Cal Poly

Caltech

SCEC
AN NSF+USGS CENTER

UC Irvine

UCLA

UC Santa Barbara

USC

Non-ergodic Methodology and Modeling Tools

Grigorios Lavrentiadis, Ph.D.
Nicolas Kuehn, Ph.D.
Yousef Bozorgnia, Ph.D.
Natural Hazards Risk and Resiliency Research Center
The B. John Garrick Institute for the Risk Sciences
University of California, Los Angeles

Elnaz Seylabi, Ph.D.
Department of Civil and Environmental Engineering
University of Nevada, Reno

Xiaofeng Meng, Ph.D
Christine Goulet, Ph.D.
Southern California Earthquake Center (SCEC)
University of Southern California

Albert Kottke, Ph.D.
Geosciences
Pacific Gas & Electric Company

A report on research supported by the California Department of Transportation and Pacific Gas & Electric Company

Report GIRS-2022-04
DOI: 10.34948/N35P4Z

University of California, Los Angeles (headquarters)

Natural Hazards Risk & Resiliency Research Center

The B. John Garrick Institute for the Risk Sciences

Non-ergodic Methodology and Modeling Tools

Grigorios Lavrentiadis, Ph.D.
Nicolas Kuehn, Ph.D.
Yousef Bozorgnia, Ph.D.

Natural Hazards Risk and Resiliency Research Center
The B. John Garrick Institute for the Risk Sciences
University of California, Los Angles

Elnaz Seylabi, Ph.D.

Department of Civil and Environmental Engineering
University of Nevada, Reno

Xiaofeng Meng, Ph.D.
Christine Goulet, Ph.D.

Southern California Earthquake Center (SCEC)
University of Southern California

Albert Kottke, Ph.D.

Geosciences
Pacific Gas & Electric Company

A report on research conducted with support from
the California Department of Transportation and Pacific Gas & Electric Company

Report GIRS-2022-04

Natural Hazards Risk and Resiliency Research Center
The B. John Garrick Institute for the Risk Sciences
University of California, Los Angeles (Headquarters)

August 2022

ABSTRACT

Non-ergodic ground-motion models (NGMMs) can potentially reduce the ground-motion aleatory variability significantly, and dramatically impact the seismic hazard, especially at large return periods which is important for critical infrastructure. This reduction in aleatory variability is accompanied by epistemic uncertainty in regions with sparse recordings or a systematic shift in the median ground motion in regions with dense recordings. This report summarizes NGMMs methodologies, commonly-used software platforms for NGMMs, and provides instructions for users of such software platforms. Gaussian Process Regression (GPR) – with spatially varying coefficients for modeling the source and site systematic effects and cell-specific anelastic attenuation for modeling the systematic path effects – is a flexible and robust modeling technique for developing NGMMs, which is elaborated in this report. As part of this work, open-source computer tools and instructions for users have been developed to show the steps toward developing NGMMs in the GPR framework. Two commonly-used statistical software packages, STAN and INLA, are used and compared. The developed software packages were verified against synthetic data sets with known non-ergodic effects, and different implementations of the developed software were evaluated for scalability, universality, precision, and model complexity. The computer codes developed as part of this report and the synthetic datesets used for the verification process are made available to the public through the UCLA NHR3 Git-hub repository (https://github.com/NHR3-UCLA/ngmm_tools). The Docker image to assist in the cross-platform compatibility of the provided tools can be built using the Git-hub repository (https://github.com/NHR3-UCLA/docker-ngmm_tools). The metadata and generated synthetic datasets used in the verification exercises can be accessed through the Products Section in https://www.risksciences.ucla.edu/nhr3/ngmm.

ACKNOWLEDGMENTS

Support for the development of this report and non-ergodic methodology and modeling tools was provided by the California Department of Transportation (Caltrans) and Pacific Gas Gas & Electric Company (PG&E).

The support of these organizations is gratefully appreciated. The opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the study sponsors, the Natural Hazards Risk and Resiliency Research Center (NHR3), or the Regents of the University of California.

We also like to thank Dr. Norman A. Abrahamson for the insightful discussions during the course of this work. Numerous individuals have been working on the development and applications of non-ergodic ground motions. Their efforts and cooperation are gratefully acknowledged.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 OVERVIEW

Traditionally, due to the small size of available regional earthquake datasets, ground-motion models (GMMs) are developed under the ergodic assumption (Anderson and Brune, 1999), which states that the ground-motion variability in space (i.e., between many stations from few earthquakes) is equal to the ground-motion variability in time (i.e., at one station from many earthquakes). This assumption has been convenient as it allows to combine data from similar tectonic environments around the world to develop a model for scaling with magnitude, distance, and site condition. The median and aleatory variability of a GMM are assumed to be applicable to any location within the broad tectonic category.

The traditional approach of developing ergodic GMMs leads to a stable global average of the ground motion for a given scenario, but a large aleatory variability between an observation and the global average. With the large increase in the number of ground-motion instruments and recordings over the last decade, it has become clear that there are significant systematic differences in ground motion based on the location of the site and the source. As a result, the ergodic GMMs generally may not work well for a specific site/source location. This has prompted the development of non-ergodic ground-motion models in which these location-specific effects are modeled explicitly, which reduces the aleatory variability. The uncertainty in the estimate of the site-specific effects is then part of the epistemic uncertainty. As a general classification, uncertainties are treated as epistemic if they are expected to be reduced by gathering more data. Variabilities are treated as aleatory if the increase of data is not expected to systematically reduce their range (Der Kiureghian and Ditlevsen, 2009).

An important difference between the application of statistics in GMMs as compared to most other fields is the use of constraints to ensure proper extrapolation. In other fields, the assumption is that the key behaviors are represented by the available data. Thus, the goal of statistics is to find the trends in the data. However, the problem is more complicated in earthquake GMMs as they are often applied to earthquake scenarios outside the range that is well constrained by the data (i.e., it is an extrapolation problem). For instance, Figure 1.1 shows the magnitude-distance distribution of the California subset of the NGA-West2 database (Ancheta et al., 2014), which is often used for the development of GMMs for California. In this dataset, the magnitudes range from 3 to 7.2, with the majority of the events being between magnitude 3 and 5, and the distance ranges from 1 to $400 km$, with the majority of the recordings being between 20 and $200 km$. However, in PSHA,

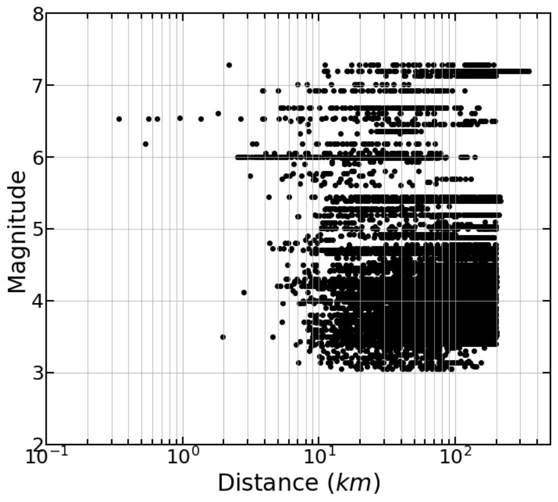


Figure 1.1. Magnitude-distance distribution of the California subset of the NGAWest2 dataset.

large-magnitude and short-distance scenarios often control the hazard. For example, in the San Francisco Bay Area, it is common to have faults that are less than $10 km$ away from a site and are capable of producing larger than $M7$ earthquakes. It is the difference between the range of the scenarios that are used to derive a GMM and the range of scenarios on which a GMM is applied that makes the proper extrapolation of the ground motion an important aspect of a GMM.

This report aims to review NGMM modeling. Our emphasis is on Gaussian process regression as a powerful method for developing non-ergodic ground-motion models and on providing software tools for the development and application of NGMMs. Potential modelers are encouraged to treat the provided computer tools as building blocks and modify them to fit their project needs. For example, users can adjust the prior distributions and modify the NGMM functional form as they wish.

The software accompanying this report can be found at the UCLA NHR3 Git-hub repository (https://github.com/NHR3-UCLA/ngmm_tools). The Docker image to assist in the cross-platform instalation of the provided tools can be built using the Git-hub repository (https://github.com/NHR3-UCLA/docker-ngmm_tools). The metadata and generated synthetic datasets used in the verification exercises can be accessed through the Products Section in https://www.risksciences.ucla.edu/nhr3/ngmm.

1.2 ORGANIZATION

This report is organized into the following chapters:

- Chapter 2 discusses the formulation of non-ergodic ground-motion models as a Gaussian Process (GP). It starts by presenting the methodologies used in GMM practice to estimate the components (coefficients and aleatory standard deviations) of an ergodic GMM, how these methodologies are extended to estimate the components of a partially non-ergodic GMM, and how they are modified to estimate the components of a fully non-ergodic GMM. In particular, Section 2.4 presents: (i) the components of a GP regression, (ii) commonly used covariance functions for modeling the spatial variability of the non-ergodic effects, (iii) the formulation of the cell-specific anelastic attenuation as a GP, and (iv) the conditional prediction of non-ergodic ground-motions on existing records.

- Chapter 3 summarizes the NGMM functional forms and kernel function options in the developed tools for the non-ergodic and anelastic attenuation coefficients. Three types of non-ergodic models can be developed using the provided tools. Type1 NGMM includes an intercept and three spatially varying constants to model the systematic source and site effects. Type2 NGMM, in addition to the previous terms, includes the cell-specific anelastic attenuation to capture a part of the systematic path effects. Lastly, type3 NGMM, adds a spatially varying geometrical spreading and V_{S30} scaling to the regression.
- Chapter 4 summarizes the input files needed to run regression tools. The regression input files are the ground-motion flatfile, the cell information flatfile, and the cell-path length flatfile. A computer script for estimating the cell-path segment lengths is also presented.
- Chapter 5 presents the synthetic datasets used to verify the GMM regression tools. Section 5.1 describes the NGAWest2 CA and NGAWest3[*] CA metadata used as the basis for the development of the synthetic datasets. Section 5.2 describes the methodology and different types of synthetic datasets generated for the verification exercise.
- Chapter 6 and Chapter 7 present the developed tools for developing the non-ergodic ground-motion models using the statistical software platforms STAN and INLA. In both cases, a general overview of the statistical environments is presented first, the computer codes for estimating the type1 NGMM terms are then explained, and the modifications for the type2 and type3 NGMM are summarized in the end.
- Chapter 8 presents the application of the developed NGMM tools on a CyberShake simulation dataset and summarizes the lessons learned from this test. The comparison showed that the developed tools can effectively capture the site effects. The source effects are not accurately captured due to the self-similar assumption for the CyberShake events. A more sophisticated representation of wave propagation in NGMMs is necessary to recover the anelastic attenuation path effects.
- Chapter 9 summarizes the proposed notation used in this report for the development and application of NGMMs.
- Chapter 10 summarizes the regression methodology and presented tools as well as directions for future research.
- Appendix A summarizes the links for the different electronic supplements.

2 OVERVIEW OF DEVELOPMENT OF ERGODIC AND NON-ERGODIC GMMS

Contents of this chapter are primarily from the Lavrentiadis, G., Abrahamson, N.A., Nicolas, K.M., Bozorgnia, Y., Goulet, C.A., Babič, A., Macedo, J., Dolšek, M., Gregor, N., Kottke, A.R., Lacour, M. ... (2022) "Overview and Introduction to Development of Non-Ergodic Earthquake Ground-Motion Models" Bulletin of Earthquake Engineering journal article.

This chapter serves as an overview and introduction to the development of non-ergodic GMMs, with an emphasis on the varying coefficient models (VCM) developed with Gaussian Processes (GPs) regression. The combination of a VCM GMM with the cell-specific anelastic attenuation and considerations regarding the extrapolation of non-ergodic GMMs are also discussed. An updated notation for key elements of non-ergodic GMMs is also presented.

2.1 PROPOSED NOTATION

The proposed notation is intended to help the reader understand the role of different terms in a GMM and facilitate the comparison of the different non-ergodic models presented in this report.

The model variables are categorized into two groups: the model parameters ($\vec{\theta}$) and model hyperparameters ($\vec{\theta}_{hyp}$). The $\vec{\theta}$ includes the ergodic and non-ergodic terms that directly affect the ground motion, while $\vec{\theta}_{hyp}$ includes the set of variables that control the behavior of the ergodic and non-ergodic terms and have an indirect effect on the ground motion. An example of a model parameter is the coefficient for the linear magnitude term, and an example of a hyperparameter is the between-event standard deviation.

The ergodic coefficients of the GMM are denoted as c_i where i is the number of the ergodic term, and the non-ergodic coefficients are denoted as $c_{i,X}$ or $\delta c_{i,X}$. The subscript X (subscript after the comma) can be the letters E, P, or S depending on whether the non-ergodic coefficient in question is intended to capture systematic effects related to the source (earthquake), path, or site. The notation δ is used to differentiate between non-ergodic coefficients that have a zero mean and act as adjustments to ergodic coefficients, and stand-alone non-ergodic coefficients that encompass both the average scaling and the systematic effects. For instance, the non-ergodic term $\delta c_{1,E}$ acts on top of c_1 to capture the systematic effects related to the source. Alternatively, the same behaviour can be modeled with $c_{1,E}$ which is equal to the sum of the ergodic coefficient and the non-ergodic adjustment ($c_{1,E} = c_1 + \delta c_{1,E}$). The non-ergodic adjustments, $\delta c_{i,X}$, are typically used when a non-ergodic GMM is developed with an ergodic GMM as a "backbone" model, in which case the non-ergodic adjustments are estimated based on the total ergodic residuals. For example, Kuehn et al. (2019) and Lavrentiadis et al. (2021) are non-ergodic GMMs derived with this approach. The non-ergodic coefficients, $c_{i,X}$, are used when a non-ergodic GMM is directly estimated with the log of the ground motion as a response variable as in the case of Landwehr et al. (2016).

Many of the non-ergodic GMMs in this issue used the cell-specific anelastic attenuation, first proposed by Dawood and Rodriguez-Marek (2013), to model the systematic path effects. In the proposed notation, the vector of attenuation coefficients of all the cells is denoted as $\vec{c}_{ca,P}$, and cell path segments are denoted as $\Delta\vec{R}$. The total anelastic attenuation is equal to $\vec{c}_{ca,P} \cdot \Delta\vec{R}$.

The terms $\delta L2L$, $\delta P2P$, and $\delta S2S$ introduced by Al Atik et al. (2010) are used here to describe the total non-ergodic effects related to the source (they used L for location), path, and site, respectively. For instance, if the constant c_1 is modified by two site adjustments, $\delta c_{1a,S}$ and $\delta c_{1b,S}$, to express the systematic site effects, then $\delta S2S$ is equal to $\delta c_{1a,S} + \delta c_{1b,S}$. Similarly, if the non-ergodic adjustment to the geometrical spreading coefficient, $\delta c_{3,P}$, and the cell-specific anelastic attenuation, $\vec{c}_{ca,P}$ are used to express the systematic path effects, then $\delta P2P = \ln(R)\delta c_{3,P} + \vec{c}_{ca,P} \cdot \Delta\vec{R} - c_7 R$, where, in this example, c_7 is the ergodic anelastic attenuation coefficient. The $c_7 R$ term is subtracted from the median prediction from the GMM to remove the systematic effects that are included in the cell-specific anelastic attenuation.

The scale and correlation length which control the spatial distribution of the non-ergodic terms are denoted as $\omega_{i,X}$ and $\ell_{i,X}$. An in-depth discussion on modeling the non-ergodic terms as GPs, where $\omega_{i,X}$ and $\ell_{i,X}$ are defined, is provided in section 2.4.1. In keeping with the Al Atik et al. (2010) notation, the total epistemic uncertainty of the non-ergodic source, path, and site effects are denoted as τ_{L2L}, ϕ_{P2P}, and ϕ_{S2S}. Expanding from the example above, if $\omega_{1a,S}$ and $\omega_{1b,S}$ correspond to the scales of $\delta c_{1a,S}$ and $\delta c_{1b,S}$, then the epistemic uncertainty of the site effects is $\phi_{S2S}^2 = \omega_{1a,S}^2 + \omega_{1b,S}^2$.

The response variable of the regression is denoted as y. For a pseudo-spectral acceleration (PSA) or Effective Amplitude Spectrum (EAS) GMM, y is equal to $\ln(PSA)$ or $\ln(EAS)$.

The location of the source, site, etc. are required in the non-ergodic GMMs included in this issue to define the spatially-varying non-ergodic terms. The coordinates of the earthquake, site, and mid-point between the source and site are denoted as: t_E, t_S, t_{MP}, respectively. The definition of the location of the earthquake, t_E (e.g. epicenter, the closest point to the site, etc.) is defined in each study. Similarly, the cell coordinates are denoted as t_C; the exact point of the cell (e.g. center, lower left corner, etc.) to which t_C corresponds is defined in each study.

The star superscript is used to denote the new scenarios and values of non-ergodic coefficients predicted for the new scenarios. For instance, t_E^* corresponds to the source locations of the new scenarios where systematic source effects will be predicted, and $\delta c_{i,E}^*(t_E^*)$ corresponds to the non-ergodic source adjustments of these scenarios.

A list of abbreviations and a glossary of all terms used in this report are summarized on Chapter 9.

2.2 DEVELOPMENT OF ERGODIC GROUND-MOTION MODELS

A typical GMM has a model for the base magnitude, distance, and linear site scaling, and may include more complicated features for non-linear site response, hanging-wall effects, basin effects, etc. For example, the median for the ASK14 (Abrahamson et al., 2014) GMM has the following form:

$$\begin{aligned} f_{erg}(M, R_{rup}, V_{S30}, ...) =& c_1 + c_2 M + c_3 (8.5 - M)^2 + (c_4 + c_5 M) ln(R_{rup} + c_6) + c_7 R \\ & + c_8 F_{RV} + c_9 F_N + c_{10} \ln(V_{S30}/V_{ref}) \\ & + f_{NL}(V_{S30}, \mu_{1100}) + f_{HW}(M, R_{rup}, R_x, Dip) \end{aligned} \tag{2.1}$$

where f_{erg} is the ergodic median ground-motion, c_i are the ergodic scaling coefficients, F_{RV} and F_N are the reverse and normal fault scaling factors, f_{NL} is the non-linear site effects scaling, f_{HW} is the hanging wall scaling, M is the moment magnitude, R_{rup} is the closest point on the rupture-to-site distance, R_x is the horizontal distance from the top edge of the rupture measured perpendicular to the fault strike, and V_{S30} is the time-average shear wave velocity at the top $30m$.

A key aspect of GMMs used for seismic hazard studies is that in engineering applications, they need to be extrapolated outside the data range. Although a GMM is developed through a regression analysis, constraints are often imposed on the coefficients to ensure that the GMM extrapolates consistently with a physical-based scaling.

Because the recordings for a single earthquake are correlated, it is common to use a mixed-effects regression when developing GMMs:

$$y_{es} = f(M, R_{rup}, V_{S30}, ...) + \delta W_{es} + \delta B_e \tag{2.2}$$

in which the left-hand-side is the observed ground motion and δB_e is the between-event aleatory term for the e^{th} earthquake and δW_{es} is the within-event aleatory term for the s^{th} station and from the e^{th} earthquake. δB_e and δW_{es} are assumed to be normally distributed with zero mean and τ and ϕ standard deviations, respectively.

2.2.1 Maximum Likelihood Estimation

In GMM development, the maximum likelihood estimation (MLE) is often used to obtain point estimates of the GMM coefficients (fixed terms) and standard deviations of the aleatory terms (random terms). In the past, the procedure outlined in Abrahamson and Youngs (1992) was commonly used to estimate the mixed-effect terms. More recently, statistical packages such as LME4 (Bates et al., 2015) in the statistical software R (R Core Team, 2020) are used to obtain point estimates and significance test statistics of the mixed terms.

In MLE, the parameters of the model are estimated by maximizing the log-likelihood function, or more commonly by minimizing the negative of the log-likelihood function. The log-likelihood is a measure of how likely it is to observe the data given the model parameters. With the assumption that δB_e and δW_{es} independent with no spatial correlation and that they are normally distributed, the log-likelihood function is given by:

$$\ln \mathcal{L} = \frac{N}{2} \ln(2\pi) - \frac{1}{2} \ln |\mathbf{C}| - \frac{1}{2} (y - \mu)^T \mathbf{C}^{-1} (y - \mu) \tag{2.3}$$

and the covariance matrix is given by:

$$\begin{aligned}\mathbf{C} =&\phi^2\mathbf{I}_N + \tau^2\Sigma_{i=1}^{N_e}\mathbf{1}_{n_i} \\ =&\begin{bmatrix} \phi^2+\tau^2 & \tau^2 & \tau^2 & & & & \\ \tau^2 & \phi^2+\tau^2 & \tau^2 & & & \mathbf{O} & \\ \tau^2 & \tau^2 & \phi^2+\tau^2 & & & & \\ & & & \phi^2+\tau^2 & \tau^2 & & \\ & & & \tau^2 & \phi^2+\tau^2 & & \\ & \mathbf{O} & & & & \phi^2+\tau^2 & \tau^2 \\ & & & & & \tau^2 & \phi^2+\tau^2 \end{bmatrix}\end{aligned} \tag{2.4}$$

where $\mathbf{I_N}$ is the identity matrix of size N, which is the total number of recordings, $\mathbf{1}_{n_i}$ is a matrix of ones, N_e is the number of events, n_i is the number of recordings of the i^{th} event, and μ is the mean predicted value.

