements and rotations
y=y(:,1); % trimming the y matrix into a vector
yd=y(1:2:end,1); % isolating the displacements from the y vector
% calculating the py-described soil reaction from the calculated y vector
if PY.UnlModel==2
[pc]  = pyunllin(Pu,femesh.nel,yd,ystart,pstart,Eunl);
Es=pc./yd;
else
pc=Es.*yd;
end
if PY.UnlModel==1
    dif=0;
else
dum1=sum(yd.^2);
dum2=sum(yn.^2);
dif=abs(dum1-dum2);
end
yn=yd;
reiter=0;
Elast=Es;
end %end unload loop
%% Loading and reloading logical test
else
count=count+1;
if Pileload(jj,1)<Fmax

    for ii=1:femesh.nel+1
    Es(ii)=Elast(ii)+Einc(ii)*(reiter+1);
    end
    reiter=reiter+1;
%% Begin unloading loop
while dif>tol % loop for convergence of soil reaction with p-y curves
K = getK(femesh.nel,neq,EIc,he,Es); % forming the stiffness matrix..
% based on the last secant stiffness calculated in the loop
F(Loading.node*2-1)=Pileload(jj,1); % forming the force vector from applied loads
F(Loading.node*2)=Pileload(jj,2);% forming the force vector from applied loads
y=gety(K,F); % solving for the displacements and rotations
y=y(:,1); % trimming the y matrix into a vector
yd=y(1:2:end,1); % isolating the displacements from the y vector
% calculating the py-described soil reaction from the calculated y vector
pc=Es.*yd;
if PY.UnlModel==1
    dif=0;
else
dum1=sum(yd.^2);
dum2=sum(yn.^2);
dif=abs(dum1-dum2);
end
yn=yd;
end %end unload loop
end
iter=0;
```

```
%% Begin loading loop
while dif>tol % beginning of increased loading loop
K = getK(femesh.nel,neq,EIc,he,Es); % forming the stiffness matrix..
% based on the last secant stiffness calculated in the loop
F(Loading.node*2-1)=Pileload(jj,1); % forming the force vector from applied loads
F(Loading.node*2)=Pileload(jj,2); % forming the force vector from applied loads
y=gety(K,F); % solving for the displacements and rotations
y=y(:,1); % trimming the y matrix into a vector
yd=y(1:2:end,1); % removing the displacements from the y vector
pc  = py(Pu,PY.k,A,femesh.nel,xc,yd,Pile.D,PY.Depth);% calculating the py-described...
% soil reaction from the calculated y vector
for qq=1:length(pc) %indexing for all values in pc vector
    if xc(qq)<PY.Depth %if the node is located above the ground
        pc(qq)=0; %there is no soil reaction
    end % end if statement
end % end for statement
Esn=pc./yd; % calculating the new secant stiffness value for use in
%stiffness calculations
Es=Esn; %set the secant stiffness to the new value
dum1=sum(yd.^2); %defining the first of two dummy variables used for convergence
dum2=sum(yn.^2); %defining the second of two dummy variables used for convergence
dif=abs(dum1-dum2);% calculating the absolute difference between dummies
yn=yd; %setting yn as the last values of the displacements for convergence purposes
ystart=yd; %setting ystart to the last calculated value of displacement..
%this will be used to determine the unloading line
pstart=pc;%setting pstart to the last calculated value of soil resistance..
%this will be used to determine the unloading line
Estart=Es;%setting Estart to the last calculated value of secant stiffness
Fmax=Pileload(jj,1);
count=count+1;
if count==100
    disp('The soil maximum capacity my be exceeded')
    break
end
iter=0;
end % end of while loop
end% end unload logical test if statement
theta=y(2:2:end,1); % organizing rotation values
mc=getm(theta,EIc, xc); % organizing moment values
mc=mc./1000; % changing moment units
for ii=1:length(mc)
    if mc(ii)>Pile.mcrack
        EIold=EIc(ii);
        EIc(ii)=getEIcrack(mc(ii),Pile.EIcrack);
        EIc(ii)=EIc(ii)*1000;
        if EIc(ii)>EIold
            EIc(ii)=EIold;
        end
    end
end
vc=getv(mc,xc); %calculating shear in pile
ptest=ismember(jj-1,plot.step);
%if ptest==1
%plotpile( yd, theta, pc, mc, vc, xc,jj );
%end
if s(1)==1
    break
end % end of if statement for single load case

displacement(:,jj)=yd;
slope(:,jj)=theta;
moment(:,jj)=mc;
shear(:,jj)=vc;
soil(:,jj)=pc;
end % end of load history loop
%% Output
for xx=2:s(1)
 plotpile(displacement(:,xx),slope(:,xx),soil(:,xx), moment(:,xx),shear(:,xx),xc, xx)
end
%%%%% save output data as columns of .mat files %%%
filename='coordinates.mat';
```

