A PREDICTIVE MODELING APPROACH FOR ASSESSING SESIMIC SOIL LIQEFACTION POTENTIAL USING CPT DATA

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Civil and Environmental Engineering

by

Jonathan Paul Schmidt

June 2019

COMMITTEE MEMBERSHIP

TITLE: A Predictive Modeling Approach for Assessing Seismic Soil Liquefaction Potential Using CPT Data

AUTHOR: Jonathan Paul Schmidt

DATE SUBMITTED: June 2019

COMMITTEE CHAIR: Robb Moss, Ph.D., P.E., F. ASCE
Professor of Civil Engineering

COMMITTEE MEMBER: Gregg Fiegel, Ph.D., P.E., GE
Professor of Civil Engineering

COMMITTEE MEMBER: Benjamin Turner, Ph.D., P.E.
Adjunct Lecturer, Consulting Engineer, Dan Brown and Associates

COMMITTEE MEMBER: Kevin Ross, Ph.D.
Associate Professor of Statistics

ABSTRACT

A Predictive Modeling Approach for Assessing Seismic Soil Liquefaction Potential Using CPT Data

Jonathan Paul Schmidt

Soil liquefaction, or loss of strength due to excess pore water pressures generated during dynamic loading, is a main cause of damage during earthquakes. When a soil liquefies (referred to as triggering), it may lose its ability to support overlying structures, deform vertically or laterally, or cause buoyant uplift of buried utilities. Empirical liquefaction models, used to predict liquefaction potential based upon in-situ soil index property measurements and anticipated level of seismic loading, are the standard of practice for assessing liquefaction triggering. However, many current models do not incorporate predictor variable uncertainty or do so in a limited fashion. Additionally, past model creation and validation lacks the same rigor found in predictive modeling in other fields.

This study examines the details of creating and validating an empirical liquefaction model, using the existing worldwide cone penetration test liquefaction database. Our study implements a logistic regression within a Bayesian measurement error framework to incorporate uncertainty in predictor variables and allow for a probabilistic interpretation of model parameters. Our model is built using a hierarchal approach account for intra-event correlation in loading variables and differences in event sample sizes that mirrors the random/mixed effects models used in ground motion prediction equation development. The model is tested using an independent set of case histories from recent New Zealand earthquakes, and performance metrics are reported.

We found that a Bayesian measurement error model considering two predictor variables, $q_{c,1}$ and CSR, decreases model uncertainty while maintaining predictive utility for new data. Two forms of model uncertainty were considered – the spread of probabilities predicted by mean values of regression coefficients (apparent uncertainty) and the standard deviations of the predictive distributions from fully probabilistic inference. Additionally, we found models considering friction ratio as a predictor variable performed worse than the two variable case and will require more data or informative priors to be adequately estimated.

ACKNOWLEDGMENTS

This work would not have been possible without the support of everyone who has been a part of this journey.

First, I would like to thank Dr. Robb Moss for his mentorship both in and out of the classroom. This work would have not been possible without him and his encouragement to pursue graduate studies. I would also like to extend thanks to the rest of my thesis committee: Dr. Gregg Fiegel, Dr. Benjamin Turner, and Dr. Kevin Ross for helping me grow as both a student and an engineer during their classes.

To my friends from the Triathlon Club: thank you for making my defense possibly the best attended one the department has seen in a while. You all have been a wonderful part of my college experience and I cherish the time we have spent together on and off the bike, even if a lot of that time involved listening to me talk about soil mechanics.

And last but not least, to my family: I can never repay you for all your love and support. From as early as I can remember you have always stood by me during trying times and encouraged me to follow my dreams. Thank you.

For this work I received financial support through Cal Poly’s Graduate Assistant position. Additional funding was provided from CalTrans (California Department of Transportation) and NRC (Nuclear Regulatory Commission) via PEER (Pacific Earthquake Engineering Research center) and SWRI (South West Research Institute) as part of the NGL (Next Generation Liquefaction) program. The opinions expressed here are the author's and the author's advisor and are not necessarily those of the funding or research agencies.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

1 INTRODUCTION

1.1 Research Motivation

Seismic soil liquefaction is a major cause of earthquake damage to the built environment, second only to tsunamis in overall cost. Many West coast US metropolitan cities (and plenty others worldwide) are located in regions of high seismicity and have sizeable developments built upon potentially liquefiable soils. Recently, the Canterbury earthquake sequence in New Zealand and the Tōhoku earthquake in Japan (both in 2011) have shown how devastating the consequences of liquefaction can be in an urban environment.

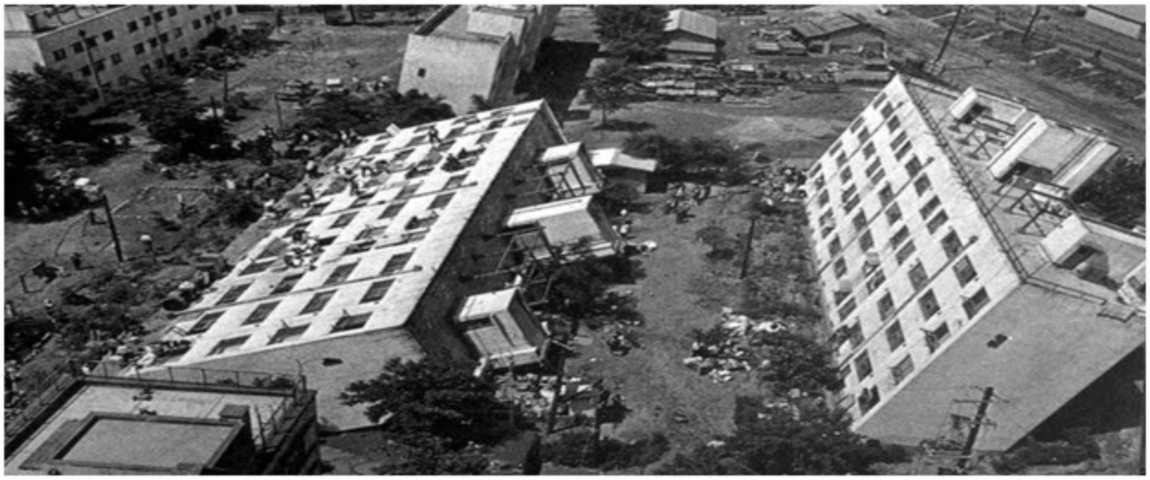


Figure 1 – Damage Resulting from Liquefaction Induced Ground Failure in the 1964 Niigata Earthquake

Broadly speaking, seismic soil liquefaction is when a loose, saturated, granular soil loses strength due to dynamic earthquake loading (NAE, 2016). Although seismic soil liquefaction is often foremost in engineers minds, liquefaction can also occur from blasting or pile driving, groundwater seepage, wave action, and other loading situations.

When shaken, a loose soil will tend to densify – much like pressing your knuckles on your two hands together, then sliding them past one another into the grooves between. However, if this space (the soil pores) is fully saturated with water the soil cannot immediately densify and will instead “squeeze” the pore water. This practically incompressible pore water will then push back on the soil particles and move them apart. This loss of contact (and associated friction) between soil particles manifests itself as a loss of shear strength. If the loads are high enough and applied quicker than the pore water can flow out of the soil the particles will become almost completely separated and the strength will eventually drop to near zero. When this happens (referred to as liquefaction triggering), the soil will behave like a fluid.

Liquefaction damages engineered features in a variety of ways (NAE, 2016):

- Foundations that rely on the strength of the competent soil to support structures can tilt and sink into the ground (undergoing a bearing capacity failure), damaging the structures they support.
- Soil on inclined ground may flow, referred to as lateral spreading, and damage supported structures that cannot tolerate lateral movements.
- The fluidized soil will have a different dynamic response and may amplify seismic waves transmitted to surface structures.
- When pore pressures dissipate soils may densify, causing vertical settlement and potential structural damage.
- Underground structures will become buoyant, causing damage to utility lines or other buried features resulting in significant economic damage despite often not directly threatening life safety.
- Soil and water may erupt at the ground surface (sand boils) causing cracking of pavements and burying objects like vehicles, fire hydrants, or sidewalks. This not only impedes clean-up efforts but can delay critical emergency first responders.

To assess the potential for liquefaction triggering, current practice relies on empirical liquefaction models (ELM’s) (NAE, 2016). These ELM’s are developed by measuring or estimating soil properties and seismic loads at sites of observed liquefaction or nonliquefaction following earthquakes. Modelers then use a variety of statistical methods ranging from simple regressions to complex machine learning techniques to determine the relationship between soil properties/seismic loads and liquefaction potential. These relationships are used to make predictions of potential liquefaction occurrence at future sites during engineering design and analysis. However, as discussed at length by the Committee on State of the Art and Practice in Earthquake Induced Soil Liquefaction Assessment there are significant shortcomings of current ELM’s (NAE, 2016).

Ideally, liquefaction assessment will eventually be conducted in a fully performance based engineering (PBE) approach that evaluates engineered features over the entire range of possible loadings rather than a single or discrete group of seismic events (NAE, 2016). This approach, which is how ground acceleration hazards are currently handled, requires a probabilistic description of liquefaction potential (NAE, 2016). Currently, only two models used in common practice (Moss et al., 2006, and Idriss and Boulanger, 2016) provide this.

Furthermore, a PBE approach requires explicit quantification of the uncertainty associated with all levels of liquefaction assessment, from triggering to soil-structural system response to financial/life losses from failure (NAE, 2016). In this framework, reducing uncertainty associated with the potential for liquefaction triggering will naturally result in a better assessment of liquefaction (NAE, 2016). The principal components of this uncertainty come from imperfect measurements or estimations of input variables (variable uncertainty) and from an imperfect model fit to the training data (model uncertainty). While no model can perfectly capture the triggering relationship, there is potential for choices during the modeling process to reduce model uncertainty while maintaining (or improving) predictive ability. By explicitly incorporating uncertainties associated with predictor variable, a modeler can often reduce overall uncertainty in the final product (NAE, 2016). A primary goal of this thesis is to apply a Bayesian measurement error model to liquefaction triggering.

Another significant limitation of existing ELM's is a lack of openness regarding the model building process. In this process, a modeler must make choices regarding which predictor variables to include, how to pre-process these predictors, what model forms to use, and many more. When a modeler makes these decisions, either manually or as part of pre-programmed algorithm, they will naturally have some idea of what "good model performance" is and make their choices to achieve it. This notion of good performance can be concrete such as some statistical metric, based on engineering judgement and experience, or most often some combination of the two. While necessary to the modeling process, this introduces bias. Because these training methods and metrics are often not reported along with the finished product, practitioners and code writers cannot currently evaluate these model biases when selecting which relationships to use or recommend in guidance documents. Within this context of our own definition of "good model performance" we discuss the impacts of modeling choices and how they can be used in future model development.

Furthermore, when reporting performance metrics most use the same data as was used to build the model to validate it. This will lead to optimistically biased performance metrics and possibly overfit models (Kuhn and Johnson, 2013). To date, only several relevant models (Oomen et al., 2010, Rezania et al., 2011, etc.) split their databases into training and testing sets through cross validation or other methods to develop relatively unbiased metrics of model performance. As will be discussed in the literature, these models unfortunate do not offer a useable product from a practical engineering standpoint. A second goal of this thesis is to show how a model validation strategy that seeks to minimize bias can be justified and implemented.

In closing, although all popular triggering models more or less follow the same "simplified method" framework they differ (often notably) in both their inputs and results. Currently, practitioners must choose between models without clear guidance on their limitations or applicability (NAE, 2016). To address this, the Next-Generation Liquefaction (NGL) project funded by the Pacific Earthquake Engineering Research Center, California Department of Transportation, and Nuclear Regulatory Commission (among others) aims to rebuild consensus and develop new, usefully sophisticated models for liquefaction assessment (Stewart et al., 2019).

The NGL project envisions an "open, collaborative process for model development in which developer teams share ideas and results during model development, so as to reduce the potential for mistakes and to mutually benefit from best practices" (Stewart et al., 2019). This thesis aims to contribute to this ongoing research in the following ways:

- Develop transparent model validation strategies to be used when examining the effects of modeling choices including:
    - Predictor variable selection
    - Predictor variable transformations
    - Treatment of predictor variable uncertainty
    - Mode form and complexity
- Reduce model uncertainty, while maintaining (or improving) predictive capability.
- Present a novel model framework that can be built upon in future efforts.

1.2 Organization of Thesis

This thesis is written with the hope that readers with a basic understanding of probability and statistics and the principles of geotechnical earthquake engineering can understand the work.

Chapter 2, following this introduction, presents relevant background information regarding predictive modeling, soil mechanics and liquefaction assessment, and the progression of ELM's in common engineering practice up to the present day. The first part of Chapter 2 deals with the fundamentals of liquefaction mechanics and the commonly used "Simplified Procedure" method of analysis. The second part summarizes relevant statistical and mathematical concepts useful for understanding the models used in earlier work and the ones developed in this thesis. The final part of Chapter 2 reviews prior probabilistic CPT liquefaction triggering models to provide context and motivation for our present efforts. Chapter 3 details our modeling process, focusing on the choices a researcher must make before arriving at a final product. The effect of these choices on the models' relative performances is presented in Chapter 4, followed by conclusions and recommendations for future research in Chapter 5.

2 BACKGROUND AND LITERATURE REVIEW

This aim of this chapter is to give background knowledge of empirical models developed to predict liquefaction potential and present statistical theory relevant to our model building. Although the overview is written with the intent that readers with an undergraduate understanding of geotechnical engineering and statistics should be able to understand our modeling techniques, further reading of the provided reference will give more depth.

2.1 A Review of Soil Mechanics Relevant to Liquefaction

Soil can be thought of as a three phase material -- in a given volume there will be some solid particles, (usually between 0.001 and 75mm), some liquid (typically water), and some amount of gas (typically air) (Holtz ET AL., 2011). The macroscopic material response of soil is a combination of the complicated interactions of all of these. While it is prohibitively complicated to account for all of the forces from these soil-soil and soil-fluid interactions, the principle of effective stress states that the engineering behavior of soil is governed by the following equation (after Holtz et al., 2011):

$$\sigma_v' = \sigma_v - u$$

Where σ_v is the vertical total stress (an averaged inter-particle contact stress on a horizontal plane), u is the pore water pressure, and σ_v' is the vertical effective stress (visualized in Figure 2, following)

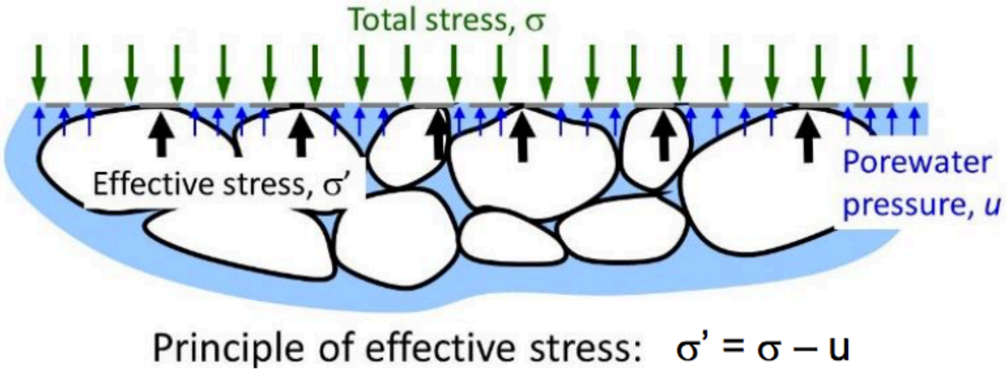


Figure 2 – A Visualization of the Principle of Effective Stress. Reproduced from NAE, 2016

Nearly every relevant engineering property (strength, stiffness, dynamic properties, etc.) is affected to some degree by the current stress state.

Because soil is a particulate material, shear failure occurs when the particles slide or roll past one another (Holtz et al., 2011). Thus, a soil's strength depends primarily on the interactions between the particles and pore water, although particle crushing can occur under very high confining stresses. These interactions are broadly grouped into frictional, resulting from physical contact between particles, and cohesive, the "stickiness" that results from electrostatic attraction, chemical bonding, capillary action, and other small-scale forces (Holtz et al., 2011). The Mohr-Coulomb shear strength equation is the most common model for soil strength soil strength, expressed as (after Holtz et al., 2011):

$$\tau = \sigma' \tan(\phi') + c'$$

where τ is the shear strength along a plane of interest, σ' is the effective stress normal to that plane, ϕ' is the soil's friction angle, an intrinsic, stress independent property which defines its frictional strength, and c’ is the soil’s cohesion, an intrinsic, stress independent property which defines its “stickiness”.

From these two simple equations alone one can understand a great deal about liquefaction behavior. Firstly, as pore water pressure increases effective stress will decrease—leading to a reduction in shear strength. If it increases high enough shear strength will drop to near zero, provided the soil has negligible cohesion typical of many sands. Secondly, soils that have enough intrinsic cohesion, typically from high fines content and plasticity), will not liquefy because of the stress independence of this property. They may undergo a different type of failure called cyclic softening but because it is fundamentally different type of failure than liquefaction we will not discuss it in this work (NAE, 2016).

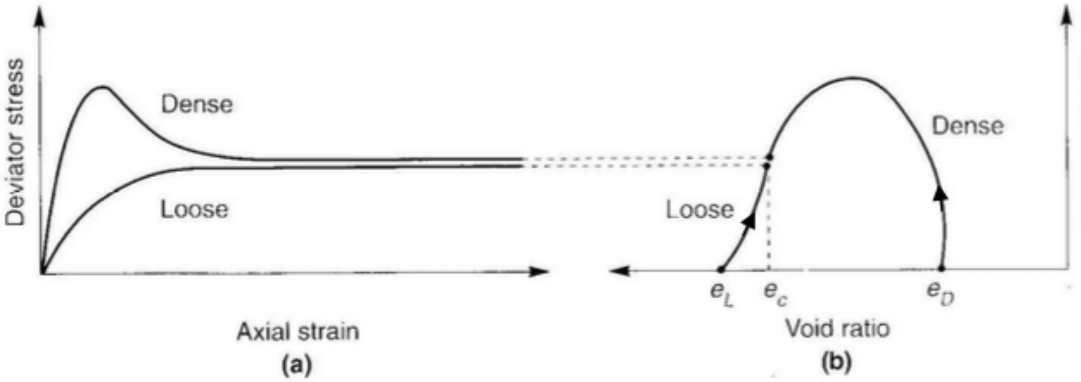


Figure 3 – A Visualization of Idealized Dry Sand Stress-strain Behaviors Under Monotonic Loading with Different Starting Void Ratios. Reproduced from NAE, 2016

Naturally, determining the boundary between predominantly sand-like and predominantly clay-like behavior (i.e. which soils are susceptible to liquefaction) is nontrivial and the subject of ongoing research (NAE, 2016). For the remainder of this section, the discussion will be focused on soils exhibiting sand like behavior.

2.1.1 Soil Behavior Under Dynamic Loading

Under monotonic loading (sheared only in one direction), a dry sand will undergo different volume changes depending on its initial void ratio (Figure 3).

Sands that are initially very loose will densify, called contractive behavior, and sands that are initially dense with become less dense, called dilative behavior. However, even very dense sands will still densify to some degree before dilating. This is because it is highly unlikely they are at their theoretical maximum density and thus have some small amount of room to contract.

Importantly, at very large shear strains both dense and loose sands tend to the same asymptotic strength and void ratio. This void ratio, at which the soil shears with continuous deformation and no change in principal stresses, is termed the critical void ratio and separates contractive and dilative states (Holtz et al., 2011). The critical void ratio (CVR) has been found to be a function of effective stress, and follows a general trend shown below (Figure 4).

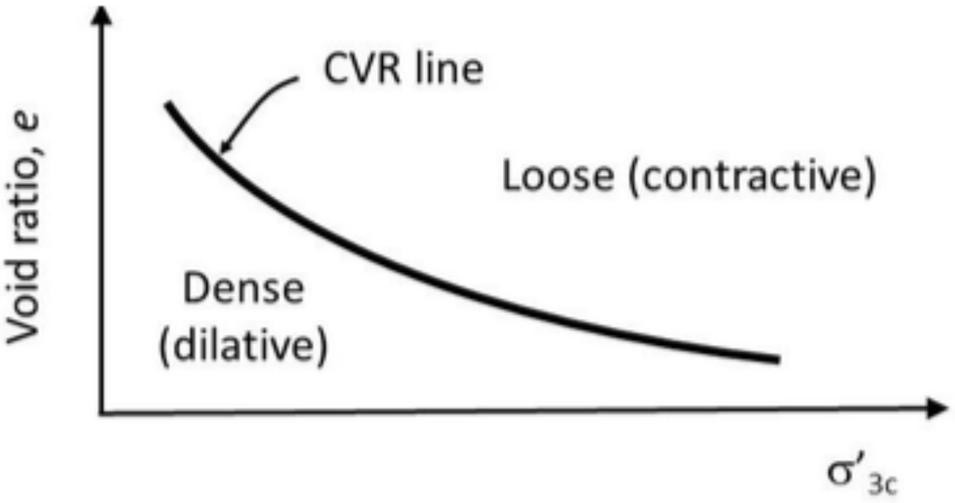


Figure 4 – The Relationship Between Critical Void Ratio and Effective Stress That Separates Contractive and Dilative Soil States. Reproduced from NAE, 2016

Another important distinction made when describing soil strength and deformation behavior is that of drained versus undrained behavior. When drained conditions apply, the loading rate is slow relative to the permeability of the soil. Water can drain from the voids during shear and the soil skeleton is able to change volume leading to no significant changes in pore water pressure. Conversely, when undrained conditions apply the loading rate is fast relative to the permeability of the soil. Water cannot drain from the voids and significant changes in pore water pressure do occur. Returning to the knuckles analogy described in the first chapter, one can see that a tendency for contraction will cause positive pore pressure generation and a tendency for dilation will cause negative (suction) pore pressure generation. The different "paths" a soil takes during loading, either through void ratio change or effective stress change, are shown conceptually in Figure 5, following.

When drained, a loose sand will decrease in void ratio (contract) and a dense sand will increase in void ratio (dilate). However, in undrained loadings these tendencies to volume change will instead cause reduced or increased effective stress (respectively) from pore water pressure generation. The state parameter, ψ, is a measure of how contractive or dilative a soil is.

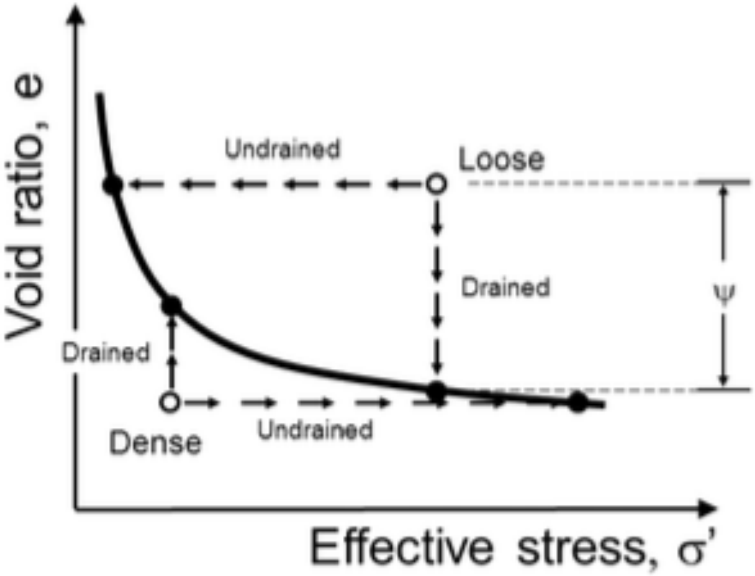


Figure 5 – The Behavior of a Saturated Sand in Both Drained and Undrained Loading.
Reproduced from NAE, 2016

2.1.2 Behavior Under Shear Reversal (Cyclic Loading)

Unlike many typical loading scenarios (surcharge, slopes, etc.) seismic loads are characterized by repeated application of shear stresses that change in direction and intensity as earthquake waves move through the soil profile and are reflected by boundaries. The more complicated behavior is shown and described for a typical liquefiable soil during a cyclic laboratory test in Figure 6, following.

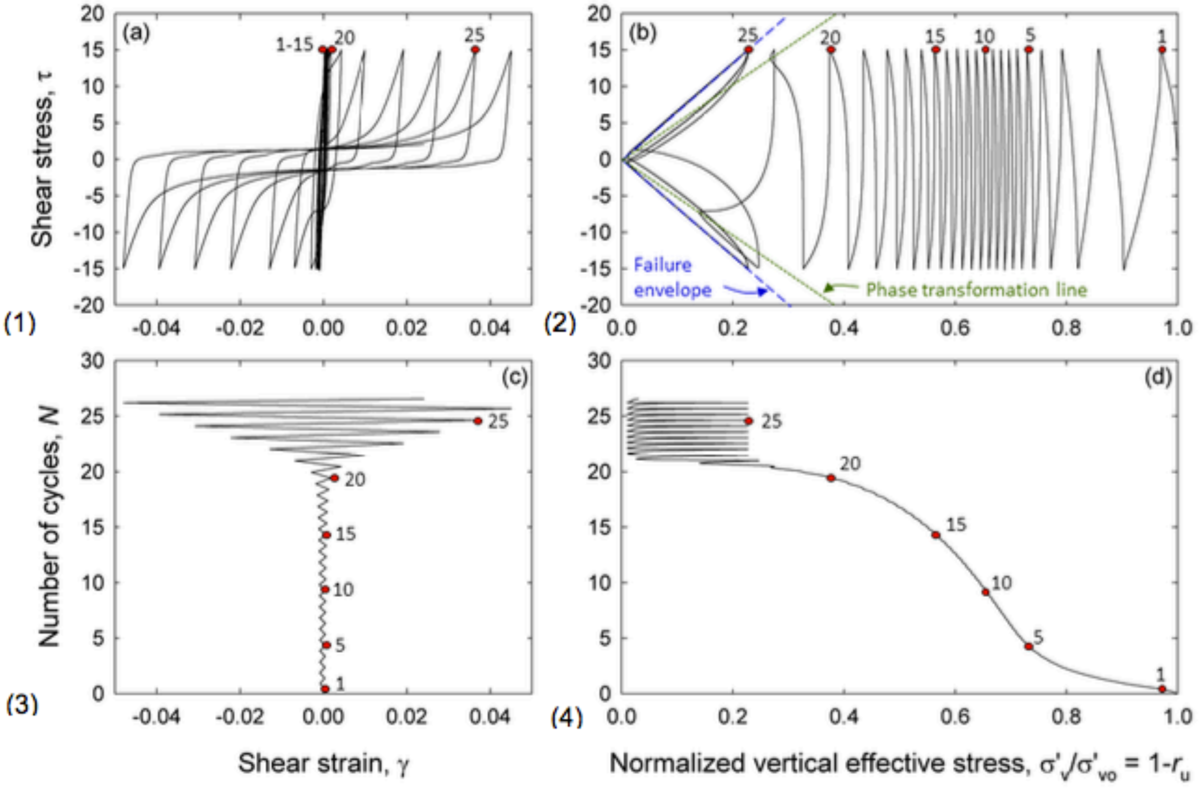


Figure 6 – Cyclic Loading Response of a Typical Loose, Liquefiable Soil. Reproduced from NAE, 2016

(1) The soil begins at an initial stiffness, and softens dramatically as the test progresses (the numbers above plot points indicate cycle number). (2) The soil is initially contractive and effective stress decreases throughout the test as pore pressures rise then becomes dilative when it reaches the phase transformation line (in green). (3) The initial maximum amplitude of shear strain for each cycle is relatively small, then becomes larger and larger as the soil approaches

failure. (4) The effective stress decreases with increasing cycles, and exhibits oscillatory behavior when the soil begins to dilate. Eventually, it reaches near zero and liquefaction is triggered.

2.1.3 Factors That Affect Liquefaction Potential

A wide variety of in-situ soil properties and earthquake loading characteristics can affect the potential for a susceptible soil to liquefy. The simplest of these is saturation, or the percentage of pore space filled with water – most research indicates that a saturation level of near 100% (i.e. below the groundwater table) is required for a soil to liquefy (NAE, 2016). Load magnitude, duration, and soil relative density have a similarly intuitive relationship with liquefaction potential (Figure 7).

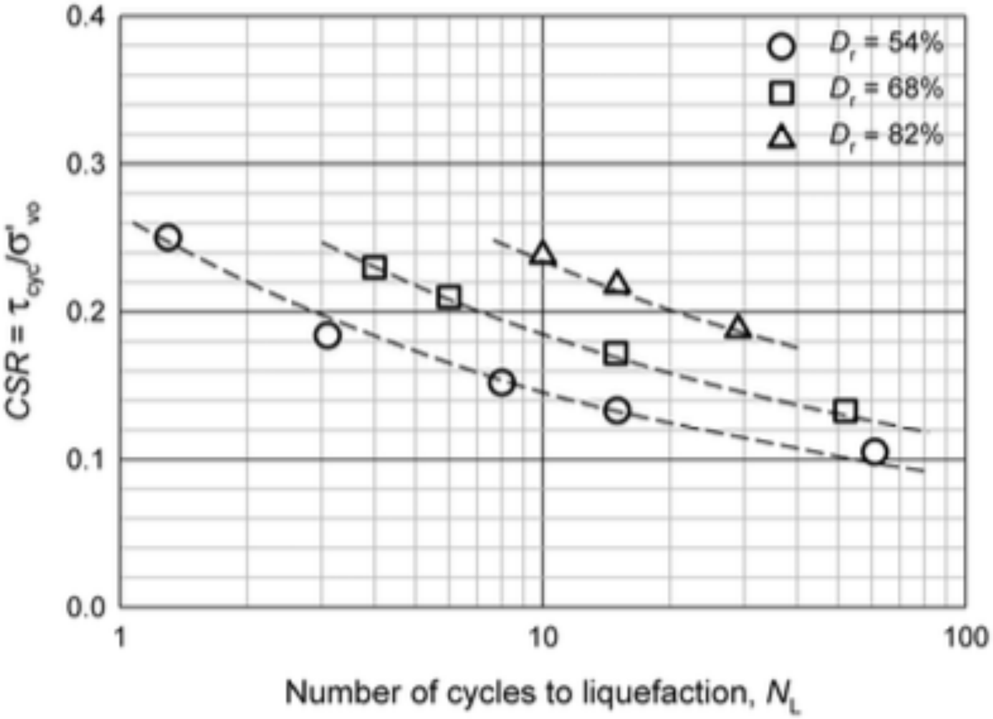


Figure 7 – Relationship Between Load Magnitude, Duration, and Cycles to Liquefaction. Expressed as shear stress required to trigger liquefaction (y-axis), duration, expressed as number of cycles to liquefaction (x-axis), and density, shown as three different curves. Reproduced from NAE, 2016

Increased load requires fewer cycles to failure while denser soils require more. Importantly, even a somewhat dense soil will liquefy with a high enough load and long enough shaking. Similarly, greater magnitudes of maximum strain amplitude are also associated with larger pore pressure development.

Soil type is also extremely important, as plastic fine grained soils have significantly less contractive behavior under loading. Beyond a certain fines content and plasticity index a soil will no longer be susceptible to liquefaction (NAE, 2016). Initial shear stresses, from sloping ground or foundations, can either increase the rate of pore pressure generation, for very loose soils, or suppress it, for medium dense to dense, though research is not yet decided on the exact magnitude of these effects (NAE, 2016). Aging and cementation also effect liquefaction potential — Holocene age younger deposits have been observed to liquefy more often than Pleistocene age or other older deposits (NAE, 2016). Finally, the spatial variability in pore pressure development within the deposit, such as when it is capped by an impermeable layer or makes up the core of an earthen dam, will also effect liquefaction potential (NAE, 2016).

2.1.4 Measurement of Soil Properties Relevant to Liquefaction Triggering

Laboratory testing is typically not used to assess liquefaction triggering potential because of the potentially unconservative effect of sample disturbance. Even with extremely careful sampling, liquefiable sands unavoidably densify some amount before testing which will under predict liquefaction potential (NAE, 2016). Instead, liquefaction assessments use representative in-situ test parameters from tests such as the standard penetration test (SPT), shear wave velocity (V_s) testing, or the cone penetration test (CPT). Of all of all the methods for determining soil liquefaction resistance, CPT measurements are preferred because of their high quality, repeatable, near-continuously sampled data, and their insensitivity to operator error which is a drawback of SPT blow counts (NAE, 2016). A cone penetrometer is equipped with load cells and friction transducers to measure the force per unit area on both the tip of the cone and sleeve (Figure 8).

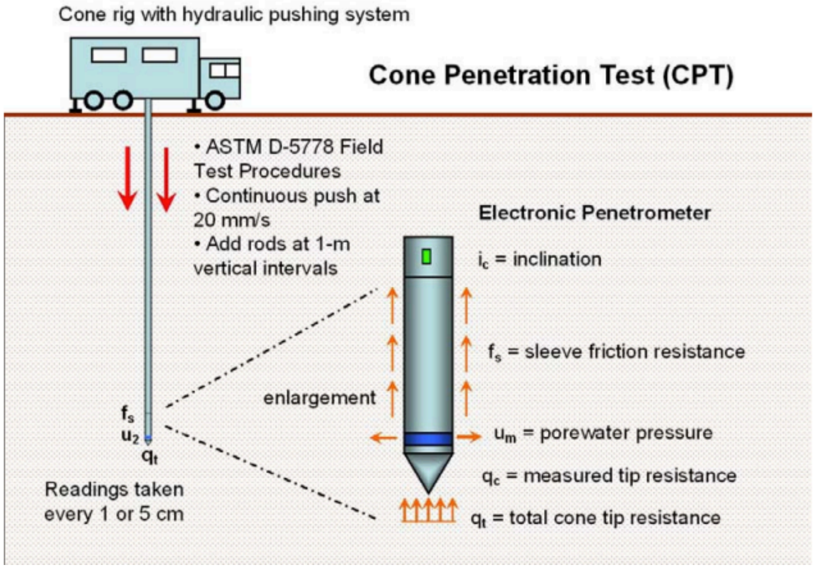
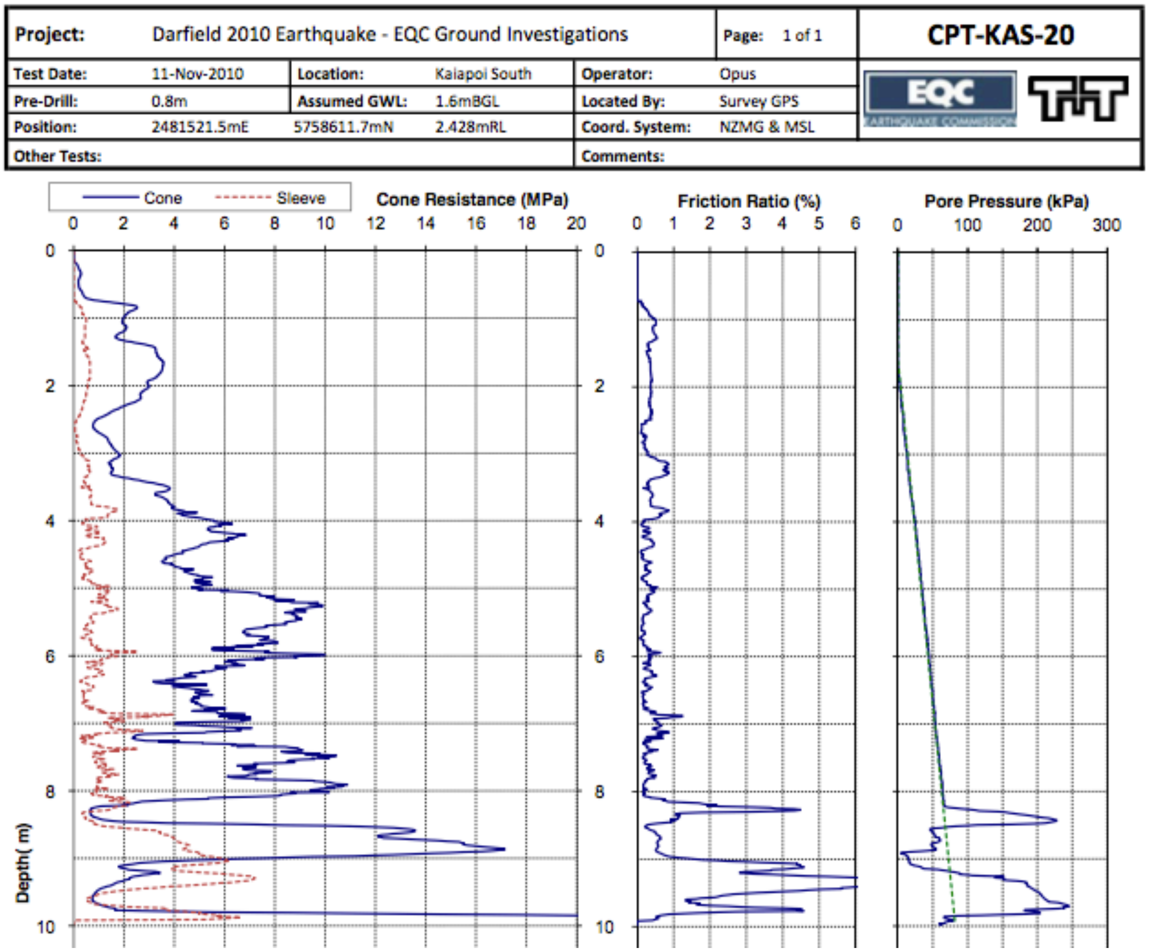

Cone rig with hydraulic pushing system
Cone Penetration Test (CPT)
• ASTM D-5778 Field Test Procedures
• Continuous push at 20 mm/s
• Add rods at 1-m vertical intervals
Electronic Penetrometer
i_c = inclination
f_s = sleeve friction resistance
enlargement
u_m = porewater pressure
q_c = measured tip resistance
q_t = total cone tip resistance
f_s
u_2
q_t
Readings taken every 1 or 5 cm

Figure 8 – An Overview of CPT Procedures and the Data Obtained During Testing. Many rigs are also equipped with geophones to measure V_s. Reproduced from the NHCRP report " Cone Penetration Testing: A synthesis of highway practice".

These are referred to in practice as tip resistance (q_c) and sleeve friction (f_s). For ease of interpretation, often times the friction ratio (R_f) of sleeve friction divided by tip resistance is reported instead. Pore water pressure measurements (u) are also often recording to correct tip/friction measurements for dynamic effects. A typical CPT sounding is shown in Figure 9, following. The data obtained from the test are empirically correlated with many useful engineering properties. In certain cases, the penetration resistance is normalized to correct for overburden stresses and referred to as $q_{c,1}$. Many ELM's will further modify the penetration resistance, often denoted by additional subscripts following the 1.



Project:	Darfield 2010 Earthquake - EQC Ground Investigations			**Page:** 1 of 1	**CPT-KAS-20**
Test Date:	11-Nov-2010	**Location:**	Kaiapoi South	**Operator:**	Opus
Pre-Drill:	0.8m	**Assumed GWL:**	1.6mBGL	**Located By:**	Survey GPS
Position:	2481521.5mE	5758611.7mN	2.428mRL	**Coord. System:**	NZMG & MSL
Other Tests:				**Comments:**	

Figure 9 – A Typical CPT Sounding Showing q_c, R_f, and u. Many reports may also include correlations for typical soil properties such as behavior type, shear strength, unit weight, and others. Reproduced from Green et al., 2011.

In closing, liquefaction behavior is highly complex and usually very dependent on specific site conditions and earthquake loads. It is unlikely that any model derived entirely from soil mechanics would be able to be generalized to a suitable breadth of field conditions to be useful in general practice. Instead, engineers turn to a semi-empirical "simplified method" when performing liquefaction assessment. That statistical methods used to develop these ELM's are discussed in the following section.

2.2 Review of Statistical Methods Commonly Used in ELM Development

The following section presents a brief theoretical background of the statistical methods used to develop popular ELM's and in our study.

2.2.1 Introduction to Regression Modeling

Regression models relate the expectation, or mean, of each observed outcome to a function of predictor variables and regression parameters and each response is assumed to come from some specified distribution (Dunn and Smyth, 2018). Formally, (after Dunn and Smyth, 2018):

$$E[y] = \mu_y = f(\boldsymbol{x}; \boldsymbol{\beta})$$

$$y \sim \text{Some Distribution}(\mu, \sigma^2)$$

$$(\mu, \sigma^2)$$

Where y is an observed outcome associated with two vectors (not necessarily the same length); **x** of predictor variables and **β** model parameters. Although the input to *f* could be any combination of x's and β's, it is often convenient to assume that is some linear combination of the two, as expressed by the following equation (after Duncan and Smyth, 2018):

$$\eta = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

Where η is commonly called a linear predictor encompassing n predictor variables.

2.2.2 Generalized Linear Models: An Overview

Generalized linear models (GLM) are a specific class of regression models that are widely used to model binary outcomes, though they are also useful for count, proportion and other data (Dunn and Smyth, 2018). In this model, observed outcomes are assumed to have a distribution from the exponential dispersion model family -- a broad group of discrete and continuous probability functions that include the normal, binomial, and Poisson distributions (Dunn and

Smyth 2018). Further, the linear predictor is related to the mean via a special function called a link function (after Dunn and Smyth, 2018):

$$g(\mu_y) = \eta \; or \; \mu_y = g^{-1}(\eta)$$

Where $g(\ )$ is monotonic and differentiable. In this model each predictor variable is associated with a unique regression parameter.

Any GLM can be fully specified by the link function and specific distribution for each data point, often referred to as the variance family (Dunn and Smyth, 2018). For example, setting the link function to simply be the identity function and the error distribution to be normal recovers the classic linear least squares regression (Dunn and Smyth, 2018). Of specific interest to this study is the logistic regression, a widely used statistical tool for modeling binary outcomes (Oomen et al., 2010).

2.2.3 Logistic Regression and Bernoulli Random Variables

Logistic regression is a GLM with a logit link function and a binomial variance family (Dunn and Smyth, 2018). Although many link functions exist for binary data, Zhang et al. 2013 showed that for liquefaction triggering the logit link performs as good as or better than others as measured using a Bayesian model comparison method, described in their paper. The logit link function is defined as (after Liao et al., 1988):

$$g(\mu_y) = \ln\left(\frac{\mu_y}{1-\mu_y}\right)$$

$$y \sim Binomial\ (n = 1, p)$$

The binomial probability mass function for a single trial is referred to as the Bernoulli distribution. A Bernoulli random variable parametrized by a probability of occurrence p has the following mass function (after DeGroot and Schervish, 2012):

$$f(x|p) = \left\{\begin{matrix} p^x(1-p)^{1-x} \text{ for } x = 0{,}1 \\ 0 \text{ else} \end{matrix}\right\}$$

or in tabular form:

x	0 (no)	1 (yes)
f(x)	1-p	p

The mean is calculated as:

$$E[x] = \ p(1) + (1 - \square)(0) = p$$

Usefully, the mean of a Bernoulli variable is simply the probability of a positive outcome. The variance is:

$$Var[x] = p(1-p)$$

Variance is maximized at p = 0.5 and approaches 0 as p approaches 0 or 1, expressing the intuition that when speaking about a yes or no outcome we are more certain when giving it a high or low probability.

Returning to our formulation of a logistic regression, it is natural to replace the mean with the probability of the desired outcome when formulating the regression model. Considering probability of liquefaction (denoted P_L), we can state (after Liao et al., 1988):

$$\ln\left(\frac{P_l}{1-P_l}\right) = \eta$$

Inverting the logit transformation, we now have a direct formula for computing the probability of liquefaction for a generic data point, given n predictor variables and regression parameters (after Liao et al., 1988):

$$P_L = \frac{1}{1 + \exp\{-(\beta_0 + \ \beta_1 x_1 + \cdots + \beta_n x_n\ )\}}$$

To improve model utility, it is common to use transformations, typically powers or logarithms, of the original predictor variables (Liao et al., 1988).