In this approach, assuming that all recordings of the same earthquake are grouped together, the covariance matrix has a simple block diagonal form. The diagonal elements of $\mathbf{C}$ are equal to $\tau^2 + \phi^2$, the off-diagonal elements that are associated with recordings of the same earthquake are equal to τ^2, and the remaining elements are equal to zero.

Maximum likelihood regressions are computationally inexpensive, as there are efficient methods to minimize the negative of the log-likelihood (e.g. Bound Optimization BY Quadratic Approximation (BOBYQA), Powell (2009)). The most involved step at each iteration is to compute the inverse of the covariance matrix; however, due to its block diagonal and sparse nature, the process is computationally efficient to perform.

2.2.2 Other Methods for Ergodic Models

Ergodic GMMs have also been developed using Bayesian regression. Bayesian models have been used successfully in the development of a FAS GMM for Mexico City (Ordaz et al., 1994), in deriving a PSA GMM that includes the correlation between spectral periods and the correlation between the GMM coefficients (Arroyo and Ordaz, 2010a,b), in capturing the uncertainty of model parameters, such as V_{S30} (Kuehn and Abrahamson, 2018), and in the development of ergodic GMMs with truncated data (Kuehn et al., 2020). More closely related to the non-ergodic GMMs, Hermkes et al. (2014) used a Bayesian GP regression to derive a non-parametric ergodic GMM for shallow crustal events. Bayesian regression has a higher computational cost than MLE which is why it is less commonly used in GMM development.

GMMs have also been derived through artificial neural networks (ANNs). Derras et al. (2014) proposed an ANN that partitions the residuals into within-event and between-event terms and used it to develop an ergodic GMM for Europe. Withers et al. (2020) applied an ANN to develop an ergodic GMM with ground motions from the CyberShake simulations for Southern California. This is a promising approach, especially for large databases, as the method scales well to many GBs of data that are frequently produced from simulation outputs ($> 10^8$ records). However, extra prudence is required as the modeler does not have direct control over the model behavior (such as interdependency among input terms) and which may limit the accurate extrapolation outside the range of training predictor variables. These concerns can be mitigated by applying physics-based constraints on the model or by augmenting the trained synthetic databases with empirical records but requires additional validation to ensure that conditions within the synthetic ground motions are consistent with empirical records and do not introduce any inherent bias within the data.

2.3 DEVELOPMENT OF PARTIALLY NON-ERGODIC GROUND-MOTION MODELS

The term “partially non-ergodic” has sometimes been used for GMMs that include mean regional differences. Here, we use the term only for GMMs that include differences due to the location of the site and/or the location of the source, not for average differences between broad regions. One such partially non-ergodic GMM approach consists of capturing systematic (i.e., site-specific) site effects (Stewart et al., 2017). Every site has its own velocity profile which leads to a repeatable site amplification relative to the reference profile of a GMM for the same V_{S30} (Lavrentiadis, 2021). This amplification is the same for all ground motions at the site of interest and is not applicable to different sites. However, in an ergodic model, any misfit between a ground-motion observation and the median ground-motion estimate is considered aleatory in nature (i.e. random). That is, ergodic GMMs are based on an assumption that the range of site amplification between different sites with the same V_{S30} is the same as the range of site amplification at the site of interest. It is the goal of partially non-ergodic GMM to properly categorize the systematic site-specific site amplification effects and remove them from the aleatory terms.

In the partially non-ergodic model, the ergodic within-event residual is partitioned into a site-specific site term and the new remaining within-event within-site residual. Using the Al Atik et al. (2010) notation:

$$\delta W_{es} = \delta S2S_s + \delta WS_{es} \tag{2.5}$$

$\delta S2S_s$ term represents the systematic difference between the site amplification at the s^{th} site and the site amplification in the ergodic GMM.

The parameters of a partially non-ergodic GMM can be formulated as a mixed-effects model with three random terms (δB_e, δWS_{es}, and $\delta S2S_s$):

$$y_{es} = f_{erg}(M, R, V_{S30}, ...) + \delta S2S_s + \delta B_e + \delta WS_{es} \tag{2.6}$$

δB_e, δWS_{es}, $\delta S2S_s$ are assumed to be normally distributed with zero means and τ_0, ϕ_{SS}, and ϕ_{S2S} standard deviations, respectively. This leads to a more complicated covariance matrix with more non-zero off-diagonal terms:

$$\begin{aligned}\mathbf{C} &= \phi_{SS}^2 \mathbf{I}_N + \phi_{S2S}^2 \Sigma_{i=1}^{N_s} \mathbf{1}_{n_i} + \tau_0^2 \Sigma_{i=1}^{N_e} \mathbf{1}_{n_i} \\ &= \begin{bmatrix} \phi_{SS}^2 + \phi_{S2S}^2 + \tau_0^2 & \tau_0^2 & \phi_{S2S}^2 & 0 \\ \tau_0^2 & \phi_{SS}^2 + \phi_{S2S}^2 + \tau_0^2 & 0 & \phi_{S2S}^2 \\ \phi_{S2S}^2 & 0 & \phi_{SS}^2 + \phi_{S2S}^2 + \tau_0^2 & \tau_0^2 \\ 0 & \phi_{S2S}^2 & \tau_0^2 & \phi_{SS}^2 + \phi_{S2S}^2 + \tau_0^2 \end{bmatrix}\end{aligned} \tag{2.7}$$

The main difference to the covariance matrix of the ergodic GMM (Equation 2.4) is that the elements that are associated with the same station include the ϕ_{S2S}^2 variance. In this framework, $\sqrt{\tau_0^2 + \phi_{SS}^2}$ is the aleatory variability of the GMM, and, ϕ_{S2S} is the epistemic uncertainty of the site term at a site without site-specific data to constrain the site term.

Alternatively, $\delta S2S_s$ can be estimated directly by partitioning the ergodic within-event residuals, δW_{es}, into $\delta S2S_s$ and δWS_{es}. This approach is expected to give similar results, but it can be problematic if some of the systematic site effects have been mapped into the ergodic event terms.

The $\delta S2S_s$ of such a partially non-ergodic GMM is spatially independent. This is a contrast with the GP-based approach (Section 2.4.1), which allows for $\delta S2S_s$ to be spatially correlated.

2.4 DEVELOPMENT OF NON-ERGODIC GROUND-MOTION MODELS

The fully non-ergodic GMM extends the partially non-ergodic GMM to account for systematic and repeatable source and path effects in addition to the systematic site effects. For that, two additional non-ergodic terms are added:

$$y_{es} = f_{erg}(M, R_{rup}, V_{S30}, ...) + \delta S2S_s + \delta P2P_{es} + \delta L2L_e + \delta B_e^0 + \delta WS_{es}^0 \tag{2.8}$$

The $\delta L2L_e$ term is the systematic source-specific adjustment to the median ground motion in the base ergodic model. It is related to repeatable effects in the release of seismic energy from a source in a region. For instance, $\delta L2L_e$ will be positive if the average stress drop of earthquakes in a region (i.e. fault system) is systematically larger than the global average. Supporting this argument, Trugman and Shearer (2018) found a strong correlation between the stress drop and between-event term of an ergodic GMM. Similarly, the $\delta P2P_{es}$ term represents the repeatable difference in the propagation of the seismic waves between a source and site and the ergodic GMM. The $\delta P2P_{es}$ term will be positive if the attenuation in a geographical region is less than the global average.

The non-ergodic terms $\delta L2L_e$ and $\delta P2P_{es}$ are assumed to be normally distributed with zero means and τ_{L2L} and ϕ_{P2P} standard deviation, respectively. The remaining aleatory terms, δB_e^0 and δWS_{es}^0, are assumed to be normally distributed with zero means and τ_0 and ϕ_0 standard deviations.

The different GMM paradigms (e.g. ergodic, partially non-ergodic, non-ergodic GMMs) should have similar size total aleatory variability and epistemic uncertainty: $\sqrt{\phi^2 + \tau^2} \approx \sqrt{\phi_{S2S}^2 + \phi_{SS}^2 + \tau^2} \approx \sqrt{\tau_{L2L}^2 + \phi_{P2P}^2 + \phi_{S2S}^2 + \phi_0^2 + \tau_0^2}$ as there is no change in the amount of information – what is different between the three approaches is how the provided information is treated (i.e. repeatable or random). This is a useful check for ensuring that the epistemic uncertainty and aleatory variability of a GMM are not overestimated or underestimated. However, it should be noted that the size of aleatory variability and epistemic uncertainty also depends on the modeling approach. For example, a single-station partially non-ergodic GMM, such as SWUS15 (Abrahamson et al., 2015) has, largely, a constant epistemic uncertainty, whereas, a non-ergodic GMM developed as GP has a scenario-dependent epistemic uncertainty. Therefore, this check is primarily applicable at the center of the ground motion data, not at the model extrapolation.

Lin et al. (2011) estimated the standard deviations for all three non-ergodic terms using ground-motion data from Taiwan. The $\delta S2S_s$ was modeled as a random term based on the site ID, and $\delta L2L_e$ was modeled as a spatially correlated random variable based on the site location using standard geostatistics. A more complex spatial correlation model was used for $\delta P2P_{es}$, as it depends both on the source and site location. For a single site, the $\delta P2P_{es}$ correlation is stronger if the earthquakes are closer together, as the seismic waves travel through the same part of the crust. The systematic path effects were found to result in the largest reduction of the aleatory variability followed by the systematic site effects. Overall, including all three effects led to about a 40% reduction in the total aleatory standard deviation compared to the ergodic GMM.

In the previous formulation, the non-ergodic effects were modeled with normal distributions, which may not always be appropriate, particularly, for the path terms; for similar variations in the earth's crust, a far apart source-site pair will have more pronounced path effects than a source-site pair that is closer together. The distance dependence of the path effects is not significant if all records in the dataset have similar rupture distances, but it can be important if the range of R_{rup} is large.

An alternative option is to describe the non-ergodic GMM as a Varying coefficient model (VCM). In this approach, the non-ergodic terms are scaled by different model variables (e.g. R_{rup}, V_{S30}) which provides a more flexible framework to model the systematic effects. More details on the development of non-ergodic GMMs as GP VCMs are provided in the next section.

2.4.1 Gaussian Process Models

The non-ergodic GMMs developed with the tools presented in this report are classified as VCM, as the non-ergodic terms are dependent on the earthquake and site locations in addition to any other input parameters (e.g. V_{S30}):

$$\begin{aligned} y_{es} =& f_{nerg}(M, R_{rup}, V_{S30}, ..., t_S, t_E) + \delta B^0_e + \delta W S^0_{es} \\ =& f_{erg}(M, R_{rup}, V_{S30}, ...) + \delta S2S(V_{S30}, ..., t_S) + \delta P2P(R_{rup}, ..., t_E, t_S,) \\ &+ \delta L2L(M, ..., t_E) + \delta B^0_e + \delta W S^0_{es} \end{aligned} \tag{2.9}$$

with f_{nerg} corresponding to the median non-ergodic ground motion for a particular pair of source and site, and f_{erg} corresponding to the median ergodic ground motion (i.e. the median ground motion for all sources and sites). The systematic source term, $\delta L2L$, is modeled as a function of the earthquake coordinates (t_E), and the systematic site term, $\delta S2S$, is modeled as a function of the site coordinates (t_S). The systematic path term, $\delta P2P$, is more complex as it depends both on the earthquake and site location. The cell-specific anelastic attenuation which is used to capture the systematic path effects is described in Section 2.4.1.3.

At first sight, the development of a non-ergodic VCM GMM may seem futile due to the large number of non-ergodic terms that need to be estimated. If the state of California is broken into a 5×5 km grid, there would be approximately $20{,}000$ grid points and so, at minimum, $60{,}000$ non-ergodic coefficients that would need to be estimated; that is the simplest non-ergodic model where the systematic source, site, and path effects are captured with one coefficient each. It is unfeasible to derive such a model with the existing datasets as they contain, at best, in the order of $10{,}000$ recordings. Fortunately, this is not a problem in VCM due to the spatial correlation structure imposed on the non-ergodic coefficients.

In the statistical approaches described so far, the GMM coefficients are treated as fixed parameters. That is, every coefficient has a single value which is estimated by the MLE or another frequentist approach. In a GMM that is developed as a VCM GP, the model coefficients are treated as random variables that are assumed to follow Normal (Gaussian) distributions. The choice of the mean and covariance function of these distributions is what controls the behavior of each coefficient; for instance, whether a coefficient is constant over a domain, whether it varies continuously on some finite length scale, or whether it is spatially independent (i.e. the value of the coefficient at some location is independent of the value of the coefficient at some other location). In this sense, in a GP regression, the GMM coefficients are modeled similarly to the aleatory terms in the mixed-effects regression (Section 2.2.1). It is these constraints on the GMM coefficients imposed by the covariance function that make the development of a non-ergodic VCM GP GMM tractable. Due to this, the non-ergodic GMM coefficients do not have to be estimated directly; instead, only the hyperparameters that control the distributions of the non-ergodic terms need to be estimated by the regression. With the current size of datasets, the number of hyperparameters is typically about 10.

Furthermore, this formulation leads to a scenario-dependent epistemic uncertainty that is more appropriate than the constant epistemic uncertainty assumed in earlier studies. In a VCM GP GMM, the non-ergodic coefficients have a constant epistemic uncertainty, but the epistemic uncertainty of the ground motion is scaled by the GMM input variables. For example, consider a non-ergodic GMM based on the base model (Equation 2.1) where the systematic path effects are modeled with a spatially varying geometrical spreading coefficient that is a function of the earthquake coordinates ($c_{4,E}(t_E)$). In this case, the epistemic uncertainty of $c_{4,E}(t_E)$ will be equal to $\psi_{4,E}(t_E)$ and the epistemic uncertainty of the ground motion due to the systematic path effects will be equal to $\phi_{P2P} = \psi_{4,E}(t_E)\ln(R + c_6)$. This results in a distance-dependent epistemic uncertainty, the epistemic uncertainty is higher for sites farther from the source, which is different from the ϕ_{P2P} values of Lin et al. (2011) which are independent of the source-to-site distance.

GP is a particular case of a hierarchical Bayesian model as it is expressed on multiple levels. At the base level are the GMM coefficients and aleatory terms which have a direct impact on the response variable y and are defined in terms of some distributions; the variables that constitute this level are called model parameters ($\vec{\theta}$). At the next level is the set of variables that control the distributions of θ. The variables of the upper level are called model hyperparameters ($\vec{\theta}_{hyp}$), which in turn could be defined in terms of some other distributions or they could be fixed. As an example, in this context, ϕ_0 and τ_0 are hyperparameters that control the distributions of the parameters: δWS^0_{es} and δB^0_e.

There are two general approaches for developing a non-ergodic GMM with a GP regression. In the first approach, which was followed by Landwehr et al. (2016), all the coefficients, ergodic and non-ergodic, were modeled as GPs. In that case, the non-ergodic GMM is developed from the beginning and the response variable is typically the log of the ground-motion parameter (e.g. $log(PSA)$). An alternative approach, which was followed by Kuehn (ress) and Lavrentiadis et al. (2021), is to model the non-ergodic coefficients or non-ergodic coefficient adjustments as GPs and keep the ergodic terms fixed. Here, the non-ergodic GMM is based on an existing ergodic GMM and the response variable is the ergodic residual. An advantage of this approach is that the extrapolation to large magnitudes and short distances from the underlying ergodic GMM is preserved in the non-ergodic GMM.

The remaining parts of this section summarize the different elements of the VCM GP GMM development: Bayesian regression, covariance functions of the prior distributions commonly used in VCM GP GMM, cell-specific anelastic attenuation, prediction of the median, and epistemic uncertainty of the non-ergodic coefficients and median ground motion at new locations.

2.4.1.1 Bayesian Regression

In Bayesian statistics, the uncertainty of the model parameters and hyperparameters, $\vec{\theta}$ and $\vec{\theta}_{hyp}$, before observing the data is expressed by the prior distribution ($p(\vec{\theta}, \vec{\theta}_{hyp})$). The uncertainty of $\vec{\theta}$ and $\vec{\theta}_{hyp}$ is updated based on the ground-motion observations, $\vec{y}$, and ground-motion parameters (such as M, R_{rup}, V_{S30}, etc., collectively for all records noted as $\vec{x}$) to produce the posterior distribution ($p(\vec{\theta}, \vec{\theta}_{hyp}|\vec{y}, \vec{x})$). The Bayes theorem provides the means for this calculation:

$$p(\vec{\theta}, \vec{\theta}_{hyp}|\vec{y}, \vec{x}) = \frac{\mathcal{L}(\vec{\theta}, \vec{\theta}_{hyp}) p(\vec{\theta}, \vec{\theta}_{hyp})}{p(\vec{y}, \vec{x})} \tag{2.10}$$

Often, the normalizing distribution $p(\vec{y}, \vec{x})$ is omitted for computational efficiency as it is not required to sample or compute the maximum of the posterior. In this case the posterior distribution is expressed as:

$$p(\vec{\theta}, \vec{\theta}_{hyp}|\vec{y}, \vec{x}) \propto \mathcal{L}(\vec{\theta}, \vec{\theta}_{hyp}) p(\vec{\theta}, \vec{\theta}_{hyp}) \tag{2.11}$$

The influence of the ground-motion data in the posterior distribution is expressed through the likelihood function ($\mathcal{L}(\vec{\theta}, \vec{\theta}_{hyp})$) — it corresponds to the likelihood (i.e probability) of observing the data given some values for $\vec{\theta}$ and $\vec{\theta}_{hyp}$. The likelihood for a single observation can be estimated with the functional form of the GMM as:

$$\mathcal{L}(\vec{\theta}, \vec{\theta}_{hyp}) = pdf(\vec{y}|f_{nerg}(\vec{x}, \vec{\theta}, \vec{\theta}_{hyp}) + \delta\vec{B}_e, \phi_0^2) \tag{2.12}$$

Because all correlated terms (i.e. non-ergodic terms and between event residuals) are included in the mean, the misfit: $y - (f_{nerg}(x, \vec{\theta}, \vec{\theta}_{hyp}) + \delta\vec{B}_e)$, which corresponds to $\delta W S_{es}^0$, is independently and identically distributed, thus, the joint likelihood of all observations is the product of the likelihoods of individual observations:

$$\begin{aligned} \mathcal{L}(\vec{\theta}, \vec{\theta}_{hyp}) &= pdf(\vec{y}|f_{nerg}(\vec{x}, \vec{\theta}, \vec{\theta}_{hyp}) + \delta\vec{B}_e, \phi_0^2) \\ &= \Pi_{e=1}^{n_e} \Pi_{s=1}^{n_s} pdf(y_{es}|f_{nerg}(x_{es}, \vec{\theta}, \vec{\theta}_{hyp}) + \delta\vec{B}_e, \phi_0^2) \end{aligned} \tag{2.13}$$

improving computational efficiency. The likelihood function is written in vector notation in the first line and expanded in the second line of Equation 2.13.

The prior distributions express our knowledge and beliefs about $\vec{\theta}$ and $\vec{\theta}_{hyp}$. They may come from prior experience in building non-ergodic GMMs or based on a desired model behavior (i.e.

penalize model complexity if not supported by the data (Simpson et al., 2017)). When there is little information about $\vec{\theta}$ and $\vec{\theta}_{hyp}$, weakly informative priors can be used. These are chosen as wide priors distributions so that the posterior distribution is primarily controlled by the likelihood function.

The prior distributions of the non-ergodic effects are spatially uniform with zero means and large standard deviations because prior to interrogating the ground-motion data, the systematic effects are unknown. With the aid of the likelihood function and ground-motion data, the non-ergodic effects can be estimated close to stations and past earthquake locations. This results in posterior distributions that are spatially varying with non-zero means and smaller standard deviations where the non-ergodic effects have been estimated. Zero posterior standard deviations would imply that the non-ergodic effects are known with absolute certainty.

Historically, Bayesian inference has seen limited use due to its high computational cost compared to point-estimate inference with MLE. However, in recent years, with the increase in computational speed, Bayesian models have been gaining wider adoption. There are three main computationally-tractable approaches to obtain the posterior distributions of complex models that do not have analytical solutions; they are summarized below.

The maximum a posteriori (MAP) approach finds the values of $\vec{\theta}$ and $\vec{\theta}_{hyp}$ that correspond to the mode of the posterior. The posterior distribution is proportional to the product of the likelihood function and the prior distribution (Equation 2.11). MAP can be found by minimizing the negative of this product, which can be numerically computed easily with gradient-based methods. In this sense, MAP is equivalent to a penalized MLE where the prior distribution acts as a regularization on the likelihood function. MAP is computationally faster than the other numerical solutions of Bayesian models, but its main shortcoming is that it provides a point estimate not the entire posterior distribution; thus, the uncertainty of the model cannot be assessed. The GPML toolbox (Rasmussen and Nickisch, 2010) available in Matlab and Octave provides such MAP estimates for Gaussian Process models.

The Markov Chain Monte Carlo (MCMC) approach generates samples from the posterior distributions that are used in the inference of $\vec{\theta}$ and $\vec{\theta}_{hyp}$. This approach is able to recreate the full posterior distribution, but it is computationally slow. An in-depth review of this method can be found at Brooks et al. (2011). Widely used statistical software that have implemented this approach are: JAGS (Plummer, 2003), BUGS (Lunn et al., 2009), and STAN (Stan Development Team, 2022) for general Bayesian models, and GPflow (van der Wilk et al., 2020) in Python for GPs.