```
save(filename,'xc')
filename='displacement.mat';
save(filename,'displacement')
filename='slope.mat';
save(filename,'slope')
filename='moment.mat';
save(filename,'moment')
filename='shear.mat';
save(filename,'shear')
filename='soil.mat';
save(filename,'soil')
plotprog(displacement,slope,moment,soil,plot.nd,Pileload)
end% end of driver funciton
```

APPENDIX III: NINERPILE STIFFNESS MATRIX FUNCTION

```
function K = getK(nel,neq,EIc,he,Es)
% getK calculates the element stiffness matrix components and assembles
% them to form the global stiffness matrix K
% nel = no. of elements
% nen = no. of element nodes
% n = no. of equations/unknowns
% Ec = nodal values of Young's modulus
% Ic = nodal values of second moment of area
% he = vector of element lengths
% nint = no. of integration points
K     = zeros(neq,neq);
k     = zeros(4,4);
for ii=2:nel+1
 %%%%%%%%%%%%%%%%%%%%%%%%% k(1,1) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))...  %Interpolation of EI
        .*(3/2.*z)...                                    %(N1'')
        .*(3/2.*z)...                                    %(N1'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1-3/4.*(1+z).^2+1/4.*(1+z).^3)...                %(N1)
        .*(1-3/4.*(1+z).^2+1/4.*(1+z).^3)...                %(N1)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                %(he/2)
    dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(1,1)=dummy1+dummy2+dummy3+dummy4;        %element stiffness matrix placement
 %%%%%%%%%%%%%%%%%%%% k(1,2) and k(2,1) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
     fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(3/2.*z)...                                    %(N1'')
        .*(1/4.*he(ii-1).*(3.*z-1))...                    %(N2'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1-3/4.*(1+z).^2+1/4.*(1+z).^3)...                %(N1)
        .*(1/8.*he(ii-1).*(1+z).*(1-z).^2)...               %(N2)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
    dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(1,2)=dummy1+dummy2+dummy3+dummy4;
    k(2,1)=k(1,2);                        %element stiffness matrix placement
     %%%%%%%%%%%%%%%%%%%% k(1,3) and k(3,1) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(3/2.*z)...                                    %(N1'')
        .*(-3/2.*z)...                                   %(N3'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1-3/4.*(1+z).^2+1/4.*(1+z).^3)...                %(N1)
        .*(3/4.*(1+z).^2-1/4.*(1+z).^3)...                  %(N3)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
    dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(1,3)=dummy1+dummy2+dummy3+dummy4;
    k(3,1)=k(1,3);
        %%%%%%%%%%%%%%%%%%%% k(1,4) and k(4,1) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(3/2.*z)...                                    %(N1'')
        .*(1/4.*(3.*z+1).*he(ii-1))...                     %(N4'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1-3/4.*(1+z).^2+1/4.*(1+z).^3)...                %(N1)
        .*(1/8.*he(ii-1).*(1+z).^2.*(z-1))...               %(N4)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
   dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(1,4)=dummy1+dummy2+dummy3+dummy4;
    k(4,1)=k(1,4);
        %%%%%%%%%%%%%%%%%%%%%%%%%% k(2,2) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(1/4.*he(ii-1).*(3.*z-1))...                     %(N2'')
        .*(1/4.*he(ii-1).*(3.*z-1))...                     %(N2'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1/8.*he(ii-1).*(1+z).*(1-z).^2)...               %(N2)
        .*(1/8.*he(ii-1).*(1+z).*(1-z).^2)...               %(N2)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
   dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(2,2)=dummy1+dummy2+dummy3+dummy4;
       %%%%%%%%%%%%%%%%%%%%%%%%%%% k(2,3) and k(3,2) %%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(1/4.*he(ii-1).*(3.*z-1))...                     %(N2'')
        .*(-3/2.*z)...                                   %(N3'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1/8.*he(ii-1).*(1+z).*(1-z).^2)...               %(N2)
        .*(3/4.*(1+z).^2-1/4.*(1+z).^3)...                  %(N3)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
   dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(2,3)=dummy1+dummy2+dummy3+dummy4;
    k(3,2)=k(2,3);
         %%%%%%%%%%%%%%%%%%%%%%%%%%% k(2,4) and k(4,2) %%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(1/4.*he(ii-1).*(3.*z-1))...                     %(N2'')
        .*(1/4.*(3.*z+1).*he(ii-1))...                     %(N4'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(1/8.*he(ii-1).*(1+z).*(1-z).^2)...               %(N2)
        .*(1/8.*he(ii-1).*(1+z).^2.*(z-1))...               %(N4)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
   dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(2,4)=dummy1+dummy2+dummy3+dummy4;
    k(4,2)=k(2,4);
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% k(3,3) %%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(-3/2.*z)...                                   %(N3'')
        .*(-3/2.*z)...                                   %(N3'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(3/4.*(1+z).^2-1/4.*(1+z).^3)...                  %(N3)
        .*(3/4.*(1+z).^2-1/4.*(1+z).^3)...                  %(N3)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
   dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(3,3)=dummy1+dummy2+dummy3+dummy4;
        %%%%%%%%%%%%%%%%%%%%%% k(3,4) and k(4,3) %%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(-3/2.*z)...                                   %(N3'')
        .*(1/4.*(3.*z+1).*he(ii-1))...                     %(N4'')
        .*(2/he(ii-1)).^3 ...                            %(2/he).^3
        +(3/4.*(1+z).^2-1/4.*(1+z).^3)...                  %(N3)
        .*(1/8.*he(ii-1).*(1+z).^2.*(z-1))...               %(N4)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
```