2.2.4 Logistic Regression: Parameter Estimation

Although other methods exist, the most common means of determining the regression parameters is via maximum likelihood estimation. The estimated coefficients commonly referred to as maximum likelihood estimates (MLE) (Dunn and Smyth, 2018). For **β**, a vector of model coefficients, the likelihood of observing n_L liquefied cases and n_{NL} non-liquefied cases given data **x** associated with outcomes **y** is (after Zhang et al., 2013):

$$l(\boldsymbol{\beta}|\boldsymbol{y},\boldsymbol{x}) = \prod_{i=1}^{n_L} P_{L,i} \prod_{j=1}^{n_{NL}} (1 - P_{L,j})$$

Where $P_{L,i}$ is calculated from the equation preceding for the i^{th} instance of liquefaction using the corresponding predictor variables. Similarly, $P_{L,j}$ is computed for the j^{th} instance of nonliquefaction. The value of **β** that maximizes this function is called its maximum likelihood estimate. For efficiency, most computer programs actually maximize the logarithm of the above function (called the log-likelihood).

2.2.5 Dealing with Class Imbalance

Because post-earthquake geotechnical reconnaissance is often focused on sites that have experienced ground failure (and subsequence impacts to engineered features) liquefaction databases contain more liquefied cases than non-liquefied cases. Several papers have examined the influence of this class imbalance on various liquefaction models (e.g., Cetin et al., 2004; Hu et al., 2017; Yazdi et al,. 2013; Oomen et al., 2011), but their results are often contradictory or inconclusive. Many models (Cetin et al. 2004, Moss et al. 2006, Idriss and Boulanger, 2016, etc.) use a weighting method that assigns different weights to liquefaction and nonliquefaction cases in the parameter estimation process. One such weighted log-likelihood is formulated as follows (after Zhang et al, 2013)

$$\ln[l(\boldsymbol{\beta})] = w_L \sum_{i=1}^{n_l} \ln\left(P_{L,i}\right) + w_{NL} \sum_{j=1}^{n_{NL}} \ln\left(1 - P_{L,j}\right)$$

Where w_l and w_{NL} are weights assigned to cases of liquefaction and non-liquefaction, respectively.

Alternatively, instead of assigning case weights during the fitting process, resampling procedures attempt to balance the dataset before the model is built (Kuhn and Johnson, 2013). Up-sampling procedures randomly select observations in the minority (less frequent) class to duplicate and down-sampling procedures randomly select observations in the majority class to remove (Kuhn and Johnson, 2013). In the context of liquefaction modeling, Hu et al., 2017 showed that up-sampling procedures have the potential to improve models' predictive ability. Importantly, they also demonstrated that the best method for weighting classes differently or compensating minority/majority classes is model dependent, and should be adjusted during the fitting process (Hu et al., 2017).

The above likelihood formulations assume that liquefaction or nonliquefaction are statistically independent. In reality, there is likely correlation between the predictor variables of interest. Whereas resistance data can be (and for this database- are) selected to be statistically independent there will be some unavoidable correlation in the loading variables. Specifically, a group of observations in an event are all subjected to the same earthquake loading (although each site response will be different due to local soil conditions). Thus, it is natural to believe that certain groups will be more likely to liquefy and certain groups will not. If this is not accounted for, the resulting model may be poorly fit and give misleading results (Clark and Linzer, 2015).

2.2.6 An Introduction to Mixed Effects and Multilevel Modeling

Multilevel models, sometimes referred to as mixed effects, random effects, or hierarchal models, extend classical regression models by allowing model parameters (slopes and intercepts) to change between groups (Jiang, 2007). For the purpose of this paper we will refer to parameters that do not change between groups as fixed effects and parameters that do as random effects, recognizing that literature on the subject does not always agree on these definitions (Gelman and Hill, 2007). Mixed effects models have seen noticeable use in ground motion attenuation relationship development (e.g. Brillinger and Preisler, 1985, Abrahamson and Youngs, 1992, Kuehn and Scherbaum, 2015).

The motivation for mixed models typically arises from data that has some natural grouping or hierarchy. For example, a modeler investigating student test performance may group students into classes, classes into schools, and schools into districts. Or, a modeler investigating voter preferences may group voters by ethnicity, gender, class, and geographic location. There will be natural correlation in these groups – students in a better school will likely perform better than others or voters from a certain economic class may tend to vote one way or another. A classic regression approach that does not account for data grouping is unable to determine these group effects and the impact they have on population mean trends (Gelman and Hill, 2007). However, a hierarchal approach (mixed effects model) has a systematic way of estimating and accounting for this inter-group variability without having to explicitly model its causes (Abrahamson and Youngs, 1992).

Recommendations for when to use a mixed model are often unclear or not immediately applicable to the research problem of interest (Clark and Linzer, 2015). Commonly, the recommendation is to use a mixed model when interest is in the underlying population as a whole or when the population includes groups not in the data, both of which would make sense in the context of liquefaction modeling (Gelman and Hill, 2007). However, with the current model fitting abilities of software it is possible to almost always fit both a mixed and classical model especially with the rise of powerful Bayesian inference programs (e.g. Stan). By doing so the researcher can investigate the scale of the inter-event variances, the impact of the differing coefficients, and decide for themselves the value added by the mixed model.

A properly defined multilevel model will have the following benefits over a classical model that are useful in the context of our work. A multilevel model accounts for both inter and intra group variation resulting in better estimates for both group level parameters and population means (Gelman and Hill, 2007). They also allow for better prediction of events in new groups – a model that ignores group variability will tend to understate the variability in predictors for new groups (Gelman and Hill, 2007).

Models fit to grouped data conceptually fall between two extremes – no pooling and complete pooling (Gelman and Hill, 2007). A completely pooled model, as the name implies, groups all the data together and estimates a single set of parameters. However, this completely ignores the impacts of group level variability and may potentially violate assumptions of independent data due to its grouped nature (Gelman and Hill, 2007). On the other hand, a no pooling model fits a regression for each group separately. Whereas the completely pooled model understates the group level variability, this approach will often over estimate it (Gelman and Hill, 2007). For example, if a group has a small number of data points compared to others in the data set, its estimations of a predictors effect on the outcome may diverge significantly from the true effect due to chance alone (Clark and Linzer, 2015).

Mixed models can be visualized as a compromise between no pooling and complete pooling (Gelman and Hill, 2007). They allow parameters to vary between groups but constrain these parameters to come from a population level distribution with hyperparameters estimated from the data, hence their “hierarchical” description. To illustrate this concept, we consider the most basic multilevel model: a linear regression where the intercept is allowed to vary by event and the slope is fixed. A model with i data points, j groups, a single slope (β), an intercept (α) that varies between groups, and a normally distributed error term (ϵ) can be expressed as follows (after Gelman and Hill, 2007):

$$y_i = \alpha_{j[i]} + \beta * x_i + \epsilon_i$$

$$\epsilon \sim \text{Normal}(0, \sigma_e)$$

This is equivalent to stating:

$$y_i \sim \text{Normal}\big(\alpha_{j[i]} + \beta * x_i, \sigma_e\big)$$

The subscript on the intercept term $\alpha_{j,[i]}$indicates that is for the j$^{\text{th}}$ event. The event level intercepts are given the further constraint that they come from a normal distribution with mean and standard deviation estimated from the dat. The full model is typically stated as:

Prior:

$$\alpha_j \sim \text{Normal}(\mu_\alpha\,, \sigma_\alpha)$$

Data:

$$y_i \sim \text{Normal}\big(\alpha_{j[i]} + \beta * x_i, \sigma_e\big)$$

Technically, the model for y is conditional on the α's. An alternate but equivalent “random effects” formulation would be to state:

$$y_i \sim \text{Normal}\big((\beta_0 + \alpha_{j[i]}) + \beta_1 * x_i, \sigma_e\big)$$

$$\alpha_j \sim \text{Normal}(0, \sigma_\alpha)$$

In this model the population level intercepts, the mean of the event level terms, are included explicitly and the random effect is modeled as a normally distributed deviation from this value. However, because the two models are functionally identical we prefer the first formulation because with many varying coefficients stating an adjustment term for each becomes tedious. Furthermore, we believe that the first formulation is a more natural way of expressing the data’s hierarchy.

Returning to our varying intercepts linear model, each group’s intercept can be expressed as a weighted average of the no pooling estimate and the mean intercept of all the groups. In the following formula the following new terms are introduced:, n_j the number of data points in the j^{th} event, $\bar{y}_j$ and the $\bar{x}_j$ the j^{th} event's mean response and predictor respectively, σ_y^2 the pooled data's variance, and σ_α^2 the inter-group intercept variance :

$$\alpha_j \approx \frac{\frac{n_j}{\sigma_y^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}} * \left(\bar{y}_j - \beta * \bar{x}_j\right) + \frac{\frac{1}{\sigma_\alpha^2}}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}} * \mu_\alpha$$

Because equation requires a simultaneous estimate of the variance parameters for σ_α^2 and σ_y^2 it is typically solved using some algorithmic technique (e.g Abrahamson and Youngs, 1985). While the estimation of group level effects becomes significantly more complicated outside of this example, the behavior described by this equation remains the same.

Averages from groups with smaller sample sizes carry less information and the weighting pulls the multilevel estimate closer to the population average. This effectively "shrinks" anomalous parameter estimates from sparse data groups closer to a better estimate and mitigates the effects of sample size disparity (Clark and Linzer, 2015). Conversely, averages from groups with more data carry more weight and the multilevel estimate is pulled towards the group's value. Two limiting cases naturally arise -- If a group has no data, its estimate is the population average and if it has an extremely large amount of data its estimate will be almost exactly the group's average. Intermediate cases will result in a multilevel estimate between the extremes (Gelman and Hill, 2007).

Multilevel models can be easily extended to GLM's as well. In our case, we would like to define a multilevel model that allows coefficients to vary between events Formally, we take the original equation for probability of liquefaction and state it for a data point in the j^{th} event as (after Gelman and Hill, 2007):

$$P_{L[j]} = \frac{1}{1 + \exp\left\{-(\beta_{j0} + \beta_{j,1}x_1 + \cdots + \beta_{j,n})\right\}}$$

$$\beta_{0,j} \sim \text{Nomal}(\mu_{\beta_0}, \sigma_{\beta_0})$$
$$\beta_{1,j} \sim \text{Nomal}(\mu_{\beta_1}, \sigma_{\beta_1})$$
$$\dots$$
$$\beta_{n,j}, \sim \text{Nomal}(\mu_{\beta_n}, \sigma_{\beta_n})$$

In this model, each event gets its own set of parameters which are constrained to come from a normal population distribution. The population means and standard deviations for each parameter, referred to as the coefficent's hyperparameters, are also estimated.

The resulting likelihood function does not have a closed form for most generalized linear models and is approximated through numerical integration, typically Gaussian quadrature (Breslow and Clayton, 1993). Most books on the subject (e.g. Jiang, 2007) include more detailed descriptions of these methods for the interested reader. However, when data is sparse at the group or the number of groups is small level these numerical maximum likelihood methods become unstable or wholly unusable (Gelman and Hill, 2007). Bayesian inference often can commonly be used to estimate hierarchal models that cannot be solved by maximum likelihood techniques (Gelman and Hill, 2007).

2.2.7 Introduction to Bayesian Modeling

A major limitation of any frequentist (or maximum likelihood) based approach is that, philosophically, parameters cannot be treated in a probabilistic fashion (Kruschke, 2015). A Bayesian analysis, on the other hand solves for the probability of observing different parameter values given experimental data and prior knowledge. The value in this approach first lies in ability to quantify model uncertainty in readily understandable terms. Secondly, the inclusion of prior information provides a consistent and mathematically sound framework for allowing expert consensus and physical behavior of the system being studied to inform models when data are sparse.

Bayes' Rule is the mathematical framework for updating our prior beliefs about the probability of an event occurring based upon observed evidence (Christensen et al., 2011). In a data analysis setting, we begin with a hypothesis about our data, typically regarding values of some population parameter or regression model coefficients. We then update the prior based upon the evidence in the data set Bayes rule is formally stated as (after Kruschke, 2015):

$$p(H|E) = \frac{p(E|H)p(H)}{p(E)}$$

where the prior that our hypothesis is correct *p(H)* is updated according to the probability *p(E|H)* of observing the evidence if our hypothesis was true, referred to as the likelihood. The result is a posterior probability *P(H|E)* that our hypothesis is correct given the evidence we have observed. The term *p(E),* the unconditional probability of the evidence, serves only as a normalization constant to ensure that our calculated values obey the axioms of probability (Kruschke, 2015). Thus, it is conceptually useful to think of Bayes rule as:

$$\text{Posterior } \alpha \text{ Prior} \times \text{Likelihood}$$

In a typical data analysis setting there are many possible parameter values, so we have many different hypotheses we wish to evaluate. In this case, the denominator, *p(E)* is often calculated using the total probability rule as follows (after Kruschke, 2015):

$$p(E) = \sum\nolimits_m p(E|H_m)p(H_m)$$

Where *p(H_m)* is the prior regarding the mth hypothesis, *p(E|H_m)* is the likelihood of observing the data if the mth hypothesis is true, which are summed over all possible hypotheses. Combining the two above equations, we arrive at the practical statement of Bayes rule in for discrete variables (after Kruschke, 2015):

$$p(H_m|E) = \frac{p(E|H)p(H_m)}{\sum_m p(E|H_m)p(H_m)}$$

However, even for discrete data (counts, binary etc.) the parameters themselves are often continuous (i.e., a population mean or regression coefficients). Additionally, because we have many observations and possibly many parameters, both the likelihood and posterior distributions will be multivariate (i.e., the return a single probability density value for an input vector of data **x** or parameters **θ).** Thus, in full generality, the posterior density function (pdf), is computed as follows (after DeGroot and Schervish, 2012).

$$f(\boldsymbol{\theta}|\boldsymbol{x}) = \frac{f(\boldsymbol{x}\ |\boldsymbol{\theta})f(\boldsymbol{\theta})}{\int \dots \int f(\boldsymbol{x}|\boldsymbol{\theta})f(\boldsymbol{\theta})d\theta}$$

The total probability rule becomes the iterated integral of the product of the prior and likelihood over the support of all θ's. Using this framework, it is now possible to cast our logistic regression in a Bayesian setting. The Bayesian model uses the same likelihood function as before, but now includes Bayes rule to calculate the probability distributions of the regression parameters.

For most realistic problems, this Bayesian formula is quite difficult to compute. In a small dimension (i.e. only one or two model parameters) it is possible to approximate the density functions as mass functions on a fine enough grid, but in a larger space we simply cannot create a dense enough grid. For example, a six parameter model with a grid of 1000 values for each parameter has 1000^6 parameter combinations to evaluate; far beyond the ability of modern computers (Kruschke, 2015). Therefore, we are limited to: choosing "nice" parametric forms that have a known analytic solution for the posterior, a numerical integration (e.g., Gaussian quadrature or Laplace approximations), or simulation approaches (randomly generating values from the posterior). Simulation is often preferred to numerical integration due to the poor scaling of numerical integration with higher dimensions—a grid of length N in D-dimensional space will require N^d evaluations of the integrand (Betancourt, 2018). As such, simulation warrants further discussion here.

2.2.8 Markov Chain Monte Carlo (MCMC) Methods

Mote Carlo simulation, qualitatively, involves generating many representative values (called draws) of a random variable (Kruschke, 2015). For some random variable X, with pdf f(x) which is often called the target distribution, we can imagine a spinner marked with the possible values of X that is biased to point at various values of X exactly according to f(x) (Kruschke, 2015). Spin it enough times and record what values are chosen, and we have a large number of representative values of X. With enough of these, we can approximate many useful distributional characteristics of the original random variable such as mean, standard deviation, and cumulative probability (Kruschke, 2015).

In a practical implementation, all simulation really requires is an ability to generate a random number and some criteria to determine if it should be included in your collection of representative values of the target distribution (DeGroot and Schervish, 2012). Many methods of specifying an acceptance/rejection criteria exist (envelope methods, importance sampling, etc.) but MCMC methods are the most useful for simulating from higher dimension distributions (Christensen et al., 2011).

A Markov chain is a series of random vectors $\boldsymbol{\theta}^{(1)}$, $\boldsymbol{\theta}^{(2)}$, $\boldsymbol{\theta}^{(3)}$, … drawn from a set A, with conditional densities $q^{(1)}(\boldsymbol{\theta}^{(1)})$,$q^{(2)}(\boldsymbol{\theta}^{(2)})$,$q^{(3)}(\boldsymbol{\theta}^{(3)})$,… that satisfy the following property (after Christensen et al., 2011):

$$\Pr\left(\boldsymbol{\theta}^{k} \in A \middle| \boldsymbol{\theta}^{1}, \dots, \boldsymbol{\theta}^{k-1}\right) = \Pr\left(\boldsymbol{\theta}^{k} \in A | \boldsymbol{\theta}^{k-1}\right)$$

Suppose k-1 represents the current step in the process and k the next step. Then given the current value θ^{k-1} , the next value θ^{k} is conditionally independent of past values $(\theta^{1}, \dots, \theta^{k-2})$. This expresses what is called a Markov property; what state you transition to is only dependent on your current state (Christensen et al., 2011). To construct a Markov chain all we need to do is specify some initial distribution $q^{(1)}(\boldsymbol{\theta}^{(1)})$ and the conditional distributions $q_{j|j-1}(\boldsymbol{\theta}^{j}|\boldsymbol{\theta}^{(j-1)})$ (Christensen et al., 2011). While not immediately obvious, with appropriate choices of the initial distribution and transition densities (and appropriate mathematical justification) we can construct algorithms that will eventually end up sampling from the target posterior distribution.

The mathematical proofs required are summarized in (Christensen et al., 2011) in the context of the popular Metropolis-Hastings algorithm, described below.

2.2.9 The Metropolis – Hastings (MH) Algorithm and Hamiltonian Monte Carlo

The general idea behind the MH algorithm, first developed in 1953 to solve statistical mechanics problems in nuclear chemistry, is to perform a random walk through the parameter space of a target distribution (in our case, the posterior). To retain the Markov property, where the jump lands on the next step is determined only by its current location. At each jump, if the ratio proposed to the current posterior probability is compared. Importantly, this only requires product of the prior and likelihood (un-normalized posterior) because normalizing constants will cancel out. To ensure that the algorithm can repeat indefinitely, not all jumps with a ratio of less than one are rejected. By doing so, parameter values are sampled in proportion to their probability density (Metropolis et al, 1953). This style of accepting/rejecting proposals requires that the method of generating proposals is symmetric i.e the probability of going from state x to y is the same as going from y to x. Formally, at step k where we have already sampled ($\boldsymbol{\theta}^{(1)}$, $\boldsymbol{\theta}^{(2)}$, $\boldsymbol{\theta}^{(3)}$, …, $\boldsymbol{\theta}^{(k)}$) (after Christensen et al., 2011):

1) Generate a proposed value $\boldsymbol{\theta}^{(*)}$ from some proposal method conditional on the last draw.

2) Define $\alpha \equiv min\left\{1, \frac{\pi(\boldsymbol{\theta}^{(*)})}{\pi(\boldsymbol{\theta}^{k})}\right\}$, where $\pi(\ )$is the target distribution. In Bayesian statistics this is the unnormalized posterior i.e. prior times likelihood (both known).

3) Simulate a random number U between 0 and 1.

4) Then, $\boldsymbol{\theta}^{k+1} = \begin{Bmatrix} \boldsymbol{\theta}^{(*)}\ if\ \alpha \geq U \\ \boldsymbol{\theta}^{(k)}\ if\ \alpha < U \end{Bmatrix}$

The software used in our study, Stan (Carpenter et al, 2012), uses an adaptively tuned Hamiltonian Monte Carlo (HMC) method, described in depth in Neal, 2012. The primary difference between HMC and MH is that the proposal mechanism is modified to produce less rejected draws, improving the algorithms computational efficiency. As a high level overview, it does so by transforming the unnormalized posterior distribution into a sort of "landscape" that a particle can move through. At each step, the algorithm gives a fictitious particle some randomly generated momentum, and solves for its trajectory via Hamiltonian dynamics (Neal, 2012). After some preset time, the particles new location is used as the proposed value in the MH algorithm and the process is repeated. The intuition behind why this improves standard MH is simple : we want to sample values with higher probability, or if we invert the distribution from the "valleys", more often. As the fictitious particle moves across the landscape it will often become "trapped" in these "valleys", corresponding to higher regions of probability density. This produces proposals that have a better chance of acceptance and improves sampling efficiency.

2.2.10 Reliability Concepts Used in ELM's

A reliability-based analysis begins by assuming that the performance of the "system" is characterized by a vector **x** of directly observable random variables (Der Kiureghian, 2004) The probability of failure can be calculated from the joint distribution of **x** by integrating over the subset of their outcome space which defines the failure event (Der Kiureghian, 2004). Defining the failure region requires formulating a limit-state function for each component, often called g(**x**), often such that when g(**x**) ≤ 0 the component is in a failure state. The failure state of the system as a whole depends on whether the components are linked in series, parallel, or some combination of the two (Der Kiureghian, 2004).

To illustrate the above concepts, consider a beam failing in a single mode such as shear or flexure. The governing random variables could be a maximum stress, typically called demand, which would be a function of other random variables such as the magnitude and location of applied load and member geometry; and a minimum strength, typically called capacity, which would also be a function of other random variables such as member geometry and material properties. The natural choice for the limit-state function is g = Capacity – Demand. The failure subset would then be all possible combinations where load exceeds resistance, or when the limit-state function is negative. However, even for this simple example, performing the integral over the failure region is often difficult or impossible so practitioners usually turn to numerical methods such as first or second order reliability methods (FORM and SORM), discussed in depth in Der Kiureghian, 2004.

Reliability-based approaches to liquefaction triggering typically model the initiation of liquefaction as a single component problem and require developing an appropriate limit-state function based upon the observed data.

2.2.11 Evaluating the Performance of Probabilistic Classifiers

While a predictive model produces a probability of liquefaction, in current engineering applications we often need to classify sites as liquefiable or not to determine if it is necessary proceed with further analyses or mitigation (Oomen et al., 2010). This is usually done by setting a threshold of liquefaction risk (TH_L), based upon potential consequences, defined as the highest probability of liquefaction we can tolerate without moving to mitigation. Sites falling above that threshold are classified as liquefiable and those below are not. These threshold of risk may differ significantly – when designing a hospital we will want to mitigate at a lower probability of liquefaction than for a non-occupied warehouse. At a given threshold, the confusion matrix is a useful tool for visualizing model performance (after Kuhn and Johnson, 2013):

		Predicted	
Observed		**Yes**	**No**
	Yes	# of True Positives (TP)	# of False Negatives
	No	# of False Positives (FP)	# of True Negatives (TN)

With these counts (which will be dependent on the specified threshold), several useful metrics can be calculated. Overall model accuracy is computed as the proportion of events classified correctly:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

However, this metric can be misleading as it is sensitive to natural class frequencies. If negative outcomes are infrequent then a model can achieve near perfect accuracy by only predicting positive outcomes (Kuhn and Johnson, 2013). Thus, if there is substantial cost associated with false positives (i.e. with expensive ground improvements) this is an inappropriate performance metric. Precision of a model measures the proportion of events in a single class predicted correctly (after Oomen et al., 2011):

$$P^{+} = \frac{TP}{TP + FP} \text{ or } \mathrm{P}^{-} = \frac{TN}{TN + FN}$$

And recall of a model measures the proportion of correct predictions out of all prediction of that class (after Oomen et al., 2011):

$$R = \frac{TP}{TP + FN} \text{ or } \frac{TN}{TN + FP}$$

These two metrics can be combined into an F-Score by taking their weighted harmonic mean and specifying β, the importance of recall to precision (after Oomen et al., 2011):

$$F_{\beta} = \frac{(1+\beta^2)(P*R)}{\beta^2 * P + R}$$

A final metric is known as the Matthew's correlation coefficient which ranges from -1 to 1, with -1 being the worst and 1 being the best. It measures the correlation between actual and predicted classes and is calculated as follows (Yazdi and Moss, 2016):

$$MCC = \frac{TP * TN - FN * FP}{\sqrt{(TP+FN)(TN+FP)(TP+FP)(TN+FN)}}$$

Importantly, all of these metrics will change based upon the specified yes/no threshold. Since the threshold of acceptable risk may vary considerably from project to project we want a model that performs well at all levels of TH_{L}.

Receiver operating characteristics (ROC) curves are a useful tool for evaluating a model's performance across all possible threshold values. They compare the true positive rate (after Fawcett, 2006):

$$\mathrm{TPR} = \frac{\text{Postives correctly classified}}{\text{Total Postives}} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$$

And the false positive rate (After Fawcett, 2006):

$$\mathrm{FPR} = \frac{\text{Negatives incorrectly classified}}{\text{Total negatives}} = \frac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$$

ROC curves are plotted as FPR vs TPR (Figure 10), with each point corresponding to a specific threshold value.

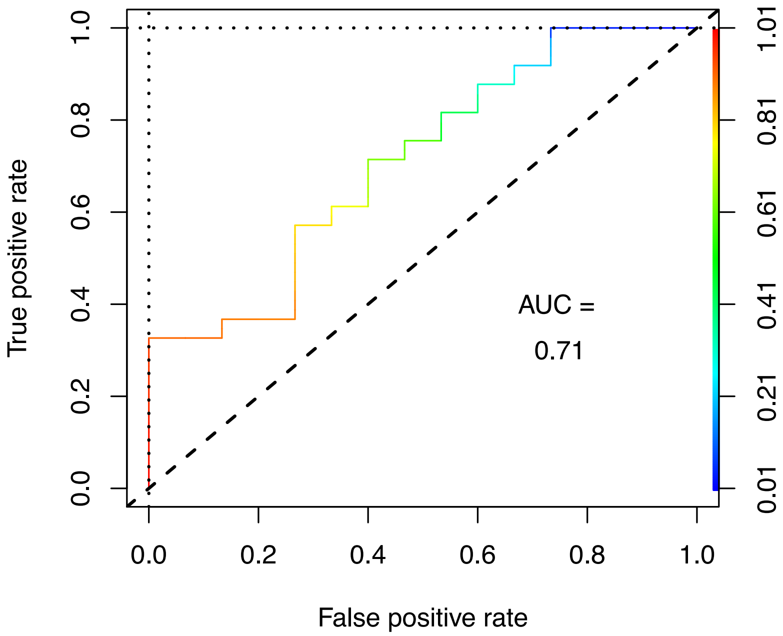


Figure 10 – A Sample ROC Curve. Showing true and false positive rates at all classification thresholds. The dashed and dotted lines represented lower and upper bounds for performance, respectively. The curve is colorized by threshold values and the area underneath reported.

A threshold of 1.0 will produce no positive classifications (the point 0,0). As the threshold is reduced, the model begins to produce true and false positive classifications until the threshold crosses the last actual positive occurrence and only false positives are produced. If a model guesses at random, its ROC curve will be a straight line 45-degree line. This represents a rational lower bound for performance, and any model that performs worse is likely flawed in its formulation. The dotted line corresponds to a model that classifies perfectly (no false positives). Qualitatively, we can think of comparing how “close” our curve is to this ideal curve as a means of describing comprehensive model performance. Practically, this is done by computing the area under the curve (AUC), which will range from 0.5 to 1.0 with higher values indicating better model performance (Fawcett, 2006). The statistical interpretation of this value is the probability that a randomly chosen positive instance will have a computed higher probability of occurrence than a randomly chosen negative one (Fawcett, 2006). ROC curves also have the benefit of being insensitive to class imbalances, a change in the ratio of positive to negative outcomes will not change the ROC curve (Fawcett, 2006).

With the rise in computational power and increased research into predictive models many current functional forms can learn the structure of a complex data set (Kuhn and Johnson, 2013). Figure 11 shows a two class, two predictor variable data set and two models fit to the data. Model 1 attempts to encircle every possible data point, producing unrealistically complex class boundaries. It has learned not just the signal in the data but its unique noise as well. Because future data is unlikely to have the same noise pattern the model will perform poorly when making future predictions. This model is said to be “over-fit”. Model 2 presents a more generalized boundary that will have greater utility when making future predictions. Importantly, if Model 1 was validated using the same data set as it was built on the estimated accuracy would be overly optimistic and potentially misleading if it was being evaluated for its usefulness in practice.

To avoid the problems associated with overfitting, modelers split their data into training and testing sets. As their names imply, training sets are the data points the model is fit on and testing sets are those that the validation metrics are developed on (Kuhn and Johnson, 2013). With a large enough data set a modeler can simply set aside a suitable number of points to create

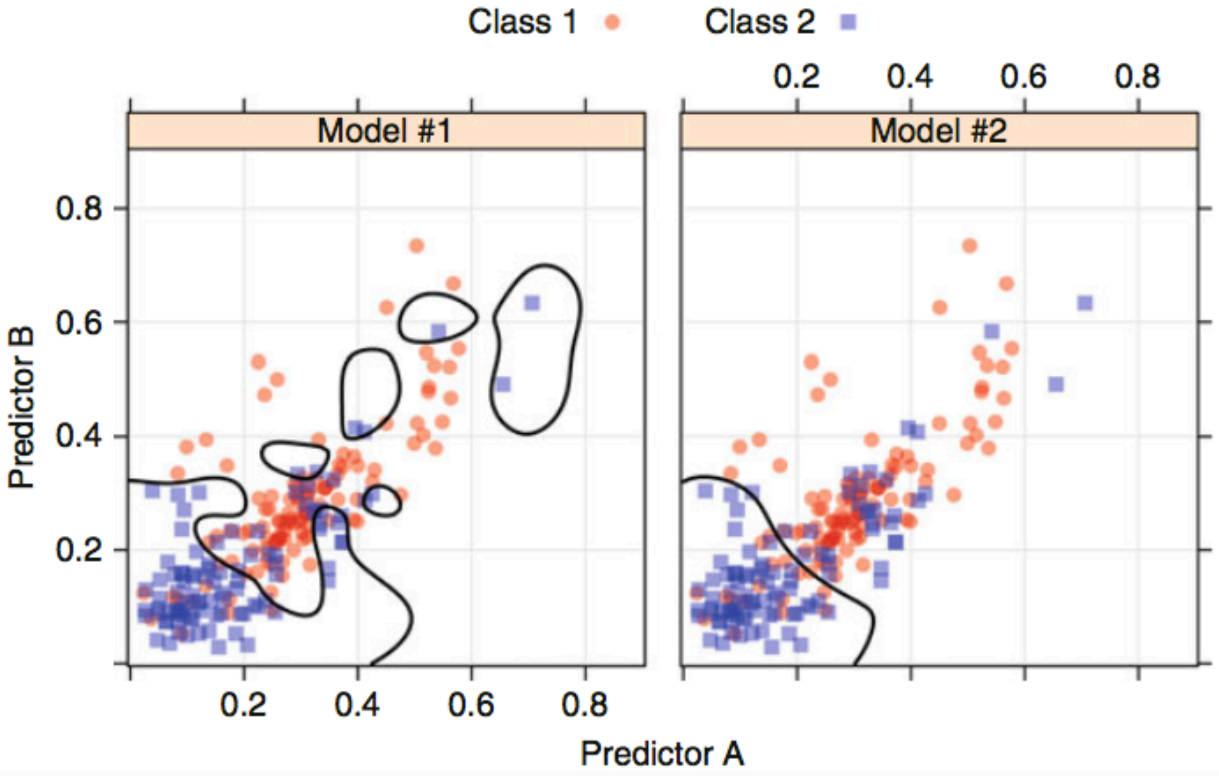


Figure 11 – A Sample Predictive Modeling Classification Problem. Reproduced from Kuhn and Johnson, 2013.

a single training set without having to worry about limiting their ability to develop the model. Usually, there is a natural method of creating these splits – a spam filter may be trained on messages collected in the past years and tested on messages collected in the current month. The sets could also be split randomly, or consciously based upon properties such as class frequency or predictor variable magnitudes (Kuhn and Johnson, 2013). Because the testing set was never used in the validation process this represents the most unbiased assessment of how a model generalizes to new data (Kuhn and Johnson, 2013). However, it is often desirable to use more than one training/testing split and resampling methods are used instead.

Resampling methods refer to repeatedly splitting the data set into training and testing subsets, fitting and validating models, and averaging model parameters and validation metrics across all runs. With an appropriate data splitting strategy all the data points can be used

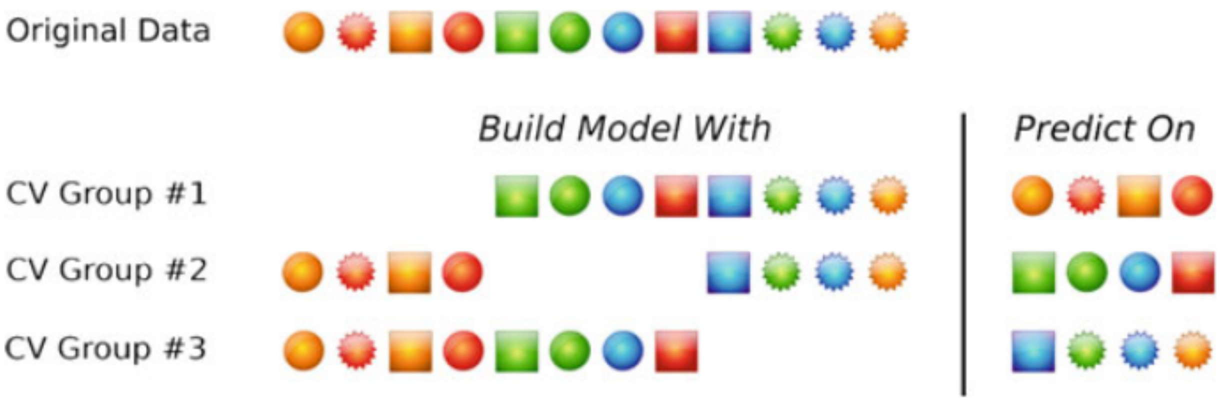


Figure 12 – A Visualization of 3-fold Cross Validation. The dataset is split into three groups of equal size then trained on 2 and tested on the third. This is repeated until all the groups have been used for both training and testing**.**

independently for training and testing. One of the most popular resampling method is known as k-fold cross validations. In k-fold cross validation the dataset is randomly split into k subsets ("folds") and in each run the model is trained on all but one of them and tested on the held out group (Figure 12).

There is no formal rule, but typically 5 or 10 folds are used (Kuhn and Johnson, 2013). The process continues until all the folds have been used for both training and testing. By doing this, the model can use all the available data points while still reporting relatively unbiased performance metrics. All the data is then used to determine the final model parameters. Because the splitting process is random many modelers will repeat the entire k-fold cross validation process several times to capture the uncertainty in performance estimates. Other methods for random resampling exist, such as bootstrapping or leave one out cross validation, but in most practical applications their performance is comparable to k-fold cross validation (Kuhn and Johnson, 2013). Additionally, researchers may also decide to use non-random splitting methods because of nuances in the data set or model (Kuhn and Johnson, 2013).

2.3 Modern Liquefaction Assessment

A complete liquefaction assessment is typically conducted in the following steps described on the following page (Figure 13). While all of these steps are critical in a properly performed liquefaction assessment, this thesis focuses on the aspect of liquefaction triggering.

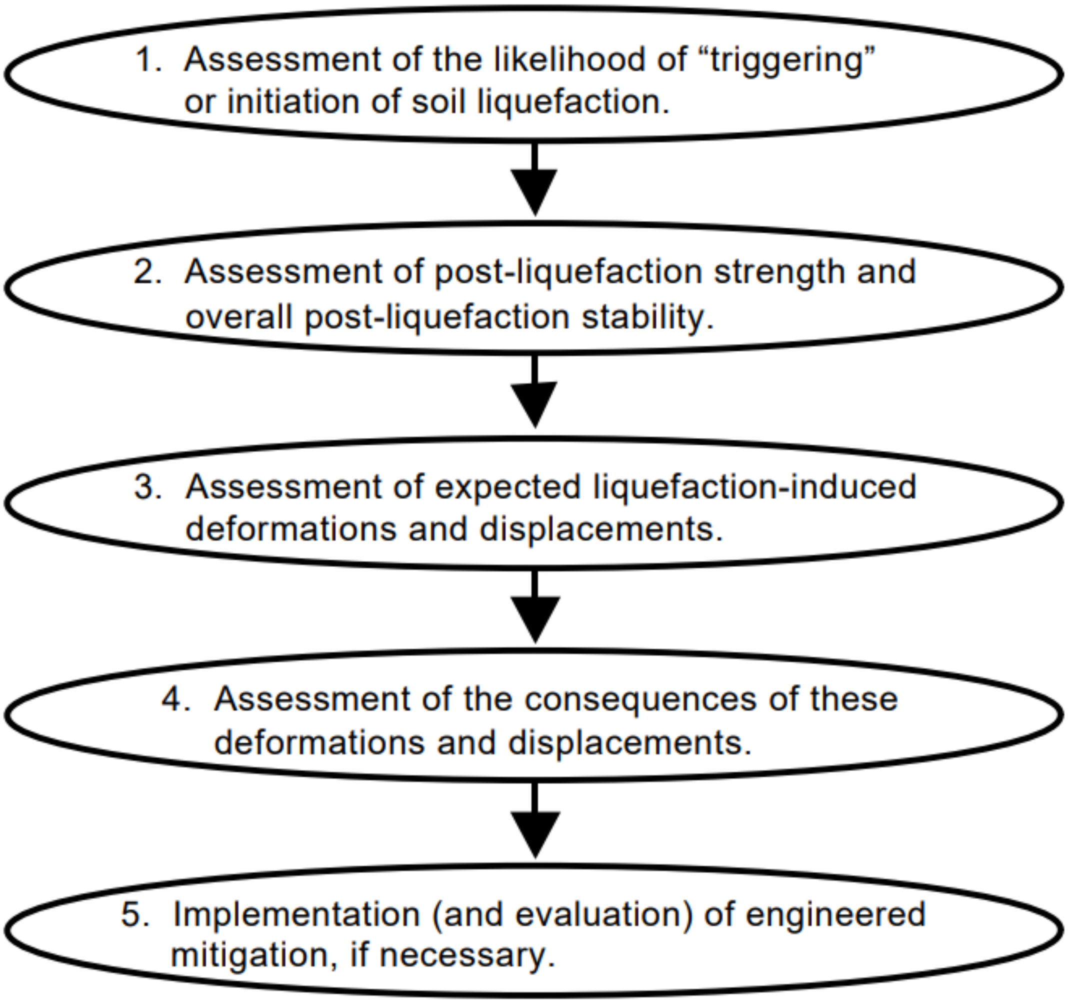

1. Assessment of the likelihood of "triggering" or initiation of soil liquefaction.
2. Assessment of post-liquefaction strength and overall post-liquefaction stability.
3. Assessment of expected liquefaction-induced deformations and displacements.
4. Assessment of the consequences of these deformations and displacements.
5. Implementation (and evaluation) of engineered mitigation, if necessary.

Figure 13 – A Modern Liquefaction Assessment Framework. Importantly, these procedures are only applied to soils that are determined to be susceptible to liquefaction. Applying triggering models to soils whose behavior is governed by physical principles that preclude liquefaction will give misleading (and wrong!) results. Reproduced from Seed et al., 2003

2.3.1 Overview of the Simplified Method

First developed in 1971 by Seed and Idriss, the “Simplified Method” is the most commonly used means for assessing liquefaction triggering in practice today. Simply put, it compares the “load” of the earthquake (expressed as the cyclic stress ratio, or CSR) to the “resistance” of the soil (expressed as the cyclic resistance ratio, CRR). CSR, as a function of depth (z) is formally defined as the ratio of the peak shear stress (τ) to the pre-existing vertical effective stress (σ'_{vo}) (after NAE):

$$CSR_{(M,\sigma'_{vo})}(z) = \frac{0.65 * \tau_{nax}(z)}{\sigma'_{vo}(z)}$$

The 0.65 scaling term serves to reduce the peak value, which is only experienced once in an event by definition, to a more representative value experienced multiple times. The subscripts indicate that it is computed for a specific moment magnitude and preexisting stress state. However, computing the peak shear stress requires a site specific dynamic analysis which can be difficult and time consuming. The “simple” part of the simplified method estimates the peak shear stress as the product of peak ground acceleration (as a fraction of gravitational acceleration, g) and total stress times a depth factor (r_d) that accounts for the nonlinear response of the soil profile (after NAE, 2016):

$$CSR_{(M,\sigma'_{vo})}(z) = \frac{0.65 * \sigma_v(z) * \frac{PGA}{g}(z) * r_d}{\sigma'_{vo}(z)}$$

To account for effects of shaking duration and adjust to reference values of M = 7.5 and $\square'_{vo}$ = 1 atmosphere, the expression is standardized with a magnitude scaling factor (MSF), effective stress correction factor (K_σ) and driving shear stress correction factor (K_α) (after Yazdi and Moss, 2016):

$$CSR_{(M=7.5,\sigma'_{vo}=1\ atm)}(z) = \frac{0.65 * \sigma_v(z) * \frac{PGA}{g}(z) * r_d}{\sigma'_{vo}(z)} * \frac{1}{MSF} * \frac{1}{K_\sigma} * \frac{1}{K_\alpha}$$

All of these correction factors, including r_d, are empirically derived and different researchers have proposed various means of calculating them. Interested readers are referred to the NAE text for further discussion of the ongoing research in this area. Importantly, because they are based upon a scatter of observed data points, even if all other terms in the equation can be calculated exactly (and often they cannot) there will always be uncertainty associated with the CSR. Furthermore, because ELM's are developed using CSR's calculated with specific methods for each coefficient it is important to use the original methods when making predictions with that model (NAE, 2016).

The cyclic resistance ratio, as its name implies, is defined as the ratio of a soil's liquefaction resistance to the earthquake loading. This naturally extends to a deterministic engineering design framework where the factor of safety (FS) is calculated as:

$$FS = \frac{CRR}{CSR}$$

Many methods for determining the CRR in this framework exist. They all correlate CRR with an in-situ soil property, typically from a standard penetration test (SPT) cone penetration test (CPT). Popular triggering curves for SPT energy corrected, normalized blow count ($N_{1,60}$) are found in Youd et al., 2001, Cetin et al., 2004, Idriss and Boulanger, 2008 etc. For normalized tip resistance ($q_{c,1}$), occasionally with other modification factors applied, curves can be found in Robertson and Wride, 1998, Moss et al. 2006, Idriss and Boulanger, 2016, etc. Correlations also exist for shear wave velocity (Andrus and Stoke, 2000, Kayen et al., 2013 etc.) and less common in-situ tests, such as the Becker penetrometer (NAE, 2016).

Probabilistic models follow the same principle of basing soil liquefaction resistance off of an index property measurement, but instead of directly providing a CRR they instead provide a probability of liquefaction given the anticipated CSR and index measurements. This nuanced treatment of whether or not liquefaction will occur is necessary for a performance based engineering approach (NAE). These models are often “converted” to a deterministic approach by setting some probability of liquefaction, usually based on expert consensus, as the FS = 1.0 curve for code based design that requires a certain FS to be achieved.

2.4 Review of Existing CPT Based ELM’s

Because this work deals with a CPT based probabilistic liquefaction triggering assessment, a brief description of the existing models developed for this purpose follows. While this discussion mostly focuses on the actual statistical methods used to fit the models to the CPT case histories, important research into more accurately estimating soil resistances (often accounting for the more nuanced behavior of fines content and thin layers) and CSR modification factors was also occurring and had a similar impact on the state of practice.

2.4.1 Early Deterministic Models

Prior to the late 1990s a variety of deterministic CPT based triggering relationships had been established through the work of Olsen and Koester, 1995, Suzuki et al., 1995 Robertson and Wride, 1997, and others. These provided a curve that represented the boundary between liquefaction and nonliquefaction, usually estimated from engineering judgement to provide a conservative lower bound for occurrences of liquefaction (Juang et al., 1999). An example is provided in Figure 14, following.

To use these, an engineer first measures field CPT data and selects the value representative of the liquefiable layer. They then can use the solid CRR curve to determine the CSR required to mobilized liquefaction for their measured value. The ratio between this CSR and their design CSR is the factor of safety.