A more recent approach consists in using approximation methods to compute the posterior distributions of $\vec{\theta}$ and $\vec{\theta}_{hyp}$. These approximation solutions are applicable to specific families of Bayesian models. One such approximation method is the integrated nested Laplace approximation, INLA (Rue et al., 2009); it uses the Laplace approximation to efficiently compute the approximations to the marginal posterior distributions of Latent Gaussian Models (LGMs). In this family of models, the response variable is expressed as an additive function of the model parameters ($y = \Sigma_{i=1}^{n} \theta_i x_i$), and all θ_i follow Normal prior distributions. INLA is a useful approximation for developing GMMs as both ergodic GMMs and non-ergodic VCM GP GMMs can be formulated as LGMs. Further information regarding the INLA approximation can be found in

Krainski et al. (2019, 2021) and Wang et al. (2018). A primer for developing ergodic GMMs with INLA can be found in Kuehn (2021).

Other methods for Bayesian regression include the Variational Inference (Blei et al., 2017) which approximates the posterior distribution with a member of a closed-form probability distribution.

2.4.1.2 Covariance Functions

The covariance functions, or kernel functions as often called in the literature, of the prior distributions are a crucial ingredient of the GP regression. They impose a correlation structure which dictates how a random variable (i.e. a coefficient or the ground-motion intensity parameter) varies in space. The covariance functions described in this section are isotropic and stationary; that is, the size and rate of spatial variation they impose is independent of the direction and location. Although this is likely a simplification for the systematic ground-motion effects, most of the non-ergodic GMM do not use non-stationary and anisotropic kernel functions due to their additional computational challenge. Ground-motion studies that used non-stationary correlation structures include Kuehn and Abrahamson (2020) and Chen et al. (2021). Other studies that applied non-stationary and anisotropic correlation structures to GP regressions include Paciorek and Schervish (2006) and Finley (2011).

The four covariance functions described here are: the identity kernel function, the spatially independent kernel function, the constant kernel function, and the exponential kernel function. Examples, where these covariance functions are combined to create more complex spatial correlation structures, are provided at the end of this section. The covariance matrices, which are used in the regression and prediction of GP, are created by evaluating the covariance functions at all indices, such as the earthquake or station IDs, or coordinate pairs, such as the earthquake or station coordinates:

$$\mathbf{K}_{i\,kl} = \boldsymbol{\kappa}_i(t_k, t_l) \tag{2.14}$$

where $\boldsymbol{\kappa}_i$ and $\mathbf{K}_i$ are the covariance function and covariance matrix for the i^{th} coefficient, and t_k and t_l are the k^{th} and l^{th} indices or coordinate values. Indices are used as input to the kernel function if the correlation structure of the i^{th} coefficient depends on information such as the event or station number, while coordinates are used as input if the correlation structure depends on information like the event or site location. In vector notation the covariance matrix is defined as:

$$\mathbf{K}_i = \boldsymbol{\kappa}_i(\vec{t}, \vec{t}') \tag{2.15}$$

where $\vec{t}$ and $\vec{t}'$ are index or coordinate arrays. If $\mathbf{K}_i$ is used in the regression phase, $\vec{t}$ and $\vec{t}'$ correspond to the existing scenarios; these are the scenarios in the regression dataset. However, if $\mathbf{K}_i$ is used in the prediction phase, $\vec{t}$ and $\vec{t}'$ correspond to combinations of the existing and new scenarios. Further details on the GP prediction are provided in Section 2.4.1.4.

The identity kernel function is given by:

$$\boldsymbol{\kappa}_i(t_k, t_l) = \omega_i^2\, \delta(k - l) \tag{2.16}$$

where $\delta(x)$ is the Dirac delta function ($\delta(x = 0) = 1$ and $\delta(x \neq 0) = 0$). It is used for random variables that are statistically independent with ω being the standard deviation of the normal distribution. It generates a covariance matrix that is equal to ω^2 along the diagonal and zero everywhere else. This kernel function is used to model the within-event within-site aleatory term, δWS_{es}.

The spatially independent kernel function is given by:

$$\boldsymbol{\kappa}_i(t_k, t_l) = \omega_i^2\ \delta(\|t_k - t_l\|) \tag{2.17}$$

This kernel imposes perfect correlation between random variables at the same location or with the same index, and zero correlation between random variables at different locations or with different indices. The hyper-parameter ω defines the size of the variability, that is, how much the values of the random variable vary between points that are not collocated. If t_k and t_l are pair of coordinates, $\|t_k - t_l\|$ corresponds to the L2 distance norm between the two coordinates (i.e. the Cartesian distance between coordinates). If t_k and t_l are indices, $\|t_k - t_l\|$ corresponds to the absolute difference between the two values.

Depending on the software, the covariance matrix of a spatially independent non-ergodic term can be modeled either with the identity or spatially independent kernel function. For example, consider a spatially independent site term, $\delta\vec{c}_{i,S}$, that has a unique value at every site but zero spatial correlation. If a statistical software requires all terms to be of size N, where N is the number of records, the spatially independent kernel function should be used. That is because, if k^{th} and l^{th} recording have the same station coordinates, $\delta c_{i,S\,k}$ and $\delta c_{i,S\,l}$ should be equal (i.e. perfectly correlated). In this approach, all covariance matrices are size $N \times N$. However, if a statistical software can model terms of different sizes, the identity kernel function can be used. In this case, it is more efficient to estimate $\delta\vec{c}_{i,S}$ at unique station locations and then pass it to the associated recordings. In this approach, $\delta\vec{c}_{i,S}$ is uncorrelated, as every station coordinate is repeated only once, thus it can be modeled with an identity covariance matrix of size $N_s \times N_s$, where N_s is the number of stations, reducing the and number of operations and memory requirements.

The constant kernel function is given by:

$$\boldsymbol{\kappa}_i(t_k, t_l) = \omega_i^2 \tag{2.18}$$

with ω controlling the deviation from the mean of the prior distribution. It imposes perfect correlation between all random variables so that they all have the same offset from the mean function. As an example, in Landwehr et al. (2016), the constant kernel function was applied to all coefficients to model their deviation from the mean of the prior which was equal to zero.

Alternatively, constant offsets in the coefficients can be modeled with a one-dimensional prior distribution on the mean function of the coefficient, as in the case of $c_{ca,p}$ in Lavrentiadis et al. (2021). It depends on the modeler and software which option is more attractive. The main advantage of the first option is that it includes all information in the kernel function, while the main advantage of the second option is that it can lead to a sparse covariance matrix.

The exponential kernel function is given by:

$$\boldsymbol{\kappa}_i(t_k, t_l) = \omega_i^2\ e^{-\frac{\|t_k - t_l\|}{\ell_i}} \tag{2.19}$$

This kernel function is applied to spatially varying random variables. The hyperparameters ℓ and ω control the specific length scale and size of the spatial variation. At the two extremes of ℓ, the exponential kernel converges to a spatially independent and constant kernel function. For $\ell \to 0^+$ the spatial correlation weakens converging to a spatially independent kernel function, while for $\ell \to +\infty$ the correlation becomes stronger converging to a constant kernel function. With this kernel function, a random variable is assumed to vary continuously but not smoothly in space (i.e. the spatial variation of the random variable is continuous, but the first derivative of the spatial variation is not). This kernel function is widely used in geostatistics to model spatially varying phenomena.

Another kernel function for modeling continuously spatially varying random variables is the squared exponential. This kernel function is infinitely differentiable resulting in very smooth spatial variations that may be unrealistic for spatial processes (Stein, 1991). However, the main advantage of this covariance function is that it is separable in the X and Y coordinates which allows for efficient approximations of the kernel function for large datasets (Lacour, 2022).

More complex correlation structures can be built by combining the kernel functions described above using the properties of the Normal distribution. For example, assume a non-ergodic site adjustment $\delta c_{i,S}$ that is the combined effect of an underlying continuous adjustment over large distances and a site-specific adjustment. Such a site adjustment can be broken into individual components: $\delta c_{ia,S}$ for the underling continuous adjustment, and $\delta c_{ib,S}$ for site-specific adjustment, with $\delta c_{i,S} = \delta c_{ia,S} + \delta c_{ib,S}$. In this case, $\delta c_{ia,S}$ can be assigned a prior distribution which has a zero prior mean and an exponential kernel function ($\boldsymbol{\kappa}_{ia,S}$), and $\delta c_{ib,S}$ can be assigned a prior distribution which has a zero prior mean and a spatially-independent kernel function ($\boldsymbol{\kappa}_{ib,S}$):

$$\begin{aligned} \delta\vec{c}_{ia,S} &\sim \mathcal{N}\left(\vec{0},\ \boldsymbol{\kappa}_{ia,S}(\vec{t}_S, \vec{t}_S)\right) \\ \delta\vec{c}_{ib,S} &\sim \mathcal{N}\left(\vec{0},\ \boldsymbol{\kappa}_{ib,S}(\vec{t}_S, \vec{t}_S)\right) \end{aligned} \tag{2.20}$$

Based on the linear properties of the Normal distribution, the prior distribution of $\delta c_{i,S}$ has a mean which is equal to the sum of mean functions of the individual components, and a kernel function which is equal to the sum of the kernel functions of the individual components:

$$\delta\vec{c}_{a,S} \sim \mathcal{N}\left(\vec{0},\ \boldsymbol{\kappa}_{ia,S}(\vec{t}_S, \vec{t}_S) + \boldsymbol{\kappa}_{ib,S}(\vec{t}_S, \vec{t}_S)\right) \tag{2.21}$$

Similarly, the kernel function of the median non-ergodic ground motion can be obtained by combining the kernel functions of the GMM coefficients. For simplicity, only three terms of the ergodic base GMM (Equation 2.1) are used in this example:

$$f_{nerg} = c_1 + c_{4,E}(t_E)\ln(R_{eff}) + c_{10,S}(t_S)\ln(V_{S30}/V_{ref}) \tag{2.22}$$

where $R_{eff} = R + c_6$, c_1 is the intercept, $c_{4,E}$ is the geometrical-spreading term which is a function of the earthquake coordinates and scales with $\ln(R_{eff})$, and $c_{10,S}$ is a linear site amplification term which is a function of the site coordinates and scales with $\ln(V_{S30}/V_{ref})$.

If, the GMM coefficients are modeled as GPs with prior distributions:

$$\begin{aligned} \vec{c}_1 &\sim \mathcal{N}\left(\vec{\mu}_1,\ \boldsymbol{\kappa}_1\right) \\ \vec{c}_{4,E} &\sim \mathcal{N}\left(\vec{\mu}_{4,E},\ \boldsymbol{\kappa}_{4,E}(\vec{t}_E, \vec{t}_E)\right) \\ \vec{c}_{10,S} &\sim \mathcal{N}\left(\vec{\mu}_{10,S},\ \boldsymbol{\kappa}_{10,E}(\vec{t}_S, \vec{t}_S)\right) \end{aligned} \tag{2.23}$$

with $\vec{\mu}_i$ and $\boldsymbol{\kappa}_i$ being the mean and kernel functions of the i^{th} coefficient, respectively; the prior distribution of f_{nerg} is equal to:

$$\begin{aligned} \vec{f}_{nerg} \sim \mathcal{N}\Big(&\vec{\mu}_1 + \vec{\mu}_{4,E} \circ \ln(\vec{R}_{eff}) + \vec{\mu}_{10,S} \circ \ln(\vec{V}_{S30}/V_{ref}), \\ &\boldsymbol{\kappa}_1 + \boldsymbol{\kappa}_{4,E}(\vec{t}_E, \vec{t}_E) \circ (\ln(\vec{R}_{eff}) \ln(\vec{R}_{eff})^\intercal) \\ &+ \boldsymbol{\kappa}_{10,S}(\vec{t}_S, \vec{t}_S) \circ (\ln(\vec{V}_{S30}) \ln(\vec{V}_{S30})^\intercal)\Big) \end{aligned} \tag{2.24}$$

in which the symbol $\circ$ corresponds to the element-wise product, and $\ln(\vec{R}_{eff})$ and $\ln(\vec{V}_{S30}/V_{ref})$ are column vectors with the $\ln(R_{eff})$ and $\ln(V_{S30}/V_{ref})$ values of all recordings. A linear combination of Normal distributions follows a Normal distribution. The mean of f_{nerg} is equal to the linear combination of the means of the prior distributions of the coefficients. To get a more intuitive feeling for the kernel function of f_{nerg}, first consider the covariance between just two scenarios $cov(f_{nerg\,k}, f_{nerg\,l})$. By substituting Equation 2.22 into the covariance and assuming the coefficients of the GMM are independent with each other ($cov(c_i, c_j) = 0$ if $i \neq j$) we obtain:

$$\begin{aligned} cov(f_{nerg\,k}, f_{nerg\,l}) &= cov(c_1, c_1) \\ &+ \ln(R_{eff\,k}) cov(c_{4,E}(t_{E\,k}), c_{4,E}(t_{E\,l})) \ln(R_{eff\,l}) \\ &+ \ln(V_{S30\,k}) cov(c_{10,S}(t_{S\,k}), c_{10,s}(t_{S\,l})) \ln(V_{S30\,l}) \\ &= \boldsymbol{\kappa}_1 + \ln(R_{eff\,k}) \boldsymbol{\kappa}_{4,E}(t_{E\,k}, t_{E\,l}) \ln(R_{eff\,l}) \\ &+ \ln(V_{S30\,k}) \boldsymbol{\kappa}_{10,S}(t_{S\,k}, t_{S\,l}) \ln(V_{S30\,l}) \end{aligned} \tag{2.25}$$

The kernel function in Equation 2.24 creates the same covariance as Equation 2.25 for all recordings. For example, for the $c_{1,S}$ contribution, the $\ln(\vec{R}_{eff}) \ln(\vec{R}_{eff})^\intercal$ product creates a matrix with all $\ln(R_{eff\,k}) \ln(R_{eff\,l})$ permutations, and the element-wise product with $\boldsymbol{\kappa}_4(\vec{\chi}_e, \vec{\chi}_e)$ combines these permutations with the covariance of the coefficient.

Generalizing from previous example, the covariance function of the median ground motion between scenarios k and l is:

$$\boldsymbol{\kappa}_{nerg\,kl} = \Sigma_{i=1}^{d}\, x_{i\,k}\, \boldsymbol{\kappa}_i(t_{i\,k}, t_{i\,l})\, x_{i\,l} \tag{2.26}$$

in which $\boldsymbol{\kappa}_{nerg}$ is the kernel function for f_{nerg}, $\boldsymbol{\kappa}_i$ is the kernel function of the i^{th} non-ergodic coefficient, x_i is the independent variable in front of the i^{th} non-ergodic coefficient (e.g. $\ln(R_{eff})$), t_i is input coordinate or ID for $\boldsymbol{\kappa}_i$, and d is the number of the non-ergodic terms.

In matrix notation Equation 2.26 can be defined as:

$$\boldsymbol{\kappa}_{nerg} = \Sigma_{i=1}^{d} \boldsymbol{\kappa}_i(\vec{t}_i, \vec{t}_i) \circ (\vec{x}_i \vec{x}_i^\intercal) \tag{2.27}$$

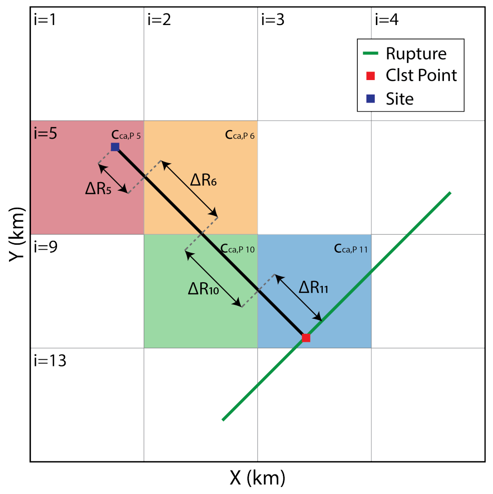


Figure 2.1. Schematic showing the calculation of the cell-path segments for the cell-specific anelastic attenuation. x_{site} is the site location, x_{cls} is the closest point on the rupture to the site, the dashed line indicates the source-to-site path, and the ΔR_i of the i^{th} cell .

2.4.1.3 Cell-specific anelastic attenuation

The cell-specific anelastic attenuation was first proposed by Dawood and Rodriguez-Marek (2013) and then extended by Kuehn et al. (2019) and Abrahamson et al. (2019) as an approach to capture the systematic effects related to the paths. In this method, the domain of interest is divided into a grid of cells and each cell is assigned its own anelastic attenuation. For each recording, the ray path that connects a point on the rupture with the site is broken into cell-path segments (ΔR_i) which are the lengths of the ray within each cell (Figure 2.1). For a given recording, the total anelastic attenuation can be calculated by $f_{atten,p} = \vec{c}_{ca,p} \cdot \Delta\vec{R}$ where $\vec{c}_{ca,p}$ is vector containing the attenuation coefficients of all the cells.

Currently, there is no consensus on the origin point for the ray path. Dawood and Rodriguez-Marek (2013) used the epicenter, while Kuehn et al. (2019) and Lavrentiadis et al. (2021) used the closest point on the rupture to the site, as the length of that path is equal to R_{rup}, which is a common distance metric for anelastic attenuation in ergodic GMMs. Additional research is needed in this area to test different options for the origin of the ray path and also investigate if there is any magnitude dependence in the location of the representative point for finite-fault ruptures.

In GP, the cell attenuation can be modeled similarly to the other spatially varying non-ergodic terms using a Truncated Normal as a prior distribution:

$$\vec{c}_{ca,P} \sim \mathcal{N}(\vec{\mu}_{ca,P}, \boldsymbol{\kappa}_{ca,P}(\vec{t}_C, \vec{t}_C))\mathcal{T}(,0) \tag{2.28}$$

the cell attenuation is limited to be equal or less than zero to ensure the proper extrapolation of the GMM. In statistical software that does not include truncated Normal distributions, the cell-specific attenuation is modeled with a Normal prior, but at a postprocessing step it is checked that no or only a small number of cells have positive attenuation.

The mean of the prior distribution, $\vec{\mu}_{ca,P}$, controls the average anelastic attenuation of the cells, while the kernel function, $\boldsymbol{\kappa}_{ca,P}$, controls their spatial correlation. In regions with sparse coverage, the cell-specific anelastic attenuation is close to $\vec{\mu}_{ca,P}$ as there are not enough data to inform the posterior. In regions with significant coverage, the cell-specific anelastic attenuation deviates from $\vec{\mu}_{ca,P}$ to capture the systematic path effects which influence the ground motion in those regions. The $\vec{\mu}_{ca,P}$ can be either fixed to the anelastic attenuation of the ergodic GMM or be assigned its own prior distribution. The second option is computationally more involved but takes into account the re-weighting of the paths. In an ergodic GMM, the anelastic attenuation is controlled by the attenuation of the areas with high-path coverage; however, in the cell-specific anelastic attenuation, the mean attenuation is determined at the cell level, the path coverage controls the mean and epistemic uncertainty of the individual cells, but it does not have a direct impact on the mean attenuation of all cells, which is why the mean of the cell-specific anelastic attenuation and the ergodic anelastic attenuation can be different.

The kernel functions that were presented in Section 2.4.1.2 can also be used to model the spatial correlation of the cell attenuation. For instance, Kuehn et al. (2019) used the spatially independent kernel function, while Lavrentiadis et al. (2021) used a combination of the exponential and spatially independent kernel function. Other approaches for modeling the spatial correlation of cell-specific anelastic attenuation are the conditional autoregressive (CAR) and simultaneous autoregressive (SAR) models (Ver Hoef et al., 2018). These models have sparse precision matrices (i.e. inverse of covariance matrices) reducing the computational cost.

The prior distribution for the total anelastic attenuation can be derived from the prior distribution for the cell attenuation using the linear transformation properties of the Normal distribution:

$$\vec{f}_{atten,P} \sim \mathcal{N}(\mathbf{R}\,\vec{\mu}_{ca,P}, \mathbf{R}\,\boldsymbol{\kappa}_{ca,p}\,\mathbf{R}^{\intercal})\mathcal{T}(,0) \tag{2.29}$$

where $\mathbf{R}$ is a matrix with the cell-path segments of all recordings, the i^{th} row of $\mathbf{R}$ is equal to $\Delta\vec{R}$ of the i^{th} recording. The $\vec{f}_{atten,P}$ prior distribution can be used in GP to make direct predictions for the median non-ergodic ground motion at new locations (Section 2.4.1.4).

2.4.1.4 Prediction

The median non-ergodic ground motion can be predicted for the new scenarios either by first predicting the non-ergodic coefficients and then substituting them at the non-ergodic functional form or by predicting the non-ergodic ground motion directly. This choice depends on how the GPs are modeled. If the GMM terms are modeled as GPs explicitly, the first method is used. However, if the GMM terms are modeled as GPs implicitly (i.e. they have been integrated out in the likelihood function), the second method is used.