```
        .*(he(ii-1)/2));                                 %(he/2)
   dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(3,4)=dummy1+dummy2+dummy3+dummy4;
    k(4,3)=k(3,4);
      %%%%%%%%%%%%%%%%%%%%%%%%%%%% k(4,4)%%%%%%%%%%%%%%%%%%%%%%%%%
    fun=@(z)((EIc(ii-1).*((1-z)/2)+EIc(ii).*((1+z)/2))... %Interpolation of EI
        .*(1/4.*(3.*z+1).*he(ii-1))...                    %(N4'')
        .*(1/4.*(3.*z+1).*he(ii-1))...                    %(N4'')
        .*(2/he(ii-1)).^3 ...                           %(2/he).^3
        +(1/8.*he(ii-1).*(1+z).^2.*(z-1))...              %(N4)
        .*(1/8.*he(ii-1).*(1+z).^2.*(z-1))...              %(N4)
        .*((Es(ii-1).*((1-z)/2)+Es(ii).*((1+z)/2)))...    %Interpolation of Es
        .*(he(ii-1)/2));                                 %(he/2)
    dummy1=(0.652145)*fun(-0.339981); %gaussian 4 point quadrature
    dummy2=(0.652145)*fun(0.339981);
    dummy3=(0.347855)*fun(-0.861136);
    dummy4=(0.347855)*fun(0.861136);
    k(4,4)=dummy1+dummy2+dummy3+dummy4;


   K((ii-1)*2-1:(ii-1)*2+2,(ii-1)*2-1:(ii-1)*2+2)=K((ii-1)*2-1:(ii-1)*2+2,(ii-1)*2-1:(ii-
1)*2+2)+k;

end
```

APPENDIX IV: NINERPILE ULTIMATE SOIL RESULTANT FUNCTION

```
function [Pu] = getPu(xc,phi,gamma,D,depth)
%This function computes the ultimate soil capacity at each node
gamma=gamma/1728; %conversion from lb/ft^3 to lb/in^3
Kp=(tand(45+phi/2)^2);
Ka=(tand(45-phi/2)^2);
Ko=(1-sind(phi));
Pu=zeros(length(xc),1);
for ii=1:length(xc)
    pu1=gamma*(xc(ii)-depth)*(D*(Kp-Ka)+xc(ii)*Kp*tand(phi)*tand(45+phi/2));
    pu2=gamma*D*(xc(ii)-depth)*(Kp^3+2*Ko*Kp^2*tand(phi)+tand(phi)-Ka);
    if pu1<pu2
        Pu(ii)=pu1;
    else
        Pu(ii)=pu2;
    end
    if xc(ii)<=depth
        Pu(ii)=0;
    end
end
end
```

APPENDIX V: NINERPILE POSITIVE LOAD STEP P-Y CURVE FUNCTION

```
function [ pc ] = py( Pu,k,A,nel,xc,yc,d,zstart)
%This function is the definition of the py curve
pc=zeros(nel+1,1);
for ii=1:nel+1
    if Pu(ii)==0
        pc(ii)=0;
    else
pc(ii)=A(ii)*Pu(ii)*tanh((k*(xc(ii)-zstart)/(A(ii)*Pu(ii)))*yc(ii));
    end
end
```