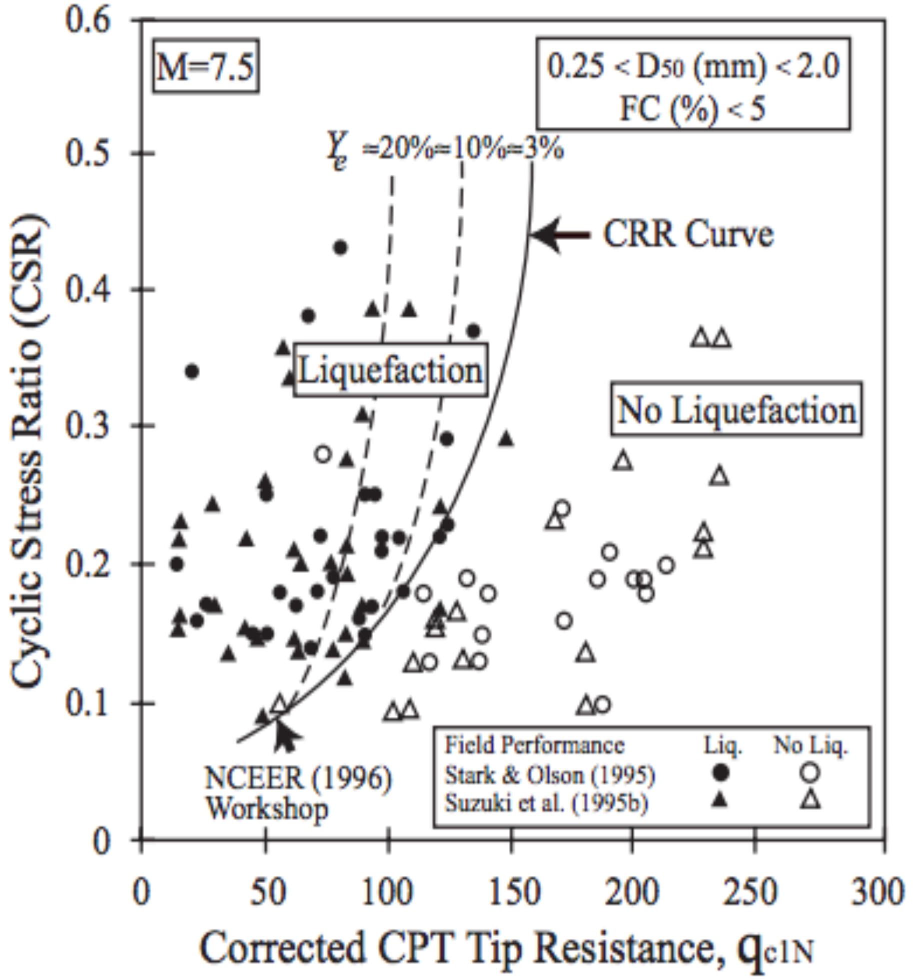


Figure 14 – A Sample Chart for Determining CRR from Corrected CPT Tip Resistance.
Reproduced from Robertson and Wride, 1997.

2.4.2 Early Probabilistic Models

Toprak et al., 1999 developed one of the first probabilistic CPT based triggering relationship by performing a logistic regression on case histories recorded after the Loma Prieta earthquake (Figure 15).

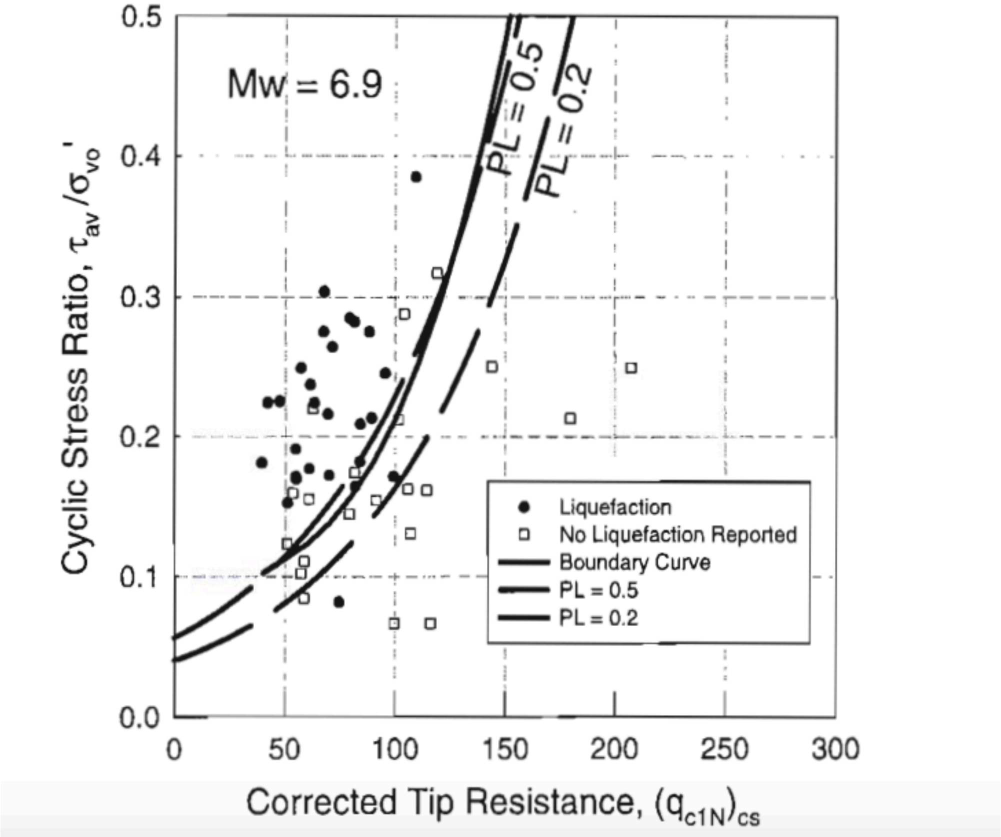


Figure 15 – One of the First Published Probabilistic CPT-based Liquefaction Triggering Relationships. Reproduced from Toprak and Bennet, 1999.

Logistic regression was actually first proposed by Liao et al., 1988 as a means of predicting liquefaction probability but was not applied to CPT case histories until this work. Because of the limited dataset, the regression is able to separate the classes fairly well but lacks generalization to predictions that fall outside the range of tip resistances and CSR's used to fit the model. Although limited to a relatively small number of cases and only a single earthquake it was still an important step in probabilistic liquefaction assessment and provided motivation for later research.

Juang et al., 2002 extended this work by applying the logistic regression to a much larger database to produce the next triggering relationship (Figure 16). An important aspect of these curves is how far apart or the spread the probability contours which represents a greater amount of model uncertainty.

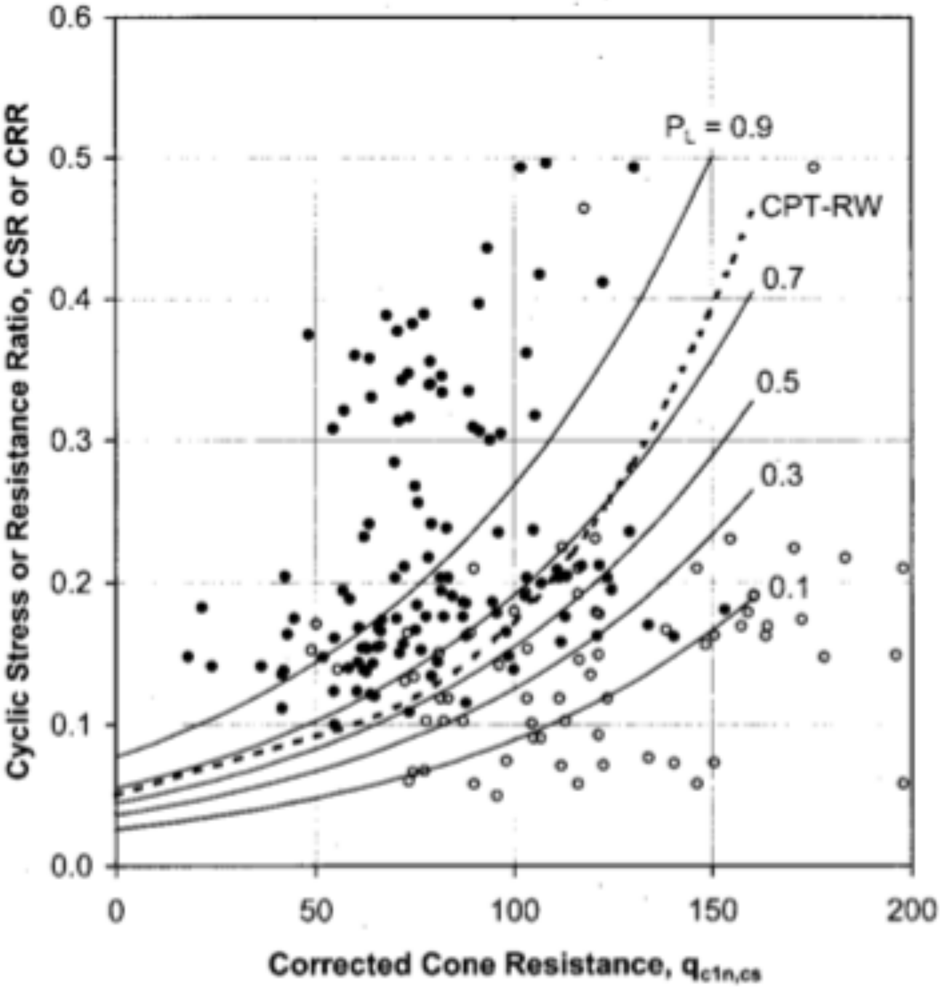


Figure 16 – An Updated Logistic Regression Model Using a Larger Set of Case Histories. Compared to previous models, it covers a wider spread of possible tip resistances and CSR's but has greater uncertainty in its predictions. Reproduced from Juang et al., 2002.

In work around the same time, researchers began to develop another method for modeling triggering relationships using reliability concepts. Juang et al., 1999 was among the first to develop probabilistic triggering models using these reliability methods. They selected a limit-state function representing the boundary between liquefaction and nonliquefaction of g(**x**) =

CSR/CRR – 1 = 0, where **x** is the vector of random variables that are transformed into CSR and CRR (Juang et al., 1999). To determine the probability distribution for CSR, they estimated standard deviations and assumed normal distributions for the relevant inputs for CSR formula to allow for an analytical solution for the transformed distribution. For the CRR, they used an artificial neural network to fit a curve similar to the earlier Robertson and Wride work and included a term to account for the uncertainty associated with this fit. Using an advanced first order second moment (FOSM) technique described in their paper they developed triggering relationships similar to the

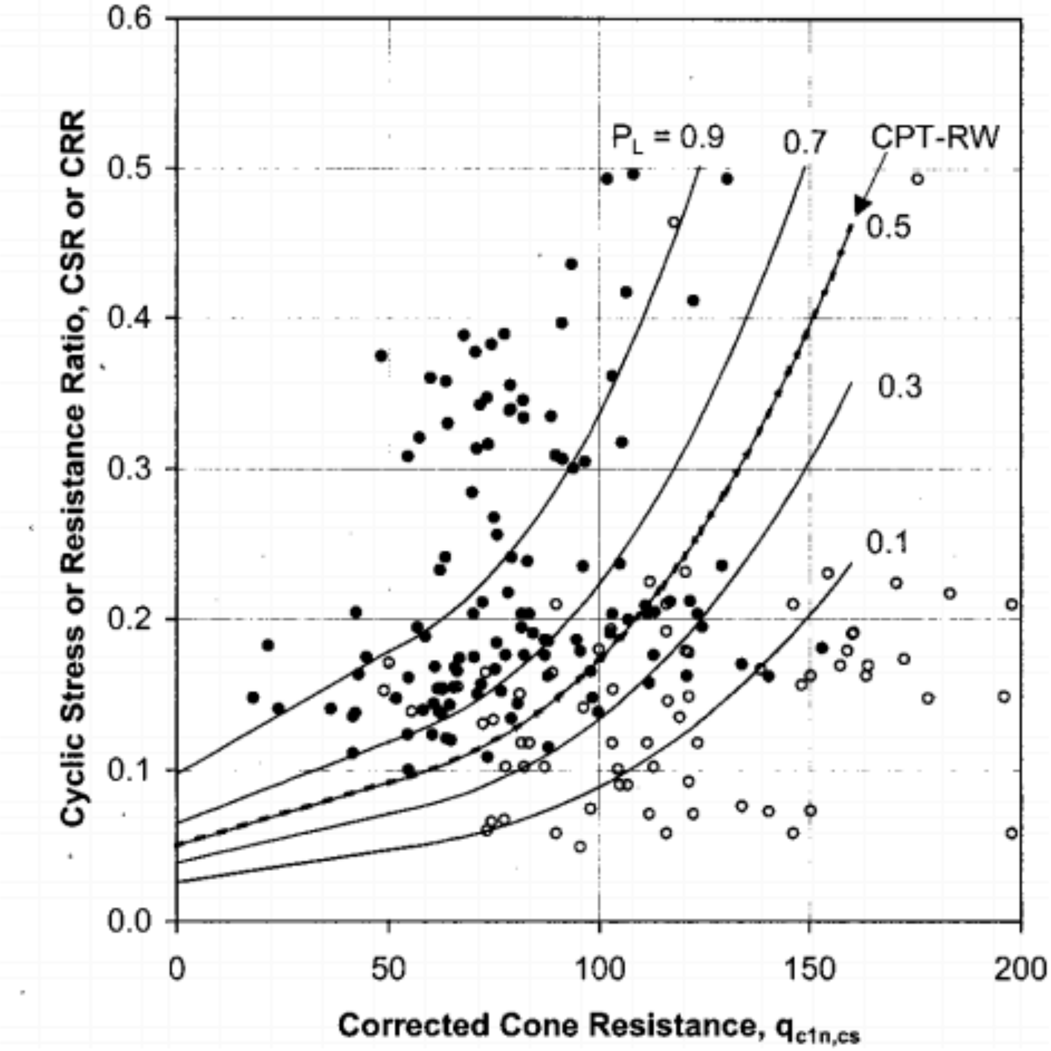


Figure 17 – The Reliability Based Bayesian Mapping Curves. Developed using the same dataset as the previous chart. Reproduced from Juang et al., 1999.

logistic regression, but with greater flexibility in the shape of the contours (Figure 17). Lai et al., 2006 extended this work using logistic regression models incorporating more case histories from earthquakes in Taiwan and China. Notably, they separated cases by CPT determined soil behavior type index (I_c) to account for the influence of fines content of liquefaction behavior

2.4.3 The Current State of Practice for CPT-based Probabilistic Models

The early 2000's saw improvements in both the size and quality of case history databases and correction factors applied to CSR or in-situ test measurements (discussed at length in Seed et al., 2003). The work of Moss et al., published in a Pacific Earthquake Engineering Research Center Report in 2003 and an ASCE journal article in 2006, was a step forward in CPT based triggering relationships and is one of the most used models in practice currently (NAE, 2016). They used a reliability-based formulation similar to the SPT-based modeling efforts of Cetin et al., 2002, employing a flexible limit-state formulation that used Bayesian updating to determine the posterior distributions of model parameters based upon the data. The general form of the limit-state function they selected, with predictor variables (**x**) and model parameters (**Θ**) was (after Moss et al., 2006):

$$\hat{g}(\boldsymbol{\theta},\boldsymbol{x}) = q_{c,1} * \left(1 + \theta_1 * R_f\right) + \left(\theta_2 * R_f\right) + c * \left(1 + \theta_3 * R_f\right) - \theta_4 * \ln(CSR) - \theta_5 * \ln(\mathrm{M_w}) - \theta_6 * \ln(\sigma_v') - \theta_7)$$

Where CSR is the simplified cyclic stress ratio calculated using correction factors described in the paper, M_w is the moment magnitude; σ_v' is the vertical effective stress; $q_{c,1}$ is the normalized CPT tip resistance; R_f is the CPT friction ratio, c is the CPT normalization exponent; and the Θ's are model parameters. They also reported that this form was chosen because it minimized the standard deviation of ε, minimized the cross correlations of predictor variables, and provided mathematical flexibility. As discussed in the previous chapter, this reporting of what metrics were used to judge the utility of changes to the model made during the modeling process (training) is lacking in other works.

To account for an imperfect formulation of the limit-state function, they also included an error term (ε) so that liquefaction and nonliquefaction states can be expressed as (after Moss et al., 2006):

$$\hat{g}(\boldsymbol{x},\boldsymbol{\theta}) + \epsilon \leq 0 \text{ and } \hat{g}(\boldsymbol{x},\boldsymbol{\theta}) + \epsilon > 0$$

If ε is taken to be normally distributed with mean 0 and standard deviation σ_ϵ and distributions are specified for the predictor variables for **x**, the likelihood of observing n_L liquefied cases and n_{NL} non-liquefied cases is (after Moss et al., 2006):

$$l(\boldsymbol{x},\boldsymbol{\theta}) \propto \prod_{i=1}^{n_l} \phi\left(-\frac{\hat{g}(\boldsymbol{x_i},\boldsymbol{\theta})}{\sigma_\epsilon}\right)^{0.8} * \prod_{j=1}^{n_{nl}} \phi\left(\frac{\hat{g}(\boldsymbol{x_j},\boldsymbol{\theta})}{\sigma_\epsilon}\right)^{1.2}$$

Where φ is the standard cumulative normal distribution function and the limit-state function is evaluated for the i^{th} instance of liquefaction and j^{th} instance of nonliquefaction using the appropriate values of predictor variables. Additionally, the included weighting terms to account for the imbalance between liquefied and nonliquefed case histories. Using the above likelihood and a non-informative prior they used importance sampling to solve for the posterior distributions of model parameters and the error standard deviation. They then used a mean value FOSM approach, validated by FORM and SORM methods, to carry out the reliability integral and develop the triggering relationships that follow (Figure 18). These represented a significant reduction in uncertainty over previous efforts while using improved correction factors and a better curated database (Moss et al., 2006).

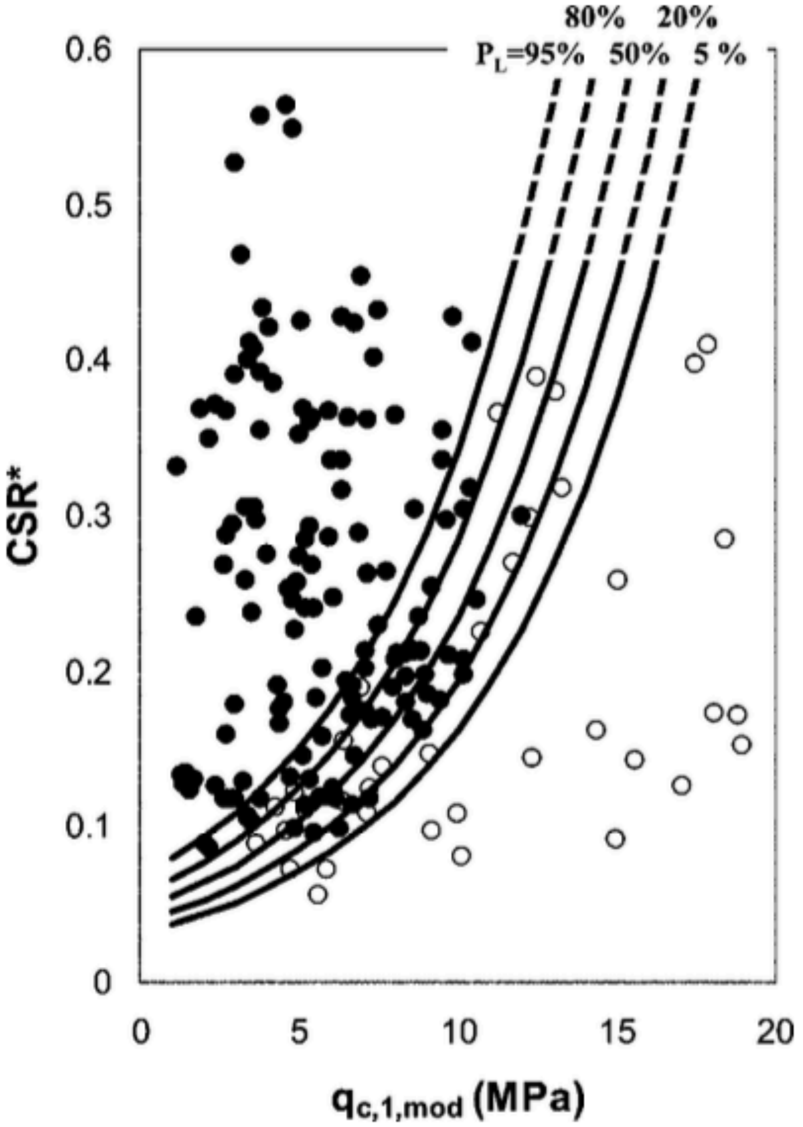


Figure 18 – The Current Standard of Practice in CPT Based Probabilistic Liquefaction Triggering. This model has appreciably less uncertainty than previous efforts while also incorporating substantial improvements in database curation and empirical correction factors. Reproduced from Moss et al., 2006.

2.4.4 Interim Models

In the years between the publication of the updated CPT database in 2003 and the completion of post-earthquake reconnaissance for the recent Chile, New Zealand, and Japan events most of the effort was focused on developing novel triggering relationships. These often used modern pattern recognition techniques such as support vector machines, artificial neural networks, or evolutionary polynomials developed in computer science and other predictive modeling fields. Examples of these works include: Goh and Goh, 2007, Hanna et. al. 2007, Rezania et. al., 2010 and 2011, Oomen et. al., 2010, and Yazdi et. al., 2012. Most of these focused on developing a deterministic separation between liquefaction and nonliquefaction cases and used older databases than the current work at the time. Because of this, they typically did not make their way into engineering practice and a detailed discussion is not particularly relevant in the scope of this study. However, many of them did use predictive modeling techniques not seen previously. Almost all used some form of splitting the data into training and testing sets – a critical step in avoiding bias associated with model validation metrics (Kuhn and Johnson, 2013).

Oomen et. al., 2010 was one of the only ones to develop a probabilistic model – though they did not provide equations or figures that engineers could readily use. Their paper did introduce new concepts regarding how probabilistic models are created and validated. They developed their model using k-fold cross validation to ensure the model used all of the available data, while reporting validation metrics that are not optimistically biased (Oomen et. al., 2010). Figure 19, below shows a comparison of the probabilistic performance of their new SVM model and the Moss et. al., 2006 model.

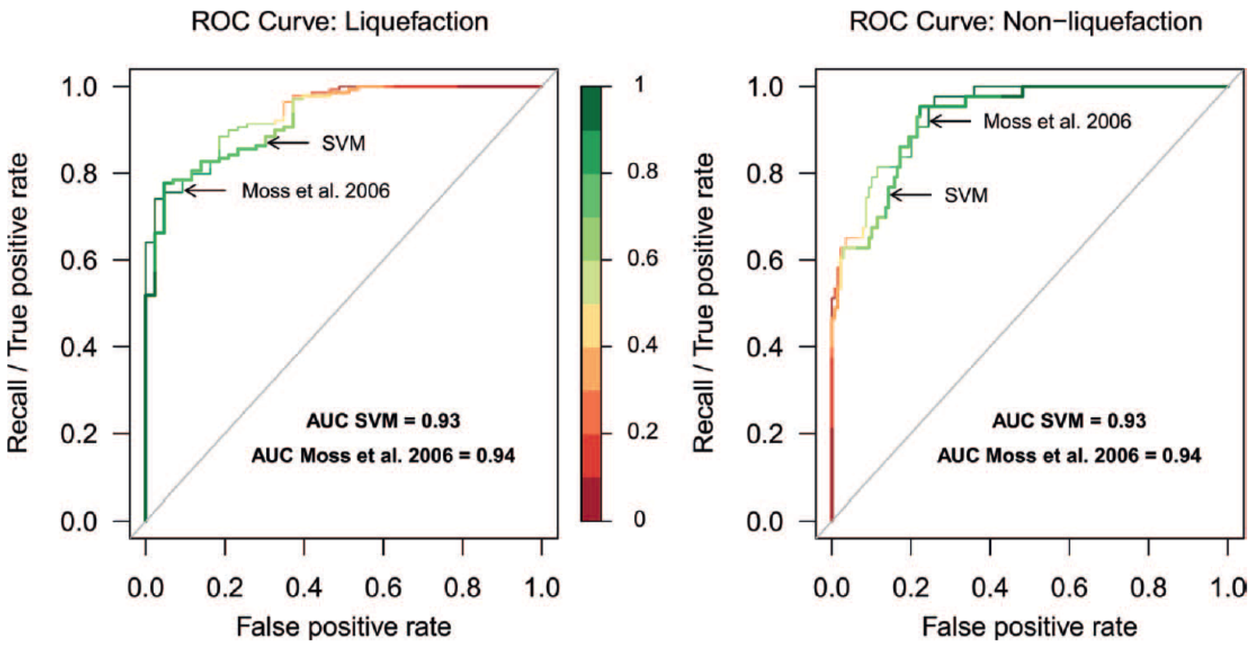


Figure 19 – A Comparison of the Performances of the Moss et al. and the Support Vector Machine Model. Both models perform fairly well, with AUC's above 0.9. Reproduced from Oomen et al., 2010.

Notably, the performance between the two is very similar, although the Moss et. al. work was developed and tested on the same data and its performance metrics are likely optimistically biased. This work also illustrates an important motivation for our study. Publishing performance metrics such as these is useful when practitioners are deciding on a threshold of unacceptable liquefaction risk and need to know the probability that the model will misclassify a liquefaction occurrence at that threshold.

In 2013, Zhang et al. extended research using logistic regression (and some other closely related models). They investigated the effects of using different link functions in the formulation of a generalized linear model and different strategies of weighting liquefaction/nonliquefaction cases. They observed that the effects of functional form choices were more prominent for small probabilities of liquefaction and when the CSR is high. They also found that the effect of sampling bias and methods for accounting for it are more noticeable in the regions of high CSR. Using a Bayesian model comparison, which develops posterior model probabilities in support of each model based upon their individual likelihoods and possible prior beliefs, they found the greatest support for the logit and complementary log-log link functions. However, the database they used for their study is not as thoroughly developed as the current standards of practice and their results are mainly useful for model building strategies.

2.4.5 Recent Probabilistic Models

In 2014, Idriss and Boulanger published a new probabilistic triggering model incorporating new case histories from the Canterbury earthquake sequence in New Zealand. Their paper also included modifications to the empirical correction factors. They used a similar reliability-based approach to Moss et. al., 2006 but employed a formulation of the limit-state function that only included a single model fitting parameter (After Boulanger and Idriss, 2016):

$$\hat{g}(q_{c1Ncs}, C_0, CSR) = \ln(CRR) - \ln(CSR)$$

$$CRR = exp\left[\frac{q_{c1Ncs}}{113} + \left(\frac{q_{c1Ncs}}{1000}\right)^2 - \left(\frac{q_{c1Ncs}}{140}\right)^3 + \left(\frac{q_{c1Ncs}}{137}\right)^4 - C_0\right]$$

Where CSR is calculated as normal using correction factors described in the paper, and the normalized clean sand equivalent cone penetration values $q_{c1N,cs}$ and unknown model parameter C_0 are used to calculate CRR. Similar to the Moss et. al. work they created error terms to incorporate uncertainty in measured predictor variables and account for imperfect model behavior. They assumed standard deviations for the normally error terms associated with CSR and $q_{c1N,cs}$ but left the standard deviation of the CRR error term to be estimated by the model. They combined all these uncertainties into a single model standard deviation, σ_t. To estimate the unknown model parameter C$_0$ and CRR relationship uncertainty $\epsilon_{\ln(r)}$ they used the following likelihood function (after Boulanger and Idriss, 2016):

$$l(\boldsymbol{x}, \boldsymbol{\theta}) \propto \prod_{i=1}^{n_l} \phi\left(-\frac{\hat{g}(q_{c1Ncs}, C_0, CSR)}{\sigma_t}\right)^{0.8} * \prod_{j=1}^{n_{nl}} \phi\left(\frac{\hat{g}(q_{c1Ncs}, C_0, CSR)}{\sigma_t}\right)^{1.2}$$

They used a maximum likelihood solution to develop a series of triggering relationships, with an example shown below (Figure 20). Their paper did not discuss the process they used to develop the CRR limit-state formulation, nor did it provide model validation metrics.

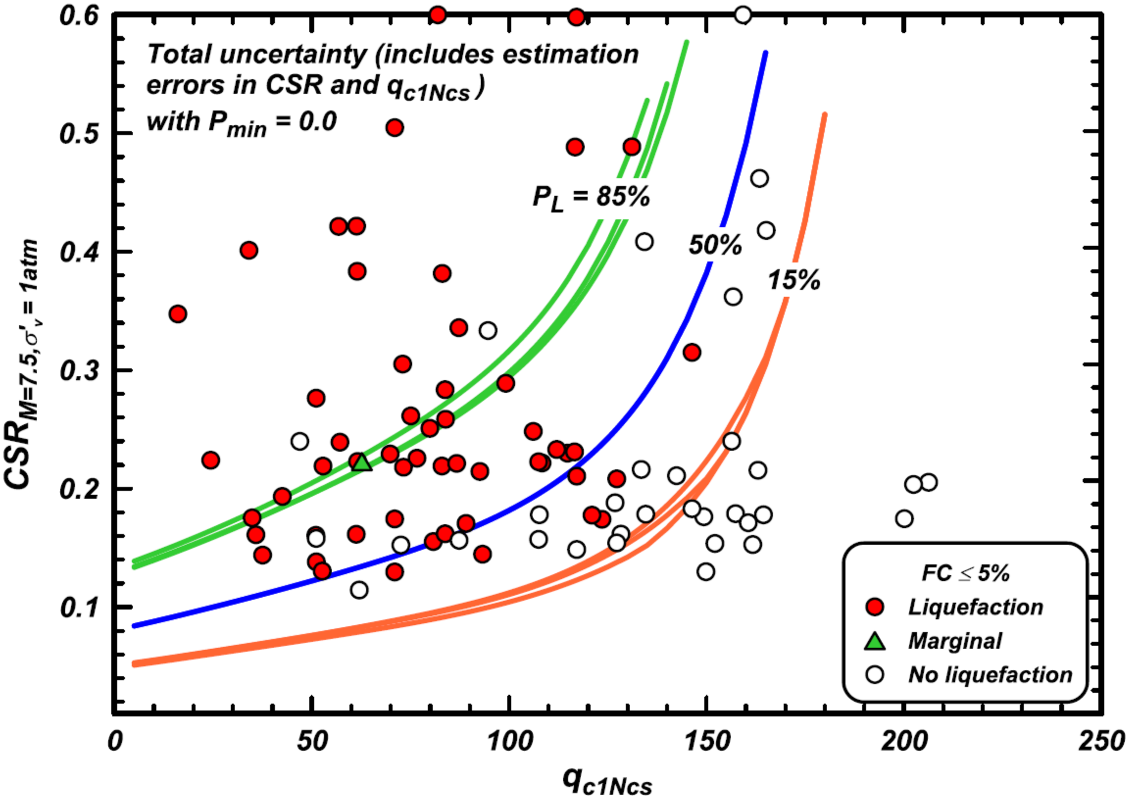


Figure 20 – The Most Recent Widely Used Probabilistic CPT Based Liquefaction Triggering Relationships. These use a simpler model functional form than the Moss et al. work but do incorporate a larger database (augmented by case histories from New Zealand and Japan). Reproduced from Idriss and Boulanger, 2016.

In 2016, Yazdi and Moss published the most current applicable probabilistic ELM (Figure 21). Their work was based a slightly updated version of the Moss et al., 2006 dataset (Yazdi and

Moss, 2016). They used a Bayes classification method, which expresses the probability of liquefaction given an event X as (after Yazdi and Moss, 2016):

$$p(L|X) = \frac{p(X|L)p(L)}{p(L)(X|L) + P(NL)P(X|NL)}$$

where P(L) and P(NL) are the prior probabilities of liquefaction and nonliquefaction, and P(X|L) and P(X|NL) are the likelihoods for liquefaction and nonliquefaction. To determine these likelihood functions instead of assuming a functional form (such as independent Bernoulli outcomes as with a logistic regression), they used a nonparametric approach employing a kernel density estimator to numerically estimate the likelihood functions (Yazdi and Moss, 2016). The kernel density estimation function they used has several tuning parameters, described in their paper, which were optimized for the likelihood shape and Matthew's correlation coefficient.

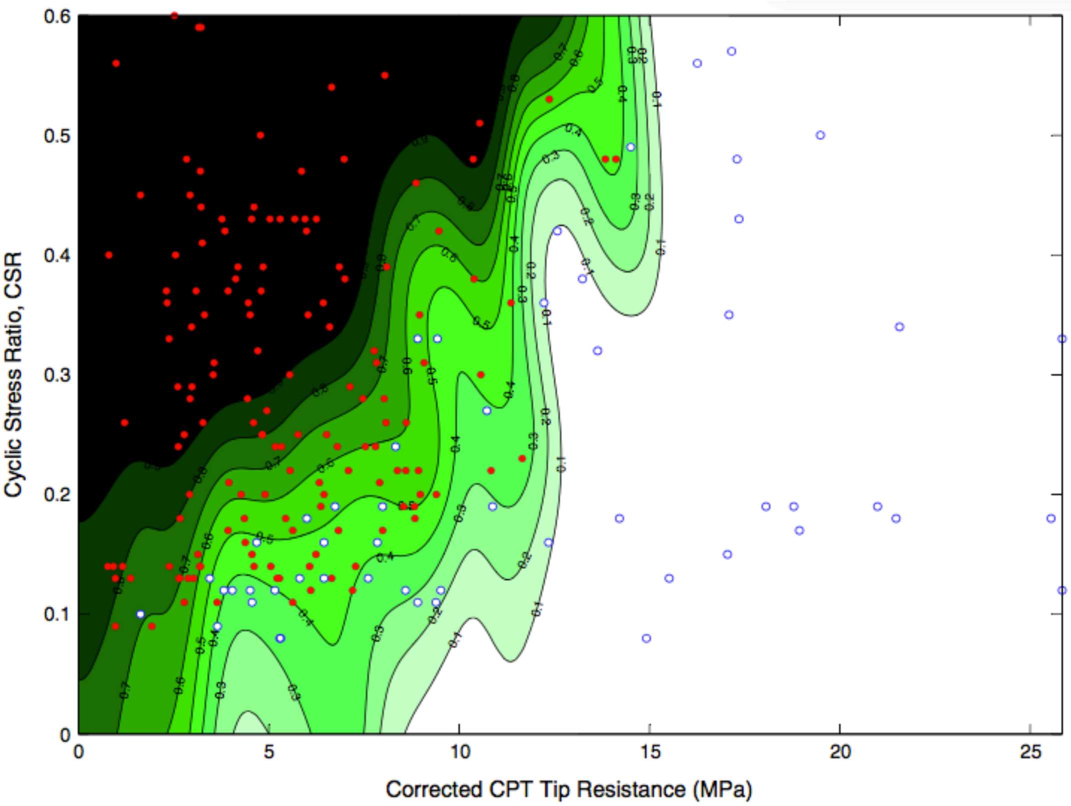
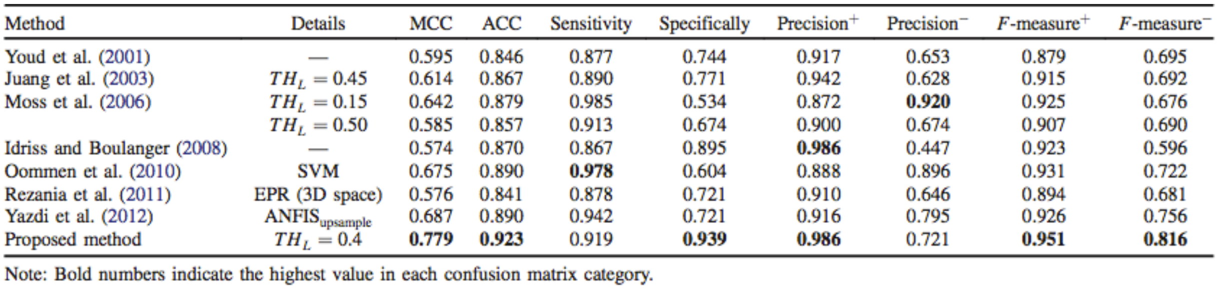


Figure 21 – The Nonparametric Triggering Curves from Yazdi and Moss, 2016.

With their estimated likelihood functions, they carried out the Bayes rule computation and produced the triggering curves.

The shape of the contours is dramatically more flexible than previous work, but follows the same general trend. This work demonstrates how modern sophisticated predictive models can capture highly nonlinear relationships in the data. They also developed model validation metrics for a single cutoff threshold and compared results to previous models. (Figure 22).



Method	Details	MCC	ACC	Sensitivity	Specifically	Precision$^+$	Precision$^-$	F-measure$^+$	F-measure$^-$
Youd et al. (2001)	—	0.595	0.846	0.877	0.744	0.917	0.653	0.879	0.695
Juang et al. (2003)	$TH_L = 0.45$	0.614	0.867	0.890	0.771	0.942	0.628	0.915	0.692
Moss et al. (2006)	$TH_L = 0.15$	0.642	0.879	0.985	0.534	0.872	**0.920**	0.925	0.676
	$TH_L = 0.50$	0.585	0.857	0.913	0.674	0.900	0.674	0.907	0.690
Idriss and Boulanger (2008)	—	0.574	0.870	0.867	0.895	**0.986**	0.447	0.923	0.596
Oommen et al. (2010)	SVM	0.675	0.890	**0.978**	0.604	0.888	0.896	0.931	0.722
Rezania et al. (2011)	EPR (3D space)	0.576	0.841	0.878	0.721	0.910	0.646	0.894	0.681
Yazdi et al. (2012)	$\text{ANFIS}_{\text{upsample}}$	0.687	0.890	0.942	0.721	0.916	0.795	0.926	0.756
Proposed method	$TH_L = 0.4$	**0.779**	**0.923**	0.919	**0.939**	**0.986**	0.721	**0.951**	**0.816**

Note: Bold numbers indicate the highest value in each confusion matrix category.

Figure 22 – Model Comparison of a Variety of ELM's. The metrics reported are for a single classification threshold listed for each model in the second column. Reproduced from Yazdi and Moss, 2016.

The Juang et al. and Idriss and Boulanger models are deterministic so they only have a single threshold curve. While the validation metrics have some degree of variation, the models generally perform comparably. All models were tested on the Moss et al., 2006 dataset which could arguably introduce bias in favor of the models built using it.

3 Model Building Process

The following chapter describes our model building process. As a general overview, we built our models following a predictive model building workflow outlined by Kuhn and Johnson (Figure 23).

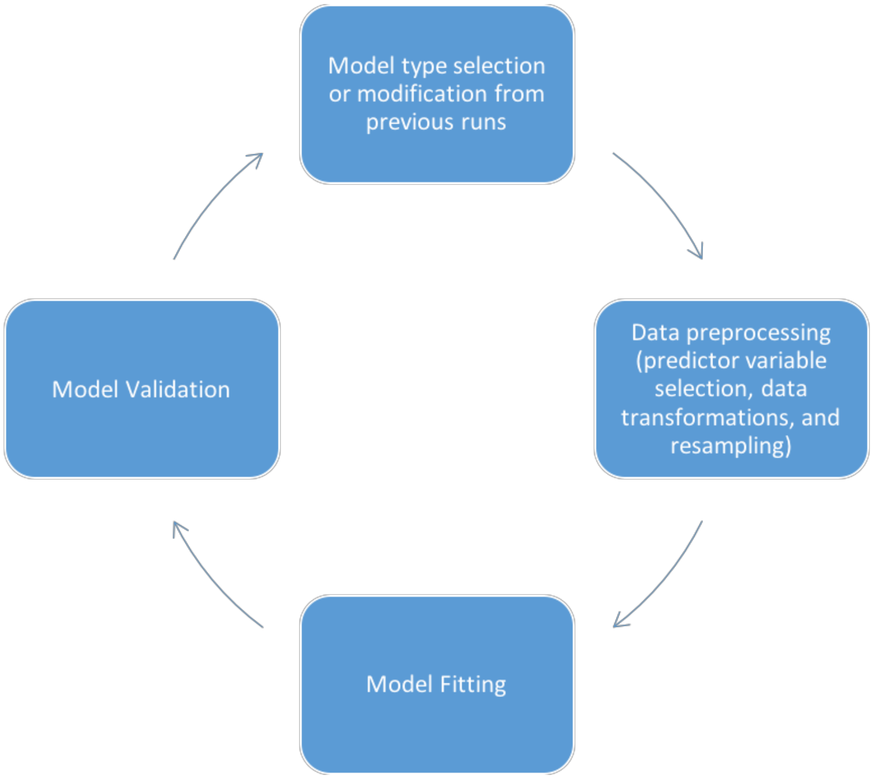

Model type selection or modification from previous runs
Data preprocessing (predictor variable selection, data transformations, and resampling)
Model Fitting
Model Validation

Figure 23 – Our Predictive Model Building Process.

We considered sequentially more complex functional forms using the results from simpler model runs to inform later computational choices (Figure 24). The following sections give a detailed description of these steps.

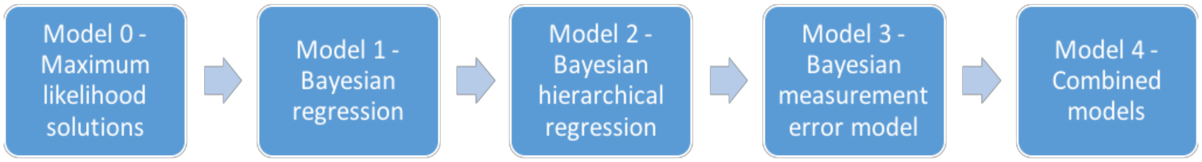

Model 0 - Maximum likelihood solutions
Model 1 - Bayesian regression
Model 2 - Bayesian hierarchical regression
Model 3 - Bayesian measurement error model
Model 4 - Combined models

Figure 24 – Model Complexity Progression

For this study, we used the R packages of caret, lme4, and ROCR for model development and validation, ggplot2 for visualization, rstan to interface with Stan (our MCMC Bayesian inference engine), and shinystan for visualizing MCMC diagnostics. The following code will check if the proper packages are installed:

```
require(caret)
## Loading required package: caret
## Loading required package: lattice
## Loading required package: ggplot2
require(ggplot2)
require(ROCR)
## Loading required package: ROCR
## Loading required package: gplots
##
## Attaching package: 'gplots'
## The following object is masked from 'package:stats':
##
##     lowess
require(lme4)
## Loading required package: lme4
## Loading required package: Matrix
require(rstan)
## Loading required package: rstan
## Loading required package: StanHeaders
## rstan (Version 2.18.2, GitRev: 2e1f913d3ca3)
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
require(shinystan)
## Loading required package: shinystan
## Loading required package: shiny
##
## This is shinystan version 2.5.0
require(caret)
```

If any packages are not installed or are out of date, call install.packages("package name") or update.packages().

3.1 Model Development – Preliminary steps

This section describes the steps taken before any models are fit, including visualizing the liquefaction database, predictor variable selection, and predictor variable transformation.

3.1.1 Importing Data

The models in this study were developed using the database from Moss et al., 2006. This database includes CPT data from 18 events, with 139 instances of liquefaction and 43 instances of no liquefaction (nonliquefaction). The original paper estimated mean values and standard deviations for σ_v', peak ground acceleration (a_{max}), CSR, $q_{c,1}$, R_f, and $M_{w.}$ The first step in R is to import this .csv database and store the values in a data frame. Note that the variable and file names/locations are arbitrary choices on the part of the programmer. To easily subset the data frame by predictor variable for later modeling, it is useful to re-name the columns as necessary.

```
setwd("/Users/appleuser/downloads/1-Thesis Stuff")
datan <- read.csv("mosDAt.csv")
datan$event <- as.factor(datan$event)
datan$liq <- as.factor(datan$liq)
head(datan)
##   liq CSR_mean CSR_sd qc1_mean qc1_sd rf_mean rf_sd  event
## 1 Yes     0.36   0.10     4.46   2.07    1.11  0.06 chichi
## 2 Yes     0.59   0.15     3.22   1.19    0.96  0.08 chichi
## 3 Yes     0.59   0.16     3.16   0.73    1.84  0.08 chichi
## 4 Yes     0.56   0.16     0.99   0.38    2.14  0.12 chichi
## 5 Yes     0.60   0.18     2.52   1.36    2.18  0.09 chichi
## 6 Yes     0.25   0.07     2.78   0.54    1.08  0.11 chichi
```

The database file also includes a label for liquefaction/nonliquefaction, and a label for event.