Prediction of non-ergodic coefficients

The non-ergodic coefficient adjustments for the new scenarios can be predicted based on the hyperparameters and posterior distribution of the coefficient adjustments of the existing scenarios. Initially, we consider the case where the non-ergodic coefficient adjustments of the existing scenarios have zero epistemic uncertainty. The joint prior distribution between the non-ergodic coefficient adjustments of the existing and new scenarios is:

$$\begin{bmatrix} \delta\vec{c}_i \\ \delta\vec{c}^*_i \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \vec{0} \\ \vec{0} \end{bmatrix}, \begin{bmatrix} \mathbf{K}_i & \mathbf{k}_i \\ \mathbf{k}_i^{\intercal} & \mathbf{K}^*_i \end{bmatrix} \right) \tag{2.30}$$

in which δc_i are the non-ergodic coefficient adjustments of the existing scenarios, δc^*_i are the non-ergodic coefficient adjustments of new scenarios, $\mathbf{K}_i$ is the prior covariance between all pairs of existing scenarios ($\mathbf{K}_{i\,kl} = \boldsymbol{\kappa}_i(t_k, t_l)$), $\mathbf{K}^*_i$ is the prior covariance between all pairs of new scenarios ($\mathbf{K}^*_{i\,kl} = \boldsymbol{\kappa}_i(t^*_k, t^*_l)$), and $\mathbf{k}_i$ is the prior covariance between all pairs of existing and new scenarios ($\mathbf{k}_{i\,kl} = \boldsymbol{\kappa}_i(t_k, t^*_l)$).

Because of the cross-correlation between the non-ergodic coefficient adjustments of the existing and new scenarios, described by $\mathbf{k}$, the posterior distributions of $\delta\vec{c}^*_i$ can be predicted by ensuring that they are in agreement with $\delta\vec{c}_i$. A naive approach for that would be to generate multiple realizations of $\delta\vec{c}^*_i$ from the joint prior distribution (Equation 2.30) and reject those that are inconsistent with $\delta\vec{c}_i$. The distribution of the accepted realizations $\delta\vec{c}^*_i$ would correspond to the posterior distribution of $\delta\vec{c}^*_i$. Although this is theoretically correct, it would be computationally inefficient. In statistics, this can be performed easily by conditioning $\delta\vec{c}^*_i$ on $\delta\vec{c}_i$, which corresponds to predicting $\delta\vec{c}^*_i$ based on the values of $\delta\vec{c}_i$. The conditional distribution of a joint Normal prior distribution is also a Normal distribution:

$$\delta\vec{c}^*_i|\delta\vec{c}_i \sim \mathcal{N}(\vec{\mu}_{\delta\vec{c}^*_i|\delta\vec{c}_i}, {}_{\delta\vec{c}^*_i|\delta\vec{c}_i}) \tag{2.31}$$

with $\vec{\mu}_{\delta c^*_i|\delta c_i}$ and ${}_{\delta c^*_i|\delta c_i}$ being the mean and covariance of the posterior distributions of δc^*_i given by (Rasmussen and Williams, 2006):

$$\mu_{\delta c^*_i|\delta c_i} = \mathbf{k}_i^{\intercal}\mathbf{K}_i^{-1}\delta c_i \tag{2.32}$$

$${}_{\delta c^*_i|\delta c_i} = \mathbf{K}^*{}_i - \mathbf{k}_i^{\intercal}\mathbf{K}_i^{-1}\mathbf{k}_i \tag{2.33}$$

In other fields where GP regression is used, one is typically interested only in the point-wise uncertainty which means that the mean and epistemic uncertainty of δc^*_i can be calculated independently for each scenario reducing the computational cost. However, in PSHA it is necessary to calculate the full covariance for the new scenarios, as the spatial correlation of $\delta\vec{c}^*_i$, which described by the off-diagonal term of ${}_{\delta c^*_i|\delta c_i}$, needs to be included in the logic tree.

A more realistic case is for there to be some uncertainty in the estimation of the non-ergodic coefficient adjustments of the existing scenarios described by the posterior distribution ($p(\delta\vec{c}_i|\vec{y}, \vec{x})$). This uncertainty can be propagated in the prediction of the non-ergodic coefficient adjustments of the new scenarios by predicting $\delta\vec{c}^*_i$ using all possible values of $\delta\vec{c}_i$ and considering how likely each $\delta\vec{c}_i$ is ($p(\delta\vec{c}_i|\vec{y}, \vec{x})$). In statistics, this is defined as marginalization of $\delta\vec{c}^*_i$:

$$p(\delta\vec{c}^*_i|\vec{y}, \vec{x}) = \int p(\delta\vec{c}^*_i|\delta\vec{c}_i)p(\delta\vec{c}_i|\vec{y}, \vec{x})\, d\delta\vec{c}_i \tag{2.34}$$

where the probability density function $p(\delta c^*_i|\delta c_i)$ can be obtained from the conditional distribution in Equation 2.31.

A closed-form solution for the posterior distribution of $\delta\vec{c}_i^*$ which includes the uncertainty of $\delta\vec{c}_i$ can be obtained if the posterior distribution of $\delta\vec{c}_i$ is assumed to be Normal (Lavrentiadis et al., 2021):

$$\delta\vec{c}_i|y,x \sim \mathcal{N}(\vec{\mu}_{\delta c_i|y,x}, {}_{\delta c_i|y,x}) \tag{2.35}$$

where $\mu_{\delta c_i|y,x}$ is the mean, and ${}_{\delta c_i|y,x}$ is the covariance of the posterior distribution of δc_i. With this assumption, Equation 2.34 results in a Normal distribution:

$$\delta\vec{c}_i^*|\vec{y},\vec{x} \sim \mathcal{N}(\vec{\mu}_{\delta c_i^*|y,x}, {}_{\delta c_i^*|y,x}) \tag{2.36}$$

with the mean and covariance given in Equations (2.37) and (2.38), respectively (Bishop, 2006).

$$\vec{\mu}_{\delta c_i^*|y,x} = \mathbf{k}_i^\intercal \mathbf{K}_i^{-1} \vec{\mu}_{\delta c_i|y,x} \tag{2.37}$$

$${}_{\delta c_i^*|y,x} = \mathbf{K}_i^* - \mathbf{k}_i^\intercal \mathbf{K}_i^{-1} \mathbf{k}_i + \mathbf{k}_i^\intercal \mathbf{K}_i^{-1} {}_{\delta c_i|y,x} (\mathbf{k}_i^\intercal \mathbf{K}_i^{-1})^\intercal \tag{2.38}$$

The assumption that the posterior distribution of δc_i is Normal is considered reasonable because in a GP regression all non-ergodic terms have Normal prior distributions. If the hyperparameters are fixed or follow Normal prior distributions, this assumption would be absolutely valid; however, because some of the hyperparameters are assigned different prior distributions, the posterior distribution of δc_i may slightly deviate from this assumption. For the prediction of $\delta\vec{c}_i^*$, Kuehn (ress) showed that this approximation gives consistent results with Equation 2.34 where the full posterior distribution of $\delta\vec{c}_i$ is used.

The non-ergodic coefficients of the new scenarios (c_i^*) can be computed similarly to δc_i^*, however, the non-zero prior means needs to be considered:

$$\vec{\mu}_{\delta c_i^*|y,x} = \mathbf{k}_i^\intercal \mathbf{K}_i^{-1} (\vec{\mu}_{\delta c_i|y,x} - \vec{\mu}_{\delta c_i}) + \vec{\mu}_{\delta c_i^*} \tag{2.39}$$

$${}_{\delta c_i^*|y,x} = \mathbf{K}_i^* - \mathbf{k}_i^\intercal \mathbf{K}_i^{-1} \mathbf{k}_i + \mathbf{k}_i^\intercal \mathbf{K}_i^{-1} {}_{\delta c_i|y,x} (\mathbf{k}_i^\intercal \mathbf{K}_i^{-1})^\intercal \tag{2.40}$$

where $\vec{\mu}_{\delta c_i}$ and $\vec{\mu}_{\delta c_i^*}$ are the prior means of the non-ergodic coefficients for the existing and new scenarios.

Prediction of non-ergodic ground motion

An alternative approach to predict the median non-ergodic ground motion for the new scenarios, $\vec{f}_{nerg}^*$, is to directly obtain it from the ground-motion observations of the exciting scenarios, $\vec{y}$ (Landwehr et al., 2016). The main difference between this approach and the previous approach is that $\vec{y}$ includes an aleatory component which must be considered in the predictions. The joint prior distribution between ground-motion observations of the existing scenarios and median non-ergodic ground motion of the new scenarios is:

$$\begin{bmatrix} \vec{y} \\ \vec{f}_{nerg}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \vec{\mu}_f \\ \vec{\mu}_f^* \end{bmatrix}, \begin{bmatrix} \mathbf{K}_f + \phi_0^2 \mathbf{I} + \tau_0^2 \mathbf{1} & \mathbf{k}_f \\ \mathbf{k}_f^\intercal & \mathbf{K}_f^* \end{bmatrix} \right) \tag{2.41}$$

where μ_f is the prior mean of the ground-motion of the existing scenarios, and μ_f^* is the prior mean of the ground-motion of the new scenarios. The $\mathbf{K}_f$, $\mathbf{K}_f^*$, and $\mathbf{k}_f$ are the prior covariance for the epistemic uncertainty of the ground-motion between all pairs of existing, new, and existing/new scenarios, respectively, and $\phi_0^2 \mathbf{I}$ and $\tau_0^2 \mathbf{1}$ are the covariance for the within-event and between-event aleatory variability.

The $\vec{\mu}_f$ and $\vec{\mu}_f^*$ depend on how the non-ergodic GMM is being developed. If it is based on a backbone ergodic model, $\vec{\mu}_f$ and $\vec{\mu}_f^*$ are equal to $\vec{f}_{erg}$ for the existing and new scenarios. That is, without any knowledge of the non-ergodic effects, both the mean of the observations and non-ergodic ground motion are equal to the means the ergodic ground motion. However, if the non-ergodic GMM is developed from the beginning, $\vec{\mu}_f$ and $\vec{\mu}_f^*$ are equal to zero.

The prior covariance for the epistemic uncertainty of the ground motion can be obtained by combining the kernel functions of the non-ergodic coefficients as shown in Section 2.4.1.2, $\mathbf{K}_{f\,kl} = \Sigma_{i=1}^d x_{i\,k}\, \boldsymbol{\kappa}_i(t_{i\,k}, t_{i\,l})\, x_{i\,l}$. Similarly, $\mathbf{K}^*_{f\,kl} = \Sigma_{i=1}^d x^*_{i\,k}\, \boldsymbol{\kappa}_i(t^*_{i\,k}, t^*_{i\,l})\, x^*_{i\,l}$, and $\mathbf{k}_{f\,kl} = \Sigma_{i=1}^d x_{i\,k}\, \boldsymbol{\kappa}_i(\vec{x}_k, \vec{x}^*_l)\, x^*_{i\,l}$. The prior covariance for $\vec{y}$ includes $\phi_0^2 \mathbf{I}$ and $\tau_0^2 \mathbf{1}$ because the deviation of ground-motion observations from $\vec{\mu}_y$ is the result of both aleatory variability and epistemic uncertainty. There is not aleatory variability in covariance between the ground-motion observations of the existing scenarios and the mean ground-motion of the new scenarios as any correlation between the two comes from the systematic non-ergodic terms. Similarly, the covariance of $\vec{f}^*_{nerg}$ does not include an aleatory component, as it corresponds to the median prediction of the non-ergodic ground motion.

Once the joint prior distribution is defined, the median non-ergodic ground motion can be predicted by expressing it as a conditional distribution on the ground-motion observations:

$$f^*_{nerg}|y \sim \mathcal{N}(\mu_{f^*_{nerg}|y}, {}_{f^*_{nerg}|y}) \tag{2.42}$$

with the mean and the covariance of the conditional distribution given in Equations 2.43 and 2.44.

$$\mu_{f^*_{nerg}|y} = \mu^*_f + \mathbf{k}_f^\intercal(\mathbf{K}_f + \phi_0^2\mathbf{I} + \tau_0^2\mathbf{1})^{-1}(y - \mu_f) \tag{2.43}$$

$$_{f^*_{nerg}|y} = \mathbf{K}^*{}_f - \mathbf{k}_f^\intercal(\mathbf{K}_f + \phi_0^2\mathbf{I} + \tau_0^2\mathbf{1})^{-1}\mathbf{k}_f \tag{2.44}$$

2.4.2 Model Extrapolation Constraints and Epistemic Uncertainty

In developing any type of GMM — ergodic or non-ergodic — attention must be paid to its proper extrapolation. That is because GMMs are typically derived on datasets primarily composed of small-to-moderate earthquakes at medium-to-large distances, but in PSHA, are applied to large earthquakes at short distances. For that, the trends in the dataset are insufficient to guide the extrapolation of a GMM and additional constraints need to be introduced.

These constraints can be imposed both on the model parameters as well as the model hyperparameters. Two common constraints for the model parameters are related to the magnitude saturation at short distances and anelastic attenuation.

Full magnitude saturation at short distances means that, close to the fault, the ground motion does not scale with magnitude. Similarly, over saturation with magnitude means that, close to the fault, the ground motions reduce as the magnitude increases. This is a controversial issue because empirical datasets, such as NGAWest2, show trends of oversaturation for large magnitudes at short periods and small distances, but the results of numerical simulations support positive magnitude scaling (Abrahamson and Silva, 2007; Collins et al., 2006). Due to the limited number of empirical data from large events, and practical design purposes most GMMs do not allow oversaturation and impose full saturation as a lower limit on the regressions. One such practical consideration is that, if oversaturation is allowed, a structure would need to be designed not only for the largest magnitude but for the smaller events too as they could lead to higher seismic demands. This is straightforward to model in PSHA, but it becomes more complicated when selecting conservative deterministic scenarios.

In a GMM, the magnitude saturation of short periods at zero distance from the rupture is controlled by the combination of the linear magnitude scaling coefficient, the geometrical spreading coefficient, the magnitude scaling coefficient for the geometrical spreading, and the pseudo-depth coefficient in geometrical spreading. In the example GMM provided in Equation 2.1, the coefficients control magnitude saturation are: c_2, c_5 and c_6. Full magnitude saturation at zero distance is achieved by:

$$c_5 = \frac{-c_2}{\ln(c_6)} \tag{2.45}$$

With this functional form, it is easy to derive a non-ergodic GMM with a full saturation constraint, as the c_2, c_5, and c_6 coefficients are treated as fixed terms. However, it may more difficult to impose this constraint with other common functional forms. For example, Chiou and Youngs (2014) (CY14) uses a different functional form, and full saturation is achieved by:

$$c_2 = -c_4\, c_6 \tag{2.46}$$

where c_2 is the linear magnitude scaling, c_4 is the near-source geometrical spreading, and c_6 controls the magnitude dependence of the geometrical spreading. In this functional form, it is harder to include a spatially varying non-ergodic geometrical spreading as the value of c_4 would also affect the magnitude saturation. A non-ergodic GMM developer should consider factors like this when deciding on the functional form and statistical software to use.

The anelastic attenuation is intended to capture the reduction of the amplitude of the seismic waves due to the dissipation of energy as they travel through the earth's crust; thus, the anelastic attenuation coefficient or cell-specific anelastic attenuation must be negative to make physical sense. However, it should be noted that due to the correlation between the linear distance term and the geometrical spreading term, the physical interpretation of the linear distance term as anelastic attenuation depends on using a realistic geometrical spreading term. Similarly to the magnitude saturation, the GMM developer should either use statistical methods and software that allows them to impose an appropriate constraint on these terms, or if that is not feasible to ensure that the model has reasonable distance scaling when used in forward calculations.

Constraints can also be applied to hyperparameters to impose a desired model behavior. For instance, if a VCM GMM contains both a spatially varying site constant and a spatially varying V_{S30} coefficient as a function of the site coordinates, it may be deemed reasonable to constrain the correlation length of the site constant to be smaller than the correlation length of the V_{S30} coefficient. That is because, the repeatable effects related to the site amplification due to the underlying geologic structure, which the V_{S30} intends to model, are broader than the repeatable effects related to the site-specific site amplification. Additionally, such a constraint will limit any trade-offs between the two coefficients as they would capture systematic site effects at different length scales.

The epistemic uncertainty of a non-ergodic GMM quantifies the confidence in estimating the systematic source, path, and site effects; however, it does not quantify the confidence in the model extrapolation. The latter is typically expressed by the model-to-model epistemic uncertainty, which reflects the range of scientifically defensible approaches for developing a GMM. In PSHA, this uncertainty is typically captured either by using multiple GMMs or by shifting the median estimate of a base GMM. As an example of the second approach, Abrahamson et al. (2019) incorporated the model-to-model epistemic uncertainty into a non-ergodic PSHA study for California by estimating the epistemic uncertainty and correlation of the coefficients of a common GMM functional based on the NGAWest2 GMMs and propagating them into the ground-motion prediction.

Another source of model-to-model epistemic uncertainty for non-ergodic GMM is related to the different statistical approaches and decisions in modeling the non-ergodic terms. For example, different covariance functions (e.g. exponential, squared exponential) can be used to model the spatially varying non-ergodic terms or even entirely different modeling approaches (e.g. GP regression, ANNs). Such choices are expected to lead to bigger differences in areas with sparse data.

Different intensity measures are affected differently by magnitude scaling. PSA, especially at short periods, is sensitive to the entire frequency content of the ground motion (i.e. spectral shape). This can be an issue when developing a GMM predominately with small earthquakes as their frequency content is different from the frequency content of large events which are more common in PSHA, potentially resulting in incorrect scaling coefficients. A solution to this is developing a GMM for an intermediary intensity parameter (IP) that is not sensitive to spectral shape and using a transformation to convert the prediction to PSA. One such example is Lavrentiadis and Abrahamson (2022) where used EAS was used as an intermediary IP and Random vibration theory was used to convert EAS to PSA.

2.4.3 Other methods for non-ergodic models

The previous sections provided an in-depth discussion on developing non-ergodic GMMs using Gaussian Process. Although it has many useful properties, it is not the only method for developing non-ergodic GMMs. This section provides a brief review of other methods that have been used for this task.

Sung and Lee (2019) built more than 700 single-station GMMs for the Taiwan region. Single-station GMMs do not include non-ergodic site terms, instead, they are independently regressed with ground motions recorded at a single station. Kriging interpolation is used to estimate the spatial distributions of the single-station GMM coefficients and aleatory terms at new locations. This is a simpler approach for developing a partially non-ergodic GMM, but it cannot provide estimates of the epistemic uncertainty at the new locations as VCM GP GMM does.

Caramenti et al. (2020) used a multi-source geographically-weighed regression (MS-GWR) to develop a non-ergodic GMM for Italy. It is similar to GP in that the spatial correlation of the non-ergodic terms is also captured through kernel functions; however, it is more efficient as it is based on the least-squares regression. The main shortcoming of this approach is that the aleatory variability is described by a single term so it is unable to capture the correlation between the recordings of the same earthquake.

Okazaki et al. (2021) developed a single-station GMM for PGA using an ANN trained on strong-motion data from the KiK-net seismograph network in Japan. In this study, the systematic site effects were expressed as a function of site ID and estimated through the ANN fitting.

3 FORMULATIONS OF NON-ERGODIC GROUND MOTION MODELS USING GAUSSIAN PROCESS

In this report, non-ergodic GMMs are developed based on backbone models. In this approach, the mean scaling is based on the ergodic backbone model while the non-ergodic effects are determined from the total residuals (ϵ_{tot}) of the ergodic model without including any terms that are treated as non-ergodic in the NGMM. Examples of calculating ϵ_{tot} for different types of NGMMs are presented in the next section. The benefits of this approach are that: (i) it can be applied to regions with limited data which may not be enough to derive the ergodic scaling, (ii) the ergodic scaling can include non-linear term which cannot be estimated with a GP regression, and (iii) the non-ergodic GMM can benefit from any seismological constraints built in the ergodic backbone model. The regression tools presented in Chapters 6 and 7 are based on such an approach.

It is beyond the scope of this report; however, a GP regression can easily be extended to model linear "fixed effects" (ergodic coefficients) of the base model if there is enough local data. An introduction for deriving an ergodic GMM in a Bayesian framework can be found in Kuehn (2021). This approach may be preferred if there is not a base model available for the region of interest.

Section 3.1 presents details of three different NGMM functional forms that can be fitted with the provided tools and Section 3.2 presents the available options for modeling of the non-ergodic coefficients.

3.1 NON-ERGODIC GROUND MOTION MODEL FUNCTIONAL FORM

A modeler may use the provided tools to develop one of three types of NGMM. Type1 includes an intercept and three spatially varying non-ergodic constants. Type2, in addition to the previous terms, includes the cell-specific anelastic attenuation. Finally, type3 also includes a spatially varying geometrical spreading and spatially varying V_{S30} scaling.