3.1.2 Database Overview

The original database included 12 predictor variables:

- Data class (A, B, or C), subjectively assigned based upon confidence in field data
- Critical depth: the depth range of the layer determined to have liquefied
- Groundwater table level: Depth below ground surface of the groundwater table
- Vertical total stress (σ_v)
- Vertical effective stress (σ_v')

- Peak ground acceleration (a_{max}), usually estimated indirectly from attenuation relationships
- Shear stress reduction coefficient (r_d) used to calculate CSR
- Cyclic stress ratio (CSR)
- CPT normalization exponent (c), an input to the equation for normalizing CPT measured tip resistance
- Normalized CPT tip resistance ($q_{c,1}$)
- Friction ratio (r_f): the CPT measured sleeve friction divided by the penetration resistance
- Moment magnitude (M_w)

Importantly many of these predictor variables are functions of each other and are correlated which will become an issue during the modeling process (Figure 25).

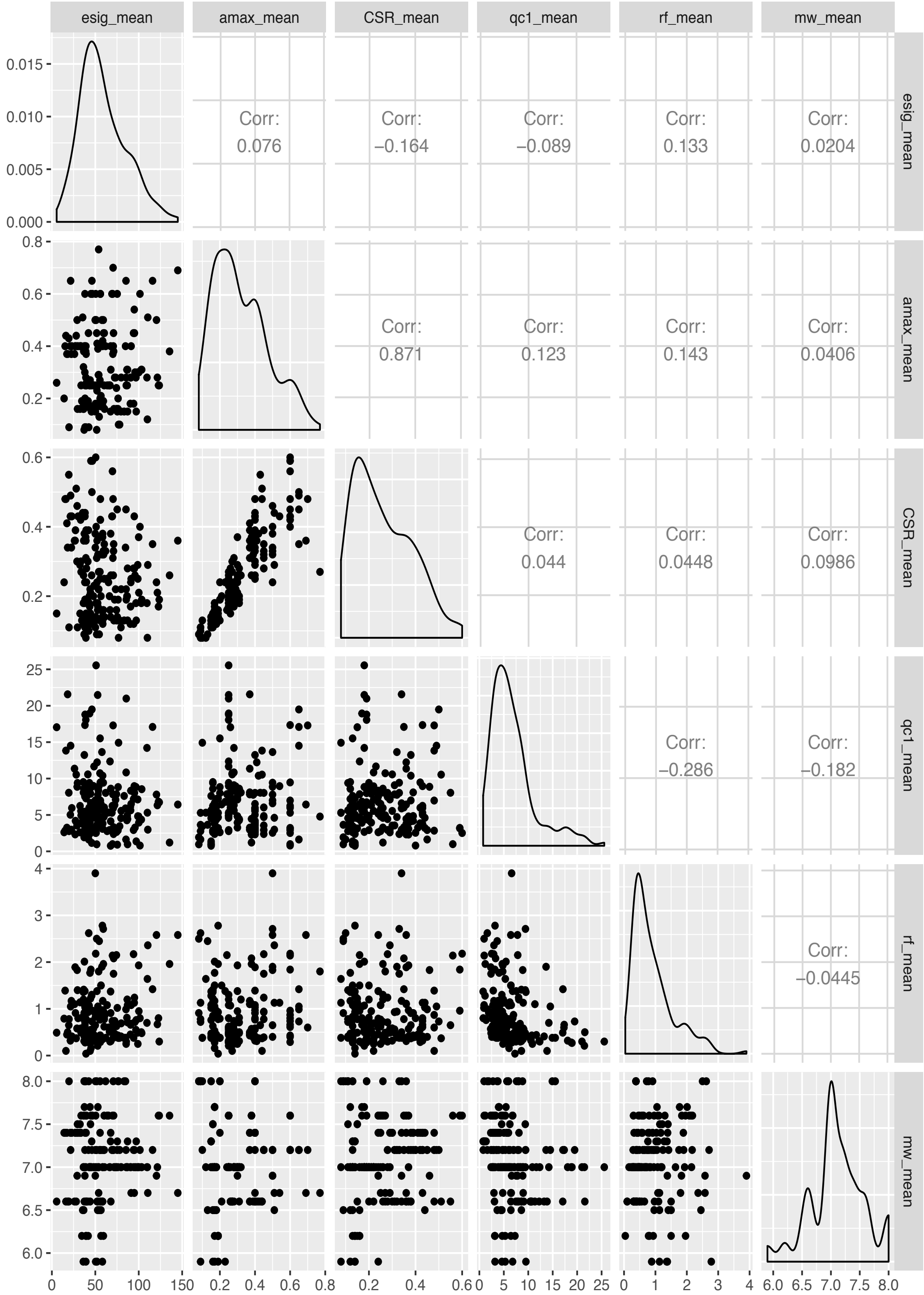


Figure 25 – The Correlation Matrix of the Database Mean Values. The diagonal shows an estimated distribution for each predictor which are plotted against each other in the lower triangle. The upper triangle reports their correlations.

To illustrate the spread of data, we first focus on a single load and resistance predictor — CSR and $q_{c,1}$. The database includes a reasonably wide range of CSR and $q_{c,1}$ mean values (Figure 26). In both cases, the data are left skewed and have a moderately high coefficient of variation ($\frac{\mu}{\sigma} \sim 0.5 - 0.7$). Additionally, Figures 27 and 28 show that each event has a slightly different distribution of load and resistance values. There is a noticeable association between event and CSR due to certain earthquakes having higher moment magnitudes and associated higher ground motions than others. Unlike CSR there is not a noticeable association between event and $q_{c,1}$. Finally, Figure 29 shows the separability between instances of liquefaction and nonliquefaction for the three predictor variables considered. Although here is no clear separation between the classes liquefaction is generally associated with lower penetration resistance and higher CSR. R_f does not show any clear trends.

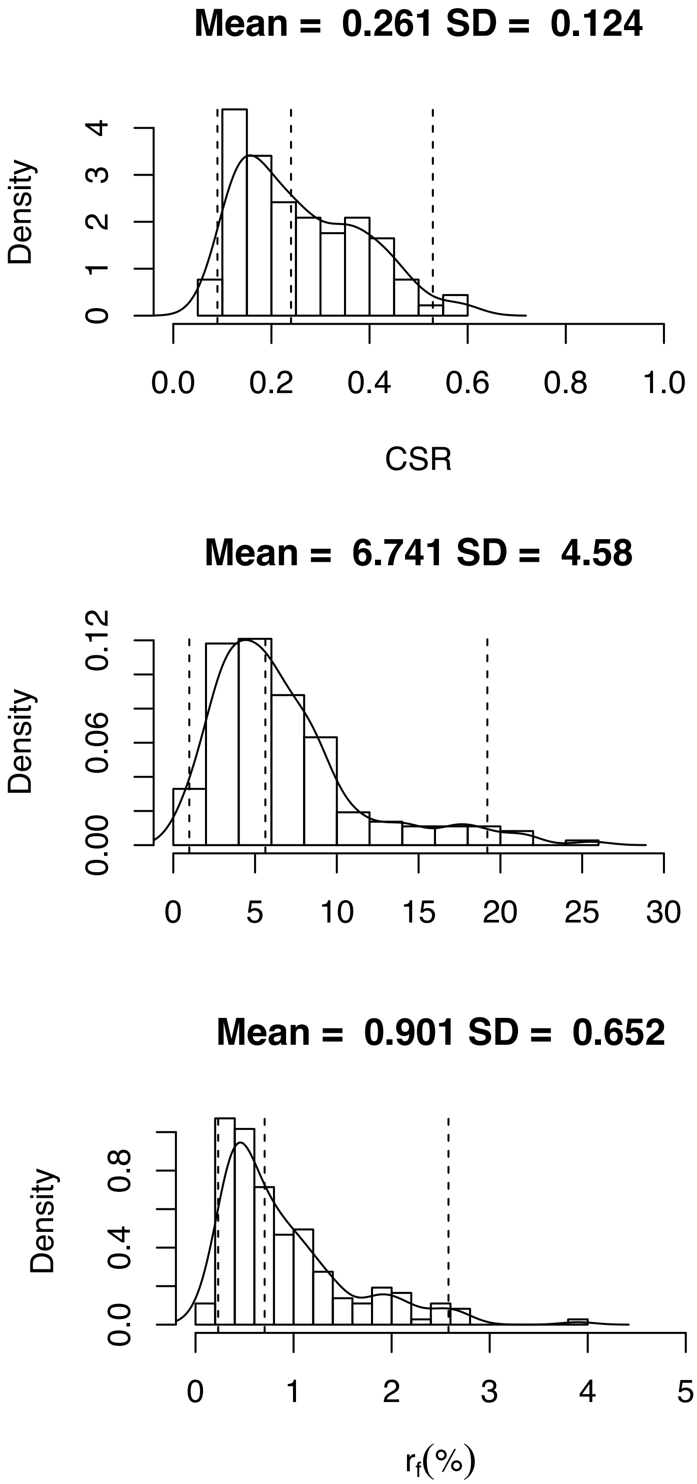


Figure 26 – Histogram and Density Approximation of the Three Predictor Variables. Including mean value and standard deviation. The dashed lines represent the 2.5th, 50th (median), and 97.5th percentiles, respectively.

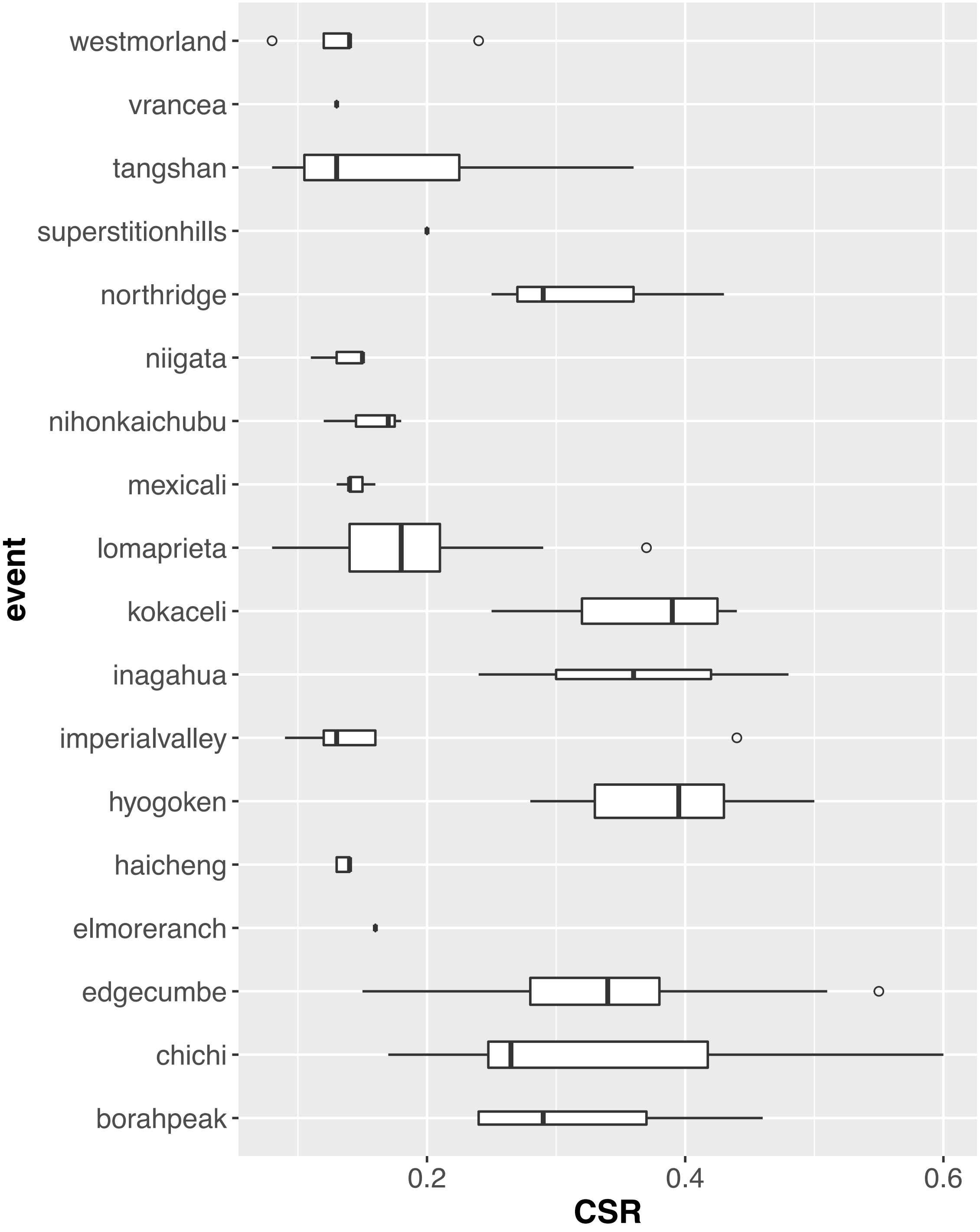
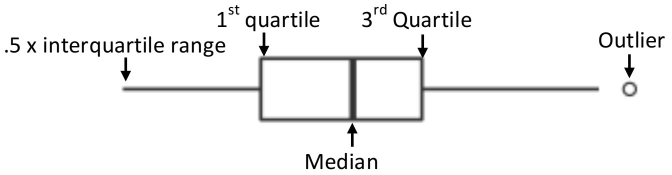


Figure 27 – Distribution of CSR by Event. The usual box-and-whiskers plot conventions are used:



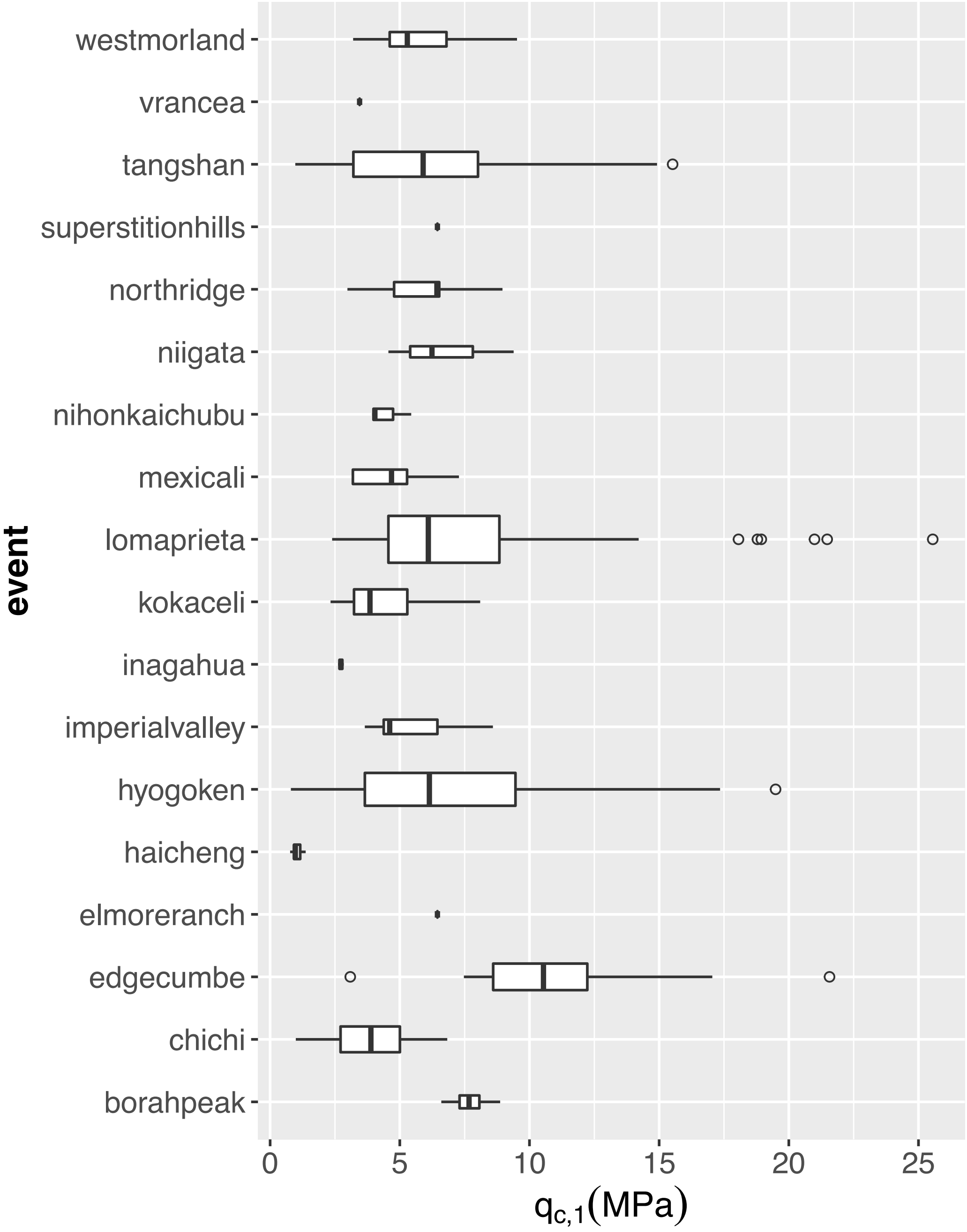


Figure 28 – Distribution of $q_{c,1}$ by Event. The usual box-and-whiskers plot conventions are used.

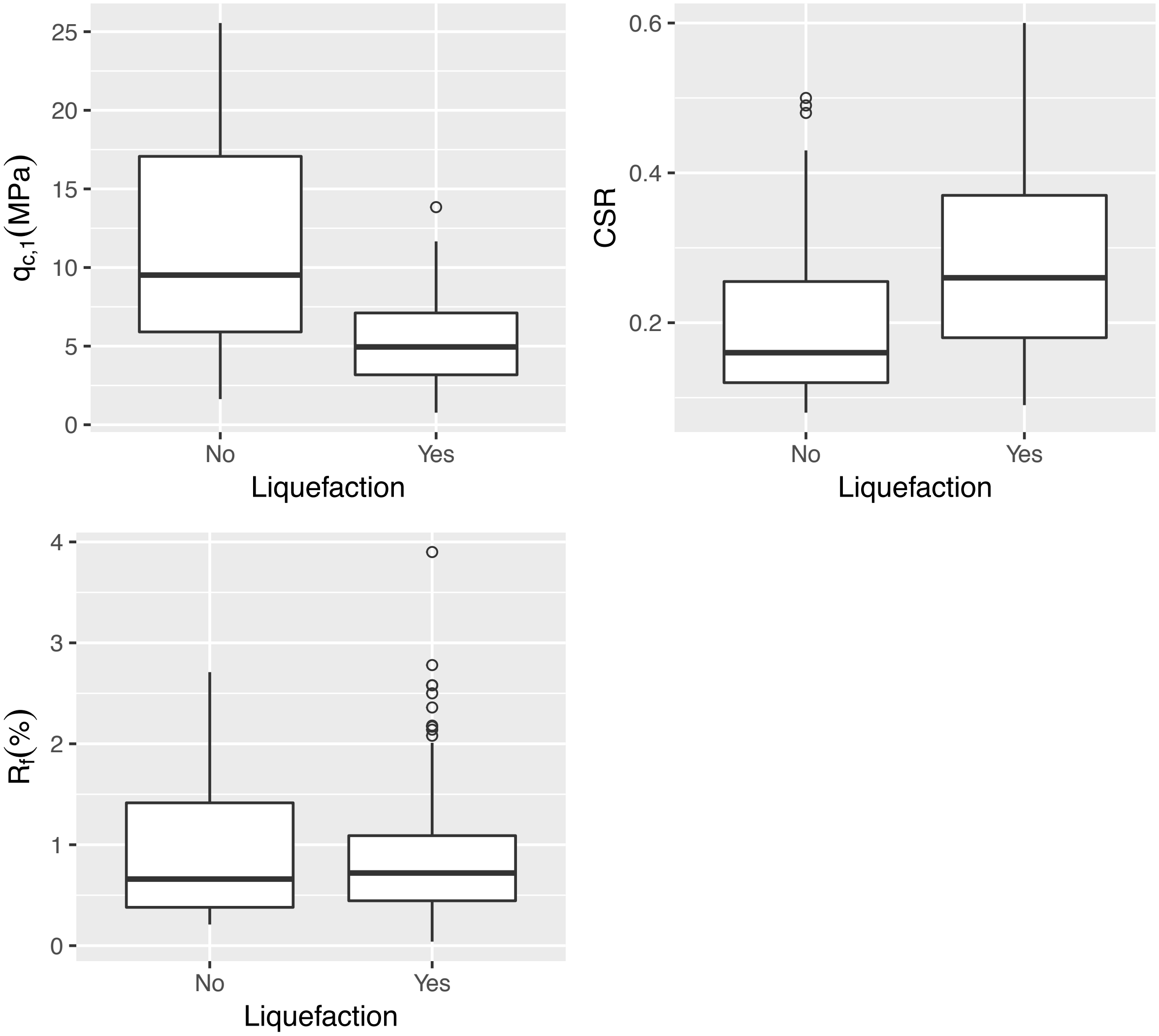


Figure 29 – Class Separability Between $q_{c,1}$, CSR, and R_f. The usual box-and-whiskers plot conventions are used.

3.1.3 Predictor Variable Selection

Predictor variable selection is a critical step in any model development—too few and the relationships in the data may not be captured, too many and the model will be over fit (Kuhn and Johnson, 2013). In particular, including predictor variables that have little or no effect on predicted outcomes will increase model uncertainty and make the model more difficult both to fit and interpret (Kuhn and Johnson, 2013). For engineering research, the utility of a model is also important – utility in this context meaning the balance between a model's predictive ability and how useable it is in practice. An overly complicated model may be confusing or require too much time and money invested in software or training for practitioner's use.

As a preliminary tool, we used a stepwise selection process to determine which predictor variables were worth including based upon the Akaike Information Criterion (AIC). AIC is a metric for making relative comparisons about model utility that estimates the tradeoff between model goodness of fit and the simplicity of the model, conceptually the tradeoff between under and over fitting (Burnham and Anderson, 2004). A lower value for AIC is better. At each step in the algorithm, the predictor variables are added or removed one by one from the model and the AIC calculated. The model with the lowest AIC is selected for the next step and the process continues until no proposed model outperforms the current.

This stepwise selection process is performed with the following code:

```
data <- read.csv("mosDATn.csv")
initial <- glm(liq ~ GWT + sig_mean + amax_mean + rd_mean + CSR_mean + qc1_mean +
CSR_mean + rf_mean + mw_mean, data = data, family = "binomial")
step(object = initial, direction = "both" )
## Start:  AIC=109.28
## liq ~ GWT + sig_mean + amax_mean + rd_mean + CSR_mean + qc1_mean +
##     CSR_mean + rf_mean + mw_mean
##
##             Df Deviance    AIC
## - amax_mean  1   91.328 107.33
## - mw_mean    1   91.725 107.72
## - GWT        1   92.397 108.40
## - rd_mean    1   92.990 108.99
## <none>           91.282 109.28
## - sig_mean   1   93.471 109.47
## - CSR_mean   1   96.593 112.59
## - rf_mean    1  105.119 121.12
## - qc1_mean   1  177.472 193.47
## 
## Step:  AIC=107.33
## liq ~ GWT + sig_mean + rd_mean + CSR_mean + qc1_mean + rf_mean + 
##     mw_mean
## 
##             Df Deviance    AIC
## - mw_mean    1   91.725 105.72
## - GWT        1   93.012 107.01
## - rd_mean    1   93.175 107.17
## <none>           91.328 107.33
## - sig_mean   1   93.653 107.65
## + amax_mean  1   91.282 109.28
## - rf_mean    1  107.280 121.28
## - CSR_mean   1  121.268 135.27
## - qc1_mean   1  181.105 195.10
## 
## Step:  AIC=105.72
## liq ~ GWT + sig_mean + rd_mean + CSR_mean + qc1_mean + rf_mean
## 
##             Df Deviance    AIC
## - GWT        1   93.151 105.15
## - rd_mean    1   93.247 105.25
## - sig_mean   1   93.671 105.67
## <none>           91.725 105.72
## + mw_mean    1   91.328 107.33
## + amax_mean  1   91.725 107.72
## - rf_mean    1  107.888 119.89
## - CSR_mean   1  122.210 134.21
## - qc1_mean   1  181.952 193.95
## 
## Step:  AIC=105.15
## liq ~ sig_mean + rd_mean + CSR_mean + qc1_mean + rf_mean
## 
##             Df Deviance    AIC
## - sig_mean   1   94.341 104.34
## - rd_mean    1   94.447 104.45
## <none>           93.151 105.15
## + GWT        1   91.725 105.72
## + amax_mean  1   92.490 106.49
## + mw_mean    1   93.012 107.01
## - rf_mean    1  108.133 118.13
## - CSR_mean   1  122.335 132.34
## - qc1_mean   1  184.774 194.77
##
## Step:  AIC=104.34
## liq ~ rd_mean + CSR_mean + qc1_mean + rf_mean
##
##             Df Deviance    AIC
## - rd_mean    1   94.507 102.51
## <none>           94.341 104.34
## + sig_mean   1   93.151 105.15
## + amax_mean  1   93.601 105.60
## + GWT        1   93.671 105.67
## + mw_mean    1   94.337 106.34
## - rf_mean    1  108.608 116.61
## - CSR_mean   1  130.115 138.12
## - qc1_mean   1  184.984 192.98
##
## Step:  AIC=102.51
## liq ~ CSR_mean + qc1_mean + rf_mean
##
##             Df Deviance    AIC
## <none>           94.507 102.51
## + amax_mean  1   93.604 103.60
## + GWT        1   93.684 103.68
## + rd_mean    1   94.341 104.34
## + sig_mean   1   94.447 104.45
## + mw_mean    1   94.504 104.50
## - rf_mean    1  108.995 115.00
## - CSR_mean   1  133.367 139.37
## - qc1_mean   1  186.074 192.07
##
## Call:  glm(formula = liq ~ CSR_mean + qc1_mean + rf_mean, family = "binomial",
##     data = data)
##
## Coefficients:
## (Intercept)     CSR_mean     qc1_mean      rf_mean
##      3.6273      20.0723      -0.7533      -1.3836
##
## Degrees of Freedom: 181 Total (i.e. Null);  178 Residual
## Null Deviance:       199
## Residual Deviance: 94.51     AIC: 102.5
```

Based upon these results there is justification for considering models of three predictor variables: $q_{c,1}$, CSR, and R_f. Qualitatively, these predictor variables have a reasonable coverage of several main factors affecting liquefaction: in-situ density, amplitude and duration of cyclic shearing, and fines content. In the absence of laboratory testing R_f serves as a proxy for the fines content because more cohesive soils will tend to have higher values. We fit all models first with only two predictor variables, $q_{c,1}$ and CSR, then included R_f to assess the value it added to the model. For brevity, all sample code that follows is for the more complicated three variable case. It is relatively simple to delete code pertaining to R_f to recover the two variable case.

3.1.4 Predictor Variable Transformations

As mentioned previously, model performance is improved by dealing with transformations of predictor variables. In this area previous research has focused on satisfying certain statistical assumptions such as normality of predictor variables or linear independence under the logit transformation (Lai et al., 2006). However, we instead searched for transformations that produced the model with the best predictive ability given a fixed set of predictors. We selected the Box-Cox family of transformations because of its flexibility and its ability to capture many common transformations such as powers and logarithms. Another useful property is that these transformations are monotonic so increases or decreases in the original variable also correspond to increases and decreases in the transformed variable. A Box-Cox transformation of a predictor variable x, indexed by the parameter λ, is defined as (after Box and Cox, 1964):

$$x' = \begin{cases} \ln(x) & \text{if } \lambda = 0 \\ \dfrac{x^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0 \end{cases}$$

We used a simple grid search method to determine the group $\lambda's$ that produced the best performing model, as measured by 5-fold cross validated AUC. Conceptually, this introduces a tuning parameter to the standard maximum likelihood logistic regression that allows for greater flexibility in the shape of the probability contours. Since the interaction between predictors can potentially be different for two and three variables the process should be repeated for both cases (only one of which is shown below).

The code below performs a grid search for all possible cases of Box-Cox transformation indices for the $q_{c,1}$, CSR, and R_f. At each grouping of possible predictor variable transformations the caret package is used to perform a logistic regression and compute its 5 fold cross validated AUC using the train function. The train function takes several arguments. The first is the model formula, expressed identically to the glm call as response ~ predictor + predictor + ... and family = “binomial” to indicate that we have binary outcomes for our response. The method argument tells train that we want to use a generalized linear model. The trainControl function returns a list that tells the train function that our validation methods are 5-fold cross validation and we want it to report the two class summary (which includes the AUC).

```
set.seed(5)
options(warn=-1)
fitControl <- trainControl(method = "cv",
                           number = 5,
                           classProbs = T,
                           summaryFunction = twoClassSummary)
dtemp <- datan
search_grid <- expand.grid(seq(-1,1,0.1) , seq(-1,1,0.1), seq(-1,1,0.1) )
iter <- length(search_grid$Var1)
results <- data.frame(matrix(ncol = 8, nrow = iter))
names(results) <- c("b0", "b1", "b2", "b3", "AUC", "l_CSR", "l_qc1", "l_rf")
for (i in 1:iter){
if (search_grid[i,1] == 0) {
  dtemp$CSR_mean <- log(datan$CSR_mean)
} else {
  dtemp$CSR_mean <- (datan$CSR_mean^(search_grid[i,1])-1)/(search_grid[i,1])
}
if (search_grid[i,2] == 0) {
  dtemp$qc1_mean <- log(datan$qc1_mean)
} else {
  dtemp$qc1_mean <- (datan$qc1_mean^search_grid[i,2]-1)/(search_grid[i,2])
}
if (search_grid[i,3] == 0) {
  dtemp$rf_mean <- log(datan$rf_mean)
} else {
  dtemp$rf_mean <- (datan$rf_mean^search_grid[i,3]-1)/(search_grid[i,3])
}
m <- train(liq ~ qc1_mean + CSR_mean + rf_mean, data = dtemp, family = "binomial",
        method = "glm",
        trControl = fitControl)
results[i,5] <- m$results$ROC
results[i,1:4] <-m$finalModel$coefficients
results[i,6:8] <- search_grid[i,]
}
newres <- results[order(results$AUC),]
write.csv(newres, "transformations.csv")
```

The results are then ranked by their cross validated AUC and stored in a .csv file for later use. We found the optimal Box-Cox parameters to be $\lambda_{CSR} = -0.6$ and $\lambda_{q_{c,1}} = 1.6$ for the two variable case and $\lambda_{CSR} = -0.6$, $\lambda_{q_{c,1}} = 1.0$, and $\lambda_{R_f} = 0.2$ for the three variable case. Importantly, these transformations do not result from physical principles nor do they have a meaningful physical interpretation. Rather, they are a result of how a logistic regression separates classes. A logistic regression is only capable of linear class boundaries, so this transformation tuning procedure can be thought of finding the transformed predictor variable space in which the liquefaction classes are most nearly linearly separable. As would be expected, these transformations cause an appreciable increase in model predictive ability. Considering the two variable case, the AUC rises from 0.664 to 0.710 when applying the transformations.

Because only the mean and standard deviation of the predictor variables are included in the database, the transformed moments cannot be calculated directly for a nonlinear transformation without assuming a distributional form for each data point. Instead, we use a first order second moment (FOSM) approximation technique that calculates the moments of the Taylor series expansion of the transformation (See Moss 2013 for a detailed derivation). Specifically, for a random variable X (in our case a predictor variable measurement) with associated mean μ and variance σ^2 and differentiable transformation Y = g(x) the FOSM approximations are given by (after Moss, 2013):

$$\mu_y \cong g(\mu_x)$$

$$\sigma_y^2 \cong g'(\mu_x)^2 \sigma_x^2$$

$$\frac{d}{dx}\left[\frac{x^{\lambda}-1}{\lambda}\right]=x^{\lambda-1}$$

For a $q_{c,1}$ observation transformed to $q_{c,1,T}$ via a Box-Cos parameter λ we have:

$$\mu_{q_{c,1,T}} \cong \frac{{\mu_{q_{c,1}}}^{\lambda}-1}{\lambda}$$

$$\sigma^2_{q_{c,1,T}} \cong \left({\mu_{q_{c,1}}}^{\lambda-1}\right)^2\sigma^2_{q_{c,1}}$$

Likewise, for a CSR observation transformed to CSR_T we have:

$$\mu_{CSR,T} \cong \frac{{\mu_{CSR}}^{\lambda}-1}{\lambda}$$

$$\sigma^2_{CSR,T} \cong \left({\mu_{CSR}}^{\lambda-1}\right)^2\sigma^2_{CSR}$$

and for R_f:

$$\mu_{R_f,T} \cong \frac{{\mu_{R_f}}^{\lambda}-1}{\lambda}$$

$$\sigma^2_{R_f,T} \cong \left({\mu_{R_f}}^{\lambda-1}\right)^2\sigma^2_{R_f}$$

Next, the FOSM approximations of the mean values and standard deviations are computed in R. The Box-Cox transformation and its derivative are written as functions to be easily called in the main section of the code without having to copy and paste the same math over and over. The following examples are shown for all three predictor variables but the code should be modified as necessary for the two variable case.

```
boxcox <- function(x,l){
 if (l == 0) {
  return(log(x))
 } else {
  return((x^(l)-1)/l)
 }
}

dboxcox <- function(x,l){
 if (l == 0) {
    return(1/x)
  } else {
    return(x^(l-1))
  }
}

l_csr <--0.6
l_qc1 <- 1.0
l_rf <- 0.2
dt1 <- data.frame(datan$liq,
                boxcox(datan$CSR_mean, l_csr),
                sqrt(dboxcox(datan$CSR_mean, l_csr)*datan$CSR_sd^2),
                boxcox(datan$qc1_mean, l_qc1),sqrt(dboxcox(datan$qc1_mean,
l_qc1)*datan$qc1_sd^2),
                boxcox(datan$rf_mean, l_rf),
                sqrt(dboxcox(datan$rf_mean, l_rf)*datan$rf_sd^2),
                datan$event)
names(dt1) <- names(datan)
```

The effects of these transformations are shown in figure 30, following.

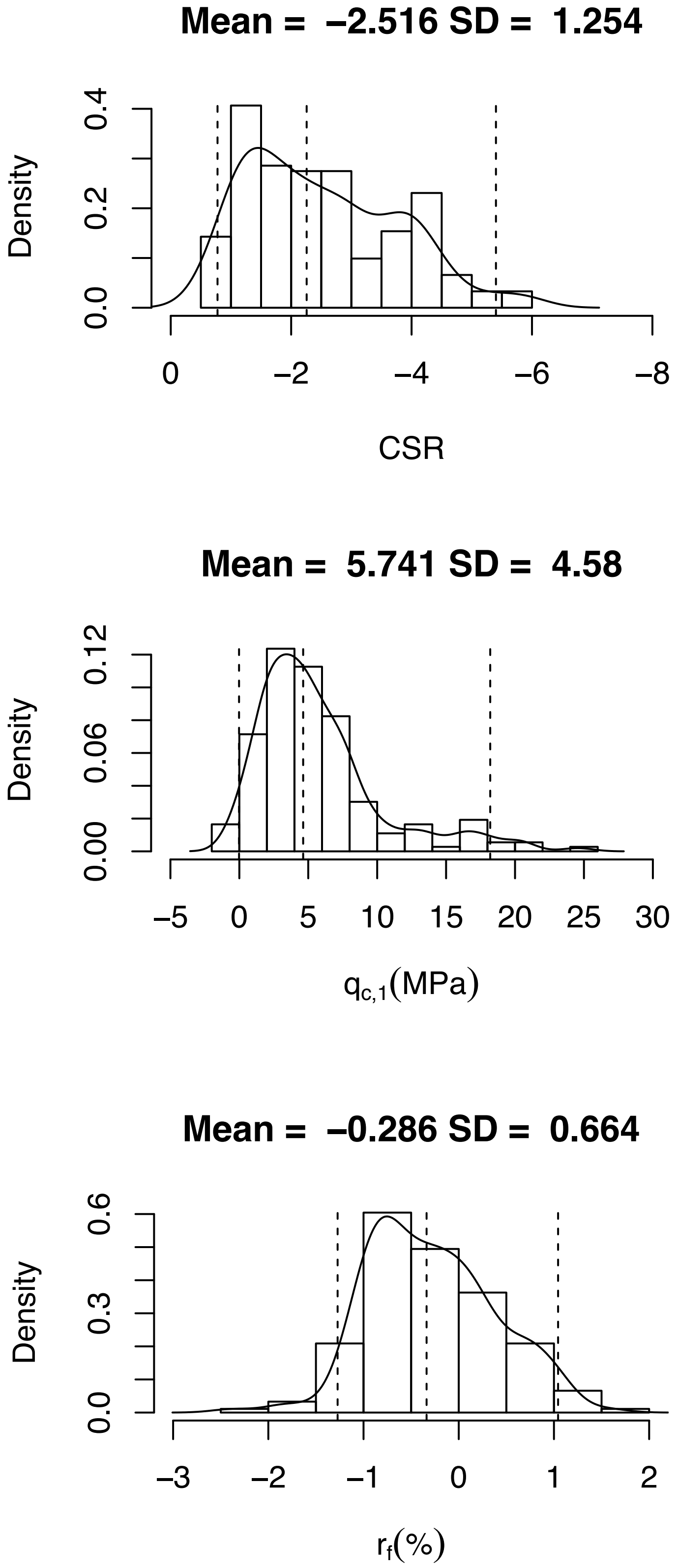


Figure 30 – Transformed Distributions of $q_{c,1}$, CSR, and R_f. The Box-Cox parameters of $\boldsymbol{\lambda_{CSR} = -0.6}$, $\boldsymbol{\lambda_{q_{c,1}} = 1.0}$, and $\boldsymbol{\lambda_{R_f} = 0.2}$ for the three variable case are shown**.**

3.2 Model development - Maximum Likelihood Fits

This section details our maximum likelihood models and the R code used to estimate parameter values. It covers the baseline model, using case weighting and up-sampling to account for class imbalance, and the mixed effects model.

3.2.1 Initial Maximum Likelihood Model

First, we use the glm function to run our initial maximum likelihood logistic regression on the standard dataset. The function call includes the regression formula of the form response ~ predictor + predictor + ..., the data frame where the data are stored, and family of generalized linear model to be used. In this case, family = “binomial” tells R that we have binary outcomes for our responses. R defaults to the logit link function but includes others.

```
m <- glm(liq ~ qc1_mean + CSR_mean + rf_mean, data = dt1, family = "binomial")
```

This stores the model results as list, m. To view a summary of model output:

```
summary(m)
##
## Call:
## glm(formula = liq ~ qc1_mean + CSR_mean + rf_mean, family = "binomial",
##     data = dt1)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.87698  -0.32937   0.00867   0.36300   2.09253
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  10.7657     1.5519   6.937 4.00e-12 ***
## qc1_mean     -0.8164     0.1187  -6.879 6.01e-12 ***
## CSR_mean      2.0875     0.3146   6.636 3.23e-11 ***
## rf_mean      -1.3206     0.3098  -4.263 2.02e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 385.39  on 277  degrees of freedom
## Residual deviance: 167.71  on 274  degrees of freedom
## AIC: 175.71
```

```
##
## Number of Fisher Scoring iterations: 7
```

This output includes many useful diagnostics and frequentist interpretations of results in addition to coefficients and their standard errors.

3.2.2 Dealing with Class Imbalances

For this study, we used both up-sampling and a case weighting technique to compensate for class imbalances. To create the up-sampled database we can use the upSample function from the caret package. The arguments passed to the upSample function are the data frame separated into the data and the class labels. It returns a new data frame with the minority class randomly compensated to achieve balanced yes and no classes. Because the data frame returned adds the class labels to the last column instead of the first, as with our original data, it is renamed accordingly.

```
up_temp <- upSample(x = dt1[,-1], y = dt1$liq)
dt1 <- up_temp
names(dt1)[8] <- "liq"
```

The original glm code can be called again to repeat the maximum likelihood regression on the up-sampled dataset.

```
m <- glm(liq ~ qc1_mean + CSR_mean, data=dt1, family = "binomial")
summary(m)
##
## Call:
## glm(formula = liq ~ qc1_mean + CSR_mean, family = "binomial",
##     data = dt1)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.76848  -0.42780   0.01844   0.42693   2.13190
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  8.57773    1.18099   7.263 3.78e-13 ***
## qc1_mean    -0.19575    0.02969  -6.592 4.33e-11 ***
## CSR_mean     1.86510    0.26459   7.049 1.80e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 385.39  on 277  degrees of freedom
## Residual deviance: 178.03  on 275  degrees of freedom
## AIC: 184.03
##
## Number of Fisher Scoring iterations: 7
```

We also performed the regression using a case weighting technique using stats4::mle(). This function requires a user defined function that accepts parameter values and returns the value of the negative log-likelihood and a list of initial parameter values, as shown in the code following. For an input vector of parameters $\boldsymbol{\beta}$ the weighted log-likelihood is defined as follows for n_l liquefied cases and n_{nl} nonliquefied cases. The probability of liquefaction for the i^{th} outcome is calculated as normal.

$$\ln[l(\boldsymbol{\beta})] = w_L \sum_{i=1}^{n_l} \ln(P_{L,i}) + w_{NL} \sum_{j=1}^{n_{NL}} \ln(1 - P_{L,j})$$

```
w_l <- 1
w_nl <- 1.5
logit.lf <- function(b0, b1, b2) {
  p_l <- 1/(1+exp(-(b0+b1*dt1$qc1_mean+b2*dt1$CSR_mean))) #probability of liquefaction w/
current parameter vector
  (w_l*sum(log(p_l[dt1$liq == "Yes"]))+w_nl*sum(log(1-p_l[dt1$liq == "No"])))*-1 #negative log-
likelihood value
}
m2 <- stats4::mle(logit.lf, start = list(b0 = 1, b1 = -1, b2 = 1))
summary(m2)
## Maximum likelihood estimation
##
## Call:
## stats4::mle(minuslogl = logit.lf, start = list(b0 = 1, b1 = -1,
##     b2 = 1))
##
## Coefficients:
```

```
##      Estimate Std. Error
## b0  8.9423534 1.42391242
## b1 -0.1928976 0.03385079
## b2  1.7285881 0.31031766
##
## -2 log L: 118.4307
```

The coefficients from this weighting method are nearly identical to the upsampled fit. The models also have a similar predictive performance, with AUC = 0.713 for the weighted likelihood and AUC = 0.727 for the upsampled model. Notably, they both outperform the original model's AUC of 0.710. Because the performances are very similar and the upsampling slightly outperforms the weighting all future models will use the upsampled dataset.

3.2.3 Mixed models

Next, we want to build our mixed modesl using the glmer function from the lme4 package. The first model only allows the intercept term to vary by event:

```
m <- glmer(liq ~ qc1_mean + CSR_mean + rf_mean + (1|event),
           data=dt1, family = "binomial",
           control = glmerControl(optimizer = "bobyqa"),
           nAGQ = 20)
```

The formula call is similar to the glm package, but include the predictor (1|event) which tells the function we want to include a random intercept by event. The options nAQG = 20 specifies that we want 20 integration points, and the optimizer = "bobyqa" specifies a non-default optimizer that tends to give better convergence. Again we can use the summary function to see the results of the fit:

```
summary(m)
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 20) [glmerMod]
##  Family: binomial  ( logit )
## Formula: liq ~ qc1_mean + CSR_mean + rf_mean + (1 | event)
##    Data: dt1
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##    166.3    184.5    -78.2    156.3      273
```

```
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.7745 -0.0973  0.0014  0.1573  3.7642
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  event  (Intercept) 4.418    2.102
## Number of obs: 278, groups:  event, 18
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  15.2628     2.6124   5.842 5.14e-09 ***
## qc1_mean     -1.0947     0.1911  -5.730 1.00e-08 ***
## CSR_mean      2.9603     0.5131   5.769 7.97e-09 ***
## rf_mean      -2.3947     0.5725  -4.183 2.88e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) qc1_mn CSR_mn
## qc1_mean -0.889
## CSR_mean  0.934 -0.788
## rf_mean  -0.594  0.639 -0.515
```

The important parts of this output are the standard deviation of the random effect, and the fixed effects coefficients. It also useful to make sure that the number of groups reported matches the number of events in the database. If they don't match something is likely mislabeled in the data frame.