The functional form for the type1 NGMM is:

$$y_{es} = f_{erg}(M, R_{rup}, V_{S30}, ...) + \delta c_0 + \delta c_{1,E}(\vec{t}_E) + \delta c_{1a,S}(\vec{t}_S) + \delta c_{1b,S}(\vec{t}_S) + \delta WS^0_{es} + \delta B^0_e \quad (3.1)$$

where y_{es} is the log of the ground-motion intensity measure from e^{th} earthquake and s^{th} station, δc_0 the constant offset to account for the re-weighting of the residuals. $\delta c_{1,E}(\vec{t}_E)$ is the spatially varying earthquake adjustment, which varies as a function of the earthquake coordinates t_E and is intended to capture the repeatable non-ergodic effects related to the source location. $\delta c_{1b,S}(\vec{t}_S)$ is the spatially varying site adjustment, which varies as a function of the site coordinates t_S and is intended to capture the repeatable regional non-ergodic site effects. $\delta c_{1a,S}(\vec{t}_S)$ is the spatially independent site adjustment, which is independent from site to site and is intended to capture the repeatable site-specific non-ergodic effects on top of $\delta c_{1b,S}(\vec{t}_S)$. Lastly, δSW^0_{es} is the non-ergodic within-event within-site residual, and δB^0_e is the non-ergodic between-event residual. The total residuals used in type1 NGMM regression correspond to the difference between the log of the ground-motion intensity measure and the median ground motion of the ergodic backbone model in log space (Equation 3.2).

$$\epsilon_{tot} = y - f_{erg}(M, R_{rup}, V_{S30}, ...) \tag{3.2}$$

The functional form for the type2 NGMM is:

$$\begin{aligned} y_{es} =& (f_{erg}(M, R_{rup}, V_{S30}, ...) - c_{a,erg}\, R_{rup}) + \delta c_0 + \delta c_{1,E}(\vec{t}_E) + \delta c_{1a,S}(\vec{t}_S) + \delta c_{1b,S}(\vec{t}_S) \\ &+ \vec{c}_{ca,P} \cdot \Delta\vec{R} + \delta WS^0_{es} + \delta B^0_e \end{aligned} \tag{3.3}$$

where $\vec{c}_{ca,P}$ is a vector with all the cell-specific anelastic attenuation coefficients, $\vec{R}$ is a vector with all cell-path segment lengths between earthquake e and stations s, and $c_{a,erg}$ is the ergodic anelastic attenuation coefficient. $c_{a,erg}\, R_{rup}$ is subtracted from the ergodic prediction as its effects are captured by the non-ergodic cell-specific anelastic attenuation. In this case, the total residuals for the regression are equal to the difference between y and the median ground motion of the ergodic backbone model in log space without the effects of the ergodic anelastic attenuation (Equation 3.4).

$$\epsilon_{tot} = y - (f_{erg}(M, R_{rup}, V_{S30}, ...) - c_{a,erg}\, R_{rup}) \tag{3.4}$$

Finally, the functional form for the type3 NGMM is:

$$\begin{aligned} y_{es} =& (f_{erg}(M, R_{rup}, V_{S30}, ...) - (c_{a,erg}\, R_{rup} + c_{2,erg}\, f_{gs}(R_{rup}, M) + c_{3,erg}\, f_{V_{S30}}(V_{S30}))) \\ &+ \delta c_0 + \delta c_{1,E}(\vec{t}_E) + \delta c_{1a,S}(\vec{t}_S) + \delta c_{1b,S}(\vec{t}_S) \\ &+ c_{2,P}(\vec{t}_E)\, f_{gs}(R_{rup}, M) + c_{3,S}(\vec{t}_S)\, f_{V_{S30}}(V_{S30}) \\ &+ \vec{c}_{ca,P} \cdot \Delta\vec{R} + \delta WS^0_{es} + \delta B^0_e \end{aligned} \tag{3.5}$$

In addition to the previous non-ergodic terms, the type3 NGMM includes a spatially varying geometrical spreading coefficient ($c_{2,P}(\vec{t}_E)$) that is a function of the earthquake coordinates, and a spatially varying V_{S30} scaling coefficient ($c_{3,S}(\vec{t}_S)$) that is a function of the site coordinates. $f_{gs}(R_{rup}, M)$ corresponds to the geometrical spreading scaling term and $f_{V_{S30}}(V_{S30})$ corresponds to the V_{S30} scaling term. The ergodic geometrical spreading and V_{S30} scaling are subtracted from the ergodic prediction as they are treated as spatially varying in the type3 NGMM. The total regression residuals are equal to the difference between y and the median ground motion of the ergodic backbone model without the effects of the ergodic geometrical spreading, V_{S30} scaling, and anelastic attenuation (Equation 3.6).

$$\begin{aligned} \epsilon_{tot} = y-& (f_{erg}(M, R_{rup}, V_{S30}, ...) \\ &-(c_{a,erg}\, R_{rup} + c_{2,erg}\, f_{gs}(R_{rup}, M) + c_{3,erg}\, f_{V_{S30}}(V_{S30}))) \end{aligned} \tag{3.6}$$

3.2 MODELING NON-ERGODIC EFFECTS

A Gaussian process is comprised of two components: the mean function and the kernel function. The choice of the mean function controls the behavior of the non-ergodic coefficient away from data while the kernel function affects the spatial correlation of a non-ergodic coefficient. The remaining section covers the modeling options for the non-ergodic terms as Gaussian Processes; details of the prior distributions of the hyper-parameters are covered in Chapters 6 and 7 as their modeling depends on the statistical package.

3.2.1 Spatially varying earthquake adjustment

The spatially varying earthquake adjustment can be modeled as a Gaussian process (Equation (3.7)) with a zero mean and a negative exponential (Equation (3.8)) or a Matérn (Equation (3.9)) kernel function.

$$\delta\vec{c}_{1,E} \sim \mathcal{GP}\left(\vec{0}, \kappa_{1,E}(\vec{t}_E, \vec{t}'_E)\right) \tag{3.7}$$

$$\boldsymbol{\kappa}_{1,E}(\vec{t}_E, \vec{t}'_E) = \omega_{1,E}^2 \ \exp\left(-\frac{||\vec{t}_E - \vec{t}'_E||}{\ell_{1,E}}\right) \tag{3.8}$$

$$\boldsymbol{\kappa}_{1,E}(\vec{t}_E, \vec{t}'_E) = \frac{\omega_{1,E}^2}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{\ell_{1,E}}||\vec{t}_E - \vec{t}'_E||\right)^{\nu} K\left(\frac{\sqrt{2\nu}}{\ell_{1,E}}||\vec{t}_E - \vec{t}'_E||\right) \tag{3.9}$$

$\delta\vec{c}_{1,E}$ is a vector with the source adjustments at all event locations. The correlation between the source effects of two earthquakes located at $\vec{t}_E$ and $\vec{t}'_E$ is given by $\kappa_{1,E}(\vec{t}_E, \vec{t}'_E)$. The negative exponential and Matérn kernel functions are similar in modeling continuously spatially varying fields but different in the smoothness of these fields. Currently, the choice of the kernel function mainly depends on the ease of implementation on the different statistical packages, negative exponential is easier to implement in STAN and Matérn is easier to use in INLA; however, there are no limitations on using different kernel functions on either software. In both kernel functions, the correlation length ($\ell_{1,E}$) controls the length scale of the spatial variation of the non-ergodic effects, and the scale ($\omega_{1,E}$) controls the size of the non-ergodic effects. Additionally, for the Matérn kernel function, ν controls the smoothness of the Gaussian field and is typically set to $\nu = 1$. K is the modified Bessel function of the second kind. More information on the Matérn kernel is provided in Section 7.1.4. With both kernel functions, away from data, the mean of $\delta c_{1,E}$ reverts to zero due to the zero mean function, and the epistemic uncertainty is equal to $\omega_{1,E}$.

3.2.2 Spatially independent site term

The spatially independent site term is modeled with a zero mean and a spatially independent kernel function:

$$\delta\vec{c}_{1a,S} \sim \mathcal{GP}\left(\vec{0}, \boldsymbol{\kappa}_{1a,S}(\vec{t}_S, \vec{t}'_S)\right) \tag{3.10}$$

$$\boldsymbol{\kappa}_{1a,S}(\vec{t}_S, \vec{t}'_S) = \omega_{1a,S}^2 \, \delta\left(||\vec{t}_S - \vec{t}'_S||\right) \tag{3.11}$$

$\delta\vec{c}_{1a,S}$ is a vector with the site-specific site adjustments at all site locations. At sites with stations that have recorded past earthquakes, the mean and epistemic uncertainty of $\delta c_{1a,S}$ is determined by the previous ground motions, whereas at sites with no ground motions, the mean of $\delta c_{1a,S}$ is zero and the epistemic uncertainty is $\omega_{1a,S}$.

3.2.3 Spatially varying site adjustment

The spatially varying site adjustment captures non-ergodic site effects that vary on a regional scale. It is modeled similarly to $\delta\vec{c}_{1,E}$ with a zero mean and a negative exponential or Matérn kernel function:

$$\delta\vec{c}_{1b,S} \sim \mathcal{GP}\left(\vec{0}, \boldsymbol{\kappa}_{1b,S}(\vec{t}_S, \vec{t}'_S)\right) \tag{3.12}$$

$$\boldsymbol{\kappa}_{1b,S}(\vec{t}_S, \vec{t}'_S) = \omega_{1b,S}^2 \ \exp\left(-\frac{||\vec{t}_S - \vec{t}'_S||}{\ell_{1b,S}}\right) \tag{3.13}$$

$$\boldsymbol{\kappa}_{1b,S}(\vec{t}_S, \vec{t}'_S) = \frac{\omega_{1b,S}^2}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{\ell_{1b,S}}||\vec{t}_S - \vec{t}'_S||\right)^{\nu} K\left(\frac{\sqrt{2\nu}}{\ell_{1b,S}}||\vec{t}_S - \vec{t}'_S||\right) \tag{3.14}$$

where $\ell_{1b,S}$ is the correlation length and $\omega_{1b,S}$ is the scale of $\delta\vec{c}_{1b,S}$.

3.2.4 Geometric spreading and V_{S30} scaling

The non-ergodic geometrical spreading (Equations 3.15 to 3.17) and V_{S30} scaling (Equations 3.18 to 3.20) coefficients are also modeled as Gaussian Processes with a negative exponential or Matérn kernel function but their difference to $\delta\vec{c}_{1,E}$ and $\delta\vec{c}_{1b,S}$ is that they are assigned a non-zero prior distribution for the mean function. The prior mean is typically centered around the ergodic values of coefficients but is given some range to accommodate the re-weighting of the data points.

$$\vec{c}_{2,P} \sim \mathcal{GP}\left(\vec{\mu}_{2,P}, \kappa_{2,P}(\vec{t}_E, \vec{t}_E)\right) \tag{3.15}$$

$$\boldsymbol{\kappa}_{2,P}(\vec{t}_E, \vec{t}'_E) = \omega_{2,P}^2 \ \exp\left(-\frac{||\vec{t}_E - \vec{t}'_E||}{\ell_{2,P}}\right) \tag{3.16}$$

$$\boldsymbol{\kappa}_{2,P}(\vec{t}_E, \vec{t}'_E) = \frac{\omega_{2,P}^2}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{\ell_{2,P}}||\vec{t}_E - \vec{t}'_E||\right)^{\nu} K\left(\frac{\sqrt{2\nu}}{\ell_{2,P}}||\vec{t}_E - \vec{t}'_E||\right) \tag{3.17}$$

$$\vec{c}_{3,S} \sim \mathcal{GP}\left(\vec{\mu}_{3,S}, \kappa_{3,S}(\vec{t}_S, \vec{t}_S)\right) \tag{3.18}$$

$$\boldsymbol{\kappa}_{3,S}(t_S, t'_S) = \omega_{3,S}^2 \ \exp\left(-\frac{||\vec{t}_S - \vec{t}'_S||}{\ell_{3,S}}\right) \tag{3.19}$$

$$\boldsymbol{\kappa}_{3,S}(t_S, t'_S) = \frac{\omega_{3,S}^2}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{\ell_{3,S}}||\vec{t}_S - \vec{t}'_S||\right)^{\nu} K\left(\frac{\sqrt{2\nu}}{\ell_{3,S}}||\vec{t}_S - \vec{t}'_S||\right) \tag{3.20}$$

$\vec{c}_{2,P}$ is a vector with the geometrical spreading coefficient at all event locations. $\vec{c}_{3,S}$ is a vector with the V_{S30} scaling coefficient at all site locations. $\vec{\mu}_{2,P}$ and $\vec{\mu}_{3,S}$ are the mean functions for $\vec{c}_{2,P}$ and $\vec{c}_{3,S}$, respectively. $\ell_{2,P}$ is the correlation length and $\omega_{2,P}$ is the scale of $\delta\vec{c}_{2,P}$. $\ell_{3,S}$ is the correlation length and $\omega_{3,S}$ is the scale of $\delta\vec{c}_{3,S}$.

3.2.5 Anelastic attenuation

In the provided software, the anelastic attenuation coefficients can be modeled either as independent or as partially spatially correlated Gaussian Processes:

$$\vec{c}_{ca,P} \sim \mathcal{GP}\left(\vec{\mu}_{ca,P}, \boldsymbol{\kappa}_{ca,P}(\vec{t}_C, \vec{t}'_C)\right) \tag{3.21}$$

where $\vec{t}_C$ is the midpoint coordinates of all anelastic attenuation cells. In both cases, the mean function ($\vec{\mu}_{ca,P}$) is centered around the ergodic coefficient for anelastic attenuation. A spatially independent covariance function (Equation 3.22) is used for the independent anelastic attenuation cells, where $\omega_{ca,P}$ controls the size of the variability. A mixture of a negative exponential and spatially independent kernel function (Equation 3.23) is used for the partially spatially-correlated case. $\ell_{ca1,P}$ and $\omega_{ca1,P}$ control the length scale and size of the underlying spatially correlated variability, respectively. $\omega_{ca2,P}$ dictates the size of the spatially independent variability

$$\boldsymbol{\kappa}_{ca,P}(\vec{t}_C, \vec{t}'_C) = \omega^2_{ca,P}\, \delta\left(||\vec{t}_C - \vec{t}'_C||\right) \tag{3.22}$$

$$\boldsymbol{\kappa}_{ca,P}(\vec{t}_C, \vec{t}'_C) = \omega^2_{ca1,P}\, \exp\left(-\frac{||\vec{t}_C - \vec{t}'_C||}{\ell_{ca1,P}}\right) + \omega^2_{ca2,P}\, \delta\left(||\vec{t}_C - \vec{t}'_C||\right) \tag{3.23}$$

4 DATA PREPARATION FOR DEVELOPMENT OF NON-ERGODIC GMMS

This chapter provides the structure of input files needed to develop non-ergodic GMMs. The regression tools for the development of non-ergodic ground-motion models read a standardized format of input files. These files include the ground-motion flatfile (Section 4.1) and, for Type-2 and Type-3 NGMM regressions, also the cell-info and cell-path length faltfiles (Section 4.2). All files are saved in a comma-separated values format (.csv) with the first row corresponding to the column names. Examples of these files can be found in the synthetic dataset directory `ngmm_tools/Data/Verification`.

4.1 GROUND-MOTION FLATFILE

The Ground-motion flatfile contains the source and site information, as well as the total ergodic residuals. The mandatory columns in the ground-motion flatfile are listed below.

- Record Sequence Number (rsn) which is a unique number of every ground-motion record in the flatfile.
- Earthquake ID (eqid) which is a unique number of every event in the flatfile. All ground motions from the same earthquake have the same eqid.
- Station Sequence Number (ssn) which is a unique number of every station in the flatfile. All ground-motions records at the same station have the same ssn.
- Earthquake location (eqX and eqY) in UTM coordinates.
- Elevation for earthquake top of rupture (eqZ).
- Station location (staX and staY) in UTM coordinates.
- Total residual without including the effect of the scaling terms that are treated as non-ergodic (tot). The definition of the total residuals for the different types of NGMM tools is described in Section 3.1.

Additionally, for a type-3 GMM, the Ground-motion flatfile needs to include:

- The scaling term for geometrical spreading ($\mathrm{x\_2}$) which is determined by evaluating the geometrical spreading functional form for all ground motions ($x_2 = f_{gs}\left(R_{rup}, M\right)$)

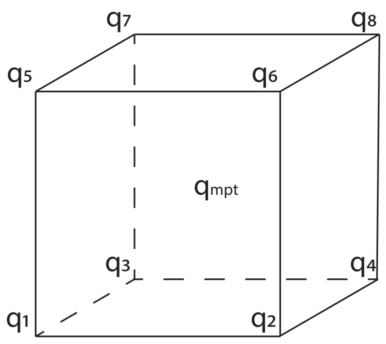


Figure 4.1. Sketch of numbering convention for vertices of attenuation cells.

- The scaling term for site amplification (i.e. V_{S30} scaling, x_3) which is determined by evaluating V_{S30} scaling functional form for all ground motions ($x_3 = f_{V_{S30}}(V_{S30})$)

4.2 CELL FLATFILES

The cell-info flatfile defines the ids and coordinates of the attenuation cells. The cell-path length flatfile contains the cell path segments of all ground motions in the ground-motion flatfile over all the attenuation cells in the cell-info flatfile.

The columns in the cell-info flatfile are:

- Cell ID (cellid) which is a unique number for every anelastic attenuation cell.
- Cell name (cellname) which is a unique character string for every cell.
- Cell vertices locations (q1X, q1Y and q1Z to q8X, q8Y and q8Z) in UTM coordinates. The numbering convention of the cell vertices is shown in Figure 4.1.
- Cell mid-point location (mptX, mptY, and mptZ) in UTM coordinates.

Similarly, the columns in the cell-path length flatfile are:

- rsn, eqid, and ssn of the corresponding ground motion in the ground-motion flatfile.
- Cell names of the cell info flatfile with the corresponding cell-path segments for every ground motion.

Both input the cell-info and cell-path flatfiles can be generated with the `compute_celldistance_matrix.ipynb` Jupyter notebook by specifying the Ground-motion flatfile and extents of the domain. `dir_flatfile` and `name_flatfile` define the directory and name of the ground-motion flatfile, while `grid_lims_x`, `grid_lims_y`, `grid_lims_z` define the longitudinal, latitudinal, and vertical extents of the domain.

As an example, a ground-motion flatfile with n_{gm} records will have $n_{gm} + 1$ rows, one row for each ground motion plus one row for the header. Similarly, a cell info flatfile with n_c cells will have $n_c + 1$ rows, one row for each cell and one row for the header. While the accompanying cell-path flatfile will have $n_{gm} + 1$ rows, a row for the header and every ground motion, and $n_c + 3$ columns, a column for rsn, eqid, ssn and a column for every cell.

5 SYNTHETIC DATASETS FOR VERIFICATION OF COMPUTER TOOLS

The verification of non-ergodic ground motion tools is performed through the use of synthetic datasets. In this approach synthetic datasets with known non-ergodic effects are generated, and the developed tools are evaluated in their accuracy to retrieve the "true" non-ergodic effects solely based on the total residuals and the source, path, and site information. No direct information about the non-ergodic effects is provided to the regression tools.

In order for the verification exercise to be applicable to actual ground-motion datasets, the synthetic datasets need to have a realistic event and station spatial coverage. Herein, the realism was ensured by basing the synthetic datasets on the metadata of the ground-motion records comprising NGAWest2 (Ancheta et al., 2014) and the ground-motion records that are expected to comprise the next phase of NGA program, NGAWest3, hereafter denoted as NGAWest3*. Additionally, by performing the verification exercise on NGAWest3*, the performance of the developed tools is evaluated for the size of datasets that are expected to be available in the coming years.

Section 5.1 presents the NGAWest2 CA and NGAWest3* CA metadata, and Section 5.2 describes the methodology for generating the synthetic datasets. The files accompanying this chapter for generating the synthetic datasets can be found at `ngmm_tools/Analyses/Code_Verification/synthetic_datasets` on the Github directory. The STAN files (`.stan`) perform the sampling of the non-ergodic coefficients. The Python files (`.py`) are used as wrapper scripts to (i) read the metadata for the synthetic datasets, (ii) call the STAN files for sampling, and (iii) save the generated datasets.

5.1 METADATA

5.1.1 Metadata of NGAWest2 CA

From NGAWest2, the ground motions that were used for the generation of the synthetic dataset correspond to the California subset, hereafter denoted NGAWest2 CA. The chosen data set is based on the selection criteria outlined in Abrahamson et al. (2014) GMM. The selected events and stations are located in California, western Nevada, and part of northern Mexico (Figure 5.1). The station density is higher in the Los Angeles, Bay Area, and San Diego metropolitan areas

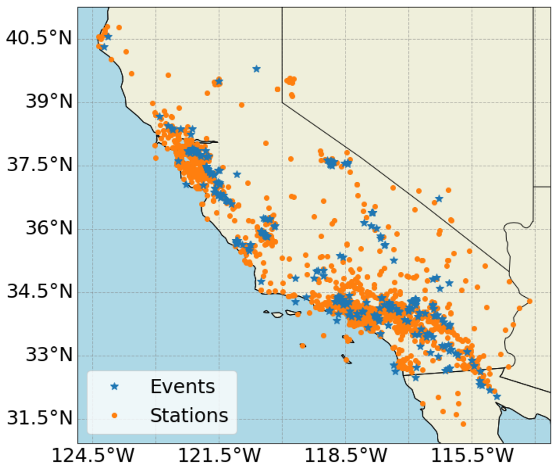


Figure 5.1. Spatial distribution for earthquakes and stations in California subset of NGAWest2 CA.

and sparser in remote areas such as northern-eastern California. It contains 12009 records from 274 earthquakes recorded at 1479 stations and spans from 1966 to 2011. The magnitude of the earthquakes ranges from 3.0 to 7.3 and the distance of most records ranges from 10 to $200km$ (Figure 5.2a). The minimum magnitude between the years 1966 and 1994 is 4.7, while from 1998 onwards is 3.0 (Figure 5.2b).

5.1.2 Metadata of NGAWest3* CA

The NGAWest3* CA metadata were developed by combining the NGAWest2 CA metadata (Section 5.1.1) with the ground-motion metadata of the post-2011 events that are expected to be part of NGAWest3 for California.

The raw catalog for the anticipated records was generated by combining the 2011-2021 IRIS event catalog for California with the IRIS station catalog for the networks: AZ, BK, CI, NC, NN, NP, SB, and, US. The minimum magnitude of the event catalog is 2.0 and the longitude/latitude limits are provided in Table 5.1. Since the time histories were not available during this phase of the project, it was not feasible to accept or reject the records based on their signal-to-noise ratio. Instead, the selection was based on the maximum distance-magnitude threshold in NGAWest2. A linear and quadratic distance limits were visually fitted to the furthest records of the entire NGAWest2 for the different magnitudes (Figure 5.3). Using this criterion $5,586,369$ data points were selected from $36,676$ earthquakes and 419 stations. Figure 5.4 shows the magnitude-distance and year-magnitude distributions of the accepted raw data points. The V_{S30} values at the station locations were obtained from Thompson et al. (2014); Wald and Allen (2007); Wills et al. (2015); Yong et al. (2012) proxy models. The weighting of the different V_{S30} proxies was adopted from Wang (2020).

To generate a more manageable dataset that is more likely to be representative of a dataset used in the development of future GMMs, the previous raw catalog was downsampled to 1000 events and a maximum distance of $300km$. The reduced dataset contains all events that are larger than

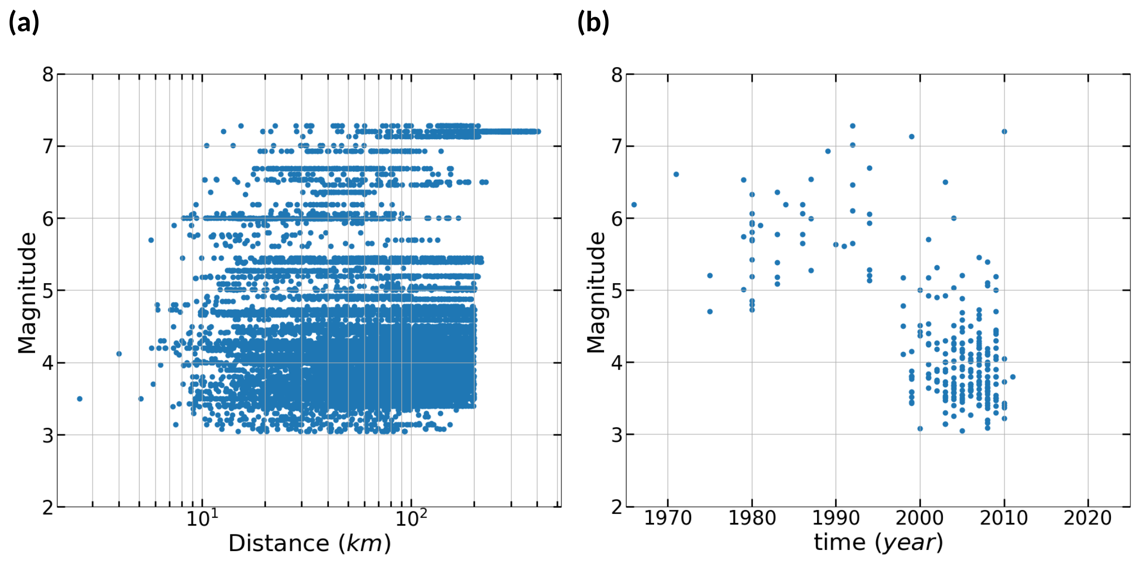


Figure 5.2. California subset of NGAWest2 CA. The left figure shows Magnitude - Distance distribution of the selected subset. The right figure shows the Magnitude - Date distribution of the selected events.

Table 5.1. Search boundaries of IRIS event catalog

Corner	lat. (deg)	lon. (deg)
SE	30.5	-113.5
SW	30.5	-125.0
NW	42.5	-125.0
NE	42.5	-113.5

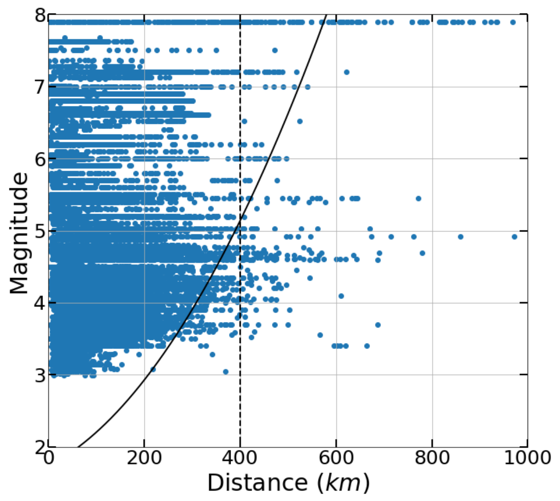
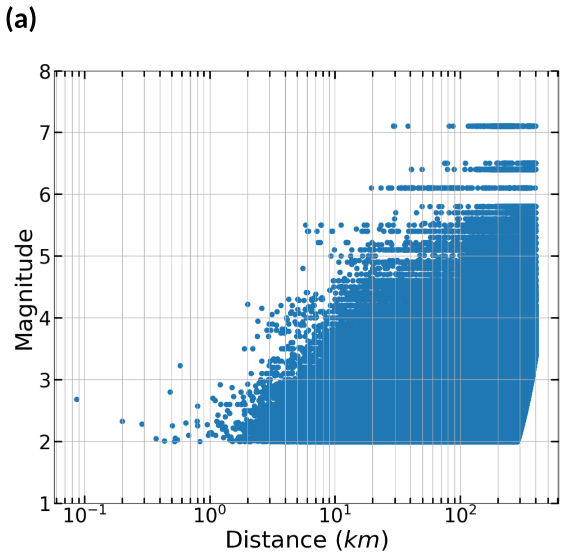


Figure 5.3. Magnitude - distance threshold based on NGAWest2.



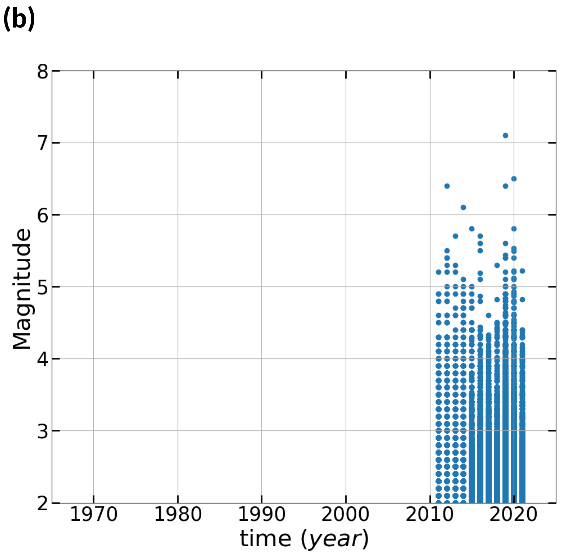


Figure 5.4. Distribution of the raw catalog of anticipated new records. The left figure shows Magnitude - Distance distribution and the right figure show the Magnitude - Date distribution of the raw catalog of anticipated new records.

$M5$, while the remaining events were randomly chosen from the $M3$ to 5 magnitude bin.

Finally, the metadata that comprises NGAWest3* CA was created by combining the NGAWest2 CA metadata with the metadata of the reduced-size new catalog. Any stations between the two catalogs that were less than $10m$ apart were assumed to be collocated and were assigned the same unique station id. Any collocated stations were assigned the average site information (e.g. V_{S30} of station coordinates) of the two datasets in NGAWest3* CA. In total, NGAWest3* CA is comprised of 157,438 ground-motion records from 1274 events recorded at 1822 stations. The earthquake magnitude ranges from 3 to 7.28 and the majority of the ground motions are 10 to 300 km away from the events (Figure 5.5). Figure 5.6 shows the spatial distribution of the events and stations. Compared to NGAWest2, there is a higher event and station coverage in North eastern California and Nevada.

Figure 5.7 shows the cells and the path coverage of the NGAWest3* dataset, and the number of paths per cell. Overall, there is dense path coverage in California, especially in the greater Bay Area and Los Angeles metropolitan areas, and poorer path coverage in Nevada.

5.2 GENERATION OF SYNTHETIC DATASETS

The synthetic datasets were generated based on the source, path, and site metadata presented in the previous section. In particular, the metadata used in the generation of the synthetic datasets includes source coordinates, site coordinates, V_{S30}, and rupture distance. Different sets of synthetic datasets were generated to evaluate the developed tools in terms of the complexity of the derived NGMMs, scalability, and universality.

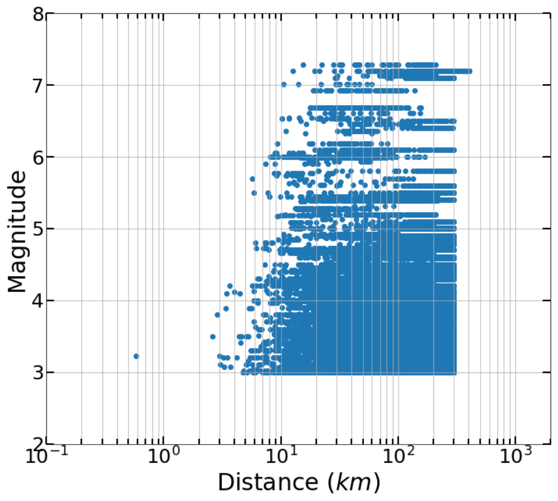


Figure 5.5. Magnitude - distance distribution of ground-motion records in NGAWest3* CA.

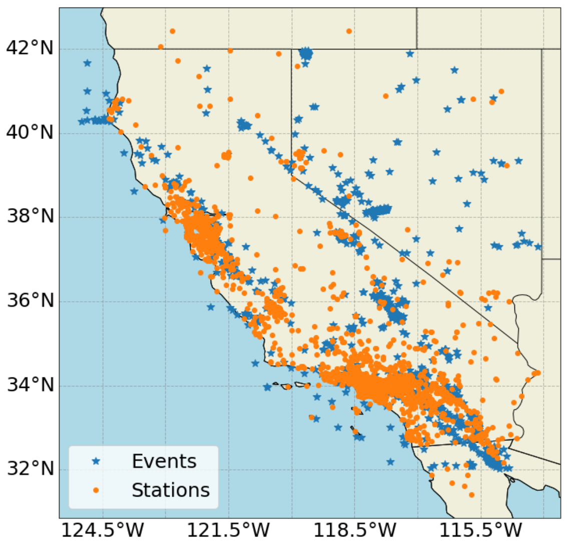


Figure 5.6. Spatial distribution for earthquakes and stations in NGAWest3* CA.

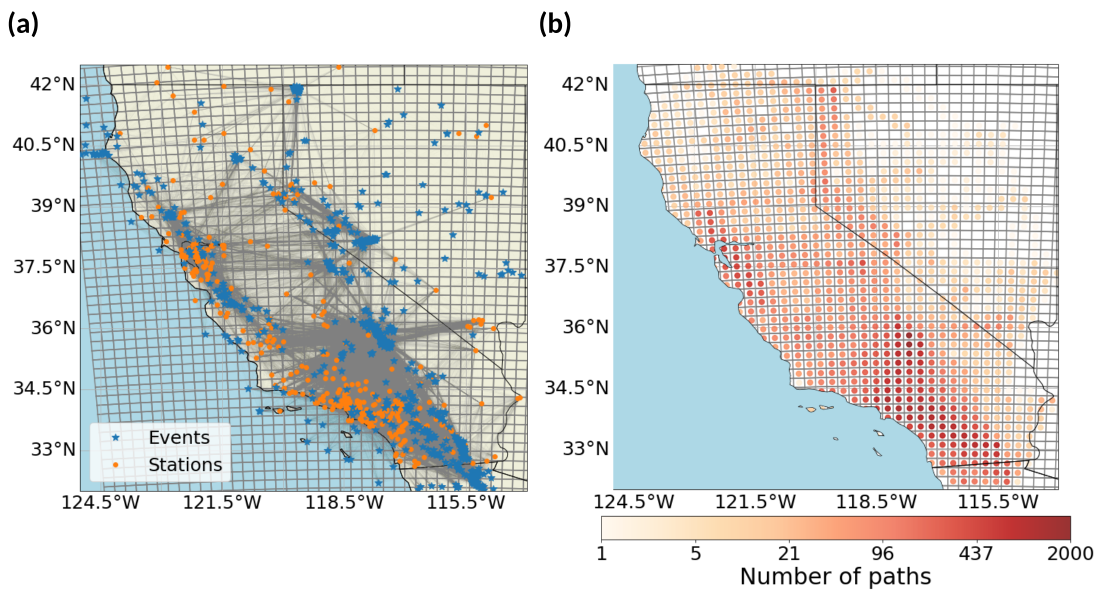


Figure 5.7. The left figure shows the path coverage for the attenuation cells in NGAWest3* CA. The right figure shows the number of paths per cell.

5.2.1 Complexity, scalability, and universality

The performance of the regression tools for different levels of GMM complexity was evaluated by generating a synthetic dataset for all types of non-ergodic ground-motion models described in Section 3.1. These range from non-ergodic GMMs that only capture the source and site repeatable effects to NGMMs that also capture the systematic path effects in terms of both geometrical spreading and anelastic attenuation.

The scalability was tested through the use of different dataset sizes which correspond to NGAWest2 CA North, NGAWest2 CA, and NGAWest3* CA flatfiles (Table 5.2 and Figure 5.8). NGAWest2 CA North is representative of regional datasets, NGAWest2 CA is representative of large-scale datasets used currently in the development of NGMMs, and NGAWest3* CA is representative of the size of datasets that are expected to be available in the coming years.

Table 5.2. Size of Synthetic Datasets for Evaluating Scalability of Developed Tools

Dataset Name	Number of ground-motions	Number of events	Number of stations
NGAWest2 CA, North	4160	150	546
NGAWest2 CA	12009	257	1479
NGAWest3* CA	157438	1274	1822

The universality of the developed tools was evaluated by creating datasets with different ranges of hyper-parameters. In particular, two sets of hyper-parameters were used, one with short and one with large correlation lengths (Table 5.3). The set with large correlation lengths is representative of

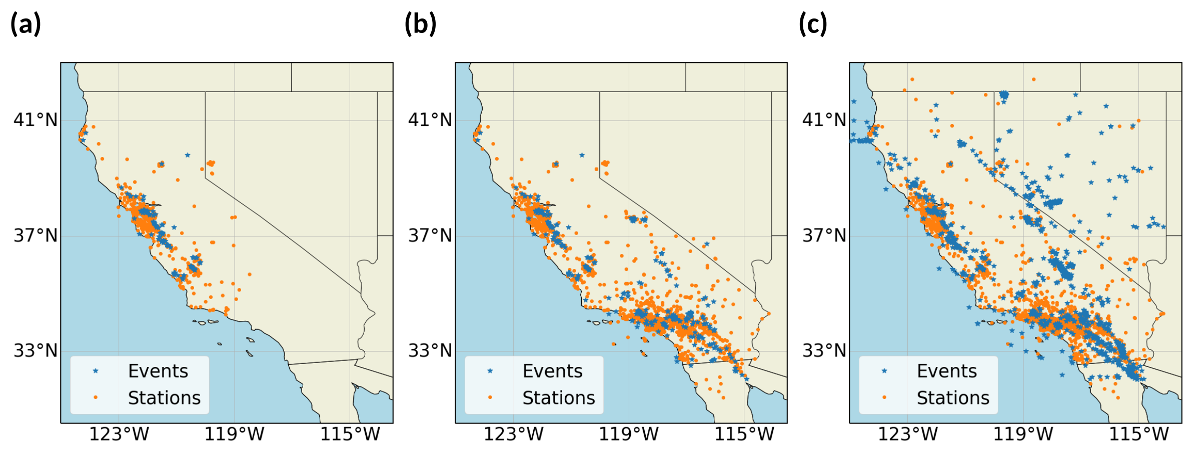


Figure 5.8. Event and Station Spatial Distribution of Synthetic Datasets for Testing Scalability of Developed Tools. (a) NGAWest2 CA, North. (b) NGAWest2 CA, and (c) NGAWest3* CA

regions with uniform geology where the non-ergodic effects are expected to be similar over large distances, while the set with short correlation lengths is representative of more heterogeneous regions where the non-ergodic effects are expected to vary over shorter distances.

5.2.2 Sampling Approach

The synthetic dataset realizations were created by first sampling the non-ergodic coefficients and cell-specific anelastic attenuation coefficients based on their correlation structure and then computing the total regression residuals for the different NGMM types based on the functional forms presented in Chapter 5. The negative exponential kernel function was used for the correlation structure of the spatially varying non-ergodic coefficients, and the partially spatially correlated kernel function was used for the correlation structure of the cell-specific anelastic attenuation coefficients. For each realization of a synthetic dataset, random samples of the coefficients were drawn by multiplying the lower Cholesky decomposition of their covariance matrix with an array of standard normally distributed samples:

$$
\begin{aligned}
\vec{z} &\sim \mathcal{N}(0,1) \\
{}_{\mathrm{i}} &= \mathbf{L}_{\mathrm{i}}\mathbf{L}_{\mathrm{i}}^{\intercal} \\
\vec{c_i} &= \mathbf{L}_{\mathrm{i}}\vec{z} + \vec{\mu_i}
\end{aligned}
\tag{5.1}
$$

where $\vec{c_i}$ is a vector with random samples of the i^{th} coefficient, $\vec{\mu_i}$ is an array with the mean value of the coefficient, $\vec{z}$ is a vector with samples from a standard normal distribution, ${}_{\mathrm{i}}$ is the covariance matrix, and $\mathbf{L}_{\mathrm{i}}$ is the lower triangular Cholesky decomposition of ${}_{\mathrm{i}}$. The covariance matrix of each coefficient is computed based on the kernel function and metadata for that coefficient:

$$
{}_{\mathrm{i}kl} = \kappa_i(\vec{t_k}, \vec{t_l}) \tag{5.2}
$$

where ${}_{\mathrm{i}kl}$ is the k^{th} row and l^{th} column element of the ${}_{\mathrm{i}}$ matrix. In total, for each synthetic dataset, five realizations were created to evaluate the accuracy of the predictions.

Table 5.3. Size of Synthetic Datasets for Evaluating Scalability of Developed Tools

		Hyper-parameter	Synthetic Dataset with Small Corr. Lengths	Synthetic Dataset with Large Corr. Lengths
Ergodic	Coeffs	$c_{2,erg}$	-2.0	-2.0
		$c_{2,erg}$	-0.6	-0.6
		$c_{a,erg}$	-0.011	-0.011
Non-ergodic	Offset	ω_0	0.10	0.10
		ω_2	0.2	0.2
		ω_2	0.10	0.20
		ω_3	0.10	0.20
		ω_{ca}	0.10	0.20
Non-ergodic	Scale	$\omega_{1,E}$	0.10	0.20
		$\omega_{1a,S}$	0.35	0.40
		$\omega_{1b,S}$	0.25	0.30
		$\omega_{2,P}$	0.10	0.20
		$\omega_{3,S}$	0.10	0.20
		$\omega_{ca1,P}$	0.004	0.005
		$\omega_{ca2,P}$	0.002	0.003
Non-ergodic	Scale	$\ell_{1,E}\ (km)$	60	100
		$\ell_{1b,S}\ (km)$	30	70
		$\ell_{2,P}\ (km)$	60	100
		$\ell_{3,S}\ (km)$	60	100
		$\ell_{ca1,P}\ (km)$	75	120
Aleat.	Var.	ϕ_0	0.30	0.30
		τ_0	0.25	0.25

6 DEVELOPMENT OF NON-ERGODIC GROUND MOTION MODELS USING STAN COMPUTER PLATFORM

This chapter provides an overview of using STAN computer platform to develop NGMMs. STAN is an open-source software that performs a full Bayesian statistical inference using Markov Chain Monte Carlo (MCMC) (Stan Development Team, 2022). This approach creates a Markov Chain whose stationary distribution corresponds to some desired distributions, and the posterior distributions of the model parameters can be numerically evaluated by sampling the stationary part of the Markov Chain. This is achieved by first drawing a long enough sequence of the Markov Chain samples to reach the steady-state (warming-up phase), the samples of which are discarded, and then continue generating new samples (sampling phase) to estimate the posterior distributions. The main advantage of this method is that it can model a wide range of functional forms (linear and non-linear), kernel functions, and prior distributions but it can be computationally slow, due to its sampling of the spatially varying terms, and prudence is required to ensure that the steady-state of the Markov Chain has been reached.

STAN can be accessed directly from a command-line terminal (CmdStan), or through interfaces for many of the popular computing environments (Python, R, MATLAB, Mathematica, ect.). For this project, Python interfaces have been developed based on the PyStan (Riddell et al., 2021) and CmdStanPy packages, however the user can execute the STAN code (`*.stan`) through any other interface, or as-is through the terminal window. Python was chosen because it is free, open-source, and widely adopted in the scientific and engineering communities.

In the following, Section 6.1 provides a general overview of the STAN, syntax, code structure, and functions & options used in the development of NGMMs, and Section 6.2 summarizes the main components of the Python wrapper functions. For more information on STAN, visit also https://mc-stan.org/.

6.1 OVERVIEW OF STAN

6.1.1 General syntax

STAN follows the C++ syntax. All expressions need to be terminated with a semicolon (;). Any characters after two forward slashes (//) are ignored by the compiler; they are understood as comments. Curly brackets ({ . . . }) may be used to group expressions and define local variables. In the remainder of the section, the ellipses points (. . .) is used to indicated omitted code that is unimportant for the presented examples.