We would next like to extend our mixed model to allow the other coefficients to vary by event. We repeat the `glmer` call but include `(0 + predictor|event)` terms in the formula to specify the varying slopes. R only allows a single Gaussian quadrature point for a model this complex.

```
m2 <- glmer(liq ~ qc1_mean + CSR_mean + rf_mean + (1|event) + (0 + CSR_mean|event) + (0 +
qc1_mean|event) + (0 + rf_mean|event),
            data=dt1, family = "binomial",
            control = glmerControl(optimizer = "bobyqa"),
            nAGQ = 1)
## singular fit
summary(m2)
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula:
## liq ~ qc1_mean + CSR_mean + rf_mean + (1 | event) + (0 + CSR_mean |
##     event) + (0 + qc1_mean | event) + (0 + rf_mean | event)
##    Data: dt1
## Control: glmerControl(optimizer = "bobyqa")
##
##      AIC      BIC   logLik deviance df.resid
##    152.8    181.8    -68.4    136.8      270
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -2.22222 -0.06505  0.00022  0.13815  2.83313
##
## Random effects:
##  Groups  Name        Variance Std.Dev.
##  event   (Intercept)  0.0000  0.0000
##  event.1 CSR_mean     0.1545  0.3931
##  event.2 qc1_mean     0.0000  0.0000
##  event.3 rf_mean     17.0545  4.1297
## Number of obs: 278, groups:  event, 18
##
## Fixed effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 14.07955    2.44644   5.755 8.66e-09 ***
## qc1_mean    -0.34255    0.06248  -5.482 4.20e-08 ***
## CSR_mean     2.91110    0.55293   5.265 1.40e-07 ***
## rf_mean     -0.88171    1.38107  -0.638    0.523
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##          (Intr) qc1_mn CSR_mn
## qc1_mean -0.885
## CSR_mean  0.929 -0.759
## rf_mean  -0.083  0.095 -0.040
## convergence code: 0
## singular fit
```

We have now run into a common problem with fitting multilevel models with maximum likelihood methods. The events with a low number of data points simply do not provide enough information to allow the model to converge (Gelman and Hill, 2007). While it is possible we could use some computational tricks or different estimation methods to get a solution we will instead move to Bayesian modeling.

3.3 Model Development - Introduction to Stan

We now want to repeat our previous regression models using Bayesian techniques. There are several goals behind this. In our final model we want to explicitly incorporate measurement uncertainty from the predictor variables with the goal of reducing overall model uncertainty. Doing so in a Bayesian context avoid the simplifying assumptions required for a maximum likelihood solution. Additionally, a Bayesian analysis gives full probability distributions for the model parameters. These can be used for the fully probabilistic analysis required by performance based engineering. Finally, appropriate prior choices can be used to ensure the model is physically well behaved.

A brief introduction to Stan, the Bayesian inference engine used in this study, follows to familiarize the reader with how the program works.

3.3.1 Stan Overview

This section is written to provide a very brief overview of the Stan programming language necessary to understand the models that follow. Reading through the Stan user manual and language reference (available on the website mc-stan.org) is highly recommended for users looking to write their own code. Stan is a probabilistic programming language, similar to BUGS or JAGS, that allows a user to code a Bayesian model and produce draws from the posterior distribution (Carpenter et al., 2017). As addressed in the summary of Hamiltonian Monte Carlo, Stan actually uses the logarithm of the posterior (referred to as log posterior).

This inference in two parts; first, a user writes the Stan code to define the model and provides the data from R or another interface which are compiled into a C++ file for the Stan backend. These both fully define the unnormalized posterior required for the Metropolis-Hastings algorithm. Simply put, the user code defines a very complicated function that takes a vector of parameter values as inputs and returns the scalar value of the logarithm of the unnormalized posterior density at that point. Next, the backend of Stan then takes this user defined log-posterior and runs its Hamiltonian Monte Carlo sampler (the NUTS sampler) to simulate the posterior distributions of the parameters of interest. It returns a R object with the posterior draws and model information.

Stan is similar to C++ and other programming languages in that it requires every variable used to have a declared data type. It supports many typical types such as integers, reals, vectors (which defaults to column vectors unless specified otherwise), matrices, and arrays. It also has special constrained classes often used in statistics such as correlation and covariance matrices and Cholesky factors. Any of the basic data types can be declared with upper and lower constraints. An example of how to declare constrained integers, reals, and a vector of length 3 follows.

```
data {
int <lower = 1> x;
real <upper = 0> y;
vector <lower = -1, upper = 1>[3] z;

}
```

3.3.2 Descriptions of Different Model Blocks

A Stan program is written as a series of “blocks”, a set of statements surrounded by brackets and preceded by the block name. Not all of these blocks are included or necessary in every Stan program – in fact an empty string is technically a valid Stan program but will raise an error from the compiler. The blocks must occur in the same order as listed in the skeleton below, and variable type declarations must come before statements.

```
functions {
// ... function declarations and definitions ...
}
data {
```

```
// ... declarations...

}
transformed data {
/// ... declarations ... statements
}

parameters {
// ... declarations ...
}
transformed parameters {
// ... declarations ... statements

}

model {
// ... declarations ... statements

}
generated quantities {
// ... declarations ... statements
}
```

A brief description of the blocks that appear in our programs follows.

Data

The data block is used for declaring types and names for the data passed to Stan from R or another interface. The names must exactly match those passed to Stan but the order does not have to be the same. In this block, constraints can be used to catch model bugs – passing values outside of the constraints or of the wrong size/type will raise an error. There are no statements in this block. Instead the transformed data block, which follows, can be used for applying transformations to or calculating means/standard deviations from data passed to Stan. However, this is often more easily done in R beforehand and passed as data directly.

Parameters

The variables declared in the parameters block are those that Stan will return samples for. These cannot be assigned values, so there are again no statements in this block. Constraints can also be applied here to prevent the sampler from drawing unrealistic values, such as requiring variances to be positive or slopes to be only positive/negative. Technically, Stan actually will transform constrained variables to be unconstrained behind the scenes but this is typically not an issue with writing code.

Model

In the model block a user specifies the priors on model parameters and the form of the likelihood for the data required to define the posterior. For notational convenience, Stan uses standard statistical sampling notation. For example, writing $y \sim \mathrm{Normal}(\mu, \sigma)$ tells Stan that that the variable y, which can be either an unknown parameter or known data, is supposed to be normally distributed with mean mu and standard deviation sigma. However, no sampling is actually done during this step. Rather the user is just adding terms to build the log posterior. Instead of using sampling statements, the user can also add terms to the log posterior directly.

A simple model could be defined as follows. We may wish to model some data y as normally distributed and estimate its mean, mu, assuming a population standard deviation of 1. We may also have some prior belief that the mean value is 5, give or take 0.5. The two sampling statements will build the model:

Prior:

$$\mu \sim \mathrm{Normal}(5, 0.5)$$

Data:

$$y \sim \mathrm{Normal}(\mu, 1)$$

It is common to write the prior before the likelihood, but because the statements are simply adding terms to the log posterior and all variables are already declared the order could be reversed.

Stan implicitly places uniform priors with support over [-∞, ∞] on all parameters without a defined prior. This becomes important for more complicated models because this default prior may cause the sampler to draw unrealistically high or low parameter values and lead to poor performance. In more complicated models weakly informative priors are preferred to uniform priors. A weakly informative prior is defined as one that ensures the model behaves properly, i.e. it converges to a reasonable answer, while contributing minimal information to the final outcome (Gelman et al., 2008).

The constraints declared in the parameters block supersede prior statements, so constraining a variable to be positive then giving it a Normal(0,1) prior will give it an “improper” half-normal prior that has the same shape as a standard normal density for positive values but is zero for negative values. This prior is improper in the sense that it does not integrate to 1 but because of the nature of its sampling algorithm it is a valid prior choice in Stan.

3.3.3 General Syntax

With a few exceptions, the syntax for mathematical operation, logical statements, and control loops (for, if/then, while etc.) is very similar to R. The Stan reference manual describes these in detail for the interested reader. A semi colon is required after every line of code and // is used to denote comments.

3.3.4 Performing a Standard Linear Regression in Stan

The following section explains how to perform a standard Bayesian linear regression with a single predictor variable. This aims to provide an illustrative example of the concepts discussed above, describe how to interface with Stan from R, and discuss model diagnostics. A linear regression assumes that the nth response is a linear function of a slope times the nth predictor variable value, an intercept, and an error term expressed as follows:

$$y_n = \beta_0 + \beta_1 * x_n + \epsilon_n$$

The error terms are assumed to come from a normal distribution with population standard deviation σ and mean 0. Recall from earlier that regression models relate the mean of observed outcomes to a function of predictor variables and regression parameters and the responses are assumed to come from some distribution with a constant variance. Thus, our linear regression can be equivalently expressed using standard normal distribution sampling notation as:

$$y_n \sim \mathrm{Normal}(\beta_0 + \beta_1 * x_n, \sigma)$$

Importantly, Stan supports vectorized statements— so instead of looping over the n outcomes and predictor variable values the following statement is equivalent (and faster):

$$y \sim \mathrm{Normal}(\beta_0 + \beta_1 * x, \sigma)$$

The complete model is written for Stan as follows. First, we use R to simulate 75 observations of predictor variable x and outcome y that have a generally linear relationship (Figure 31). We can use the lm function in R to recover the slope and intercept.

```
set.seed(5)
n <- 75 #number of data points
sigma <- 5 #population variance
b0 <- 3 #regression intercept
b1 <- 5 #regression slope
x <- rnorm(n, 10, 5)
y <- rnorm(n, b0 + b1*x, sigma)
plot(x,y)
```

```
m <- lm(y ~ x)
a <- summary(m)
a$coefficients
##             Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 4.204265  1.3519345  3.109814 2.669695e-03
## x           4.889179  0.1215198 40.233604 1.429516e-51
a$sigma
```

```
## [1] 5.124816
```

The following Stan code to estimate the model parameters is written in a text file and saved with a .stan extension in the current working directory.

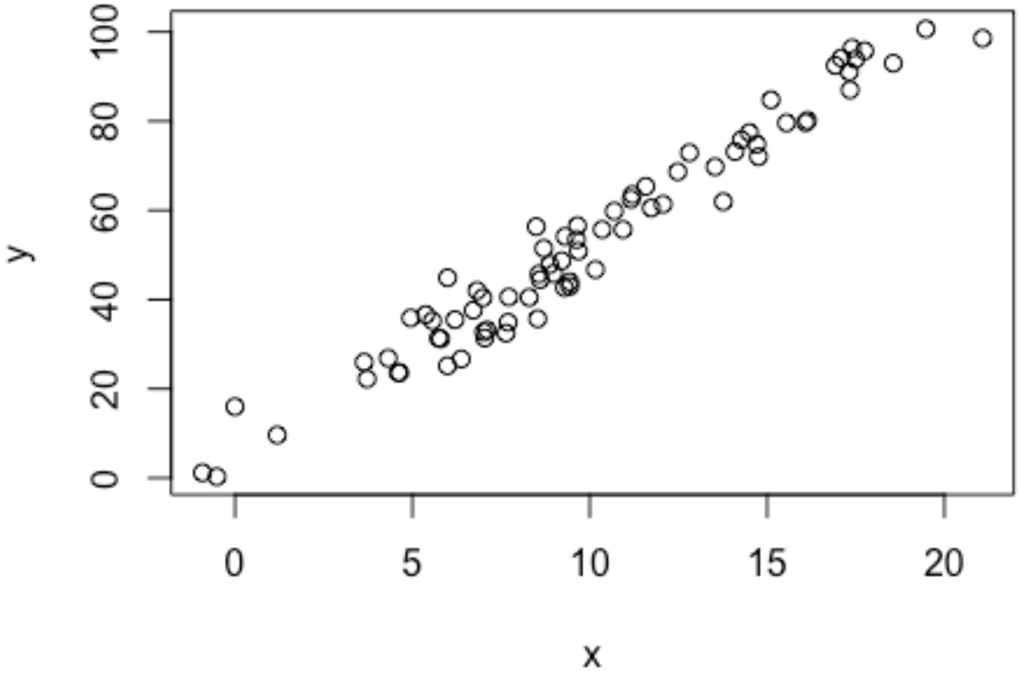


Figure 31 – Simulated Data for the Bayesian Linear Regression Example.

```
data {
 int <lower = 0> N; //number of data points
 vector[N] y; //outcomes
 vector[N] x; //predictor variable values

   }
parameters {
 real b0; //intercept
 real b1; //slope
 real <lower = 0> sigma; //population noise
 }

model {
 y ~ normal(b0 + b1*x, sigma);
 }
```

If desired, priors would be placed on the slope, intercept, or population noise parameters in the model block.

The next step is to convert the data into a named list to pass to Stan. A typical error checking step is to constrain x and y to be size N, so if erroneous data is passed to the program it will raise a flag.

```
data <- list(x,y, length(x))
names(data) <- c("x","y", "N")
```

Next, we use the `rstan` interface to compile the stan program and run the sampling algorithm. The stan argument takes the .stan file name, a named list of data, and a variety of control options. The chains and iter options specifies that we want to run 4 Markov chains with 1500 iterations each. By default, Stan will use half of these for warming-up, also called the burn-in period, in which it fine-tunes its sampling algorithm. It then discards the warm up samples and draws the final half. This is to allow the sampler to settle in to the posterior distribution. There are no strict guidelines on how many chains to run. Multiple are preferred to a single one because they can provide a check on convergence: if they do not all eventually converge to the same space the sampling is suspect. Because Stan supports parallel processing where each chain is run on a single core, a reasonable number to run is the number of cores available. Running more chains also generates more effectively independent samples. The user can also use a `control = list()` argument to directly specify certain algorithmic controls. This is usually unnecessary for most simple models. In fact, poor performance is more often the fault of a poorly coded or wrongly specified model rather than the sampling algorithm itself.

```
fit <- stan(
  file = "linear_regression.stan",  # Stan program name
  data = data,    # named list of data
  chains = 4,           # number of Markov chains
  iter = 1500           # total number of iterations per chain
  )
```

This call to `stan` may return two common errors. Divergent transitions after warm-up are the most worrisome– they indicate numerical instabilities in the sampling algorithm and the model should be re-run. As the program will recommend, increasing `adapt_delta` above the current value by calling control=list(adapt_delta = 0.99), for example, will often solve the problem at the expense of computational time. Samples hitting maximum tree depth is an efficiency concern rather than validity concern and can be fixed with a similar command control=list(max_treedepth = 20), for example. If the model continues to perform poorly it often indicates that it is poorly coded or requires a re-parametrization. The Stan user guide and discourse forums are a good resource if this is encountered.

Theoretically, a properly designed MCMC simulation that is run "forever" will always converge to the posterior distribution of interest. However, since we have stopped before "forever" it is critical to check if it has actually converged. The shinystan package is very useful for visualizing model results and performing diagnoses. The "explore" tab summarizes the basic distributional summarizes and diagnostics for each parameter (Figure 32).

```
launch_shinystan(fit)
```

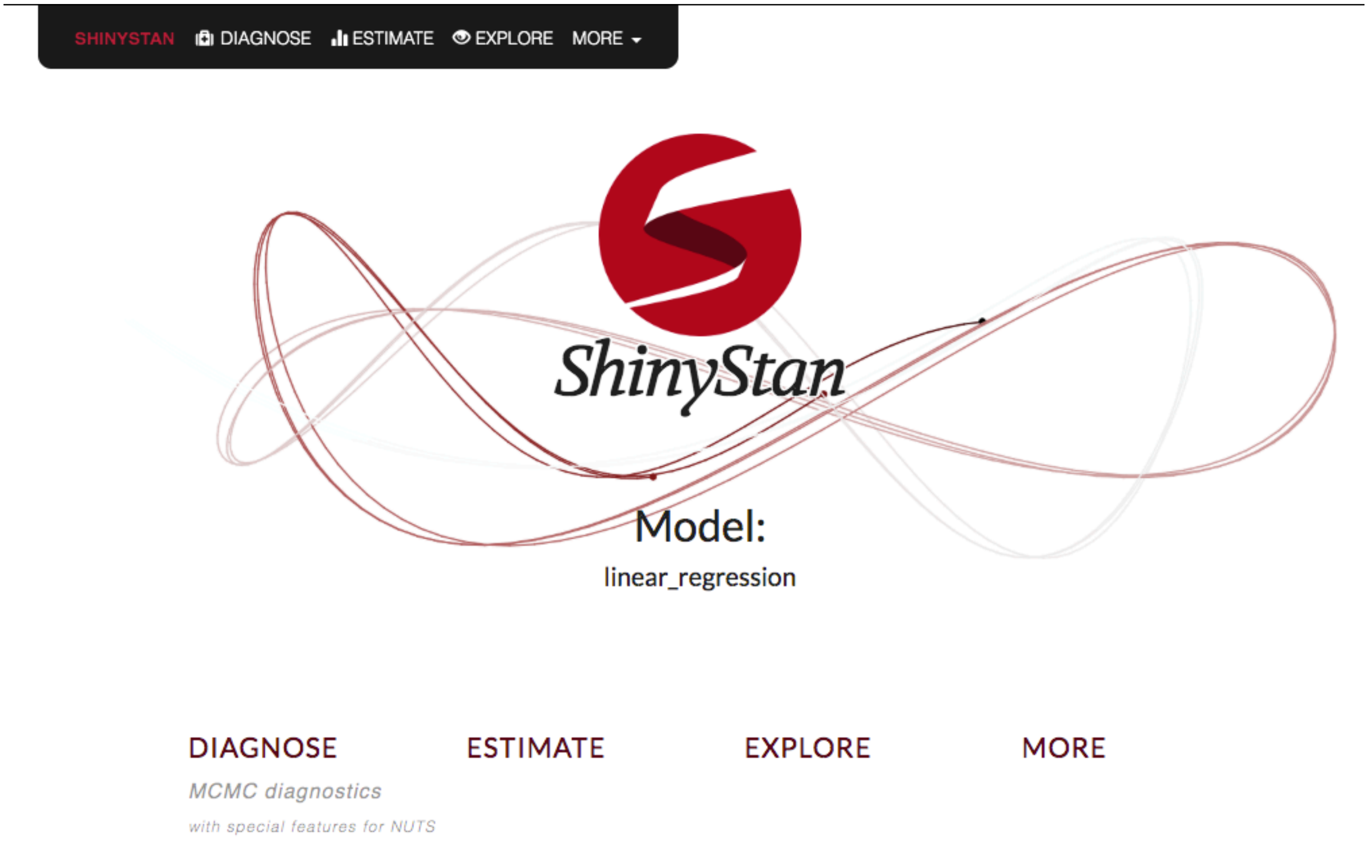


Figure 32 – The Shinystan Interface for Visualizing Model Results and MCMC Diagnostics.

The "diagnose" tab contains specific metrics for the No U-Turn Sampler typically only relevant for more complicated models experiencing sampling problems. The "estimate" tab is used for viewing distributional summary statistics and simple plots for the posterior draws. The "explore" tab, usually the most useful for preliminary model check, has all the diagnostics we look for such as traceplots, autocorrelation plots, and $\hat{R}$ in addition to distributional summary statistics and plots. Finally, the "explore" tab can be used to generate scatterplots comparing the different parameters.

Providing the model has run without errors, the first step in checking convergence should be always to visually inspect the trace plots. We are looking for the plots to look like a "fat, hairy caterpillar" that more or less stay stationary without any clear trends (Figure 33).

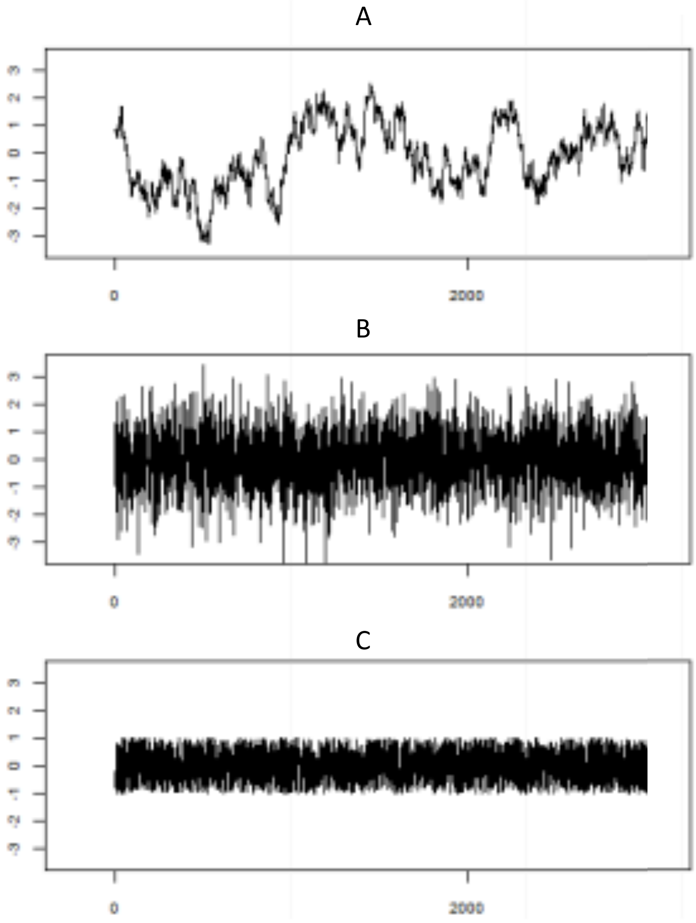
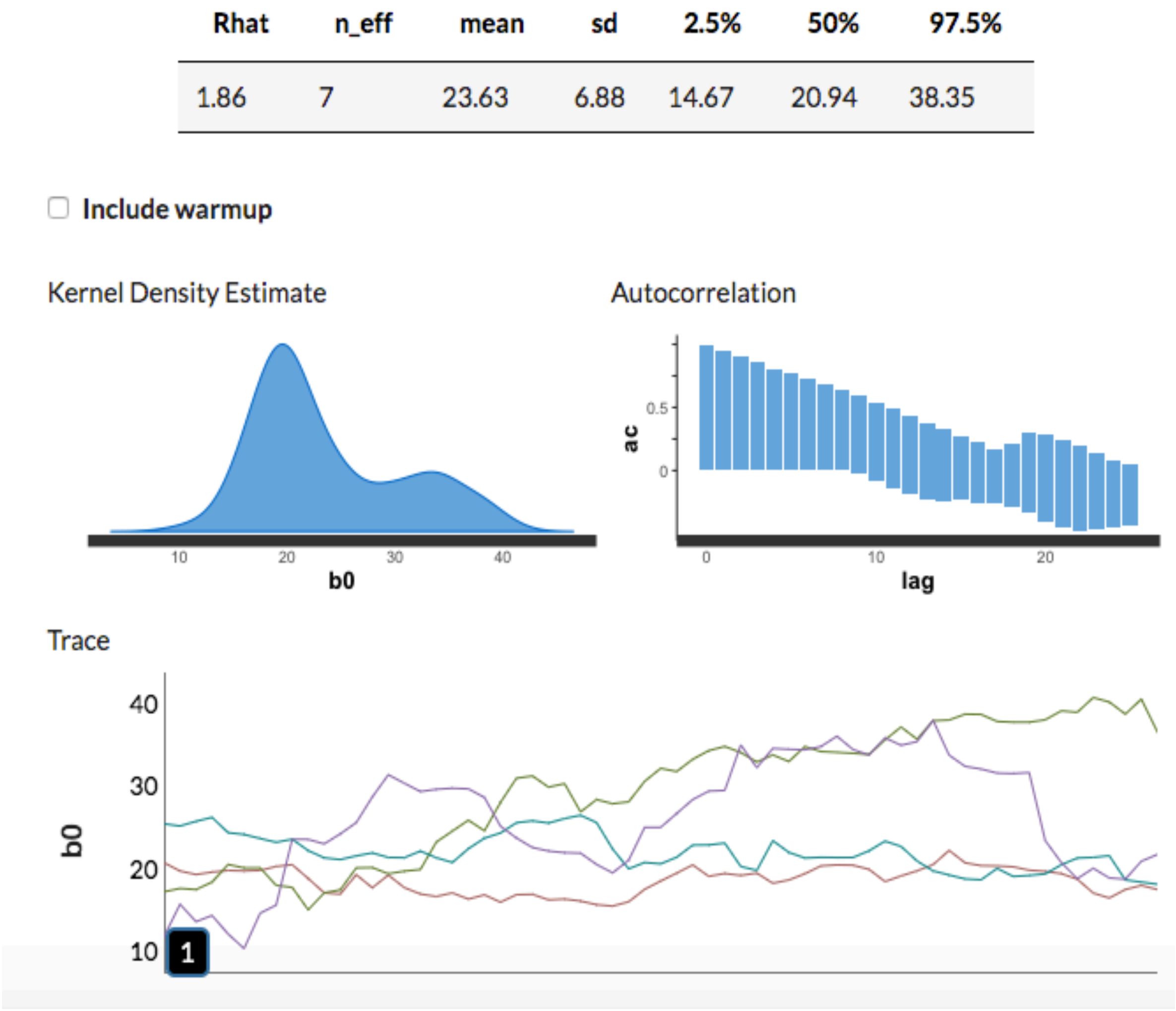


Figure 33 – Three Possibilities for Typical Traceplot Results. Reproduced from Kruschke, 2015.

Trace A explores the full parameter space but because its values stay close to each other it has high autocorrelation and low effectively independent sample size. Trace C has much lower autocorrelation but does not explore the full parameter space. Trace B is what a good MCMC result should look like. The autocorrelation plot compares how correlation the draws are we each other as the iterations progress. It should quickly drop to zero and hover around there. If it doesn't the chain is producing a low number of effectively independent samples, also indicated by a trace plot that shows clear trends (more like a snake than a caterpillar). We are looking to have as many effectively independent samples (n_{eff}) as possible to ensure we have a good approximation of the posterior distribution. Finally, the $\hat{R}$ statistic should be 1 or near 1 (± 0.1 usually). Figure 34 shows a summary of a MCMC output that has not converged to the target distribution.



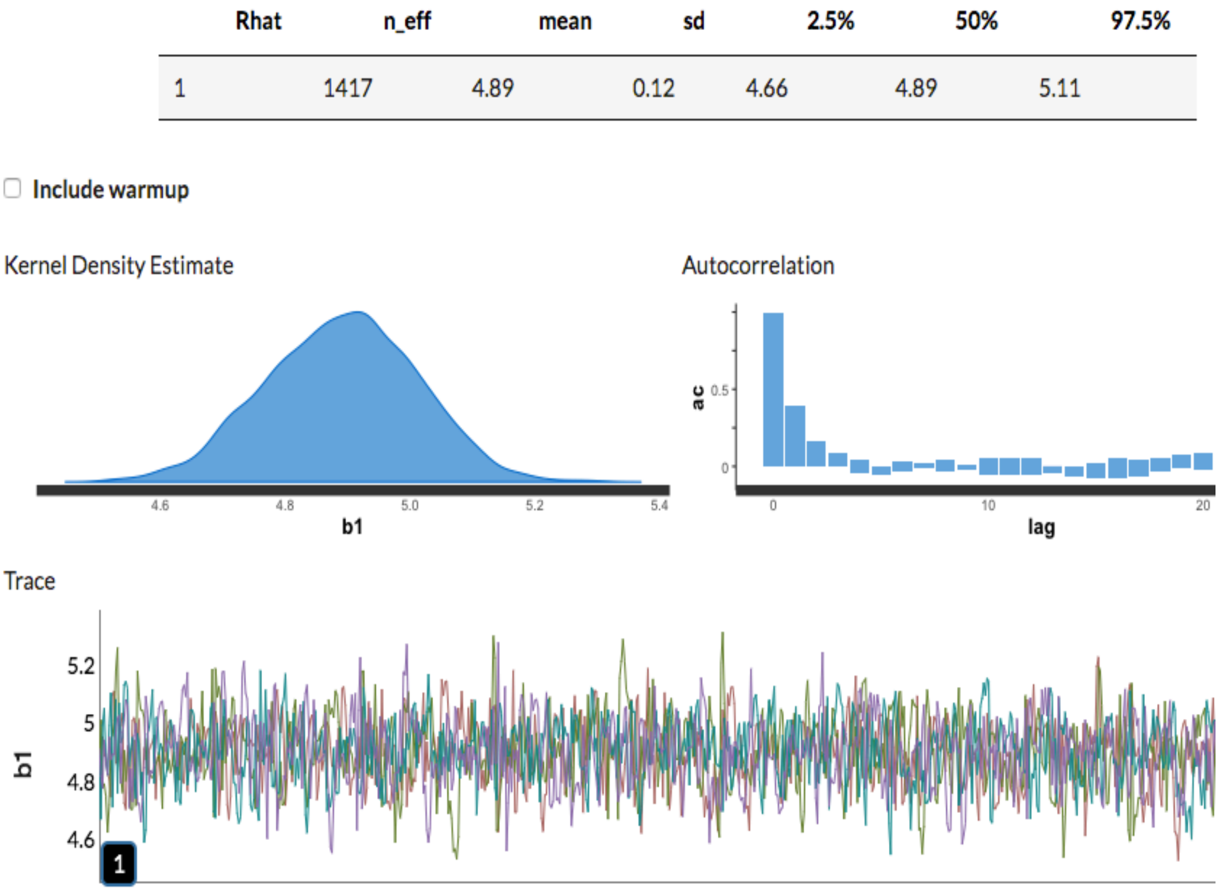
Rhat	n_eff	mean	sd	2.5%	50%	97.5%
1.86	7	23.63	6.88	14.67	20.94	38.35

☐ Include warmup

Figure 34 – The Shinystan "Explore" Tab for a Model Parameter That Has Not Converged. The autocorrelation plot shows high draw-to-draw correlation and the traceplot shows clear trends and does not explore the parameter space. The $\hat{\boldsymbol{R}}$ value is also well above 1.

Typically, any issues with the above are resolved by simply increasing the number of iterations per chain. While there are theoretically methods of determining beforehand how long to run the chains for it is easiest to start with a small number, say 125, of iterations and increase them until diagnostic criteria are met. This allows errors in the model to be caught early on and avoids running the chains for excessively long.

Now that we are satisfied with the convergence of our Markov chains (the summary for the slope parameter is shown in Figure 35), we can store all draws in a data frame or a .csv file.



Rhat	n_eff	mean	sd	2.5%	50%	97.5%
1	1417	4.89	0.12	4.66	4.89	5.11

□ Include warmup

Kernel Density Estimate

Autocorrelation

Trace

Figure 35 – The Shinystan "Explore" Tab For Our Sample Linear Regression Slope.

The MCMC draws for this parameter have converged to the proper distribution as evidenced by: the traceplot showing good mixing, low autocorrelation, and an $\hat{R}$ of approximately 1. This tab also includes a kernel density estimate and useful summary statistics such as mean, standard deviation, median, and a 95% credible interval.

```
draws <- as.data.frame(fit)
write.csv(draws, "linear_regression_results.csv")
```

A useful way of reporting this Bayesian analysis would be distributional summaries of the model coefficients. The plots show a histogram and density estimate with dashed lines representing the 2.5th and 97.5th percentiles to visualize a 95% credible interval.

```r
draws <- read.csv("linear_regression_results.csv")
b0 <- draws$b0
hist(b0, breaks = 30, freq = F, main = paste("Mean = ", round(mean(b0),3),"SD = ",
round(sd(b0),3)), xlab = "Intercept")
lines(density(b0))
abline(v = quantile(b0, c(0.025,0.975)), lty = 2, col = 'black')
draws <- read.csv("linear_regression_results.csv")
b1 <- draws$b1
hist(b1, breaks = 30, freq = F, main = paste("Mean = ", round(mean(b1),3),"SD = ",
round(sd(b1),3)), xlab = "Slope")
lines(density(b1))
abline(v = quantile(b1, c(0.025,0.975)), lty = 2, col = 'black')
draws <- read.csv("linear_regression_results.csv")
sigma <- draws$sigma
hist(sigma, breaks = 30, freq = F, main = paste("Mean = ", round(mean(sigma),3),"SD = ",
round(sd(sigma),3)), xlab = "Sigma")
lines(density(sigma))
abline(v = quantile(sigma, c(0.025,0.975)), lty = 2, col = 'black')
```

We can also plot the distribution of the many possible regression lines using the posterior draws of the regression coefficients. The dashed line is the one resulting from using the mean values of regression coefficients (Figure 36).

```r
plot(x,y)

nlines = sample(1:length(b0), length(b0))

for (l in nlines){
  abline(b0[l], b1[l], col=rgb(0.25,0.25,0.25,0.01), lwd=2)
}
abline(mean(b0),mean(b1), lty = "dotted", col = "black",lwd = 2)
```

As this example shows, for a simple case the frequentist or maximum likelihood and Bayesian solutions should be very close to each other, if not identical. This is often a useful check in early stages to ensure the Stan model is coded properly.

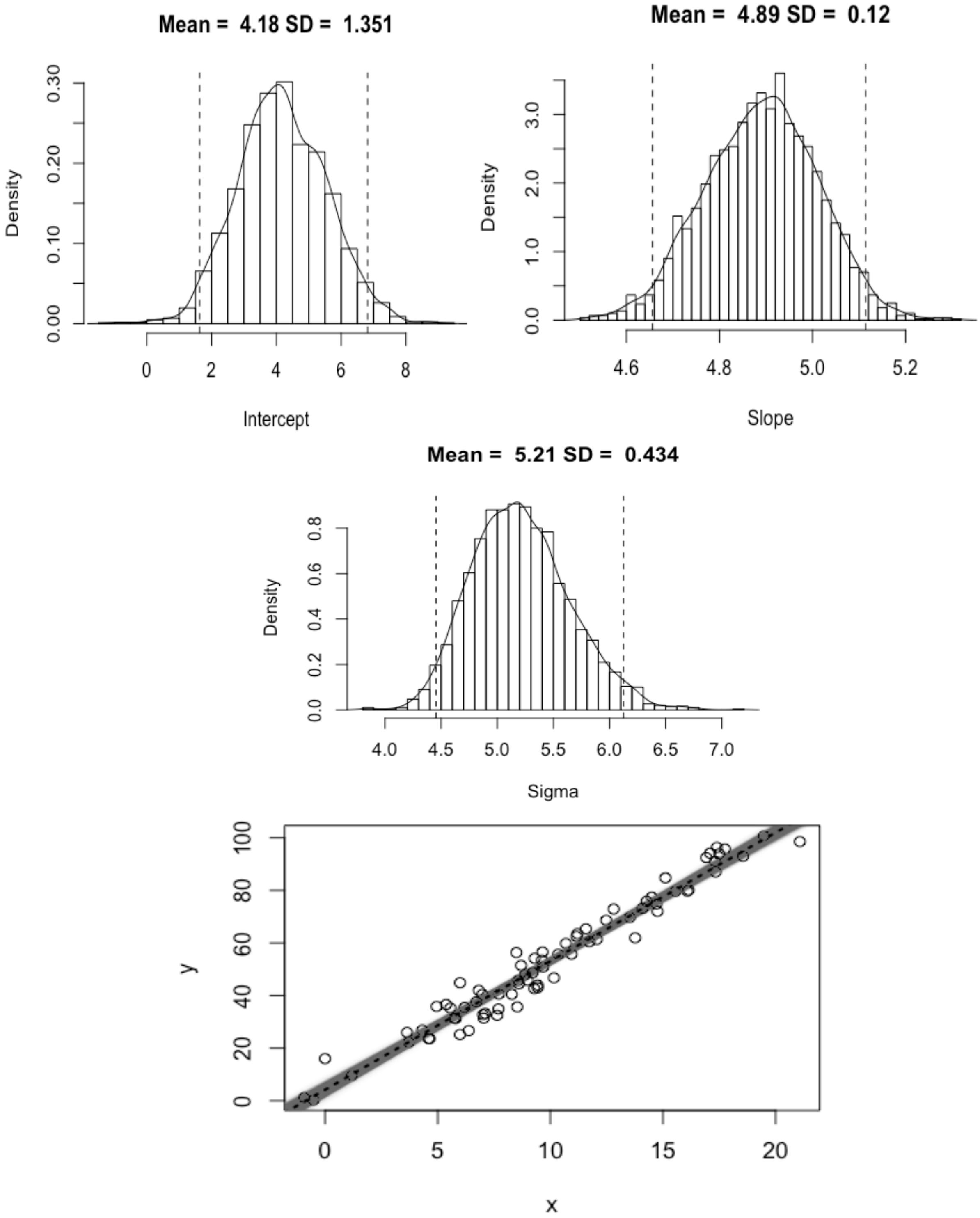


Figure 36 – The Posterior Distributions for our Linear Regression Example. Showing mean, standard deviations, and 95% credible intervals. The plot at the bottom shows the distribution for the regression lines shaded by probability density, with the mean result indicated by the dashed line.

3.3.5 A (Very Brief) Discussion of Prior Choice

Choosing a prior distribution on all model parameters is required for any Bayesian analysis. It is often the most highly scrutinized, as most of the criticisms around Bayesian analysis deal with the "subjectivity" of prior choice. We would like to highlight a few responses to this argument. First, subjective decisions cannot be avoided in statistics. Practitioners constantly exercise subjective judgment when choosing what model to use, what data to include in their analysis, and the implicit assumptions that accompany these decisions (Gelman and Hennig, 2017). Even a seemingly innocuous linear regression or p-test comes with a host of assumptions about the underlying population and experimental process (Gelman and Henig, 2017). In the words of Gelman and Hennig, "it is a mistake to consider the prior distribution as the exclusive gate at which subjectivity enters a statistical procedure." However, given a "sufficient" amount of data the resulting posterior distributions will dominated entirely by the likelihood and its associated characteristics (mean, standard deviation, etc.) will not change much with a change in prior (Kruschke, 2015). Naturally, what constitutes a "sufficient" amount of data is dependent on the questions being asked and the models used.

Broadly, a prior can be categorized into three categories: noninformative, weakly informative, and deliberately informative. A noninformative prior is built to allow inference only from the data itself, without any outside information (Gelman, 2006). Importantly, depending on the of the model a uniform prior is not always uninformative (Gelman, 2006). In fact, in cases where data is relatively limited such a prior will result in quite misleading posterior inferences – Gelman et al., 2017 discusses this in the context of parent attractiveness influencing the sex of their children. The general idea is that the posterior is not wholly dominated by the data, choosing a seemingly innocent uniform prior actually places far too much mass on unrealistic (or impossible) parameter values and prevents accurate inference. A weakly informative prior, on the other hand, contains enough information to limit the mode to “realistic” parameter values while remaining relatively uninformative over this range (Gelman, 2006). This is referred to as a “regularizing” effect. For hierarchal models, similar to the one we develop, such a prior choice is often necessary for the model to actually identify reasonable posterior distributions for parameters (Gelman and Hill, 2007). Finally, an informative prior intentionally influences parameter values and is often based upon past work or expert consensus. For example, Kuehn and Abrahamson, 2017, used computer simulation of ground motions and existing attenuation relationships to determine an appropriate distribution of possible regression coefficients to ensure their ground motion prediction model was physically well-behaved. Informative priors can also be based on expert consensus or the physical mechanisms behind the data being modeled. We believe that this is an often overlooked benefit of Bayesian analysis – it offers a consistent mathematical framework for explicitly incorporating engineering judgement and past work into new models.

3.4 Bayesian Models

Now it is time to fit our Bayesian logistic regressions. For models that follow, convergence criteria were a targeted $\hat{R}$ of 1.0 ± 0.1 and qualitative inspection of the trace and autocorrelation plots to verify independent sampling of the entire parameter space. We used 4 chains and selected an appropriate number of post-warmup iterations, which varied by model complexity, to satisfy convergence criteria. As necessary we modified the max_treedepth and adapt_delta values to ensure sampling stability and efficiency.

3.4.1 Prior Selection

For the models that follow, we use weakly informative normal distributions to constrain the scale of the regression coefficients. First, we have to estimate what a reasonable scale for these coefficients will be. Recall that our formula for calculating liquefaction probability (for two predictor variables) is:

$$q_{c,1}^{*} = \frac{{q_{c,1}}^{1.6} - 1}{1.6}$$

and

$$CSR^{*} = \frac{CSR^{-0..6} - 1}{-0.6}$$

giving

$$P_L = \frac{1}{1 + \exp\{-(\beta_0 + \ \beta_1 * q_{c,1}^{*} + \beta_2 * CSR^{*})\}}$$

This also be expressed in terms of this on the log-odds, where the odds are the ratio of the probability for and against an outcome:

$$\beta_0 + \ \beta_1 * q_{c,1}^{*} + \beta_2 * CSR^{*} = \ln\left(\frac{P_L}{1 - P_L}\right)$$

This formula illustrates the natural interpretation of logistic regression coefficients – that a one unit increase in a predictor variable results in an increase of β in the log-odds.

Table 1, following, shows the relationship between probability of occurrence, odds, and log odds for a few levels of probability.

Table 1 – The Relationship Between the Probability of an Outcome, its Odds, and its Log-Odds.

Probability	Odds	Log-Odds
0.01	0	-4.60
0.1	0.1	-2.20
0.5	1	0.00
0.9	9	2.20
0.95	19	2.94
0.99	99	4.60
0.999	999	6.91
0.9999	9999	9.21

Gelman et al., 2008 comments that is almost never seen in practical applications that a typical increase in a predictor variable would result in a jump from 1% probability 99% probability. From the table above, we see that this corresponds to an increase of 9.2 in log-odds. If we re-scaled all predictor variables to have a standard deviation of 0.5, as with Gelman et al., 2008, we would then be comfortable assigning low prior probabilities to coefficient values above 5. While this study employs different transformations that do not necessarily have this re-scaling property, we can use the same logic to determine expected scales of our regression coefficients.

Considering $q_{c,1}^{*}$ a typical change could be the standard deviation of the transformed values in the dataset: 18.4 (for two predictor variables) or 4.58 (for three predictor variables. We can then apply the logic of Gelman et al., that an increase of 1 standard deviation should at most decrease the probability of liquefaction from 99% to 1%. Holding other terms constant, we can solve for the β_1 that would result in this limiting case.

$$\beta_0 + \ \beta_1 * q_{c,1}^{*} + \beta_2 * CSR^{*} = \ln\left(\frac{0.99}{1-0.99}\right) = 4.60$$
$$\beta_0 + \ \beta_1 * (q_{c,1}^{*} + 18.4) + \beta_2 * CSR^{*} = \ln\left(\frac{0.01}{1-0.01}\right) = -4.60$$

Subtracting the two yields

$$-\beta_1 * 18.4 = 9.2$$
$$\beta_1 = -0.5$$

Repeating this process with a standard deviation of 4.58 instead gives a $\beta_1 = -2.01$.

Applying similar logic to CSR* a typical change could be the standard deviation of the transformed values in the dataset: 1.253. Holding other terms constant, we can solve for the β_2 that would result in this limiting case that an increase of 1 standard deviation should at most increase the probability of liquefaction from 1% to 99%.

$$\beta_0 + \ \beta_1 * q_{c,1}^{*} + \beta_2 * (CSR^{*} + 1.253) = \ln\left(\frac{0.01}{1-0.01}\right) = -4.60$$
$$\beta_0 + \ \beta_1 * q_{c,1}^{*} + \beta_2 * CSR^{*} = \ln\left(\frac{0.99}{1-0.99}\right) = 4.60$$

Subtracting the two yields

$$-\beta_2 * 1.253 = -9.2$$
$$\beta_2 = 7.34$$

Repeating the process for R$_f$ gives $\beta_3 = -13.86$.

In summary, we can reasonably expect that the absolute value of regression should then be on the order 10. Based upon this reasoning, we default to a Normal (0,10) prior on regression slopes in our modeling (Figure 37).

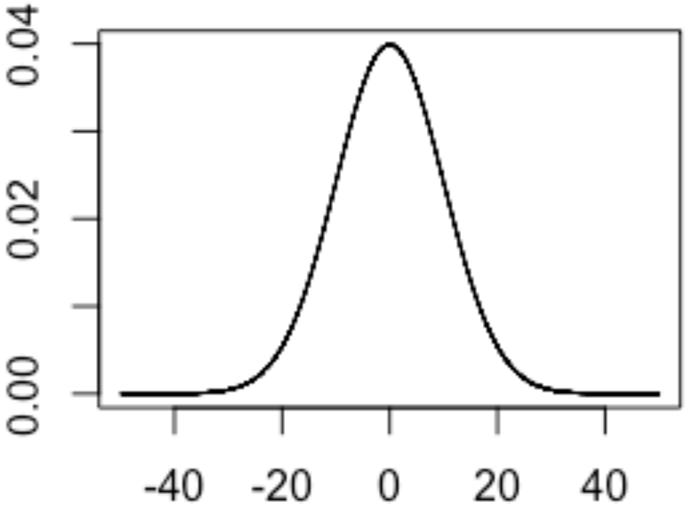


Figure 37 – Our Default Normal Prior Distribution, with Mean 0 and Standard Deviation 10.

The distribution is relatively uninformative in the range of reasonable values. For example, the probability of the coefficient being between 0 and 1 or 7 and 8 does not differ too much (0.039 vs 0.03) but there is only a probability of 0.16 that it is greater than 10 in absolute value.

For a sensitivity study, we also run the models with prior standard deviations of 25 and 100 to examine the influence of the prior distribution on the models' behavior. In the results section we discuss how the different prior choices affect posterior distribution characteristics and model performance.

For the intercept parameters, we follow the recommendations of Gelman et al., 2008 and use a slightly more diffuse distribution of Normal(0,25).

3.4.2 Standard Logistic Regression

First, we convert our data frame into a list usable by Stan. These variables are not used in all models, but it is useful to have them all included initially and have the Stan code pick out the terms it needs.

```
data <- as.list(dt1)
data <- c(data, length(dt1$liq),length(unique(dt1$event)))
names(data) <- c("liq","CSR_obs", "CSR_sd", "qc1_obs","qc1_sd","rf_obs","rf_sd","event", "N",
"E")
data$liq <- as.integer(dt1$liq == "Yes")
data$event <- as.integer(data$event)
```

Importantly, the liquefaction and event labels need to be recoded as integers.

Next, we can write the Stan model and store it under the name of our choosing (model.stan) as follows:

```
data {
 int <lower = 1> N; //number of data points
 vector[N] qc1_obs; //observed values
 vector[N] CSR_obs;
 vector[N] rf_obs;
 int<lower=0,upper=1> liq[N];
   }

parameters {
 real b0;
 real <upper = 0> b1 ;
 real <lower = 0> b2;
 real b3;
}

model {
    liq ~ bernoulli_logit(b0 + b1 * qc1_obs +
    b2*CSR_obs+b3*rf_obs);

}
```

and call it using the rstan interface

```
fit <- stan(
 file = "model.stan", # Stan program name
 data = data,   # named list of data
 chains = 4,          # number of Markov chains
 iter = 2500          # total number of iterations per chain
 )
```

The code in this model is quite similar to the linear regression example. The values for $q_{c,1}$, CSR, and R_f, and their corresponding binary coding for liquefaction or nonliquefaction are declared and accessed in the data block. The constraints on the liq variable serve as a check – if the program sees something other than a 0 or 1 it will terminate with an error. In the parameters block we specify the intercept (b0), and the slopes (b1, b2, and b3). To ensure the model is consistent with physical principles of liquefaction we constrain the CSR slope to be positive and the $q_{c,1}$ slope to be negative. These choices reflect that it would be unrealistic for a lower penetration resistance or a higher seismic load to imply a lesser probability of liquefaction. Finally, the model block specifies the regression model. The bernouli_logit likelihood is the same likelihood as discussed in Chapter 2. The sampling notation for this model is:

Model 1

Prior:

$$\beta_0 \sim \text{Normal}(0{,}25)$$

$$\beta_1 \sim \text{Normal}(0{,}10), \beta_1 \leq \ 0$$

$$\beta_2 \sim \text{Normal}(0{,}10), \beta_2 \geq 0$$

$$\beta_3 \sim \text{Normal}(0{,}10)$$

Data:

$$y \sim \text{Bernouli\_Logit}(\beta_0 + \beta_1 * \ q_{c,1} \ + \ \ \beta_2 * CSR \ + \ \beta_3 * R_f \ )$$

Because of the simplicity of this model, its posteriors are almost identical to uniform priors and do not show any appreciable change between the sensitivity study priors.

3.4.3 Multilevel/Mixed Effects Logistic Model

Next, we want to build our full mixed model. Recall from Chapter 2 that a mixed effects or multilevel model groups data and allows the coefficients to vary between group provided they are constrained to come from underlying distributions with parameters estimated from the data. The model is described using sampling notation as follows for an outcome in the jth event.

Model 2

Priors:

$$\beta_{0,j} \sim \text{Normal}\left(\mu_{\beta_0}, \sigma_{\beta_0}\right)$$

$$\beta_{1,j} \sim \text{Normal}\left(\mu_{\beta_1}, \sigma_{\beta_1}\right)$$

$$\beta_{2,j} \sim \text{Normal}\left(\mu_{\beta_2}, \sigma_{\beta_2}\right)$$

$$\beta_{3,j} \sim \text{Normal}\left(\mu_{\beta_3}, \sigma_{\beta_3}\right)$$

$$\mu_{\beta_0} \sim \text{Normla}(0{,}25)$$

$$\mu_{\beta_1} \sim \text{Normal}(0, 10),\ \mu_{\beta_1} \leq 0$$

$$\mu_{\beta_2} \sim \text{Normal}(0, 10),\ \mu_{\beta_2} \geq 0$$

$$\mu_{\beta_3} \sim \text{Normal}(0{,}10)$$

$$\sigma_{\beta_0} \sim \text{Normal}(0{,}10),\ \sigma_{\beta_0} \geq 0$$

$$\sigma_{\beta_1} \sim \text{Normal}(0{,}5),\ \sigma_{\beta_1} \geq 0$$

$$\sigma_{\beta_2} \sim \text{Normal}(0{,}5),\ \sigma_{\beta_2} \geq 0$$

$$\sigma_{\beta_3} \sim \text{Normal}(0{,}5),\ \sigma_{\beta_3} \geq 0$$

Data:

$$y_{,j} \sim \text{Bernouli\_Logit}(\beta_{0,j} + \beta_{1,j} * \ q_{c,1} \ + \ \ \beta_{2,j} * CSR \ + \ \beta_{3,j} * R_f \ )$$

This model estimates a set of parameters is for each event, with these parameters constrained to each come from some normal population with mean and standard deviation also estimated from the model. The priors for the population hyperparameters are selected so that the implicit priors on the individual parameters are close to the default priors of Normal(0,10) or Normal(0,25). As discussed earlier, because of the complexity of the model a prior distribution with some sense of expected scale for the regression coefficients is required for it to estimate reasonable posterior distributions for regression coefficients.

Furthermore, hierarchal models have a tendency to produce posteriors with geometries that make sampling difficult – mostly due to the strong local, as opposed to global correlations introduced by the hierarchal structure. This complex phenomenon is discussed at length in the paper by Betancourt and Girolami, 2013. Fortunately, a rather simple parameterization will produce much simpler posterior geometries. This computation trick makes use of the following equivalency between the first “centered” parameterization and the second “non-centered” parameterization (after Betancourt and Girolami, 2013). If we have data y with a dependence on a parameter θ that has associated hyperparameters μ and τ:

$$\theta \sim \mathrm{N}(\mu, \tau)$$

$$\leftrightarrow$$

$$\theta = \theta^* * \tau + \mu$$

$$\theta^* \sim \mathrm{Normal}(0,1)$$

In the context of our hierarchal regression, we can code this as follows for by-event varying coefficient β:

$$\beta \sim \mathrm{Normal}(\mu_\beta, \sigma_\beta)$$

$$\leftrightarrow$$

$$\beta = \beta^* * \sigma_\beta + \mu_{\ \beta}$$

$$\beta^* \sim \mathrm{Normal}(0,1)$$

As discussed in the Betancourt and Girolami paper, because of how the HMC sampler works this re-parameterization drastically improves computational speed and stability. For all models following we make use of the re-parameterization in the actual code, but for ease of understanding when describing models in the text we will use the typical centered parameterization.

The Stan code to estimate the mixed model coefficients is as follows:

```
data {
 int <lower = 1> N; //number of data points
 int <lower = 1> E; //number of events
 int <lower = 1, upper = E>  event[N]; //event id for observations
 real qc1_obs[N]; //observed values
 real CSR_obs[N];
 real rf_obs[N];
 int<lower=0,upper=1> liq[N];
   }
parameters {
 vector[E] b0_raw; // event level parameters
 vector[E] b1_raw;
 vector[E] b2_raw;
 vector[E] b3_raw;
 real <lower = 0> sigma_b0; // hierarchal standard deviation
 real <lower = 0> sigma_b1;
 real <lower = 0> sigma_b2;
 real <lower = 0> sigma_b3;
 real mu_b0; // hierarchal means
 real <upper = 0> mu_b1;
 real <lower = 0> mu_b2;
 real mu_b3;
}
transformed parameters{
 vector[E] b0;
 vector[E] b1;
 vector[E] b2;
 vector[E] b3;
 b0 = mu_b0 + sigma_b0 * b0_raw;
 b1 = mu_b1 + sigma_b1 * b1_raw; // non-centered parametrization
 b2 = mu_b2 + sigma_b2 * b2_raw;
 b3 = mu_b3 + sigma_b3 * b3_raw;
}
model {
    mu_b0 ~ normal(0,25); // priors for coefficients
    mu_b1 ~ normal(0,10); // diffuse normals
    mu_b2 ~ normal(0,10);
    mu_b3 ~ normal(0,10);
    sigma_b0 ~ normal(0,10);
    sigma_b1 ~ normal(0,5);
    sigma_b2 ~ normal(0,5);
    sigma_b3 ~ normal(0,5);
    b0_raw ~ normal(0,1);
    b1_raw ~ normal(0,1);
    b2_raw ~ normal(0,1);
    b3_raw ~ normal(0,1);
    for (i in 1:N) {
      liq[i] ~ bernoulli_logit(b0[event[i]] + b1[event[i]]* qc1_obs[i] + b2[event[i]] * CSR_obs[i] +
b3[event[i]] * rf_obs[i]);
    }
}
```

This model is noticeably more complex than the standard regression. The data used by the model now includes an event ID for each outcome, used to index into the appropriate set of regression coefficients for that event. In addition to estimating posterior distributions for each event's regression coefficients it also produces posterior draws for the population mean and standard deviation of these coefficients.

3.4.4 Measurement Error Model

Next, we implement our measurement error model. This model is based off the work by Kuehn and Abrahamson, 2017 though the original form was proposed in 1993 by Richardson and Gilks. The measurement error model treats the true values of predictor variables as parameters to be estimated during the modeling process. More formally, it introduces another hierarchal level – that the latent true values are assumed to come from a normal distribution centered at the observed mean from the database and with the database estimated standard deviation. More formally, for a generic predictor variable x, (either $q_{c,1}$, CSR, or R_{f},), an observed mean value $x_{measured}$, and known standard deviation τ_x,

Prior:

$$x_{true} \sim \text{Normal}(\mu_x, \sigma_x)$$

Data:

$$x_{observed} \sim N(x_{true}, \tau_x)$$

The full model is described as follows

Model 3

Priors:

$$CSR_{true} \sim \mathrm{Normal}(\mu_{CSR}, \sigma_{CSR})$$

$$q_{c,1,true} \sim \mathrm{Normal}(\mu_{q_{c,1}}, \sigma_{q_{c,1}})$$

$$r_{f,true} \sim \mathrm{Normal}(\mu_{r_f}, \sigma_{r_f})$$

$$\beta_0 \sim \mathrm{Normal}(0, 25)$$

$$\beta_1 \sim \mathrm{Normal}(0, 10)$$

$$\beta_2 \sim \mathrm{Normal}(0, 10)$$

$$\beta_3 \sim \mathrm{Normal}(0, 10)$$

Data:

$$CSR_{observed} \sim \mathrm{Normal}(CSR_{true}, \tau_{CSR})$$

$$q_{c,1,observed} \sim \mathrm{Normal}(\mathrm{q}_{\mathrm{c,1,true}}\ , \tau_{q_{c,1}})$$

$$r_{f,observed} \sim \mathrm{Normal}(\mathrm{r}_{\mathrm{f,true}}, \tau_{r_f})$$

$$y_{,} \sim \mathrm{Bernouli\_Logit}(\beta_0 + \beta_1 * \ q_{c,1,true} \ + \ \beta_2 * CSR_{true} \ + \ \beta_3 * R_{f,true}\ )$$

This model is implemented in Stan with the code below:

```
data {
  int <lower = 1> N; //number of data points
  vector[N] qc1_obs; //observed means
  vector[N] CSR_obs;
  vector[N] rf_obs;
  vector[N] qc1_sd; //observed measurement noises
  vector[N] CSR_sd;
  vector[N] rf_sd;
  int<lower=0,upper=1> liq[N];
    }

parameters {
  real b0; // regression parameters
  real <upper = 0> b1;
  real <lower = 0> b2;
  real b3;


}
model {
  qc1_true ~ normal(qc1_pmu, qc1_psd); // hierarchal priors on measured value
  CSR_true ~ normal(CSR_pmu, CSR_psd);
  rf_true ~ normal(rf_pmu, rf_psd);
  qc1_obs ~ normal(qc1_true, qc1_sd); //measurement model
  CSR_obs ~ normal(CSR_true, CSR_sd);
  rf_obs ~ normal(rf_true, rf_sd);
  b0 ~ normal(0,25); // weakly informative priors on coefficients
  b1 ~ normal(0,10);
  b2 ~ normal(0,10);
  b3 ~ normal(0,10);
  liq ~ bernoulli_logit(b0 + b1*qc1_true + b2*CSR_true + b3*rf_true);
  }
}
```

Again, we can break down how the model functions block by block. The data block now reads in the means and standard deviations for three predictor variables. The parameters block contains the standard regression coefficients but now also has the latent true values for all predictor variables to be estimated by the model. The transformed parameters block implements the non-centered parametrization.

Finally, the model block first places hierarchal normal priors on all true values to prevent unrealistic values from being drawn and cofounding estimation of the regression coefficients. The model then uses the observed values and the observed standard deviations to estimate distributions for the latent true values to be used in the regression model. The model coefficients are given diffuse normal priors to regularize.

3.4.5 Measurement Error Mixed Effects Logistic Regression

Finally, we can combine the above measurement error and hierarchal formulations for our final model. Again, the j subscript indicates a regression coefficient associated with the jth event.

Model 4

Priors

$$\beta_{0,j} \sim \mathrm{Normal}\left(\mu_{\beta_0}, \sigma_{\beta_0}\right)$$

$$\beta_{1,j} \sim \mathrm{Normal}\left(\mu_{\beta_1}, \sigma_{\beta_1}\right)$$

$$\beta_{2,j} \sim \mathrm{Normal}\left(\mu_{\beta_2}, \sigma_{\beta_2}\right)$$

$$\beta_{3,j} \sim \mathrm{Normal}\left(\mu_{\beta_3}, \sigma_{\beta_3}\right)$$

$$\mu_{\beta_0} \sim \mathrm{Normla}(0{,}25)$$

$$\mu_{\beta_1} \sim \mathrm{Normal}(0, 10),\ \mu_{\beta_1} \leq 0$$

$$\mu_{\beta_2} \sim \mathrm{Normal}(0, 10),\ \mu_{\beta_2} \geq 0$$

$$\mu_{\beta_3} \sim \mathrm{Normal}(0{,}10)$$

$$\sigma_{\beta_0} \sim \mathrm{Normal}(0{,}10),\ \sigma_{\beta_0} \geq 0$$

$$\sigma_{\beta_1} \sim \mathrm{Normal}(0{,}5),\ \sigma_{\beta_1} \geq 0$$

$$\sigma_{\beta_2} \sim \mathrm{Normal}(0{,}5),\ \sigma_{\beta_2} \geq 0$$

$$\sigma_{\beta_3} \sim \mathrm{Normal}(0{,}5),\ \sigma_{\beta_3} \geq 0$$

$$CSR_{true} \sim \mathrm{Normal}(\mu_{CSR}, \sigma_{CSR})$$

$$q_{c,1,true} \sim \mathrm{Normal}(\mu_{q_{c,1}}, \sigma_{q_{c,1}})$$

$$r_{f,true} \sim \mathrm{Normal}(\mu_{r_f}, \sigma_{r_f})$$

Data:

$$CSR_{observed} \sim \mathrm{Normal}(CSR_{true}, \tau_{CSR})$$

$$q_{c,1,observed} \sim \mathrm{Normal}(\mathrm{q}_{\mathrm{c,1,true}}\ , \tau_{q_{c,1}})$$

$$r_{f,observed} \sim \mathrm{Normal}(\mathrm{r}_{\mathrm{f,true}}, \tau_{r_f})$$

$$y_{,j} \sim \mathrm{Bernouli\_Logit}(\beta_{0,j} + \beta_{1,j} *\ q_{c,1,true}\ +\ \beta_{2,j} * CSR_{true}\ +\ \beta_{3,j} * R_{f,true}\ )$$

The code for this model is included below:

```
data {
  int <lower = 1> N; //number of data points
  int <lower = 1> E; //number of events
```

```
 int <lower = 1, upper = E>  event[N]; //event id for observations
 vector[N] qc1_obs; //observed values
 vector[N] CSR_obs;
 vector[N] rf_obs;
 vector[N] qc1_sd; //specified measurement noises
 vector[N] CSR_sd;
 vector[N] rf_sd;
 int<lower=0,upper=1> liq[N];
   }

parameters {
 vector[E] b0_raw; //group level parameters
 vector[E] b1_raw;
 vector[E] b2_raw;
 vector[E] b3_raw;
 real <lower = 0> sigma_b0; //hierachal standard deviatons
 real <lower = 0> sigma_b1;
 real <lower = 0> sigma_b2;
 real <lower = 0> sigma_b3;
 real mu_b0; //hierachal means
 real <upper = 0> mu_b1;
 real <lower = 0> mu_b2;
 real mu_b3;
 vector <lower = -0.4, upper = 33>[N] qc1_true; // unknown true value
 vector <lower = -7, upper = -0.2>[N] CSR_true;
 vector <lower = -3 , upper = 2 >[N] rf_true;
}
transformed parameters{
 vector[E] b0;
 vector[E] b1;
 vector[E] b2;
 vector[E] b3;
 b0 = mu_b0 + sigma_b0 * b0_raw;
 b1 = mu_b1 + sigma_b1 * b1_raw;
 b2 = mu_b2 + sigma_b2 * b2_raw;
 b3 = mu_b3 + sigma_b3 * b3_raw;
}
model {
     mu_b0 ~ normal(0,10); //priors for coefficents
     mu_b1 ~ normal(0,10); //diffuse normals
     mu_b2 ~ normal(0,10);
     mu_b3 ~ normal(0,10);
     sigma_b0 ~ normal(0,5);
     sigma_b1 ~ normal(0,5);
     sigma_b2 ~ normal(0,5);
     sigma_b3 ~ normal(0,5);
     b0_raw ~ normal(0,1);
     b1_raw ~ normal(0,1);
```

```
    b2_raw ~ normal(0,1);
    b3_raw ~ normal(0,1);
    qc1_true ~ normal(qc1_pmu, qc1_psd); // prior on measured value
    CSR_true ~ normal(CSR_pmu, CSR_psd);
    rf_true ~ normal(rf_pmu, rf_psd);
    qc1_obs ~ normal(qc1_true, qc1_sd); //measurement model
    CSR_obs ~ normal(CSR_true, CSR_sd);
    rf_obs ~ normal(rf_true, rf_sd);
    for (i in 1:N) {
      liq[i] ~ bernoulli_logit(b0[event[i]] + b1[event[i]]*qc1_true[i] + b2[event[i]]*CSR_true[i]+
b3[event[i]]*rf_true[i]);
    }

}
```

3.4.6 Prior Sensitivity Study

To assess the impact of different prior scales on our models, we repeated each run with prior standard deviations of 10, 25, and 100. The results section discusses the impacts of these changes on the posterior distributions of the regression coefficients and the model's overall predictive performance

3.5 Model Validation Framework

To develop model validation metrics, we ultimately chose to use a single training and testing set consisting of select case histories from the 2011 New Zealand Canterbury earthquake sequence (Green et al., 2014). Figure 38, following, shows the separability between the three predictors and liquefaction for this training set.

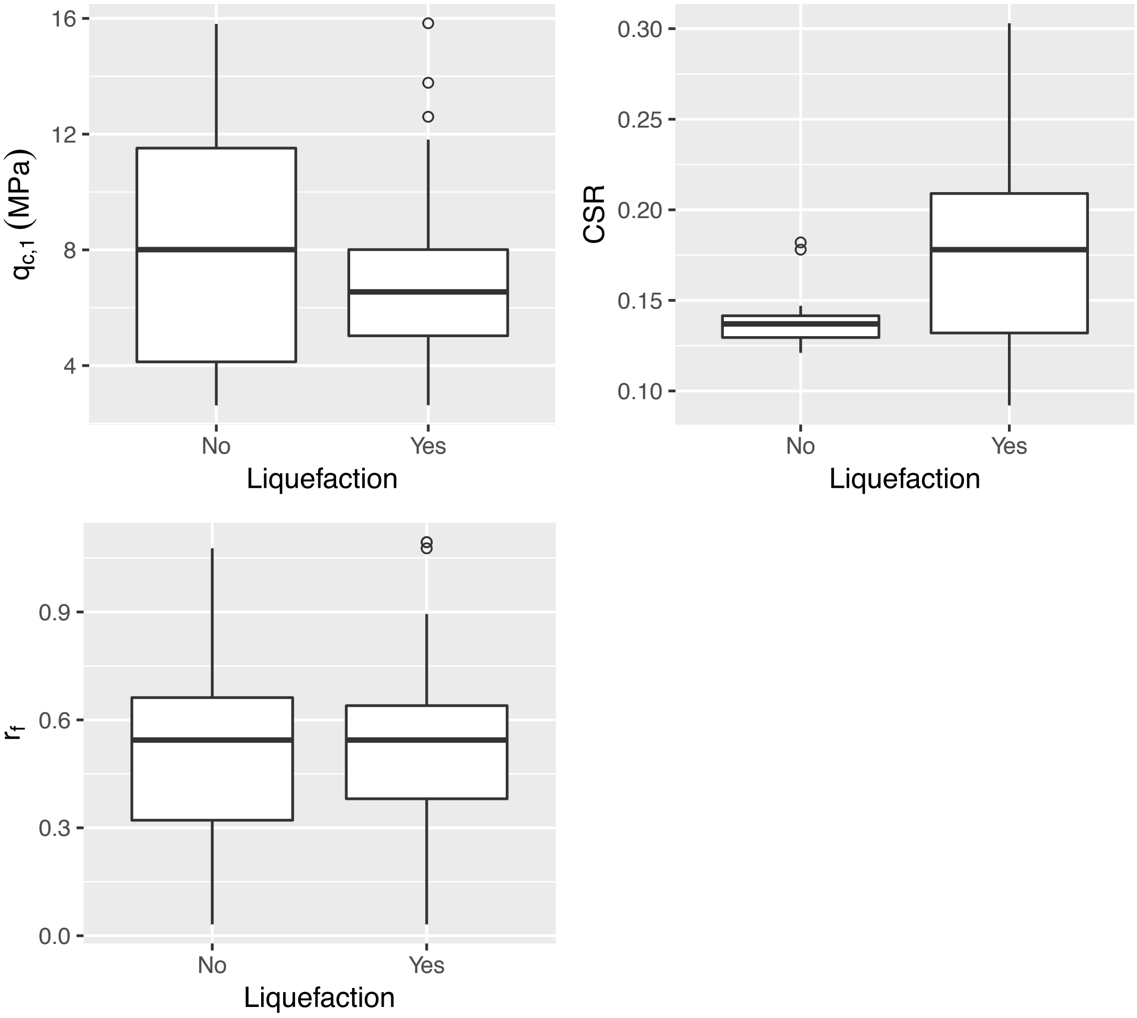


Figure 38 – Class Separability Between $q_{c,1}$, CSR, and R_f for the New Zealand Testing Set.

The range of qc,1 is similar to the original database but both CSR and R_F have considerably less variability. For CSR this would be expected because the data is limited to two events. However, this focus primarily on clean sands which is at odds with typical New Zealand cases which are known for their higher fines contents which may have an impact on apparent model performance.

There are several reasons why we chose a single testing set instead of cross validation. First, because the goal of this study is to assess the effects of modeling choices on model utility rather than provide a complete new model we wanted a completely unbiased evaluation of model performance. Additionally, implementing cross validation for the Bayesian models would require custom coding a train method to interface with Stan. With the measurement error models taking five to ten minutes to run and often requiring several adjustments per run to sampling controls implementing cross validation for these was not worth doing for the scope of this study.

The final step in each model run is to use the New Zealand testing set to validate model performance using ROC charts. We used the ROCR package for this in the code following, but there are several others that have the same capability. Before running the code, the appropriate values for b0, b1, b2, and b3 should be stored from the latest model run.

```
predict_logit <- function(b0,b1,b2,b3,qc1,CSR,rf) {
   return(1/(1+exp(-(b0+b1*qc1+b2*CSR+b3*rf))))
}
l_csr <--0.6
l_qc1 <- 1.0
l_rf <- 0.2
b0 <- coef(m)[1]
b1 <- coef(m)[2]
b2 <- coef(m)[3]
b3 <- coef(m)[4]

nz <- read.csv("nzevents.csv")
p <- predict_logit(b0 = b0, b1 = b1 , b2 = b2, b3 = b3, qc1 = boxcox(nz$qc1,l_qc1), CSR = boxcox(nz$CSR,l_csr), rf = boxcox(nz$rf, l_rf))
pred <- prediction(p, nz$Liq)
rocc <- performance(pred, measure = "tpr", x.measure = "fpr")
plot(rocc, colorize = T)
abline(a=0, b=1, lty=2, lwd=2, col="black") #45 degree line represents performance of a 'coin flip' model
abline(h = 1, lty = 3, lwd=2, col="black")
abline(v = 0, lty = 3, lwd=2, col="black")
auc.perf <- performance(pred, measure = "auc")
text(0.75,0.4, "AUC = ")
text(0.75,0.3,labels = round(auc.perf@y.values[[1]],3))
```

To create a ROC curve in the rocr package several functions need to be called. The prediction function takes vectors of predicted probabilities and true class labels. The performance function takes the output of this function and produces the performance metrics, in our case true and false positive rates. We then plot the results with the curve colorize according to the current threshold. Finally, we again call the performance function to compute the AUC and add it to the chart.

4 RESULTS

The following chapter presents the results of our modeling process. The first section summarizes each model run and predictive performance. The second section gives our metrics for model uncertainty and discusses how the models perform compared to each other. The final section discusses the effects of different prior choices on model parameter distributions and predictive performance.

In this chapter we discuss the following models:

- Model 0 – Maximum likelihood model
- Model 1 – Baseline Bayesian model
- Model 2 – Hierarchal Bayesian model
- Model 3 – Bayesian measurement error model
- Model 4 – Combine measurement error and hierarchical model

We will refer to models with three predictor variable with a -3 after the model number. For example, model 2-3 refers to the hierarchal Bayesian model with all three predictor variables.

4.1 Model Summaries

Recall that for a Bayesian analysis the result is a posterior distribution. In our case, the posteriors of interest are those of the regression coefficients ($\beta_0, \beta_1, \beta_2$, and β_3). For the hierarchal models we are mainly interested in the population mean values for the coefficients (μ_β) which will have their own posterior distributions. We are also interested in making point estimates of probability. We can do this by fixing regression coefficients to their mean values and using the following equation (assuming two predictor variables):

$$P_L = \frac{1}{1 + \exp\left\{-(\beta_0 + \ \beta_1 * q_{c,1}^* + \beta_2 * CSR^* \,)\right\}}$$

where

$$q^*_{c,1} = \frac{{q_{c,1}}^{1.6} - 1}{1.6}$$

$$CSR^* = \frac{CSR^{-0..6} - 1}{-0.6}$$

The output of this function is a logistic surface that lives above the $q_{c,1}{}^*$, CSR* space. To visualize we can reverse the Box-Cox transformations and plot contours of probability. For the three variable case, the equation for calculating probability of liquefaction becomes:

$$P_L = \frac{1}{1 + \exp\{-(\beta_0 + \ \beta_1 * q^*_{c,1} + \beta_2 * CSR^* + \beta_3 * R^*_f))\}}$$

where

$$q^*_{c,1} = \frac{{q_{c,1}}^{1.0} - 1}{1.0}$$

$$CSR^* = \frac{CSR^{-0..6} - 1}{-0.6}$$

$$R^*_f = \frac{R_f^{0.2} - 1}{0.2}$$

To visualize the three variable models, we can fix R_f at 2.5th, 50th (median) and 97.5th percentile values and plot the probability contours for each case to visualize how the curves shift. Finally, the model summaries also include the ROC curve from testing the model on the New Zealand training set and its AUC. To facilitate model comparison we present all the graphical summaries in succession, then discuss the key points in the sections following (Figures 39 through 51). In each graphic, the histograms summarize the posterior distributions for the regression intercept and the two slopes showing mean, standard deviation and the 2.5th, 50th (median), and 95th percentiles indicated by dashed lines. The scatterplot shows the probability contours resulting from the mean values of regression coefficients. These mean values are also used to generate the ROC curve and evaluate model predictive performance. For three variable models, the left is for the full three variable model and the curve on the right is for Rf fixed at its median value.

Table 2 – Model 0 (Maximum Likelihood Coefficient Estimates).

Parameter	Estimate	Standard Error
Intercept (β_0)	9.32	1.63
$q_{c,1}$ Slope (β_1)	-0.19	0.038
CSR Slope (β_2)	1.72	0.350

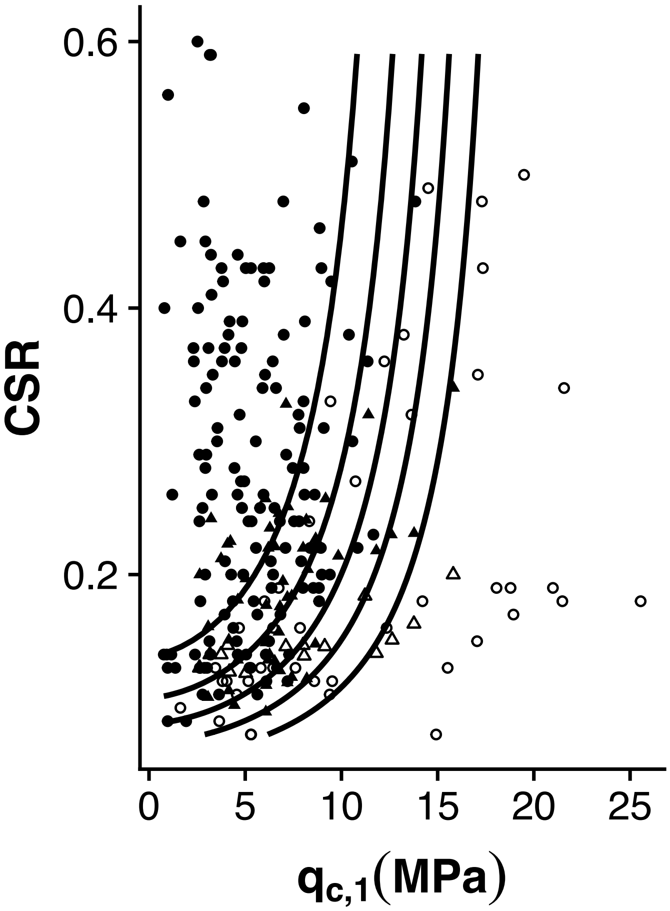


Figure 39 - Contours of 5, 20, 50, 80, and 95% Liquefaction Triggering Probability. Closed and open circles are liquefied and nonliquefied case histories, respectively, from the Moss et al., 2006 database. Closed and open triangles are liquefied and nonliquefied case histories, respectively, from the Green et al., 2011 validation database.

Model 1 – Baseline Bayesian

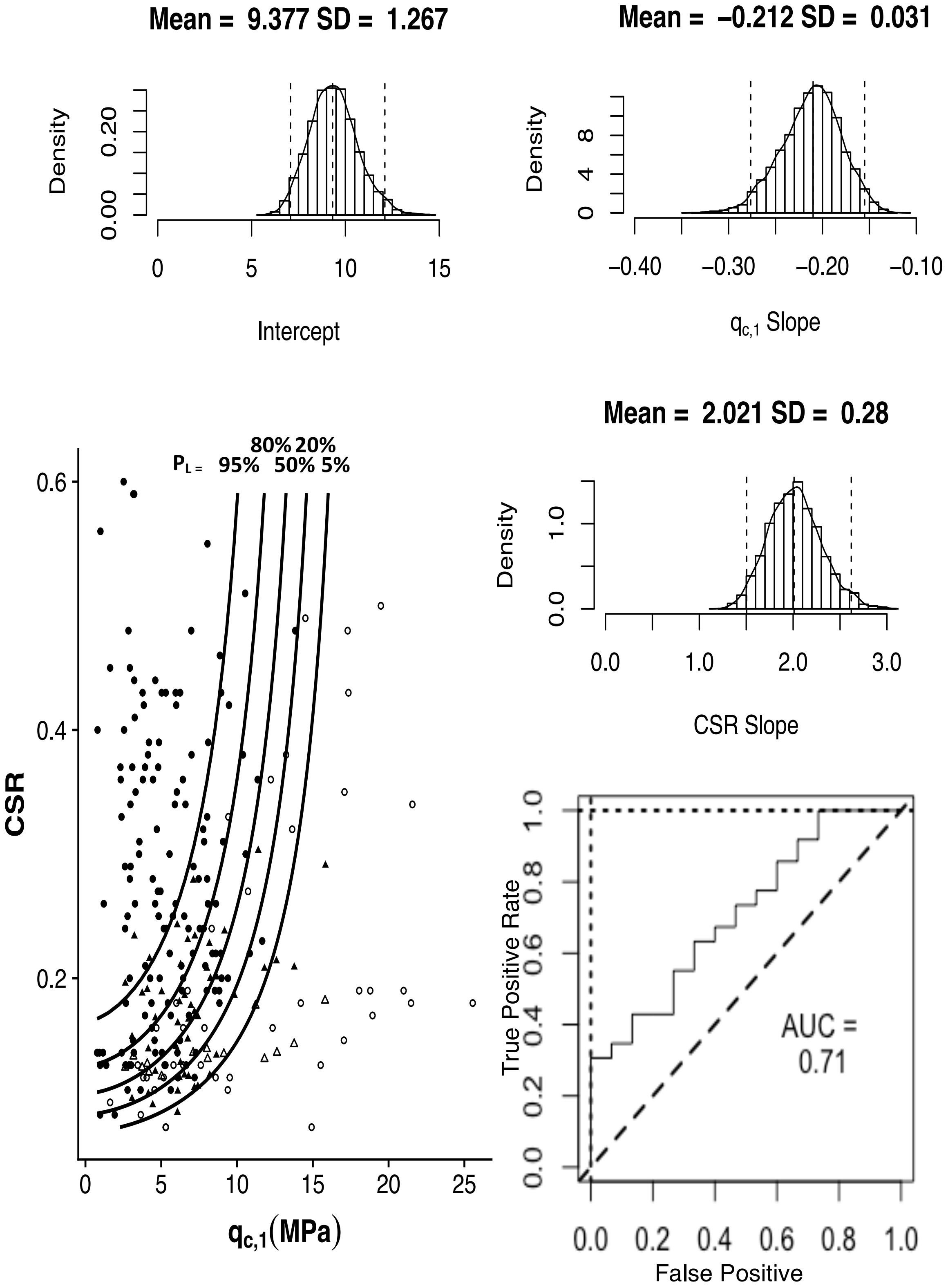


Figure 40 – Model 1 Summary.

Model 2 –Bayesian hierarchal

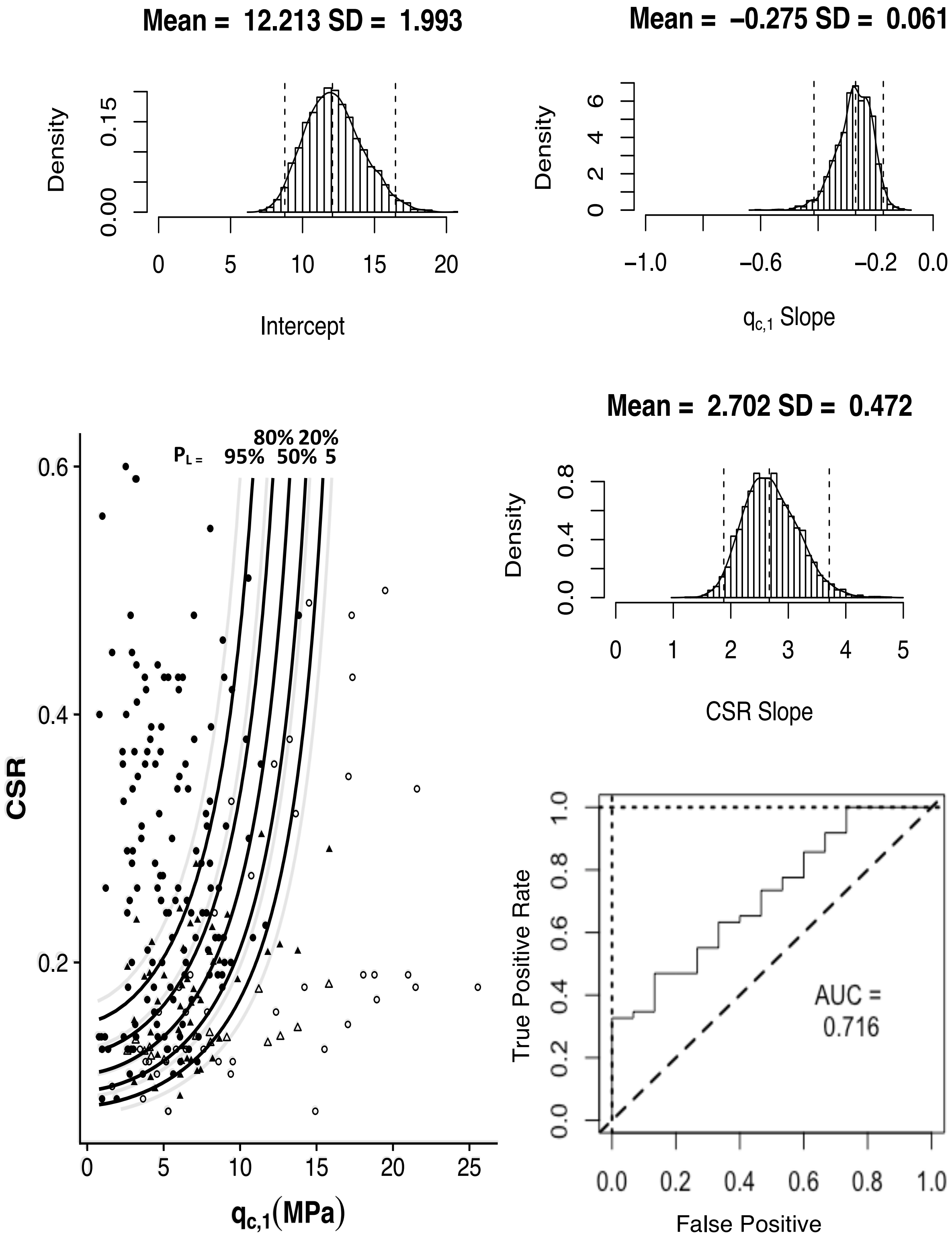


Figure 41 – Model 2 Summary. The distributions shown are for the population mean value parameters estimated from the hierarchal model.