In STAN all variables need to be declared by defining their type and size. For scalar variables, the most common types are `int VarInt` and `real VarReal` which declare `VarInt` as a integer scalar and `VarReal` real scalar, respectively. Arrays of these types can be created by starting with the `array` argument. For example, `array[N] int VecInt` is a one-dimensional array of integers of size N, while `array[N,M] real VecReal` is a two-dimensional array of real numbers of size N by M. Real-type column vectors and real-type matrices can also be declared as `vector[N] VecReal` and `matrix[N,M] MatReal`. Alternatively, following the old syntax (prior to 2.26 STAN) vectors are declared by defining the size of the vector after the variable type (e.g. `real[N] VecReal`), and matrices are declared by defining the number of columns after the variable type and the number of rows after the variable name `real[M] MatReal[N]`. The old syntax is not recommenced when writing new regression scripts, but it is mentioned here in case an old regression script is encountered.

An important variable type used in the NGMM development is a Compressed Sparse Row (CSR) matrix which is used to store the cell-path distance matrix for the cell-specific anelastic attenuation as it is sparse and contains a lot of zeros. A CSR matrix can be defined by three one-dimensional arrays: the values of the non-zero elements (V), column indices of the non-zero elements (W), and indices of V corresponding to new rows. Although there are build in function to compute V, W, and W in STAN, the current implementation of the NGMM tools performs this calculation in the Python wrapper functions, presented in Section 6.2

Upper and lower limits for the range of variables are specified as `real<lower=low_limt, upper=up_lim> A` where `low_limt` and `up_lim` are the lower and upper limit values, respectively.

6.1.2 Regression File Structure

A STAN regression file is broken into program blocks each of them with a specific scope. The most commonly used porgram blocks are: `data`, `transformed data`, `parameters`, `transformed parameters`, and `model` as follows:

```
data {
  // ... input data ...
}
transformed data {
   // ... constants and modified data  ...
}
parameters {
    // ... model parameters ...
}
transformed parameters {
    // ... modified parameters ...
}
model {
    // ... statistical model ...
}
```

The data block declares and contains all the required input variables used in the regression model. The transformed data block offers a space for the declaration and calculation of additional input variables based on the variables in the data block. It can also be used to define and declare any constants. The parameters block declares all the model parameters, such as the non-ergodic coefficients, hyper-parameters, and aleatory terms, which will be sampled directly from the MCMC STAN sampler. The transformed parameters block allows for the definition of additional parameters, used in the model fit, which are based on the variables in data, transformed data, and parameters. The parameters in transformed parameters are sampled indirectly based on the sampled values of the parameters in parameters. The model block defines the statistical model and computes the log-likelihood. The data and transformed data blocks are executed only once at the beginning, while the blocks parameters, transformed parameters, and model are executed in every iteration of the MCMC sampler. The variables declared in parameters, transformed parameters are saved at the end of every iteration and can be used to estimate the posterior distributions.

6.1.3 Specification of Prior Distributions

STAN supports many distribution families for use as prior distributions. Any parameter that is assigned a prior distribution needs to be declared in parameters block, and then, in the model block, it is is assigned a prior distribution using the tilde symbol $\sim$ followed by the name of the distribution family and the values of parameters.

Commonly used prior distributions for the development of NGMMs in STAN are: the log-normal, inverse-gamma, and exponential distributions.

Log-normal distribution:

In cases where the existence of the random effect and the range of their standard deviation is known, a log-normal prior distribution for the standard deviation is a common choice, as it is only defined in the positive part of the real numbers and most of its mass is away from zero. The probability density function of a log-normal distribution is:

$$\pi(\sigma) = \frac{1}{\sqrt{2\pi}\,\sigma_L}\frac{1}{\sigma}\exp\left(-\frac{1}{2}\frac{(\log(\sigma)-\mu_L)^2}{\sigma_L^2}\right) \tag{6.1}$$

where μ_L and σ_L is the mean and standard deviation in log scale. In arithmetic scale, the mean of a log-normal distribution is: $\exp(\mu_L + \sigma_L/2)$. Syntax for assigning the log-normal prior distribution is:

```
  sigma ~ lognormal(mu_L, sigma_L);
```

where mu_L and sigma_L are μ_L and σ_L, respectively. In NGMM development, a log-normal distribution is used as a prior for the aleatory standard deviations phi_0 and tau_0.

Inverse-gamma distribution:

The probability density function of an inverse-gamma distribution is defined as:

$$\pi(\ell) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} \sigma^{-(\alpha+1)} \exp\left(-\frac{\beta}{\ell}\right) \tag{6.2}$$

where α is the shape, β is the scale parameter, and $\Gamma(\dot{})$ is the Gamma function. Both α and β must be positive. For $\alpha > 1$ the mean of an inverse-gamma distribution is $\beta/(\alpha-1)$. An inverse-gamma distribution is assigned in STAN with:

```
sigma ~ inv_gamma(alpha, beta)
```

In NGMMs, inverse-gamma prior distributions are commonly used for modeling the correlation length of the spatially varying terms, such as: $\ell_{1,E}$ and $\ell_{1a,S}$.

Exponential Distribution:

An exponential prior distribution can be used to regularize a particular parameter so as not to have large values, if there is not significant evidence in the data. This prior distribution is often employed when modeling non-ergodic effects that their existence is unknown ahead of time. An example of using it in NGMMs is modeling the scales in the kernel function of spatially varying terms. The probability density function of an exponential distribution is:

$$\pi(\sigma) = \lambda \exp(-\lambda \sigma) \tag{6.3}$$

where λ is the rate. The mean and the standard deviation of an exponential distribution are $1/\lambda$ In STAN, an exponential prior distribution is invoked with:

```
sigma ~ exponential(lambda)
```

Hierarchical Priors:

In STAN, defining a hierarchical model, where parameters of the prior distribution of a model variable (lower level) are assigned their own prior distributions (upper level), is done simply by specifying the different levels of prior distributions in succession, starting from the highest and going towards the lowest level. The following example shows the declaration and definition of the prior distribution for the between event residuals:

```
parameters{
  //Aleatory Terms
  real<lower=0> tau_0;
  vector[NEQ]   dB;

  ...
}
model{
  ...
  //Priors for aleatory terms
  tau_0 ~ lognormal(-1,0.3);
  dB ~ normal(0,tau_0);

  ...
}
```

The standard deviation of the between-event residuals (tau_0) is declared as a real number with a zero lower limit. The between-event residuals (dB) is defined as real valued vector of size NEQ, where NEQ is the number of unique events. At the upper level, tau_0 is assigned a log-normal prior distribution with a log mean of 0.1 and standard-deviation in log space of 0.3, while at the lower level, dB is assigned a normal prior distribution with zero mean and standard deviation tau_0.

6.1.4 Specification of Random Effects

Random effects can be easily modeled in STAN as independently and identically distributed (iid) latent variables which are applied to the corresponding group of observations based on group indices. For example, δB^0_e is modeled as a vector of iid latent variables (dB) of size NEQ, which follows a normal distribution with zero mean and tau_0 standard deviation. The elements of dB are assigned to the appropriate ground motion observation based on the earthquake indices eq. As an example of the eq structure, if the twentieth to thirtieth ground motion are associated with the second event, the twentieth to thirtieth element in eq will be equal to two.

```
parameters{
  //Aleatory Terms
  real<lower=0> tau_0;
  vector[NEQ]    dB;
  ...

}
model{
  //Between event residuals
  dB ~ normal(0,tau_0);
  ...

  //Mean non-ergodic including dB
  rec_nerg_dB = ... + dB[eq];
}
```

6.1.5 Specification of Spatially Varying Coefficients

The spatially varying coefficients are defined by using a multi-normal prior distribution and explicitly computing their mean ($\vec{\mu}$) and kernel ($\kappa(\vec{t}, \vec{t'})$) functions:

$$\delta\vec{c} \sim \mathcal{MN}\left(\vec{\mu}, \kappa(\vec{t}, \vec{t'})\right) \tag{6.4}$$

The exponential and squared-exponential kernel functions are common choices for modeling spatially varying non-ergodic effects in STAN due to their simple functional form.

$$\boldsymbol{\kappa}(\vec{t},\vec{t'}) = \sigma^2 \exp\left(-\frac{||\vec{t}-\vec{t'}||}{\ell}\right) \tag{6.5}$$

$$\boldsymbol{\kappa}(\vec{t},\vec{t'}) = \sigma^2 \exp\left(-\frac{||\vec{t}-\vec{t'}||^2}{\ell^2}\right) \tag{6.6}$$

σ control the size, and ℓ controls the length scale of the non-ergodic effects. To reduce the computational cost, the kernel functions are evaluated at the unique event or station locations and are distributed to their associated ground-motions based on the event and station indices. Additionally, to improve the regression efficiency: (i) the event-to-event and station-to-station distances are precomputed in the `transformed data` block, which is only executed once, and (ii) the spatially varying terms are sampled indirectly based on standard normally distributed random samples. In particular, in STAN, the standard normally distributed random variables are sampled in the `model` block, and the spatially varying terms are computed in the `transformed parameters` block by multiplying the lower Cholesky decomposition of the covariance matrix with the aforemetnioned standard normally distributed samples.

The following example shows the calculation of the spatially varying earthquake term, $\delta c_{1,E}$:

```
transformed data {
  real delta = 1e-8;

  //compute distances
  matrix[NEQ, NEQ] dist_e;
  ...

  //compute earthquake distances
  for(i in 1:NEQ) {
    for(j in i:NEQ) {
      real d_e = distance(X_e[i],X_e[j]);
      dist_e[i,j] = d_e;
      dist_e[j,i] = d_e;
    }

    ...
}
parameters {
  //Epistemic Uncertainty Terms
  real<lower=0.0>  ell_1e;
  real<lower=0.0>  omega_1e;
  ...

  //standardized normal variables for spatially correlated coefficients
  vector[NEQ]    z_1e;
  ...
}
transformed parameters{
  //spatially correlated coefficients
  vector[NEQ]    dc_1e;    //spatially varying eq coeff
  ...

  //spatially latent variable for event contributions to GP
  {
    matrix[NEQ,NEQ]  COV_1e;
    matrix[NEQ,NEQ]  L_1e;
    for(i in 1:NEQ) {
      //diagonal terms
      COV_1e[i,i] = omega_1e^2 + delta;
      //off-diagonal terms
      for(j in (i+1):NEQ) {
        real C_1e = (omega_1e^2 * exp(-dist_e[i,j]/ell_1e));
        COV_1e[i,j] = C_1e;
        COV_1e[j,i] = C_1e;
      }
    }
    L_1e = cholesky_decompose(COV_1e);
    dc_1e = L_1e * z_1e;
  }

  ...
}
model {
  //non-ergodic hyper-parameters
  ell_1e   ~ inv_gamma(2.,50);
  omega_1e  ~ exponential(5);
  ...

  //standardized event contributions to GP
  z_1e ~ std_normal();

  ...

  //Mean non-ergodic including dB
  rec_nerg_dB = ... + dc_1e[eq] + ...;
}
```

The distances between all event pairs are computed with the `distance(X_e[i],X_e[j])` and stored in `dist_e[i,j]`, where `X_e` is a vector with the event coordinates. In the `parameters` block, the correlation length and scale hyper-parameters `ell_1e` and `omega_1e` are declared and are assigned a zero lower limit because negative values are undefined. The standard normal samples used for the calculation of the spatially varying event terms are declared as `z_1e`. In the `transformed parameters` block, the spatially varying earthquake constant is declared as `dc_1e`. To reduce the computational cost, the upper triangular part of the covariance matrix `COV_1e` is evaluated directly using the negative exponential functional form, while the lower triangular part is filled based on the values of the upper part. `L_1e`, which is lower Cholesky decomposition of `COV_1e`, is computed using the `cholesky_decompose()` function. The samples of the spatially varying earthquake constant are generated by multiplying `L_1e` with `dc_1e`. Finally, in the `model` block, the prior distributions of the hyper-parameters and standard normal variables are first defined, and then the elements of `dc_1e` are distributed to their associated ground motions using the event indices.

6.1.6 Specification of Anelastic Attenuation

In STAN, $\vec{c}_{ca,P}$ can be modeled either with spatial correlation between neighbouring cells, or as independent from cell to cell. In the first case, the covariance matrix of $\vec{c}_{ca,P}$ is computed in the model block to impose the spatial correlation, and $\vec{c}_{ca,P}$ are drawn from a multi-normal prior distribution. In the second case, $\vec{c}_{ca,P}$ are drawn as iid samples from a normal prior distribution. In both cases, a zero upper limit is applied to $\vec{c}_{ca,P}$ to ensure proper extrapolation.

$$\vec{c}_{ca,P} \sim \mathcal{MN}\left(\mu_{ca,p}, \kappa(\vec{t}_C, \vec{t'}_C)\right) \mathcal{T}(,0) \tag{6.7}$$

A composite kernel function is used for the spatially correlated case that is the sum of an exponential and spatially independent kernel function:

$$\boldsymbol{\kappa}_{ca,P}(\vec{t}_C, \vec{t'}_C) = \omega_{ca1,P}^2 \; \exp\left(-\frac{||\vec{t}_C - \vec{t'}_C||}{\ell_{ca1,P}}\right) + \omega_{ca2,P}\, \delta(||\vec{t}_C - \vec{t'}_C||) \tag{6.8}$$

The exponential kernel function captures the underlining continuous variation of anelastic attenuation over large areas, while the spatially independent kernel function captures local changes of anelastic attenuation from cell to cell. The size and length scale of the regional component of $\vec{c}_{ca,P}$ is captured by $\omega_{ca1,P}$, and $\ell_{ca1,P}$, respectively, while the size of local component of $\vec{c}_{ca,P}$ is captured by $\omega_{ca1,P}$.

The next example shows the modeling of the spatially correlated cell-specific anelastic attenuation:

```
parameters {
  ...
  //attenuation cells
  real<upper=0.0>  mu_cap;
  real<lower=0.0>  ell_ca1p;
  real<lower=0.0>  omega_ca1p;
  real<lower=0.0>  omega_ca2p;

  ...
  //cell-specific attenuation
  vector<upper=0>[NCELL]  c_cap;

  ...
}
model {
  ...
  //effect anelastic attenuation
  vector[N] inatten;

  ...
  //cell specific attenuation hyper-parameters
  mu_cap ~ normal(c_a_erg, 0.01); //mean anelastic attenuation
  ell_ca1p ~ inv_gamma(2.,50);
  omega_ca1p ~ exponential(250);
  omega_ca2p ~ exponential(250);

  ...
  //cell attenuation
  //generate latent variables for spatially correlated
  //anelastic attenuation cells
  {
    matrix[NCELL, NCELL] COV_cap;
    for(i in 1:NCELL) {
      //diagonal terms
      COV_cap[i,i] = omega_ca1p^2 + omega_ca2p^2 + delta;
      //off-diagonal terms
      for(j in (i+1):NCELL) {
        real C_cap = (omega_ca1p^2 * exp(-dist_c[i,j]/ell_ca1p));
        COV_cap[i,j] = C_cap;
        COV_cap[j,i] = C_cap;
      }
    }
    c_cap ~ multi_normal(rep_vector(mu_cap,NCELL),COV_cap);
  }

  //anelastic attenuation
  inatten = csr_matrix_times_vector(N, NCELL, RC_val, RC_w, RC_u, c_cap);

  //Mean non-ergodic including dB
  rec_nerg_dB = ... + inatten + ...;
  ...
}
```

The mean of the prior distribution (mu_cap), scale and correlation lengths of the spatially varying component (omega_ca1p and ell_ca1p), scale of the spatially independent component (omega_ca2p), and vector of anelastic attenuation coefficients (c_cap) are declared in the parameter block. mu_cap and c_cap are assigned a zero upper limit to ensure a physically defensible behaviour. omega_ca1p, ell_ca1p, and omega_ca2p are assigned a zero lower limit as negative values of these parameters are undefined.

In the model block, inatten, which is a vector of size equal to number of ground motions, is initialized to store the effect of anelastic attenuation in the median ground motion. mu_cap is given a normal prior distribution centered around the value of the ergodic anelastic attenuation; a 0.01 standard deviation is used to allow for some adjustment of mu_cap from the ergodic value based on the re-weighting of the residuals. ell_ca1p is assigned an inverse gamma prior distribution with shape and scale parameters equal to 2.0 and 50 which results in a wide prior with a 10 to $140 km$ range for the 5^{th} to 95^{th} quantiles. The scales omega_ca1p and omega_ca2p are assigned an exponential prior distribution with 150 rate to penalize unnecessary complexity if the data do not support large variations of anelastic attenuation from cell to cell. The covariance matrix of $\vec{c}_{ca,P}$ is computed next and stored in COV_cap; similar to covariance matrices of the spatially varying terms, the upper triangular part of c_cap is computed explicitly from the covariance function while the lower-triangular part is filled based on elements in the upper part. The prior distribution of $\vec{c}_{ca,P}$ is defined in c_cap ~ multi_normal(rep_vector(mu_cap,NCELL),COV_cap). In this case, $\vec{c}_{ca,P}$ is sampled directly from a multi-normal distribution with a mu_cap mean and COV_cap covariance matrix, instead of using a standard normal distribution and the Cholesky decomposition of the covariance matrix, to enforce the upper limit constraint. The effect of anelastic attenuation in the median ground motion is computed by the product of cell-path segment length matrix ($\mathbf{R}$) and $\vec{c}_{ca,P}$. To improve the computational efficiency, $\mathbf{R}$ is stored in a CSR matrix format and the multiplication is performed with the csr_matrix_times_vector(N, NCELL, RC_val, RC_w, RC_u, c_cap) function; the first two arguments (N and NCELL) define the size of $\mathbf{R}$, RC_val contains the non-zero elements of $\mathbf{R}$, RC_w contains the column indices of the non-zero elements, and RC_u contains the indices of the RC_val elements which start each new row.

6.1.7 Specification of Likelihood Function

The likelihood function is specified similarly to the prior distributions using the ~ symbol, the name of the distribution family, and the distribution parameters. Since, for the regression, the effect of the between event residuals is included in the mean, the likelihood function is equal to the probability density function of a normal distribution with the sum of the median non-ergodic ground motion and between-event residuals as the mean, and the within-event aleatory variability as the standard deviation.

```
Y ~ normal(rec_nerg_dB,phi_0);
```

where Y is the array of the regression residuals as defined in Chapter 3 for each NGMM type, rec_nerg_dB the sum of the median non-ergodic ground motion and between-event residuals, and phi_0 is the within-event standard deviation.

6.2 PYTHON-STAN REGRESSION FOR NGMMS

The Python interface functions implemented here can be divided into three main sections:

- pre-processing to format the input files for the STAN regression code,
- regression to call the STAN MCMC sampler and pass the input data, and
- post-processing to extract the regression output and summarize the posterior distributions.

The Python wrapper function for the type-1 NGMM is presented in Section 6.2.1, and the modifications for the type-2 and type-3 NGMM wrapper functions are provided in Sections 6.2.2 and 6.2.3, respectively. The wrapper functions using CMDSTAN and Pystan libraries, are located respectively in gmm_tools/Analyses/ Python_lib/regression/cmdstan, and gmm_tools/Analyses/Python_lib/regression/pystan.

6.2.1 Python wrapper function for Type-1 NGMM

The function RunStan implemented in regression_cmdstan_model1_unbounded_hyp.py is used to run the type-1 NGMM GP regression with three main sections explained next.

Pre-processing:

The mandatory input arguments are as follows.

- $\mathrm{df\_flatfile}$ is the ground-motion regression dataframe that contains the source and site information (event ID, station ID, event coordinates, site coordinates), and total regression residuals,
- $\mathrm{stan\_model\_fname}$ contains the name of the STAN regression file,
- $\mathrm{out\_fname}$ defines the names of the output files, and
- $\mathrm{out\_dir}$ defines the location of the output directory.

The optional input arguments are as follows.

- $\mathrm{res\_name}$ is the column name of the regression residuals in $\mathrm{df\_flatfile}$ (default value is 'res'),
- $\mathrm{n\_iter\_warmup}$ is the number of iterations for the MCMC warm-up phase (default value is 300),
- $\mathrm{n\_iter\_sampling}$ is the number of MCMC sampling iterations for the calculation of the posterior distributions (default value is 300),
- $\mathrm{n\_chains}$ is the number of independently sampled MCMC chains (default value is 4),
- $\mathrm{max\_treedepth}$ controls maximum number of evaluations per sample (default value is 10),
- $\mathrm{adapt\_delta}$ is the target average proposal acceptance probability of the No-U-Turn Sampler (NUTS) in STAN (default value is 0.8), and
- $\mathrm{stan\_parallel}$ is the option for multi-thread execution of the STAN regression (default is False).

The maximum number of evaluations during each iteration is defined by the $\mathrm{max\_treedepth}$ as $10\hat{}\mathrm{max\_treedepth}$. The $\mathrm{stan\_parallel}$ option has been added for future-proofing the python functions but it requires the STAN regression codes to be rewritten for multi-thread execution which has not been implemented yet. Information on the parallelization of STAN can be found on Stan Development Team (2022).