Model 3 – Bayesian measurement error

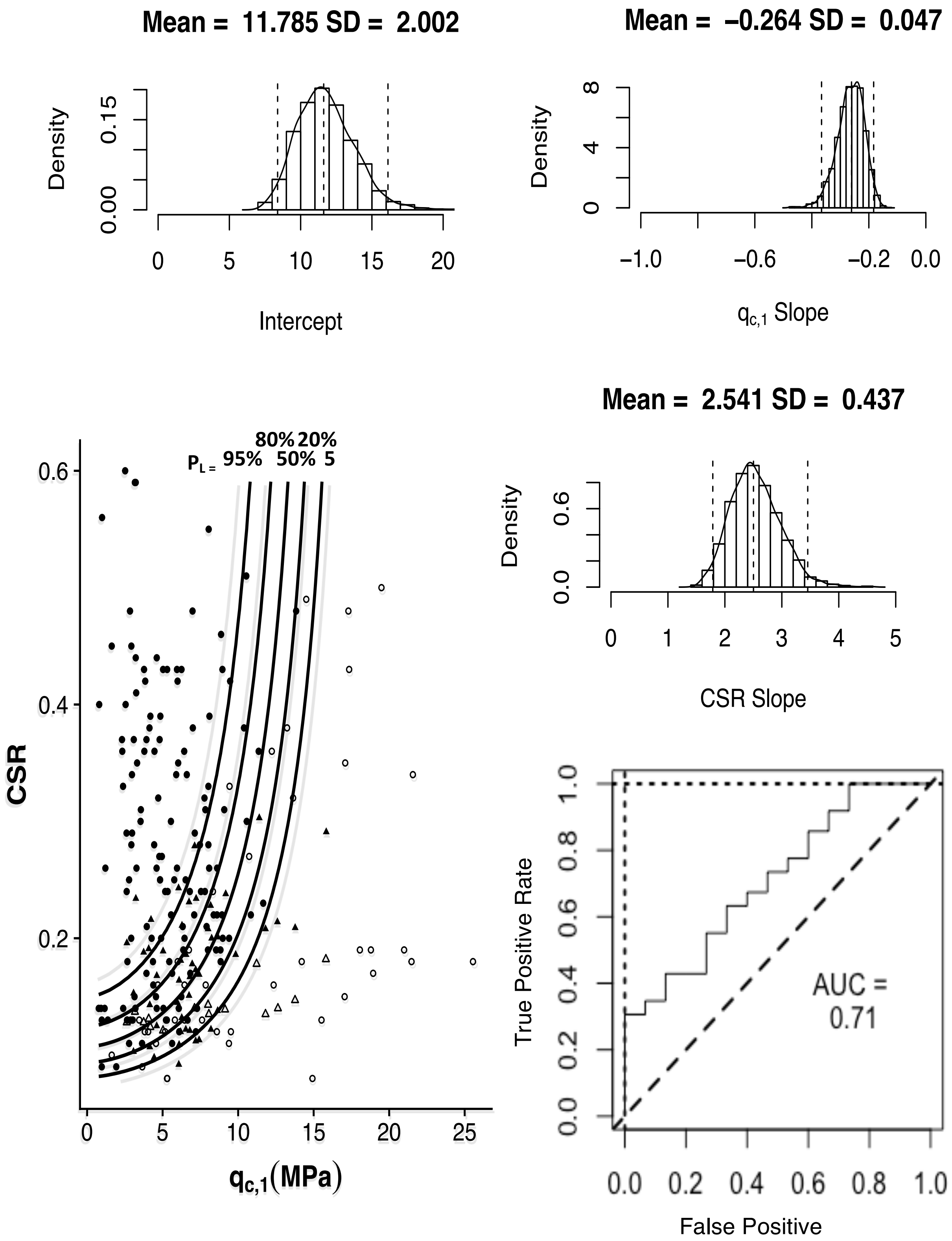
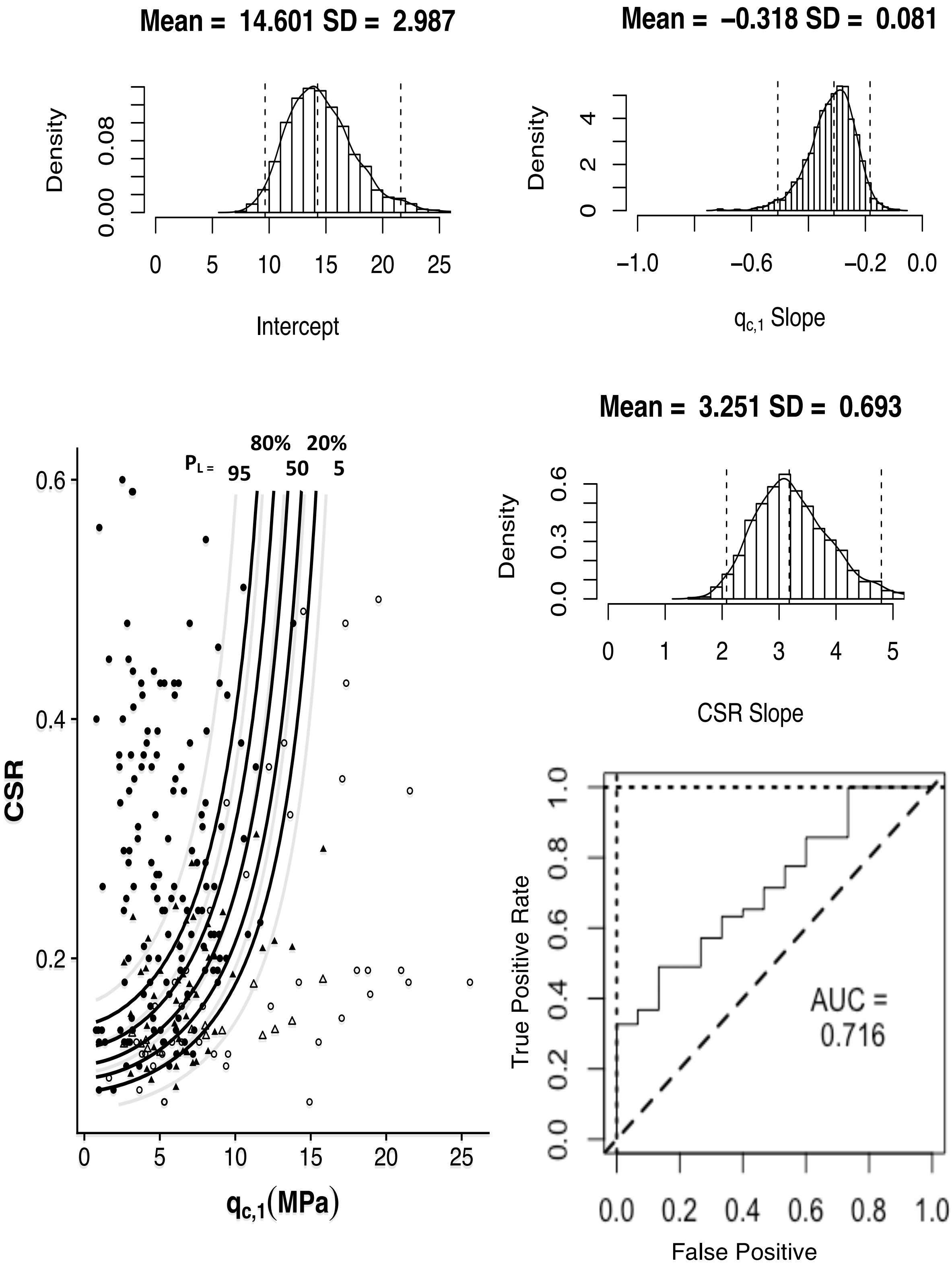


Figure 42– Model 3 Summary.

Model 4 –Bayesian combined



Mean = 14.601 SD = 2.987

Mean = –0.318 SD = 0.081

Mean = 3.251 SD = 0.693

Figure 43 – Model 4 Summary. Similar to Model 2, the distributions shown are for the population mean value parameters estimated from the hierarchal model.

Model 1-3 –Baseline three variable model

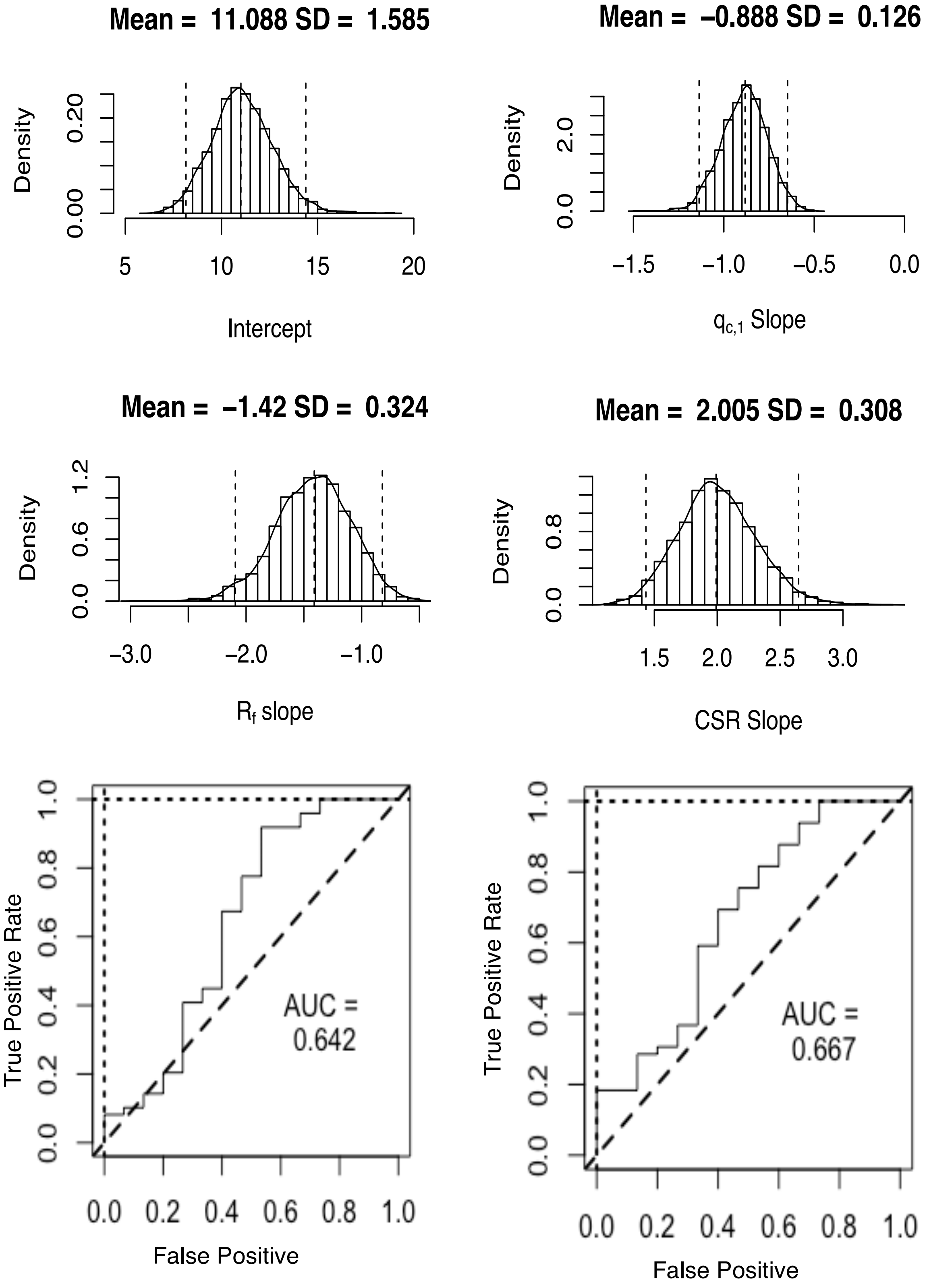


Figure 44 – Model 1-3 Summary.

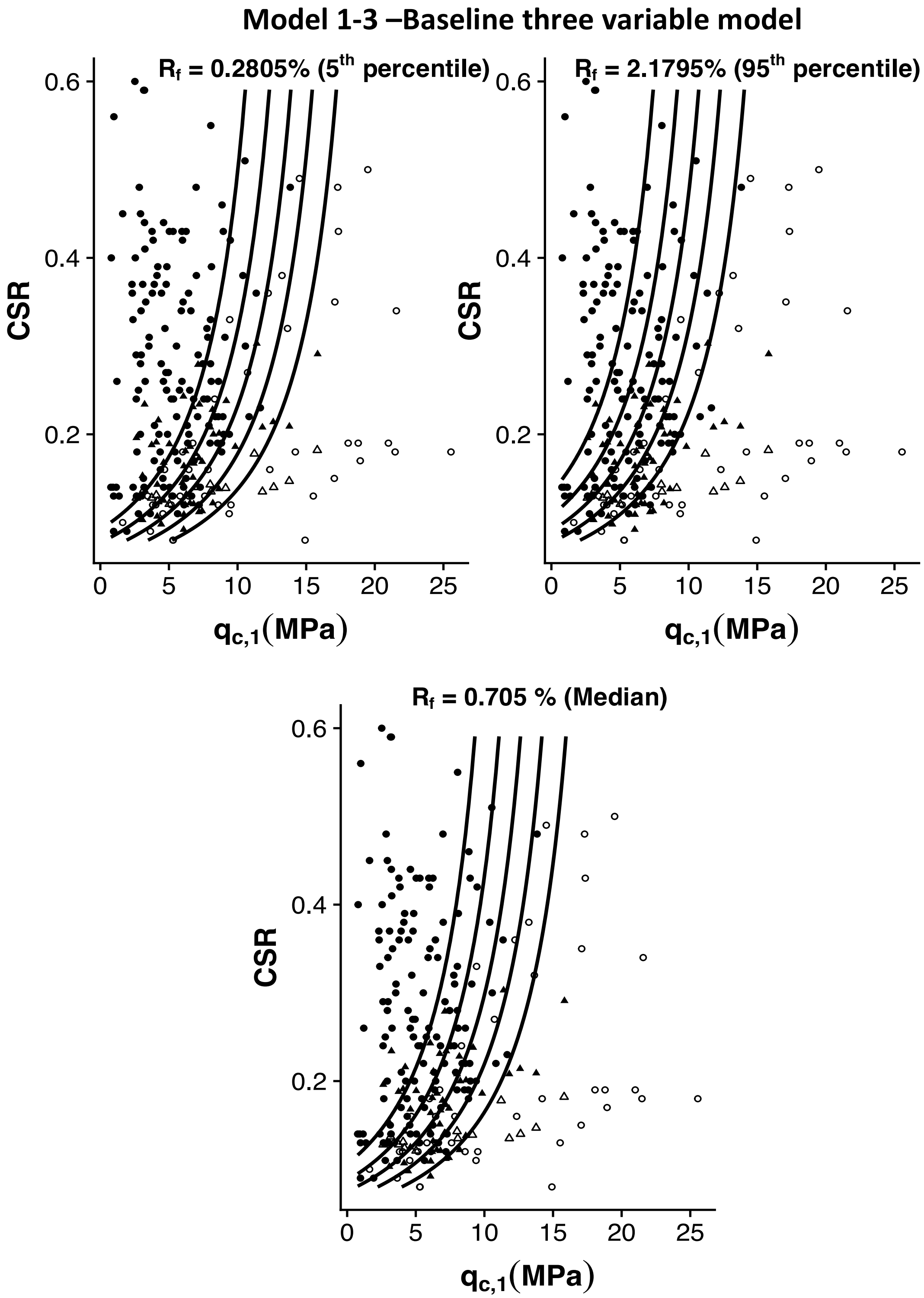


Figure 45 – Graphical Triggering Curves for Model 1-3. The contours of probability plotted are 95%, 80%, 50%, 20% and 5%, left to right.

Model 2-3 – Hierarchal three variable model

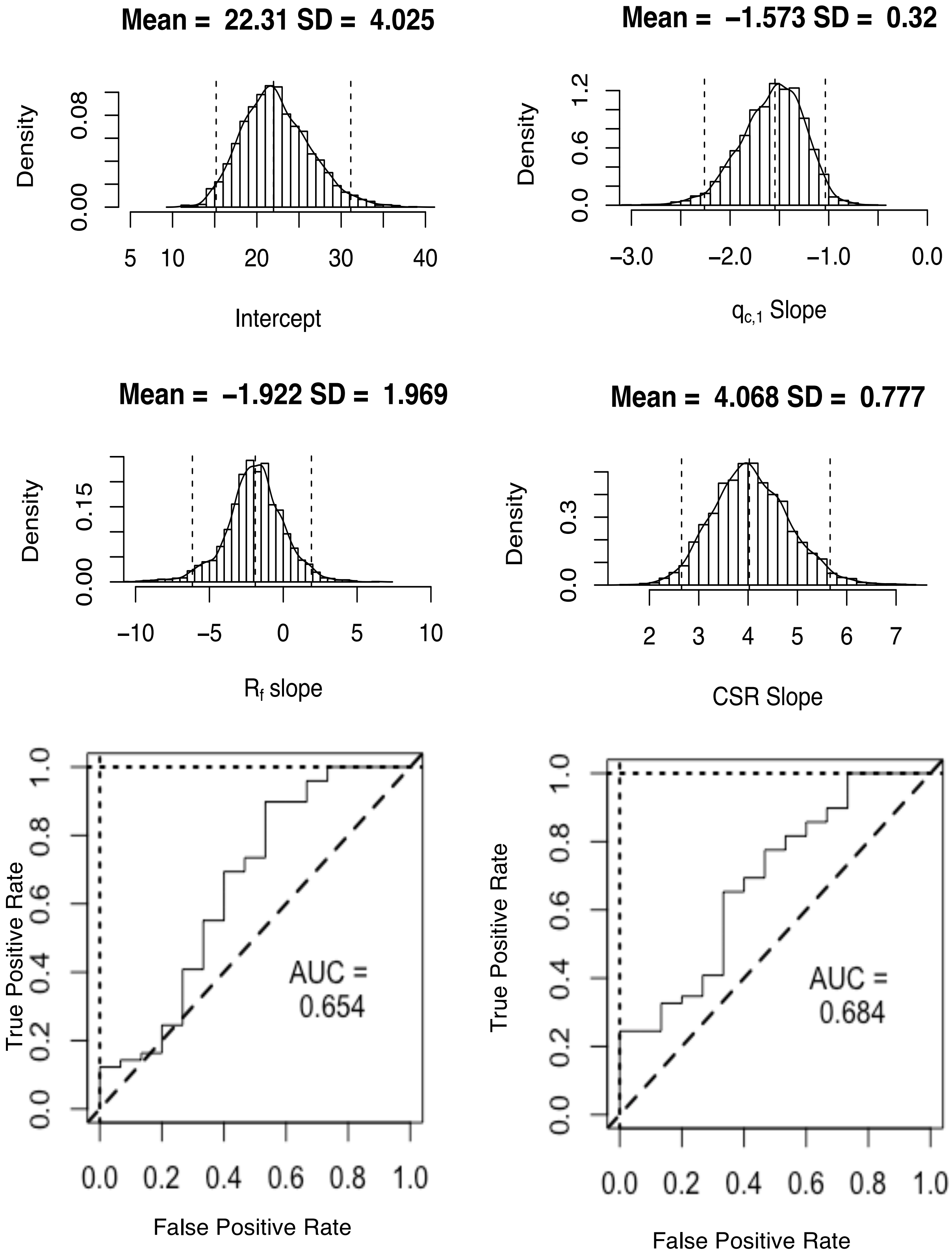
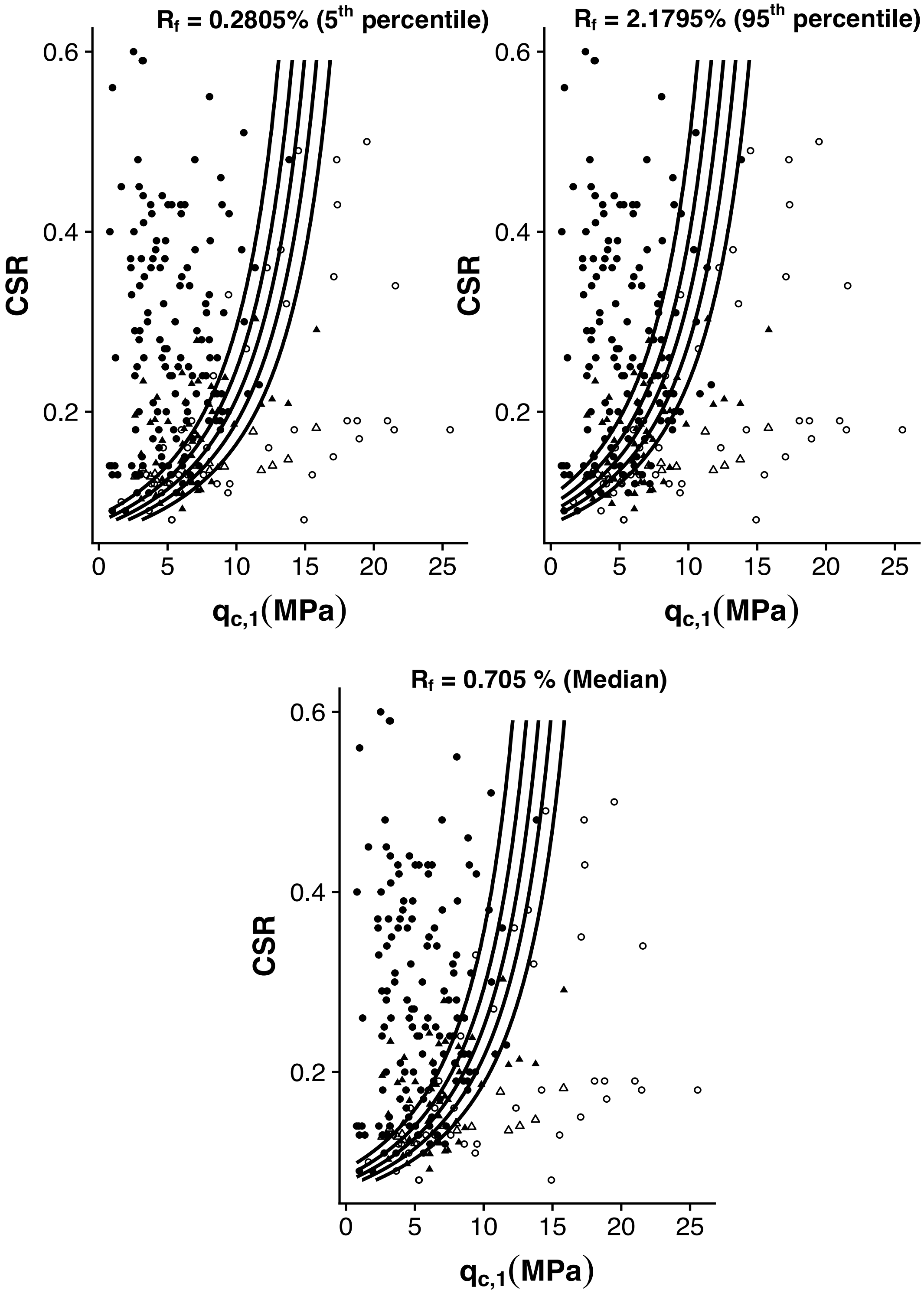


Figure 46 – Posterior Distributions and Predictive Performance for Model 2-3.

Model 2-3 –Hierarchal three variable model



R_f = 0.2805% (5th percentile)

R_f = 2.1795% (95th percentile)

R_f = 0.705 % (Median)

Figure 47 – Graphical Triggering Curves for Model 2-3.

Model 3-3 –Measurement error three variable

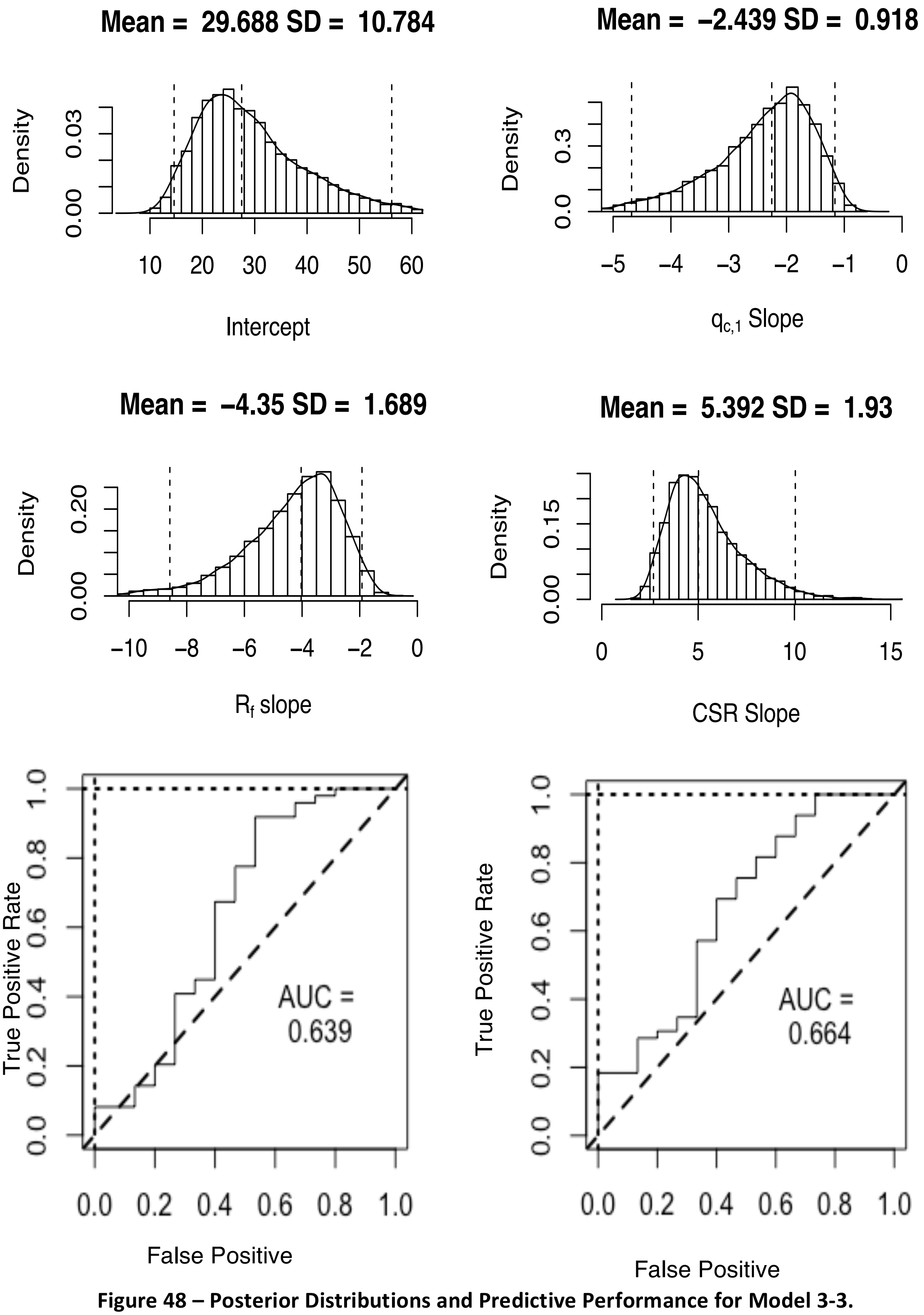
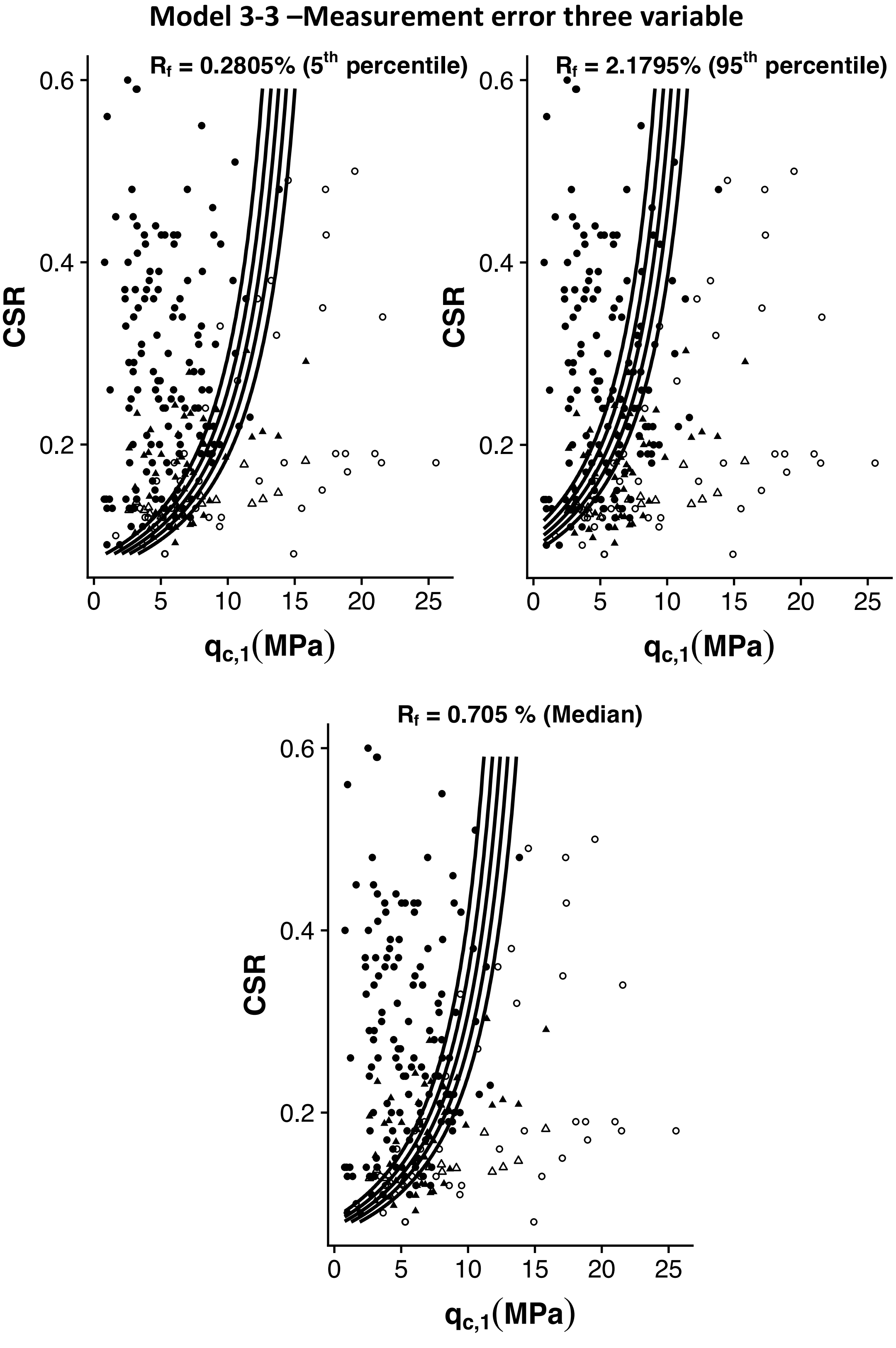


Figure 48 – Posterior Distributions and Predictive Performance for Model 3-3.



Model 3-3 –Measurement error three variable

R_f = 0.2805% (5th percentile)

R_f = 2.1795% (95th percentile)

R_f = 0.705 % (Median)

Figure 49 – Graphical Triggering Curves for Model 3-3.

Model 4-3 – Combined three

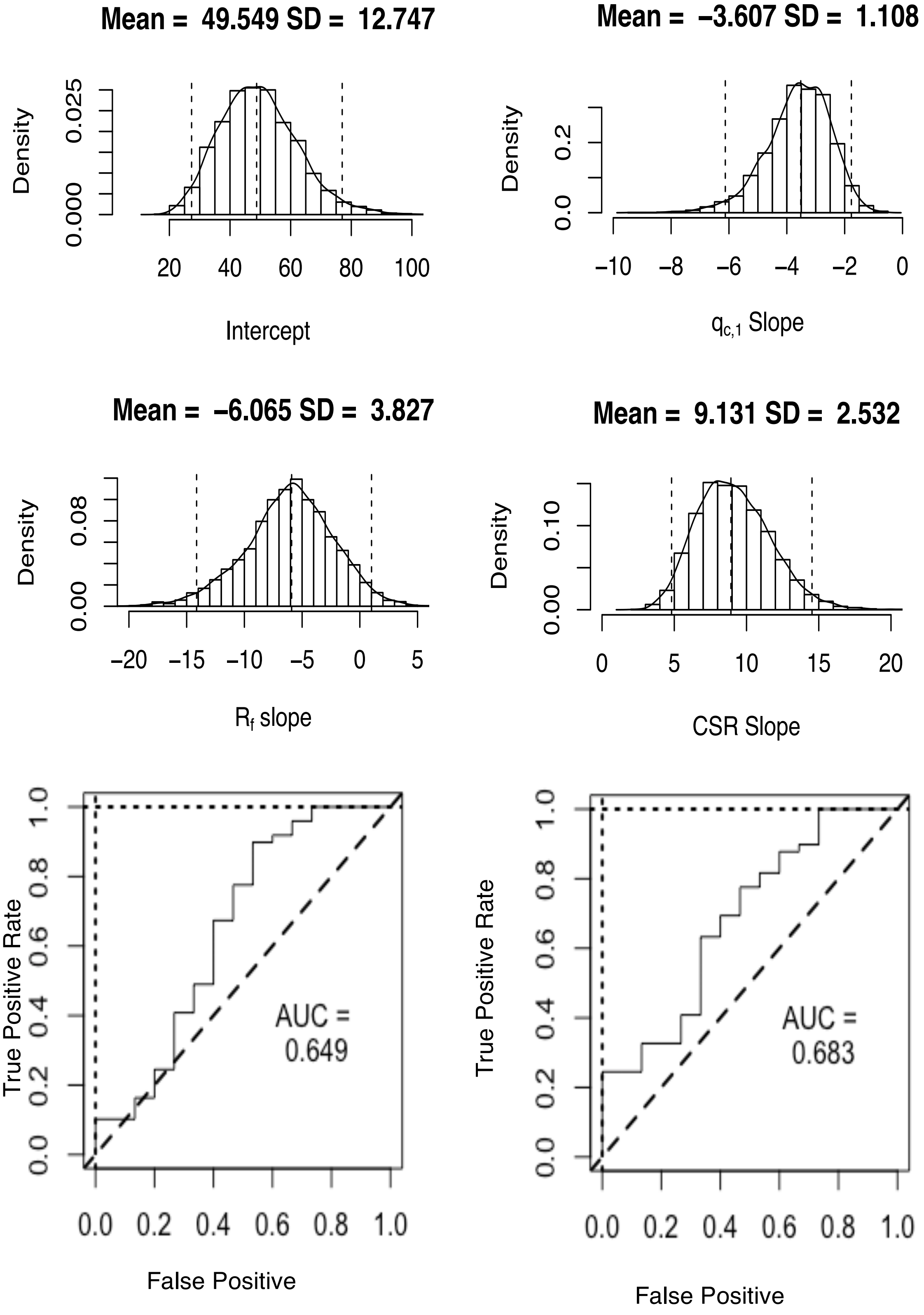


Figure 50 – Posterior Distributions and Predictive Performance for Model 4-3.

Model 4-3 – Combined three variable

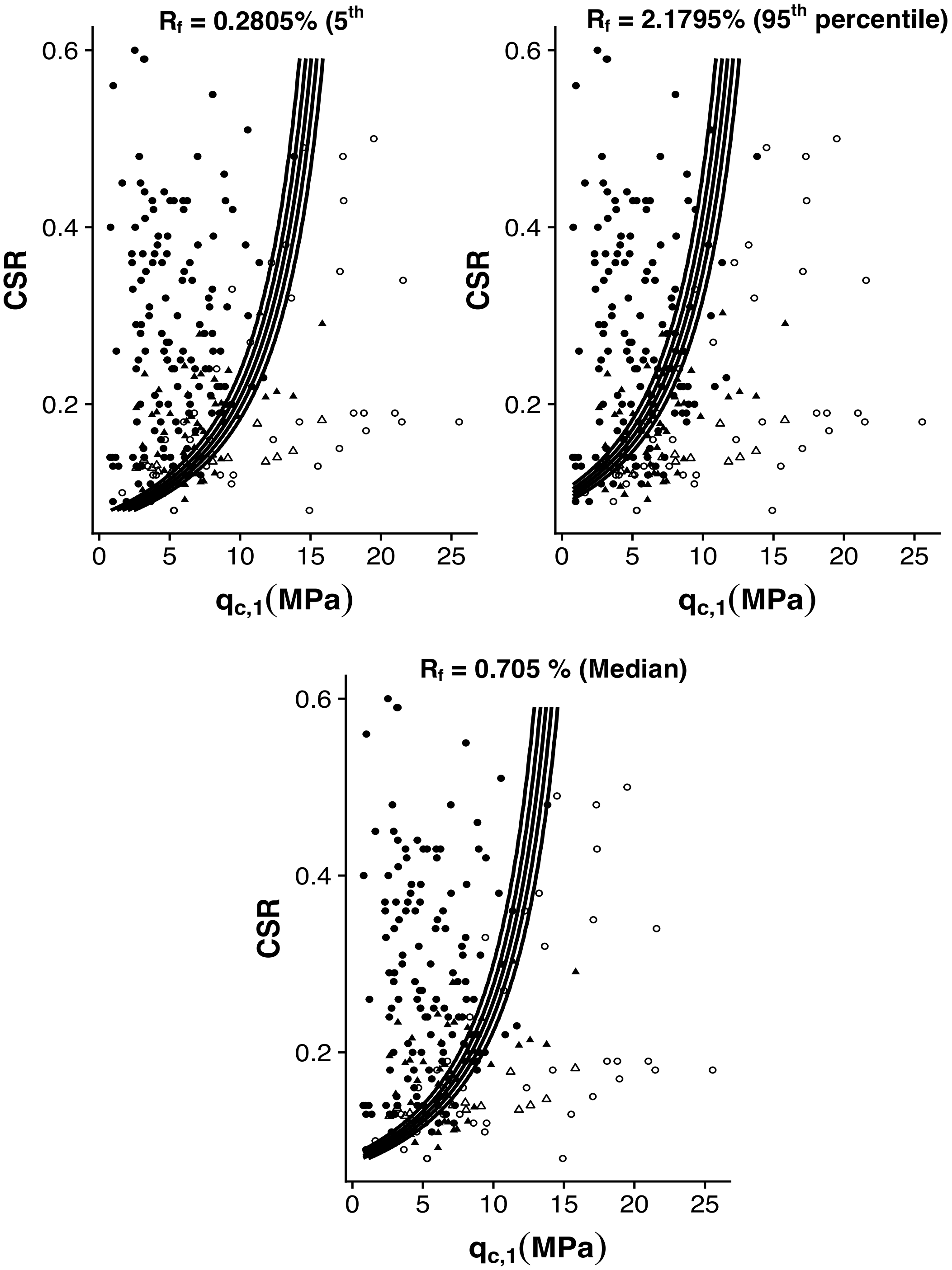


Figure 51 – Graphical Triggering Curves for Model 4-3.

4.2 Discussion of Model Summaries

Model 2 has a slightly higher AUC and coefficient standard deviations than the baseline case. Compared to the baseline curves plotted in light gray the spread of the probability contours has decreased similar to Model 3. However, for Model 3 because the shift is relatively small the AUC does not increase compared to baseline and is slightly less than Model 2. The standard deviations of the regression coefficients for Model 3 are slightly larger than the baseline model but smaller than Model 2. Finally, compared to all the previous models Model 4 has the most "squishing" of the probability contours and a tie for the highest AUC. But, it also has the largest standard deviation for model parameters thus far.

When including 3 predictor variables predictive performance decreased appreciably. For Model 1-3, both the full model and fixing R_f to its median value have a noticeably less AUC than any of the two predictor variable models, though fixing R_f outperforms the full model. Its posterior standard deviations are comparable to the 2 predictor case. Visually, as R_f increases the curves shift to the left indicating a generally lower probability of liquefaction – consistent with the belief that increased fines content will suppress liquefaction triggering. Model 2-3 has posterior standard deviations larger than Model 1-3 and its predictive performance is worse than any of the two variable models. However, its AUC is higher than model 1-3. Similar to the two variable case its probability curves have shrunk closer together relative to the three variable baseline The trend of increasing parameter posterior uncertainty continues with Models 3-3 and 4-3. They also have worse predictive performance than the baseline 3 variable case, although Model 4-3 has a higher AUC than 3-3. Both of these models exhibit dramatic shrinkage of their contours of probability, indicating low apparent model uncertainty. However, because their predictive performance is generally poor these models are "over confident", in a sense.

4.3 Defining and Assessing Model Uncertainty

To assess model uncertainty, we first consider what we will refer to as a model's "apparent uncertainty". For a point estimate of liquefaction triggering the uncertainty of that assessment is directly related to its predicted probability. As touched upon in the background section, a binary outcome that occurs with probability p has variance $p(1-p)$ or standard deviation $\sqrt{p(1-p)}$. This formula expresses that we are more confident about an event's occurrence the higher probability we assign to it or vice versa. Graphically this appears as the probability curves shifting closer together – which indicates more space being assigned very high or very low probabilities. To assign a numerical value to this, we can also define a σ_{50} as the median computed standard deviation of an evenly spaced grid of $q_{c,1}$, CSR pairs across the domain covered by the database. This metric summarizes how confident the models point estimates of probability are, on average.

However, we are also interested in a probabilistic interpretation of model uncertainty. Because the results of each model run are posterior distributions of the model coefficients we can treat the equation for probability of liquefaction as a function of random variables and use simulation to approximate a predictive distribution for probability of liquefaction, given a CSR and $q_{c,1}$ input (Figures 52 and 54). This distribution will also have a standard deviation associated with it, which we will refer to as the "posterior predictive" uncertainty. Again, we can define a σ_{50} as the median computed standard deviation of the simulated posteriors corresponding to evenly spaced grid of $q_{c,1}$, CSR pairs across the domain covered by the database. Additionally, we plot a sampling of the probable median curves for each model to visualize the posterior predictive uncertainty (Figure 53 and 55). The mean trend is shown in black and the sample lines are drawn in grey, shaded corresponding to their probability density.

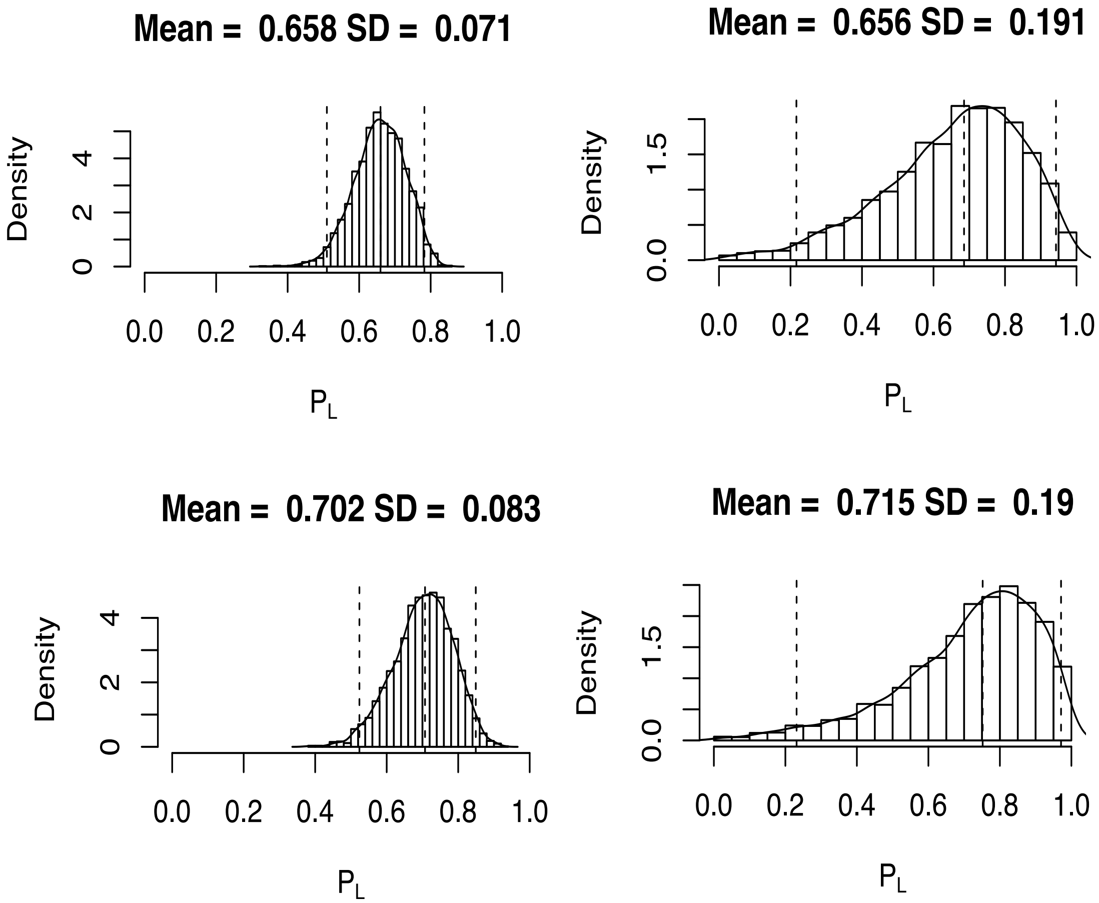


Figure 52 – Sample Posterior Predictive Distributions for Models 1 - 4, Left to Right Top to Bottom.

In these examples $q_{c,1}$ = 10 MPa and CSR = 0.3. These values were deliberately chosen because they lie in the intermediate region where there is noticeable mixing between liquefied and nonliquefied case histories. Models 1 and 3 have very similar predictive distributions, while models 2 and 4 have a much wider spread of possible probabilities. This corresponds roughly with the standard deviations of their coefficient posterior distributions – Model 1 had the tightest, followed by 3 then 2 then 4.

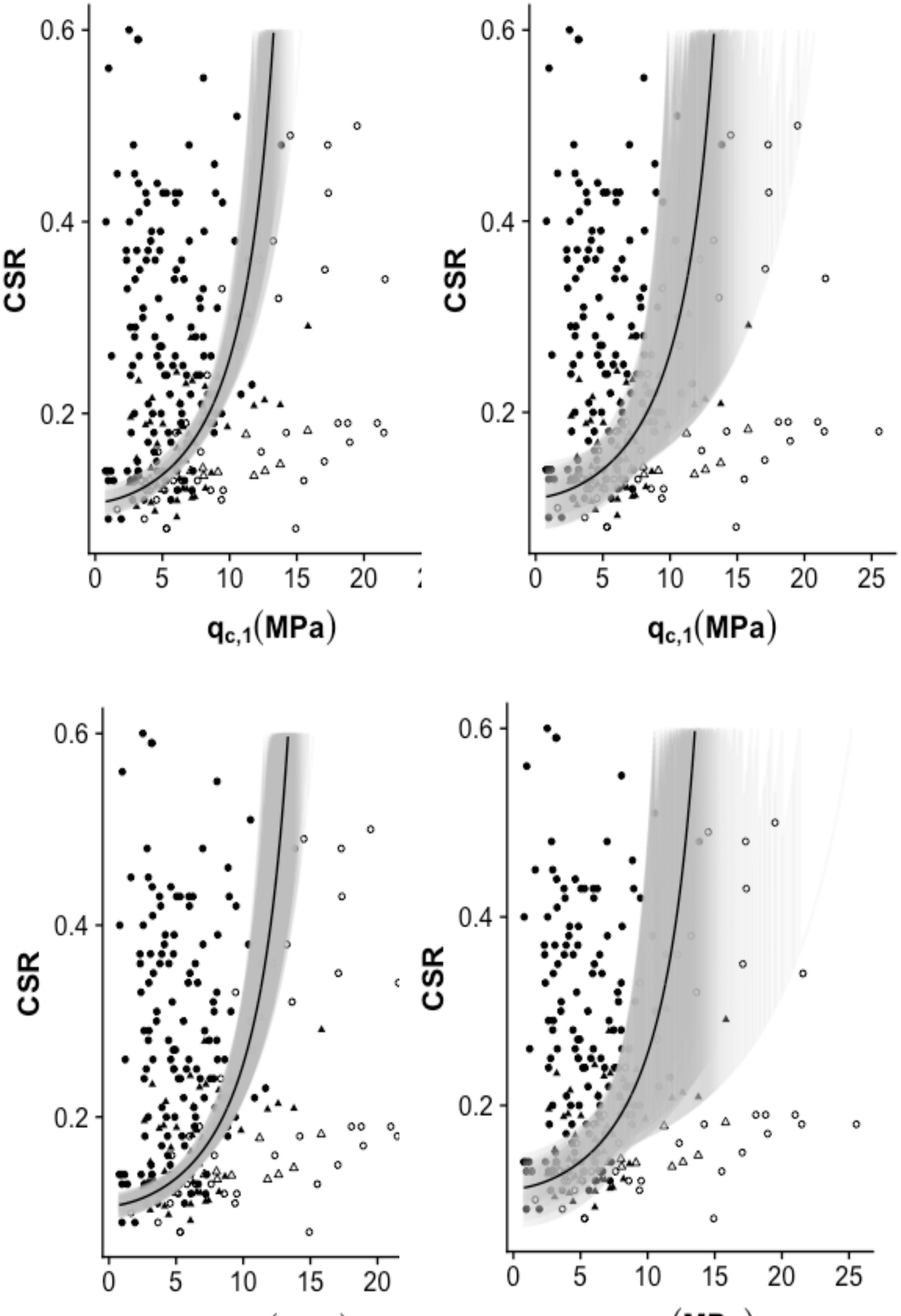
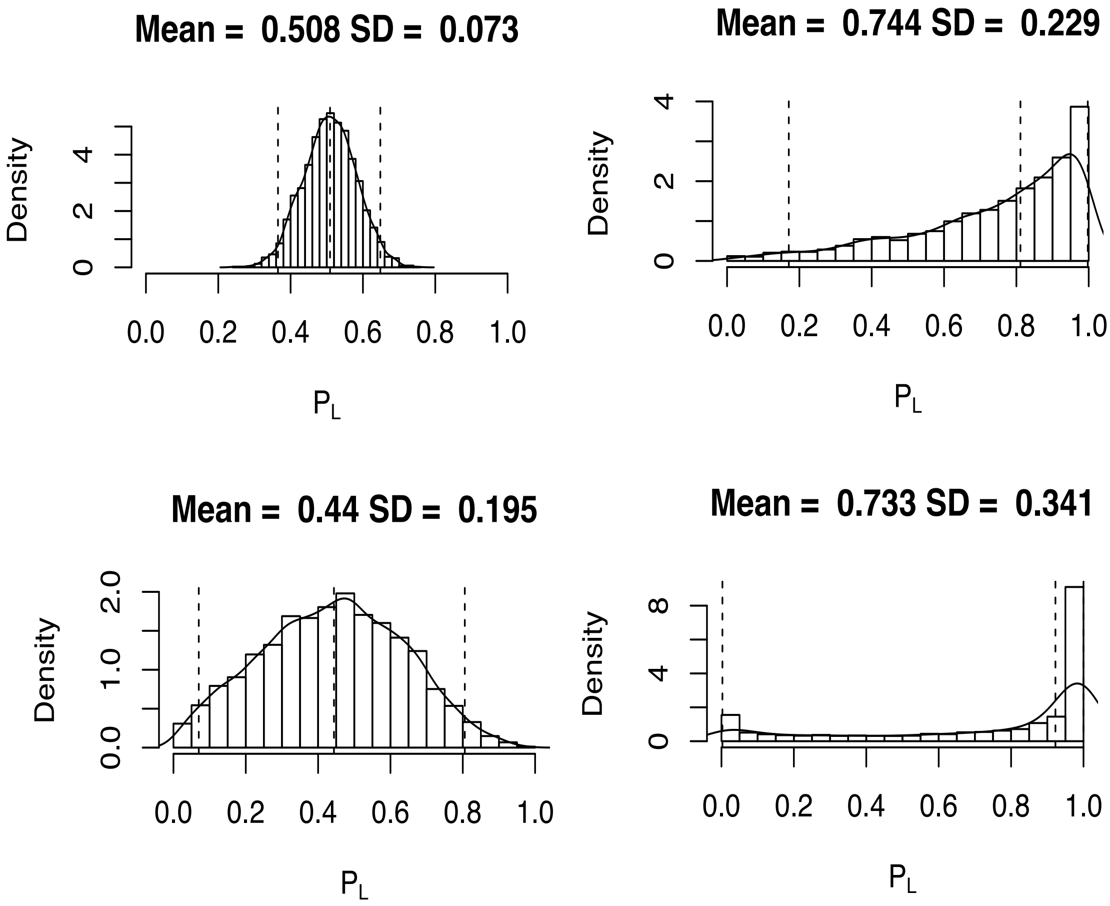


Figure 53 – Visualizing the Posterior Uncertainty of the 50% Probability Contour Location. Models 1 through 4 are shown left to right, top to bottom. Models 1 and 3 have a similar spread while the hierarchal models (2 and 4) are more diffuse.


Mean = 0.508 SD = 0.073

Mean = 0.744 SD = 0.229

Mean = 0.44 SD = 0.195

Mean = 0.733 SD = 0.341

Figure 54 – Sample Posterior Predictive Distributions for Models 1-3 Through 4-3, Left to Right Top to Bottom.

In these examples $q_{c,1}$ = 10 MPa , CSR = 0.3, and R_f = 0.705% as before**.** The standard deviations of these predictions are noticeably larger than the two variable case as a result of the higher posterior standard deviations associated with the regression coefficients. However, they still follow the general trend of standard deviation increasing 1 to 3 to 2 to 4.

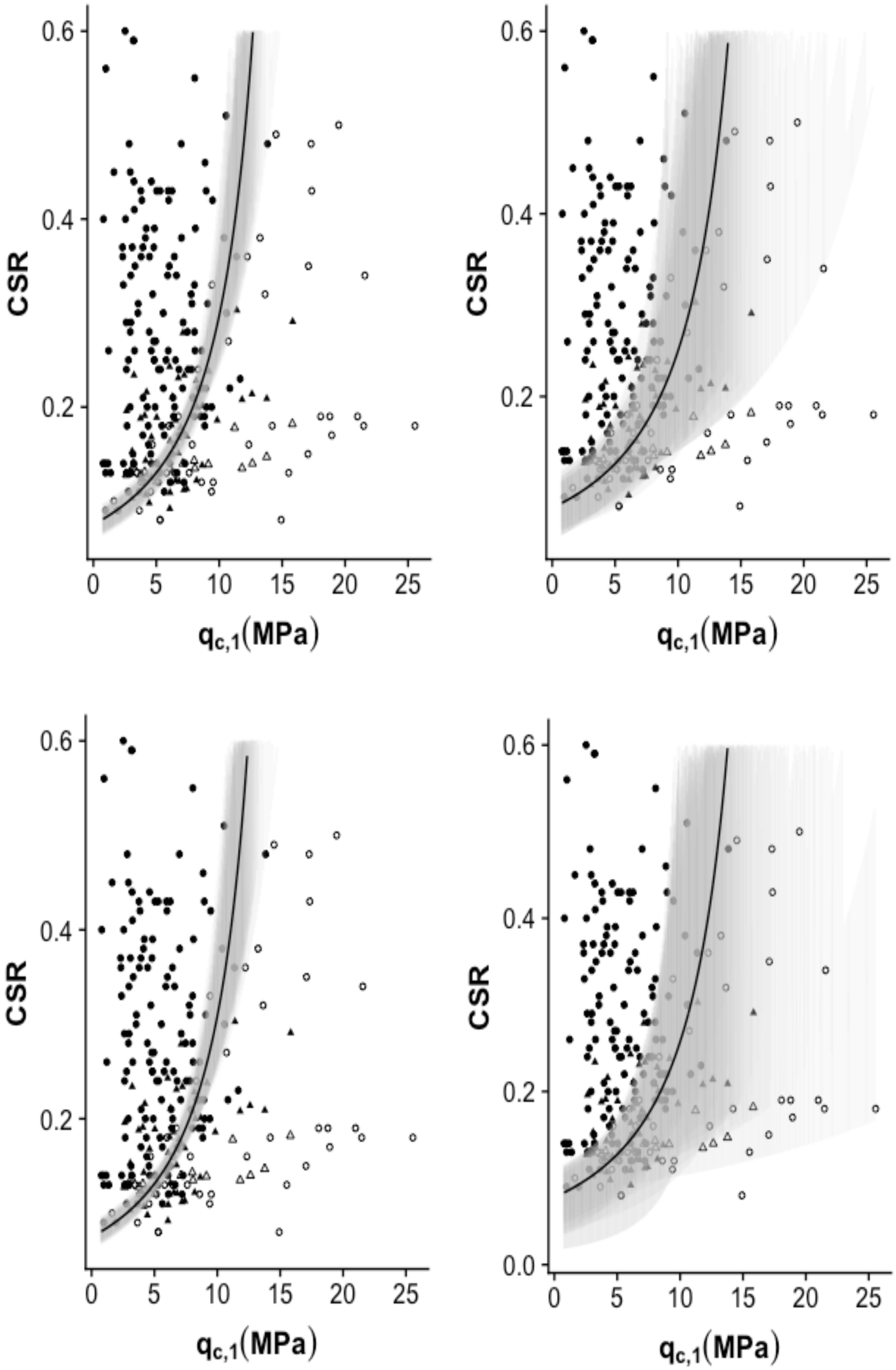


Figure 55 – Visualizing the Posterior Uncertainty of the 50% Probability Contour Location for the Three Variable Case.

Models 1-3 through 4-3 are shown left to right, top to bottom. The mean trend is shown in black and the sample lines are drawn in grey, shaded corresponding to the probability density. The spread of the curves follows similar trends to the two variable case. The odd shape is likely due to the how diffuse the posteriors for this model are, allowing for curves that do not follow the same trend.

Table 3 – Numerical Summaries of Model Uncertainty. The models denoted x-3* are the results for the three variable case with Rf fixed at its median value. The two and three variable are only compared to their own baselines i.e. model 2-3* is compared to 1-3* not 1-3 or 1.

Model	Apparent σ_{50}	% Difference from baseline	Posterior Predictive σ_{50}	% Difference from baseline
1	1.14E-02	--	3.67E-04	---
1-3	3.71E-02	--	2.68E-03	---
1-3*	3.13E-02	--	2.22E-03	---
2	2.97E-03	-117%	2.26E-03	144%
2-3	3.01E-03	-170%	1.75E-04	-176%
2-3*	2.63E-03	-169%	1.08E-02	132%
3	3.83E-03	-99%	1.42E-04	-89%
3-3	1.17E-04	-199%	3.61E-02	172%
3-3*	8.00E-05	-199%	1.52E-04	-174%
4	1.10E-03	-165%	7.01E-03	180%
4-3	1.63E-06	-200%	4.63E-02	178%
4-3*	1.19E-06	-200%	2.76E-02	170%

4.4 Overall Discussion of Model Performance

When considering all the models three major trends are apparent:

- Two predictor variables always outperform three predictor variables
- The measurement error and hierarchal models have appreciably less apparent uncertainty than the baseline models
- This added complexity often comes at the cost of less certain coefficient posterior distributions and increased predictive uncertainty.

The following sections will discuss these trends in further detail.

4.4.1 Comparing Three and Two Predictor Variables

The lower than expected predictive performance of the three variable models can be interpreted in several ways. Because The New Zealand testing set is mostly clean sand cases it is possible that adding R_f to the model does not generalize well to soils with low apparent fines content. Or, it can simply be that R_f as a predictor variable does not generalize well to new data, period.

Additionally, R_f and $q_{c,1}$ are correlated because R_f is a function of tip resistance and because R_f is an input to the equation for the $q_{c,1}$ normalization exponent (Figure 56). When predictor variables are correlated the posterior will have a high spread because many logistic surfaces can fit the data (Kruschke, 2015). This may contribute to these models' poor performance because the resulting posteriors may be too diffuse for the model to identify model coefficients.

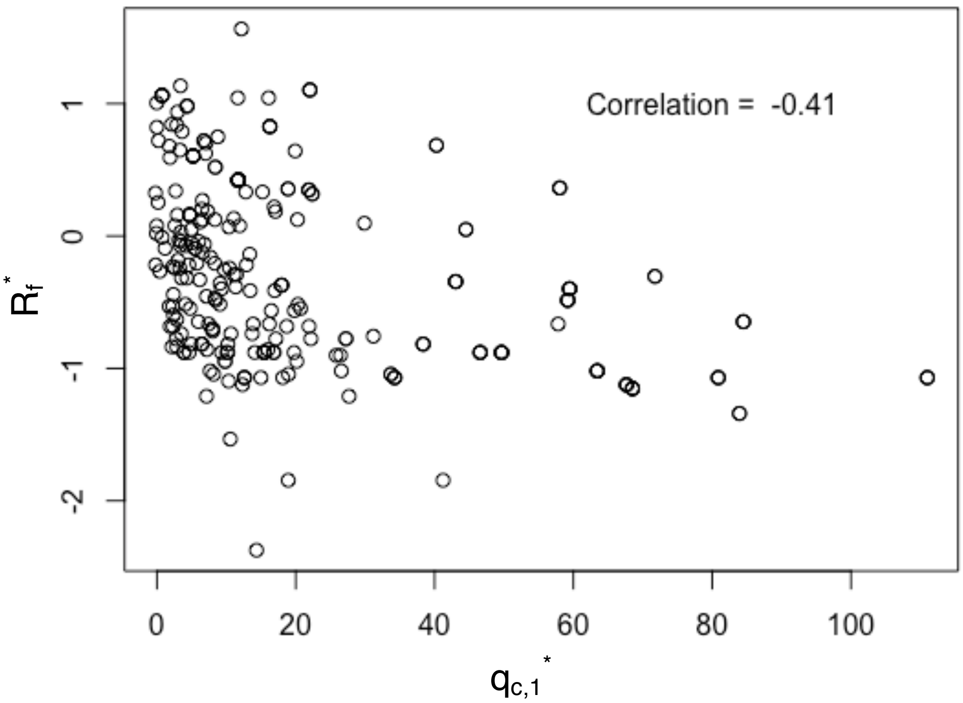


Figure 56– Scatterplot Showing Correlation of Transformed $q_{c,1}$ and R_f

These results illustrate the importance of distinguishing between model fit to the training data and its actual predictive performance. The AIC based selection indicated that there is statistical utility in including R_f as a predictor variable, but this metric is based off the model fit to the training data. When actually considering the model's predictive performance on a testing set the two variable cases consistently outperform the variable cases. However, it is possible that a testing set that includes a wider spread of R_f values will show improved predictive performance.

4.4.2 Apparent Model Uncertainty

The hierarchal, measurement error, and combined models all had lower apparent uncertainties than the baseline case. This is supported by values of percent difference in apparent σ_{50} on the order of 100 to 200% between the baseline and following models. The “squishing” of the probability curves in the previous summary figures illustrate this reduction in uncertainty. In general, the hierarchal model had a lower uncertainty than the measurement error model and the combined model outperformed them both. This trend was more pronounced in the three variable case. The three variable measurement error and combined models produced strikingly low apparent model uncertainties (both numerically and graphically) but fall into the trap of being overly certain. That is, despite having low model uncertainty they perform poorly when making predictions on new data.

4.4.3 Model Posterior Predictive Uncertainty

In general, the hierarchal models (2, 4, 2-3, and 4-3) had larger posterior standard deviations than the fully pooled cases (models 1, 3, 1-3, and 3-3). This is consistent with the notion that hierarchal models account in some way for event-to-event variability in the coefficients while the pooled models ignore it entirely. As a result, their sample posterior predictive distributions have larger standard deviations and their probable median contours have a higher spread. A result of this is that the distributions for the median line spread out in areas with fewer data points (the upper right of the scatter plot) and illustrate greater posterior predictive uncertainty in these regions as would be expected. This trend is generally considered one of the benefits of hierarchal models – they avoid underestimating the uncertainty associated with out of sample predictions (Gelman and Hill, 2007).

For the two variable case, the measurement error model and baseline models had similar variability in their coefficient posterior distributions. One would expect with more data the scale of these standard deviations would become nearly identical. On the other hand, in the three variable case the posterior uncertainties grow rapidly as complexity is added to the model. This can be partially explained by the correlation between R_f and $q_{c,1}$ -- correlated predictors leads to more diffuse posteriors as discussed in the previous section. However, for models 3-3 and 4-3 it appears that the some or all posterior standard deviations are highly influenced by the prior standard deviations. This inference is based on the rule of thumb proposed by the Stan development team that a posterior is “influenced” by a default prior if its standard deviation is greater than 10% of the prior’s (Gelman, 2019). That is, the data is not strong enough to constrain the posterior predictive uncertainty for these parameters. This effect is particularly noticeable for the intercept though present in all parameters.

However, as evidenced by the reduction in apparent model uncertainty discussed previously these models show useful mean trends and can be useful for fully probabilistic inference with tighter priors or more data. This is evidenced by the work of Kuehn and Abrahamson, who were able to use numerical techniques to justify tighter priors on regression coefficients and also had access to a much larger dataset.   It is also important to note that almost no relevant work to date (e.g Robertson and Wride, 1998, Moss et al., 2006, Boulanger and Idriss, 2016) has acknowledged the impact of parameter uncertainty on forward prediction of liquefaction. However, doing so will be necessary for a performance based approach to liquefaction assessment.

4.4.4 A Brief Discussion on the Dangers of Overfitting

Figure 57, below, shows the predictive performance Model 4-2 but this time using the original training set for testing.

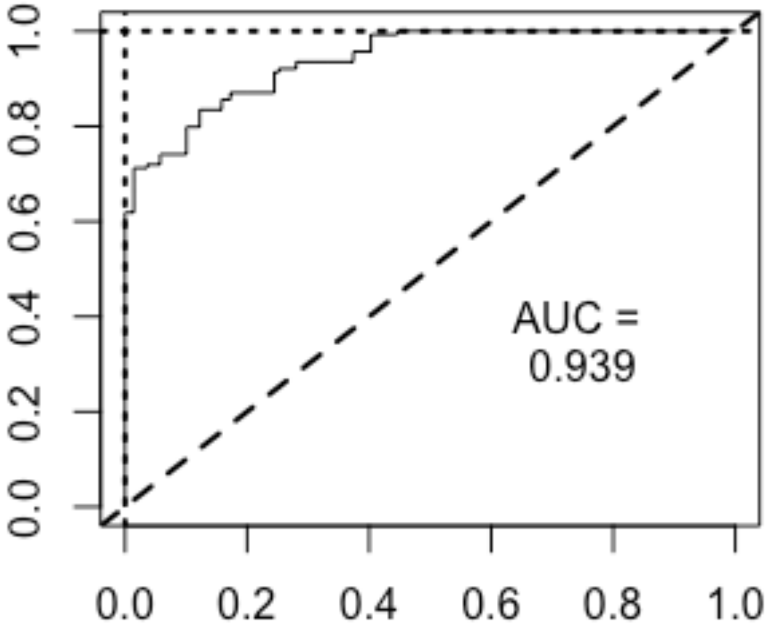


Figure 57 – Model 4's Predictive Performance. This time it is assessed only using the training set. The dramatic increase in apparent performance illustrates the optimistic, and possible dangerous, bias in this approach.

This approach, unfortunately taken by most of the previous work in the field, dramatically overestimates the model's predictive ability. This is further compounded with practitioners often having to rely on self-reported validation metrics when deciding which is a better model to use.

4.5 Prior Sensitivity Study

To further assess the influence of prior choice we performed a sensitivity study on our models. The default priors, used for the results reported above, is Normal (0,25) on the intercept parameter and Normal (0,10) on the slope parameters. We then further increased the standard deviation on these priors to 25 and 100 (for both) and recorded the change in posterior distributions and model predictive performance. The tables following report posterior means and standard deviations and AUC using the 3 different priors. We found for the 2 variable models the posterior distributions of model coefficients changed only slightly and AUC remained almost the same – though the slight increases in standard deviations will necessarily affect the posterior predictive uncertainty. The three variable baseline model also exhibited this behavior. However, for models 2-3, 3-3, and 4-3 we observed noticeable prior sensitivity as evidenced by an large increase in posteriors standard deviations (and a shift of mean values). This was accompanied by a drop in predictive accuracy, indicating that the larger priors are too diffuse to allow for good estimation of model parameters.

Table 4 -- Model 1-3 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 100
β_0	11.09 (1.59)	11.42 (1.63)
β_1	-0.89 (0.13)	-0.91 (0.13)
β_2	2.01 (0.31)	2.07 (0.32)
β_3	-1.42 (0.32)	-1.43 (0.32)
AUC	0.642	0.642

Table 5 -- Model 2 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 25	Prior SD = 100
μ_{β_0}	12.03 (1.94)	12.16 (2.00)	12.21 (1.99)
μ_{β_1}	-0.27 (0.06)	-0.27 (0.06	-0.27 (0.06)
μ_{β_2}	2.66 (0.46)	2.69 (0.48)	2.70 (0.47)
σ_{β_0}	0.94 (0.75)	0.96 (0.79)	1.00 (0.80)
σ_{β_1}	0.08 (0.05)	0.05 (0.31)	0.08 (0.06)
σ_{β_2}	0.31 (0.24)	0.31 (0.24)	0.31 (0.24)
AUC	0.716	0.713	0.716

Table 6 -- Model 2-3 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 25	Prior SD = 100
μ_{β_0}	22.31 (4.03)	23.07 (4.43)	26.31 (9.60)
μ_{β_1}	-1.57 (0.32)	-1.63 (0.35)	-1.83 (0.68)
μ_{β_2}	4.07 (0.78)	4.15 (0.85)	4.61 (1.38)
μ_{β_3}	-1.92 (1.97)	-1.93 (2.34)	-2.24 (3.33)
σ_{β_0}	1.89 (1.58)	2.06 (1.72)	3.34 (4.71)
σ_{β_1}	0.27 (0.24)	0.30 (0.28)	0.48 (0.67)
σ_{β_2}	0.51 (0.37)	0.54 (0.43)	0.79 (1.49)
σ_{β_3}	5.18 (2.04)	6.48 (2.93)	8.11 (4.78)
AUC	0.654	0.654	0.650

Table 7 -- Model 3 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 25	Prior SD = 100
β_0	11.79 (2.00)	12.14 (2.11)	12.36 (2.31)
β_1	-0.26 (0.05)	-0.27 (0.05)	-0.28 (0.5)
β_2	2.54 (0.44)	2.62 (0.46)	2.66 (0.50)
AUC	0.710	0.710	0.710

Table 8 -- Model 3-3 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 25	Prior SD = 100
β_0	29.69 (10.78)	33.13 (13.25)	165.13 (73.24)
β_1	-2.44 (0.92)	-2.73 (1.13)	-13.96 (6.26)
β_2	5.39 (1.93)	6.00 (2.36)	29.45 (13.08)
β_3	-4.35 (1.69)	-4.87 (2.07)	-24.62 (11.43)
AUC	0.639	0.639	0.631

Table 9 -- Model 4 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 25	Prior SD = 100
μ_{β_0}	14.60 (2.99)	14.67 (2.93)	14.87 (3.14)
μ_{β_1}	-0.32 (0.08)	-0.32 (0.08)	-032 (0.08)
μ_{β_2}	3.25 (0.69)	3.27 (0.67)	3.31 (0.73)
σ_{β_0}	1.15 (0.91)	1.15 (0.96)	1.16 (0.92)
σ_{β_1}	0.08 (0.06)	0.07 (0.06)	0.08 (0.06)
σ_{β_2}	0.35 (0.27)	0.35 (0.28)	0.35 (28)
AUC	0.716	0.716	0.716

Table 10 -- Model 4-3 Prior Sensitivity Results

Parameter	Default Prior	Prior SD = 25	Prior SD = 100
μ_{β_0}	49.55 (12.75)	59.23 (14.90)	211.15 (58.79)
μ_{β_1}	-3.61 (1.11)	-4.29 (1.29)	-15.36 (5.04)
μ_{β_2}	9.13 (2.53)	10.87 (3.02)	38.83 (11.71)
μ_{β_3}	-6.07 (3.83)	-7.02 (5.48)	-26.44 (17.88)
σ_{β_0}	3.61 (2.93)	4.02 (3.24)	12.90 (9.99)
σ_{β_1}	0.98 (0.74)	1.22 (0.96)	4.37 (3.32)
σ_{β_2}	1.39 (1.04)	1.81 (1.31)	6.46 (4.73)
σ_{β_3}	7.28 (3.15)	12.13 (5.22)	34.89 (14.63)
AUC	0.649	0.648	0.641

5 CONCLUSION AND RECCOMENDATIONS

The following chapter presents a summary of our research and recommendations for future work.

5.1 Study Summary

This study built a predictive modeling workflow for developing empirical models for liquefaction triggering potential. While we only considered CPT data the process would be similar, if not identical, for other in-situ tests. Using the database from Moss et al., 2006 we first selected the predictor variables that had a strong enough statistical association with liquefaction outcomes to be considered in the modeling process. These were $q_{c,1}$, CSR, and R_f. We then selected Box – Cox transformations of these variables that produced the highest cross-validated AUC for both the two and three variable models. In the modeling process, we considered four main models – a baseline simple logistic regression, a hierarchal (or mixed/random effects) model, a Bayesian measurement error model, and a combination of the last two. The performance of these models was assessed using a testing set of New Zealand case histories and report as ROC curves.

The goals of this study were two-fold. First, we wanted to develop transparent model validation strategies to be used when examining the effects of the modeling choices discussed above. Secondly, we wanted to reduce model uncertainty, while maintaining (or improving) predictive capability. For the purpose of this study, we divide model uncertainty into two components. The apparent model uncertainty referrers how high or low of a probability is assigned to predictions in general (fixing regression coefficients to their mean values). A more "confident" model will consistently assign higher or lower probabilities, i.e. classifying a yes as 90% instead of 60%, than a less "confident one". Graphically, this is seen as the spread of the contours of probability. We also considered a fully probabilistic posterior predicative uncertainty. This was defined as the standard deviation of probability of liquefaction from a fixed predictor variable pair arising from using the full posterior distributions of model coefficients and treating the equation for liquefaction potential as a function of random variables.

We found that the all cases of the measurement error model, hierarchal, and combination models reduced apparent model uncertainty based upon the graphical behavior of triggering curves and median apparent uncertainty values. In this context, the hierarchal models outperformed the stand alone measurement error models and the combinations of the two performed the best. For the two predictor variable models, predictive performance remained constant relative to the baseline or improved. In contrast, all the three predictor variable models performed worse than the two variable baseline. This indicates that, at least in the context of our training and testing sets, models incorporating R_f as a predictor variable do not generalize well to new data. Additionally, predictive performance dropped with model complexity illustrating what we call an "over confidence" effect – a model that has low apparent uncertainty but performs poorly on new data.

Posterior predictive uncertainty was noticeably higher for the hierarchal models and for all three variable cases. For the hierarchal models this is not necessarily a problem – accounting for the possible group level variability in regression coefficients will naturally produce a more variable posterior distribution. However, for three variable models the high standard deviations of the regression coefficients indicate that the data is insufficiently to adequately estimate them to an acceptable level of uncertainty for fully probabilistic inference. This can likely be remedied with more data points (such as the NGL database) or with more informative priors determined from expert consensus or laboratory testing.

5.2 Recommendations For Future Work

- Most of the posterior uncertainty can be eliminated with more data. Thus, we recommend that new models be built using the NGL database when it is released. These will be inherently be more useful for the state of practice than those built on limited and outdated data.
- Future work should take a principled approach to predictor variable selection and processing, similar to the methods described in this paper. To minimize posterior uncertainty, correlation in the selected variables should be avoided. We believe it is worth investigating predictors that haven't be seen in previous models but show strong statistical association with liquefaction outcomes (e.g. Kayen and Mitchell, 1997)
- Use informative priors to constrain regression coefficients when data is insufficient to estimate them to acceptable levels of uncertainty. This can be based on laboratory or field testing, or numerical modeling similar to the work of Kuehn and Abrahamson. Additionally, principled elicitation of expert consensus can also be used to build prior distributions.
- The logistic model can be extended by allowing the log-odds of liquefaction to be a nonlinear function of predictor variables. Similar to how we selected our transformations, future modelers can optimization strategies to determine the best performing functional form of predictors.
- The Bayesian measurement error model is not limited to logistic regression, nor is logistic regression the only means of building a probabilistic model. Because of how common logistic regression is in practice we recommend that it be used a baseline and compared to more complex functional forms. However, it is important to consider the

utility of models in general practice. A highly complicated, more accurate model that is used incorrectly will be less useful than a less accurate, but less prone to user error one.

- The final models should be cross-validated to report relatively unbiased metrics of performance

REFERENCES

Abrahamson, N. A., & Youngs, R. R. (1992). “A stable algorithm for regression analyses using the random effects model”. *Bulletin of the Seismological Society of America*, *82*(1), 505-510.

Academies of Sciences, Engineering, and Medicine. (2016). *State of the art and practice in the assessment of earthquake-induced soil liquefaction and its consequences*, The National Academies Press, Washington, DC.

Andrus, R.D., & Stokoe, K.H. II. 1997. “Liquefaction resistance based on shear wave velocity.” *Proc. of the NCEER Workshop on Evaluation of Liquefaction Resistance of Soils*, Buffalo, NY: National Center for Earthquake Engineering Research.

Bates, D., Martin, M., Bolker, B., & Walker, S. (2015). “Fitting linear mixed-effects models using lme4.” Journal of Statistical Software, 67(1), 1-48. https://doi.org/10.18637/jss.v067.i01

Betancourt, M. (2018). “A conceptual introduction to Hamiltonian Monte Carlo”. *ArXiv:1701.02434 [stat.ME]*

Betancourt, M. & Girolami, M. (2013). “Hamiltonian Monte Carlo for hierarchical models”. *ArXiv: 13112.0906 [stat.ME]*.

Boulanger, R.W., D.W. Wilson, & I.M. Idriss. 2012. “Examination and reevaluation of SPT-based liquefaction triggering case histories”. Journal of Geotechnical and Geoenvironmental Engineering 138(8),898–909. https://doi.org/10.1061/(ASCE)GT.1943-5606.0000668

Boulanger, R. W., & Idriss, I. M. (2016). “CPT-based liquefaction triggering procedure.” *Journal of Geotechnical and Geoenvironmental Engineering*, *142*(2), 04015065. https://doi.org/10.1061/(ASCE)GT.1943-5606.0001388

Breslow, N. E., & Clayton, D. G. (1993). “Approximate inference in generalized linear mixed models”. *Journal of the American Statistical Association*, 88(421), 9-25. https://doi.org/10.2307/2290687

Brillinger, D. R., & Preisler, H. K. (1985). “Further analysis of the Joyner-Boore attenuation data”. *Bulletin of the Seismological Society of America*, *75*(2), 611-614.

Carpenter, B., Gelman, A., Hoffman, M.D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., & Riddell, A. 2017. “Stan: a probabilistic programming language”. *Journal of Statistical Software,* 76(1). DOI 10.18637/jss.v076.i01

Cetin, K.O., Seed, R. B., Der Kiureghian, A., Tokimatsu, K., Harder, L. F., Kayen, R. E., & Moss, R. E. S. (2004). “Standard penetration test-based probabilistic and deterministic assessment of seismic soil liquefaction potential.” Journal of Geotechnical and Geoenvironmental Engineering 130(12),1314–1340. https://doi.org/10.1061/(ASCE)1090-0241(2004)130:12(1314)

Christensen, R., Johnson, W., Branscum, A., & Hanson, T.E. (2011). *Bayesian ideas and data analysis: an introduction for scientists and statisticians*. New York, NY: Taylor and Francis.

Clark, T. S., & Linzer, D. A. (2015). “Should I use fixed or random effects?”. *Political Science Research and Methods*, *3*(02), 399–408. https://doi.org/10.1017/psrm.2014.32

DeGroot, M.H, & Schervish, M.J. (2012). *Probability and Statistics*. Boston, MA: Pearson.

Der Kiureghian. (2004). “First and second order reliability methods”. *Engineering Design Reliability Handook,* Nikoladis, E., Ghiocel, M., & Singhal, S., eds. Abingdon, OX: Taylor & Francis.

Dunn, P.K., & Smyth, G.K. (2018). *Generalized linear models with examples in R*. New York, NY: Springer

Fawcett, T. (2006). “An introduction to ROC analysis”. *Pattern Recongition Letters*, 27(8), 861-874. https://doi.org/10.1016/j.patrec.2005.10.010.

Gelman, A. (2019). “Prior choice reccomednations.” *Stan development wiki*. < https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations>. Accessed 31 May 2019.

Gelman, A. (2006). "Prior distributions for variance parameters in hierarchical models.” *Bayesian Analysis,* 1(3), 515-534. https://doi.org/10.1214/06-BA117A

Gelman, A., & Hill, J. (2007). *Data analysis using regression and multilevel/hierarchical models.* New York: NY, Cambridge University Press.

Gelman, A., & Rubin, D. B. (1992). “Inference from iterative simulation using multiple sequences.” *Statistical Science*, 7(4), 457–472. https://doi.org/10.1214/ss/1177011136

Gelman, A., Jakulin, A., Pittau, M. G., & Su, Y. (2008). “A weakly informative default prior distribution for logistic and other regression models.” *The Annals of Applied Statistics*, 2(4), 1360-1383. https://doi.org/10.1214/08-AOAS191

Goh, A. T. C., and Goh, S. H. (2007). “Support vector machines: their use in geotechnical engineering as illustrated using seismic liquefaction data”. *Computational Geotechnics*, 34, 410-421. https://doi.org/10.1016/j.compgeo.2007.06.001

Giovinazzi, S., Black, J. R., Milke, M., Esposito, S., Brooks, K. A., Craigie, E. K., Liu, M. (2015). “Identifiying seismic vulnerability factors for wastewater pipelines after the Canterbury (NZ) earthquake sequence 2010-2011.” *Pipelines 2015: Recent Advances in Underground Pipeline Engineering and Construction,* 304-315. http://www.doi.org/10.1061/9780784479360.029

Green, R. A., Cubrinovski, M., Cox, B., Wood, C., Wotherspoon, L., Bradley, B., & Maurer, B. (2014). “Select liquefaction case histories from the 2010-2011 Canterbury earthquake sequence.” *Earthquake Spectra,* 20(1), 131-153. https://www.doi.org/10.1193/030713EQS066M

Hanna, A. M., Ural, D., and Saygili. G. (2007). “Evaluation of liquefaction potential of soil deposits using artificial neural networks”. *Engineering Computations*, 24(1), 5-16. https://doi.org/10.1108/02644400710718547

Holtz, R.D., W.D. Kovacs, & T.C. Sheahan (2011). *An Introduction to Geotechnical Engineering*. Pearson Education, Inc., Upper Saddle River, NJ.

Hu, J.L., Tang, X.W., & Qiu, J.N. (2017). “Analysis of the influences of sampling bias and class imbalance on performances of probabilistic liquefaction models.” *International Journal of Geomechanics*, *17*(6), 04016134. https://doi.org/10.1061/(ASCE)GM.1943-5622.0000808

Idriss, I.M., & R.W. Boulanger. 2010. “SPT-Based liquefaction triggering procedures.” *Report No. UCD/CGM-10-02.* Center for Geotechnical Modeling, Department of Civil and Environmental Engineering, University of California, Davis.

Jiang, J. (2007). *Linear and generalized linear mixed models and their applications.* New York, NY: Springer New York

Juang, C. H., Jiang, T., & Andrus, R. D. (2002). “Assessing probability-based methods for liquefaction potential evaluation.” *Journal of Geotechnical and Geoenvironmental Engineering*, 128(7), 580-589. https://doi.org/10.1061/(ASCE)1090-0241(2002)128:7(580)

Juang, C. H., Rosowsky, D. V., & Tang, W.H. (1999). “Reliability-based method for assessing liquefaction potential of soils.” *Journal of Geotechnical and Geoenvironmental Engineering*, 125(8), 684-689. https://doi.org/10.1061/(ASCE)1090-0241(1999)125:8(684)

Kayen R, E., and Mitchell, J. K. (1997). "Assessment of liquefaction potential during earthquakes by Arias intensity". *Journal of Geotechnical and Geoenvironmental Engineering*, 123(12), 1162-1174. https://doi.org/10.1061/(ASCE)1090-0241(1997)123:12(1162)

Kayen, R.E., Moss, R.E.S., Thompson, E.R., Seed, R.B., Cetin, K.O., Der Kiureghian, A., Tanaka, Y., & Tokimatsu, K. (2013). "Shear wave velocity-based probabilistic and deterministic assessment of seismic soil liquefaction potential." *Journal of Geotechnical and Geoenvironmental Engineering,* 139(3),407–419. https://doi.org/10.1061/(ASCE)GT.1943-5606.0000743

Kruschke, J. K. (2015). *Doing Bayesian data analysis: a tutorial with R, JAGS, and Stan*. Boston: Academic Press.

Kuehn, N. M., & Scherbaum, F. (2015). "Ground-motion prediction model building: a multilevel approach." *Bulletin of Earthquake Engineering*, *13*(9), 2481–2491. https://doi.org/10.1007/s10518-015-9732-3

Kuehn, Nicolas M., & Abrahamson, N. A. (2018). "The effect of uncertainty in predictor variables on the estimation of ground-motion prediction equations." *Bulletin of the Seismological Society of America*, *108*(1), 358–370. https://doi.org/10.1785/0120170166

Kuhn, M. (2008). "Building predictive models in R using the caret package." Journal of Statistical Software, 28(5), 1 - 26. http://dx.doi.org/10.18637/jss.v028.i05

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. New York, NY: Springer New York. https://doi.org/10.1007/978-1-4614-6849-3

Lai, S.Y., Chang, W.J., & Lin, P.S. (2006). "Logistic regression model for evaluating soil liquefaction probability using CPT data." *Journal of Geotechnical and Geoenvironmental Engineering*, *132*(6), 694–704. https://doi.org/10.1061/(ASCE)1090-0241(2006)132:6(694)

Liao, S. S. C., Veneziano, D., & Whitman, R.V. (1988.). "Regression models for evaluating liquefaction probability." *Journal of Geotechnical Engineering,* 114(4), 389-411. https://doi.org/10.1061/(ASCE)0733-9410(1988)114:4(389)

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M., N., amd Teller, E. J. (1953). "Equation of state calculations by fast computing machines". *Journal of Chemical Physics,* 21(6), 1087-1092. https://doi.org/10.1063/1.1699114

Moss, R. E. (2013). *Applied civil engineering risk analysis*. Shedwick Publishing.

Moss, R. E., Seed, R. B., Kayen, R. E., Stewart, J. P., Der Kiureghian, A., & Cetin, K. O. (2006). "CPT-Based probabilistic and deterministic assessment of in situ seismic soil liquefaction potential." *Journal of Geotechnical and Geoenvironmental Engineering*, 132(8), 1032–1051. https://doi.org/10.1061/(ASCE)1090-0241(2006)132:8(1032)

National Cooperative Highway Research Program. (2007). "Cone penetration testing: a synthesis of highway practice." *Project 20-5 (Topic 37-14)*, Transportation Research Board

Neal, R. M. (2012). "MCMC using Hamiltonian dynamics." *ArXiv:1206.1901 [physics, stat]*.

Olsen, R. S., & Koester, J. P. (1995). ''Prediction of liquefaction resistance using the CPT.'' *Proc., Int. Symp. on Cone Penetration Testing*, *Vol. 2*, Balkema, Rotterdam, 251–256.

Oommen, T., Baise, L. G., & Vogel, R. M. (2011). "Sampling bias and class imbalance in maximum-likelihood logistic regression." *Mathematical Geosciences*, *43*(1), 99–120. https://doi.org/10.1007/s11004-010-9311-8

Oommen, T., Baise, L. G., & Vogel, R. (2010a). "Validation and application of empirical liquefaction models." *Journal of Geotechnical* *and Geoenvironmental Engineering*, *136*(12), 1618–1633. https://doi.org/10.1061/(ASCE)GT.1943-5606.0000395

Rezania, M., Faramarzi, A., & Javadi, A. A. (2011). "An evolutionary based approach for assessment of earthquake-induced liquefaction and lateral displacement." *Engineering Applications of Artificial Intelligence*, 24, 142-153. https://doi.org/10.1016/j.engappai.2010.09.010

Richardson, S., & Gilks, W.R. (1993). "A Bayesian approach to measurement error problems in epidemiology using conditional independence models." *American Journal of Epidemiology,* 138(6), 430–42. https://doi.org/10.1093/oxfordjournals.aje.a116875

Robertson, P.K. & Cabal, K.L. (2015). "Guide to Cone Penetration Testing". Gregg Drilling, Signal Hill, CA.

Robertson, P. K., & Wride, C. E. (1997). ''Cyclic liquefaction and its evaluation based on the SPT and CPT.'' *Proc., NCEER Workshop on Evaluation of Liquefaction Resistance of Soils.*

Sing, T., Sander, O., Beerenwinkel, N., & Lengauer, T. (2005). "ROCR: visualizing classifier performance in R." Bioinformatics, 21(20), 7881. https://doi.org/10.1093/bioinformatics/bti623

Stan Development Team. (2018). *Stan modeling language users guide and reference manual*, version 2.18.0. http://mc-stan.org.

Stewart, J.P., Kramer, S. L., & Bozorgnia, Y. (2019). "NGL: open source global database and model development for the next-generation of liquefaction assessment procedures". *UCLA Geotechnical Group*. < https://uclageo.com/NGL/>. Accessed 19 May 2019.

Suzuki, Y., Tokimatsu, K., Koyamada, K., Taya, Y., & Kubota, Y. (1995). ''Field correlation of soil liquefaction based on CPT data.'' *Proc., Int. Symp. on Cone Penetration Testing, CPT 95, Vol. 2*, Balkema, Rotterdam, 583–588.

Toprak, S., & Bennet, M. J. (1999). "CPT- and SPT- based probablistic assessment of liquefaction potential". *Proc. 7th US-Japan Workshop on Earthquake Resistant Design of Lifeline Facilities and Countermeasures against Liquefaction*.

Yazdi, J. S., Kalantary, F., & Yazdi, H. S. (2013). "Investigation on the effect of data imbalance on prediction of liquefaction." *International Journal of Geomechanics*, *13*(4), 463–466. https://doi.org/10.1061/(ASCE)GM.1943-5622.0000217

Yazdi, J.S., & Moss, R. E. S. (2017). "Nonparametric liquefaction triggering and postliquefaction deformations." *Journal of Geotechnical and Geoenvironmental Engineering,* 143(3). https://doi.org/10.1061/(ASCE)GT.1943-5606.0001605

Youd, T.L., Idriss, I.M, Andrus, R. D., Arango, I., Castro, G., Christian, J.T., Dobry, R., Finn, W.D. L., Harder, L. F., Hayes, M. E., Ishihara, K., Koester, J. P., Liao, S. S. C., Marcuson, W. F., Martin, G. R., Mitchell, J. K., Moriwaki, Y., Power, M. S., Roberston, P. K., Seed, R. B., & Stokoe, K. H. (2001). "Liquefaction resistance of soils: summary report from the 1996 NCEER and 1998 NCEER/NSF workshops on evaluation of liquefaction resistance of soils." *Journal of Geotechnical and Geoenvironmental Engineering*, 127, 817– 833. https://doi.org/10.1061/(ASCE)1090-0241(2001)127:4(297)

Zhang, J., Zhang, L. M., & Huang, H. W. (2013). "Evaluation of generalized linear models for soil liquefaction probability prediction". *Environmental Earth Sciences*, *68*(7), 1925–1933. https://doi.org/10.1007/s12665-012-1880-z