The code in Listing 6.1 summarizes the source and site information as well as the total regression residuals. As shown, first, the record sequence number (RSN) is set as the index column in $\mathrm{df\_flatfile}$ and the number of ground motions is stored in $\mathrm{n\_data}$. $\mathrm{data\_eq\_all}$ is a matrix of $\mathrm{n\_data}$ by four, containing the event IDs, magnitude, and event coordinates in UTM for all ground motion records. Based on the event IDs, the row indices that correspond to the unique values ($\mathrm{eq\_idx}$) and inverse indices ($\mathrm{eq\_inv}$) to reconstruct the event IDs from the unique values are calculated. $\mathrm{data\_eq}$ contains the event information for all unique events, and $\mathrm{X\_eq}$ contains all unique earthquake coordinates. The event indices, which are used in STAN to distribute the non-ergodic earthquake terms to all ground motions, are stored in $\mathrm{eq\_id}$; a unit is added on all elements as Python uses a base zero index system while STAN uses a base one index system. The event-to-event distance for all pairs of unique earthquakes is computed and is checked to be larger than `5e-5` to ensure that there are no collocated events to avoid singular covariance matrices for the event terms. Similarly, `data_sta_all` matrix contains the Station IDs, V_{S30}, and station coordinates in UTM for all ground motion records. The indices of `data_sta_all` corresponding to unique stations are stored in `sta_idx`, and the inverse indices to reconstruct the station IDs array for all ground motions from the unique values are stored in `sta_inv`. `data_sta` contains the station information, and `X_sta` contains the coordinates for all unique stations. The station indices to distribute the station terms to all ground motions in STAN are stored in `sta_id`. Lastly, it is checked that there are no collocated stations based on the station-to-station distance. `y_data` is an array with the regression residuals for all ground motions.

```
## Preprocess Input Data
#==============================
#set rsn column as dataframe index, skip if rsn already the index
if not df_flatfile.index.name == 'rsn':
    df_flatfile.set_index('rsn', drop=True, inplace=True)
#number of data
n_data = len(df_flatfile)

#earthquake data
data_eq_all = df_flatfile[['eqid','mag','eqX', 'eqY']].values
_, eq_idx, eq_inv = np.unique(df_flatfile[['eqid']].values, axis=0,
                              return_inverse=True, return_index=True)
data_eq = data_eq_all[eq_idx,:]
X_eq = data_eq[:,[2,3]] #earthquake coordinates
#create earthquake ids for all records (1 to n_eq)
eq_id = eq_inv + 1
n_eq = len(data_eq)
#verify no collocated events
eq_dist_min = np.min([np.linalg.norm(x_eq - np.delete(X_eq,k, axis=0),
                                     axis=1).min()
                      for k, x_eq in enumerate(X_eq) ])
assert(eq_dist_min > 5e-5),'Error. Singular covariance matrix due
                            to collocated events'

#station data
data_sta_all = df_flatfile[['ssn','Vs30','staX','staY']].values
_, sta_idx, sta_inv = np.unique(df_flatfile[['ssn']].values, axis = 0,
                                return_inverse=True, return_index=True)
data_sta = data_sta_all[sta_idx,:]
X_sta = data_sta[:,[2,3]] #station coordinates
#create station indices for all records (1 to n_sta)
sta_id = sta_inv + 1
n_sta = len(data_sta)
#verify no collocated stations
sta_dist_min = np.min([np.linalg.norm(x_sta - np.delete(X_sta,k, axis=0),
                                      axis=1).min()
                       for k, x_sta in enumerate(X_sta) ])
assert(sta_dist_min > 5e-5),'Error. Singular covariance matrix due
                             to collocated stations'

#ground-motion observations
```

```
y_data = df_flatfile[res_name].to_numpy().copy()
```

Listing 6.1. Preprocessing section of type-1 NGMM STAN regression file. Input data organization

The Listing 6.2 summarizes the input data to be passed to STAN. All data required for the regression are grouped in the `stan_data` dictionary, with the key of each element corresponding to the variable name in the STAN regression code. The input data are saved as a Jason file on the disk with `cmdstanpy.utils.write_stan_json(stan_data_fname, stan_data)` to be read by STAN to perform the regression.

The STAN input variables are:

- N: number of ground-motion records,
- NEQ: number of unique events,
- NSTAT: number of unique stations,
- `eq`: array with event indices to distribute the earthquake terms to all ground motions,
- `stat`: array with station indices to distribute the station terms to all ground motions,
- X_e: matrix with event coordinates,
- X_s: matrix with station coordinates,
- rec_mu: array with log of median ergodic ground motion, and
- Y: array with the log ground motion observations.

The STAN regression code can be executed either with the log of ground motion observation values in Y and the log median ergodic ground motion values in rec_mu, or with the total ergodic regression residuals in Y and an array of zeros in rec_mu.

```
#stan data
stan_data = {'N':         n_data,
             'NEQ':       n_eq,
             'NSTAT':     n_sta,
             'eq':        eq_id,
             'stat':      sta_id,
             'X_e':       X_eq,
             'X_s':       X_sta,
             'rec_mu':    np.zeros(y_data.shape),
             'Y':         y_data,
            }
stan_data_fname = out_dir + out_fname + '_stan_data' + '.json'

#create output directory
pathlib.Path(out_dir).mkdir(parents=True, exist_ok=True)

#write as json file
cmdstanpy.utils.write_stan_json(stan_data_fname, stan_data)
```

Listing 6.2. Preprocessing section of type-1 NGMM STAN regression file. STAN input

Regression:

The Listing 6.3 compiles the STAN code and runs the regression. If the variable stan_parallel is True and the number of independent chains is less than the number of CPU cores, STAN is complied with the multi-threading option enabled. The link to the compiled code is represented by the stan_model object. The Bayesian regression is performed with the .sample(...) method of the STAN object, with the following arguments.

- data specifies the file name of the input data.
- chains specifies the number of independent chains.
- iter_warmup & iter_sampling specify the number of iterations for the warmup and sampling phase of the MCMC sampler.
- seed initializes the random number generator for the reproducibility of the regression.
- max_treedepth defines the maximum number of evaluations of the posterior distribution during each iteration (10^max_treedepth).
- adapt_delta specifies the average proposal acceptance probability of the NUTS STAN sampler.
- output_dir defines the directory of the temporary output files.
- show_progress can be set True to show the progress on the STAN sampler during the regression.

For the multithread execution of the STAN, the cpp_options={"STAN_THREADS": True}) argument should be specified in the .CmdStanModel(...) method, and the number of CPU cores per chain (threads_per_chain) should be specified in the .sample(...) method.

```python
## Run Stan, fit model
#===============================
#number of cores
n_cpu = max(cpu_count() -1,1)

#run stan
if (not stan_parallel) or n_cpu<=n_chains:
    #compile stan model
    stan_model = cmdstanpy.CmdStanModel(stan_file=stan_model_fname)
    stan_model.compile(force=True)
    #run full MCMC sampler
    stan_fit = stan_model.sample(data=stan_data_fname, chains=n_chains,
                                 iter_warmup=n_iter_warmup,
                                 iter_sampling=n_iter_sampling,
                                 seed=1, max_treedepth=max_treedepth,
                                 adapt_delta=adapt_delta,
                                 output_dir=out_dir+'stan_fit/',
                                 show_progress=True)
 else:
    #compile stan model
    stan_model = cmdstanpy.CmdStanModel(stan_file=stan_model_fname,
                                        cpp_options={"STAN_THREADS": True})
    stan_model.compile(force=True)
    #number of cores per chain
    n_cpu_chain = int(np.floor(n_cpu/n_chains))
    #run full MCMC sampler
    stan_fit = stan_model.sample(data=stan_data_fname, chains=n_chains,
                                 iter_warmup=n_iter_warmup,
                                 iter_sampling=n_iter_sampling,
                                 seed=1, max_treedepth=max_treedepth,
                                 adapt_delta=adapt_delta,
                                 output_dir=out_dir+'stan_fit/',
                                 show_progress=True,
                                 threads_per_chain=n_cpu_chain)
```

Listing 6.3. Regression section of type-1 NGMM STAN regression file.

Post-processing:

Listing 6.4 summarizes MCMC sampling from the posterior distributions of all NGMM parameters and hyper-parameters using `df_stan_posterior_raw`. `col_names_hyp` defines the column names in `df_stan_posterior_raw` for the samples of hyper-parameters. `col_names_dc_1e`, `col_names_dc_1as`, and `col_names_dc_1bs` specify the column names for the non-ergodic earthquake and station terms ($\delta c_{1,E}$, $\delta c_{1a,S}$, and $\delta c_{1b,S}$), and `col_names_dB` specifies the column names for the between event residuals.

The MCMC samples are extracted from the STAN regression object with `stan_fit.stan_variable(var_name)` method, where `var_name` is the name of the variable in the STAN code. The line `np.stack([stan_fit.stan_variable(c_n) for c_n in col_names_hyp], axis=1)` extracts the hyper-parameter samples and stores them in the `stan_posterior` matrix. The next three lines extract the $\delta c_{1,E}$, $\delta c_{1a,S}$, and $\delta c_{1b,S}$ posterior samples for all event and station locations (separate column for each unique event and station) and append them to `stan_posterior`. Line `np.concatenate((stan_posterior, stan_fit.stan_variable ('dB')), axis=1)` extracts the δB_e^0 samples (separate column for each unique event) and adds them to the end of the `stan_posterior`. Then, the sampled matrix is converted into a pandas dataframe in `pd.DataFrame(stan_posterior, columns = col_names_all)` and is saved as comma separated valued file (CSV) in `df_stan_posterior_raw.to_csv(out_dir + out_fname + '_stan_posterior_raw' + '.csv', index=False)` with the `'_stan_posterior_raw.csv'` suffix.

```
## Postprocessing Data
#==========================
## Extract posterior samples
# --------------------------
#hyper-parameters
col_names_hyp = ['dc_0','ell_1e','ell_1as','omega_1e','omega_1as','omega_1bs',
                 'phi_0','tau_0']

#non-ergodic terms
col_names_dc_1e  = ['dc_1e.%i'%(k)  for k in range(n_eq)]
col_names_dc_1as = ['dc_1as.%i'%(k) for k in range(n_sta)]
col_names_dc_1bs = ['dc_1bs.%i'%(k) for k in range(n_sta)]
col_names_dB     = ['dB.%i'%(k)     for k in range(n_eq)]
col_names_all = col_names_hyp + col_names_dc_1e + col_names_dc_1as +
                col_names_dc_1bs + col_names_dB
```

```
#summarize raw posterior distributions
stan_posterior = np.stack([stan_fit.stan_variable(c_n)
                           for c_n in col_names_hyp], axis=1)
#adjustment terms
stan_posterior = np.concatenate((stan_posterior,
                                 stan_fit.stan_variable('dc_1e')),  axis=1)
stan_posterior = np.concatenate((stan_posterior,
                                 stan_fit.stan_variable('dc_1as')), axis=1)
stan_posterior = np.concatenate((stan_posterior,
                                 stan_fit.stan_variable('dc_1bs')), axis=1)
stan_posterior = np.concatenate((stan_posterior,
                                 stan_fit.stan_variable('dB')),     axis=1)

#save raw-posterior distribution
df_stan_posterior_raw = pd.DataFrame(stan_posterior, columns = col_names_all)
df_stan_posterior_raw.to_csv(out_dir+out_fname+'_stan_posterior_raw'+'.csv',
                             index=False)
```

Listing 6.4. Post-processing section of type-1 NGMM STAN regression file. Hyper-parameter summary.

The Listing 6.5 shows the part of the code that organizes the posterior distributions of the hyper-parameters in the `df_stan_hyp` dataframe. The array `perc_array` defines the percentiles of the posterior distributions to be reported, and the `.quantile()` method of the `df_stan_posterior_raw` dataframe computes the hyper-parameter values at the corresponding percentiles. The mean values of the hyper-parameters are computed at `df_stan_posterior_raw[col_names_hyp].mean(axis = 0)` and appended in `df_stan_hyp`. `df_stan_hyp` is saved on the computer disk as a CSV file with the `'_stan_hyperposterior.csv'` suffix.

```
## Summarize hyper-parameters
# --------------------------
#summarize posterior distributions of hyper-parameters
perc_array = np.array([0.05,0.25,0.5,0.75,0.95])
df_stan_hyp = df_stan_posterior_raw[col_names_hyp].quantile(perc_array)
df_stan_hyp = df_stan_hyp.append(
                          df_stan_posterior_raw[col_names_hyp].mean(axis=0),
                          ignore_index=True)
df_stan_hyp.index = ['prc_%.2f'%(prc) for prc in perc_array]+['mean']
df_stan_hyp.to_csv(out_dir+out_fname+'_stan_hyperparameters'+'.csv',
                   index=True)

#detailed posterior percentiles of posterior distributions
perc_array = np.arange(0.01,0.99,0.01)
df_stan_posterior = df_stan_posterior_raw[col_names_hyp].quantile(perc_array)
df_stan_posterior.index.name = 'prc'
df_stan_posterior.to_csv(out_dir+out_fname+'_stan_hyperposterior'+'.csv',
                         index=True)

del col_names_dc_1e, col_names_dc_1as, col_names_dc_1bs, col_names_dB
del stan_posterior, col_names_all
```

Listing 6.5. Post-processing section of type-1 NGMM STAN regression file. Hyper-parameter summary.

The Listing 6.6 summarizes the posterior distributions for the non-ergodic coefficients. The coeff_0_mu, coeff_0_med, and coeff_0_sig contain the posterior mean, median, and standard deviation values for c_0. The coeff_1e_mu, coeff_1e_med, and coeff_1e_sig contain the mean, median, and standard deviation values of $c_{1,E}$ for all ground motions. The coeff_1as_mu, coeff_1as_med, and coeff_1as_sig contain the mean, median, and standard deviation values of $c_{1a,S}$. The coeff_1bs_mu, coeff_1bs_med, and coeff_1bs_sig contain the mean, median, and standard deviation values of $c_{1a,S}$ for all ground motions. The posterior mean of the aleatory standard deviations (ϕ_0 and τ_0) are stored in phi_0_array and tau_0_array.

The variables in the preceding paragraph are summarized in df_coeffs_summary along with the event ID, station ID, earthquake coordinates, and station coordinates and saved as a CSV file with the '_stan_coefficients.csv' suffix.

```
# GMM coefficients
#---   ---   ---   ---   ---   ---   ---   ---
#constant shift coefficient
coeff_0_mu  = df_stan_posterior_raw.loc[:,'dc_0'].mean()   * np.ones(n_data)
coeff_0_med = df_stan_posterior_raw.loc[:,'dc_0'].median() * np.ones(n_data)
coeff_0_sig = df_stan_posterior_raw.loc[:,'dc_0'].std()    * np.ones(n_data)

#spatially varying earthquake constant coefficient
coeff_1e_mu  = np.array([df_stan_posterior_raw.loc[:,f'dc_1e.{k}'].mean()
                         for k in range(n_eq)])
coeff_1e_mu  = coeff_1e_mu[eq_inv]
coeff_1e_med = np.array([df_stan_posterior_raw.loc[:,f'dc_1e.{k}'].median()
                         for k in range(n_eq)])
coeff_1e_med = coeff_1e_med[eq_inv]
coeff_1e_sig = np.array([df_stan_posterior_raw.loc[:,f'dc_1e.{k}'].std()
                         for k in range(n_eq)])
coeff_1e_sig = coeff_1e_sig[eq_inv]

#site term constant covariance
coeff_1as_mu  = np.array([df_stan_posterior_raw.loc[:,f'dc_1as.{k}'].mean()
                          for k in range(n_sta)])
coeff_1as_mu  = coeff_1as_mu[sta_inv]
coeff_1as_med = np.array([df_stan_posterior_raw.loc[:,f'dc_1as.{k}'].median()
                          for k in range(n_sta)])
coeff_1as_med = coeff_1as_med[sta_inv]
coeff_1as_sig = np.array([df_stan_posterior_raw.loc[:,f'dc_1as.{k}'].std()
                          for k in range(n_sta)])
coeff_1as_sig = coeff_1as_sig[sta_inv]

#spatially varying station constant covariance
coeff_1bs_mu  = np.array([df_stan_posterior_raw.loc[:,f'dc_1bs.{k}'].mean()
                          for k in range(n_sta)])
coeff_1bs_mu  = coeff_1bs_mu[sta_inv]
coeff_1bs_med = np.array([df_stan_posterior_raw.loc[:,f'dc_1bs.{k}'].median()
                          for k in range(n_sta)])
coeff_1bs_med = coeff_1bs_med[sta_inv]
coeff_1bs_sig = np.array([df_stan_posterior_raw.loc[:,f'dc_1bs.{k}'].std()
                          for k in range(n_sta)])
coeff_1bs_sig = coeff_1bs_sig[sta_inv]
```

```
# aleatory variability
phi_0_array=np.array([df_stan_posterior_raw.phi_0.mean()]*X_sta_all.shape[0])
tau_0_array=np.array([df_stan_posterior_raw.tau_0.mean()]*X_sta_all.shape[0])

#initiaize flatfile for sumamry of non-erg coefficinets and residuals
df_flatinfo = df_flatfile[['eqid','ssn','eqLat','eqLon','staLat','staLon',
                           'eqX','eqY','staX','staY','UTMzone']]

#summary coefficients
coeffs_summary = np.vstack((coeff_0_mu,
                            coeff_1e_mu,
                            coeff_1as_mu,
                            coeff_1bs_mu,
                            coeff_0_med,
                            coeff_1e_med,
                            coeff_1as_med,
                            coeff_1bs_med,
                            coeff_0_sig,
                            coeff_1e_sig,
                            coeff_1as_sig,
                            coeff_1bs_sig)).T
columns_names = ['dc_0_mean','dc_1e_mean','dc_1as_mean','dc_1bs_mean',
                 'dc_0_med', 'dc_1e_med', 'dc_1as_med', 'dc_1bs_med',
                 'dc_0_sig', 'dc_1e_sig', 'dc_1as_sig', 'dc_1bs_sig']
df_coeffs_summary = pd.DataFrame(coeffs_summary, columns = columns_names,
                                 index=df_flatfile.index)
#create dataframe with summary coefficients
df_coeffs_summary = pd.merge(df_flatinfo, df_coeffs_summary, how='right',
                             left_index=True, right_index=True)
df_coeffs_summary[['eqid','ssn']] =
                            df_coeffs_summary[['eqid','ssn']].astype(int)
df_coeffs_summary.to_csv(out_dir + out_fname + '_stan_coefficients' + '.csv',
                         index=True)
```

Listing 6.6. Post-processing section of type-1 NGMM STAN regression file. Non-ergodic coefficients summary.

Listing 6.7 organizes the non-ergodic residuals. `y_mu` computes the median non-ergodic adjustment to the ground motion that is equal to the sum of all the non-ergodic terms. The total non-ergodic residuals are calculated by subtracting `y_mu` from the total ergodic residuals. The within-event residuals (`res_within`) are extracted directly from the STAN regression, and the between event residuals are calculated by subtracting the between event residuals from the total non-ergodic residuals. The median non-ergodic adjustment, and all non-ergodic residuals are summarized in the `df_predict_summary` dataframe which is saved as a CSV file under the `'_stan_residuals.csv'` suffix.

```
# GMM prediction
#---  ---  ---  ---  ---  ---  ---  ---
#mean prediction
y_mu  = (coeff_0_mu + coeff_1e_mu + coeff_1as_mu + coeff_1bs_mu)

#compute residuals
res_tot     = y_data - y_mu
```

```
#residuals computed directly from stan regression
res_between = [df_stan_posterior_raw.loc[:,f'dB.{k}'].mean()
               for k in range(n_eq)]
res_between = np.array([res_between[k] for k in (eq_inv).astype(int)])
res_within  = res_tot - res_between

#summary predictions and residuals
predict_summary = np.vstack((y_mu, res_tot, res_between, res_within)).T
columns_names = ['nerg_mu','res_tot','res_between','res_within']
df_predict_summary = pd.DataFrame(predict_summary, columns = columns_names,
                                  index=df_flatfile.index)
#create dataframe with predictions and residuals
df_predict_summary = pd.merge(df_flatinfo, df_predict_summary, how='right',
                              left_index=True, right_index=True)
df_predict_summary[['eqid','ssn']] =
                            df_predict_summary[['eqid','ssn']].astype(int)
df_predict_summary.to_csv(out_dir + out_fname + '_stan_residuals' + '.csv',
                          index=True)
```

Listing 6.7. Post-processing section of type-1 NGMM STAN regression file. Ground motion summary.

Lastly, Listing 6.8 prints out the summary statistics of the regression fit, and plots the posterior distributions of the hyper-parameters. The summary of the regression is saved as plain text files with the `'_stan_summary.txt'` suffix. Additionally, for each hyper-parameter, the posterior distribution and trace plot of each chain is plotted using the Arviz Python package. These figures can be used to visually inspect the quality of the regression looking for convergence and for good mixing between the independent chains. As an example, Figure 6.1 shows the posterior distribution and trace plot of ϕ_0 estimated from the first synthetic ground-motion realization of NGAWest3[*]. The posterior distributions of the MCMC chains do not show big differences and the trace plots exhibit stationary behavior and overlap among different chains, showing a good regression fit.

```
## Summary regression
# ---------------------------
#save summary statistics
stan_summary_fname = out_dir  + out_fname + '_stan_summary' + '.txt'
with open(stan_summary_fname, 'w') as f:
    print(stan_fit.summary(), file=f)

#create and save trace plots
fig_dir = out_dir + 'summary_figs/'
#create figures directory if doesn't exit
pathlib.Path(fig_dir).mkdir(parents=True, exist_ok=True)

#create stan trace plots
stan_az_fit = az.from_cmdstanpy(stan_fit, posterior_predictive='Y')
for c_name in col_names_hyp:
    #create trace plot with arviz
    ax = az.plot_trace(stan_az_fit,  var_names=c_name, figsize=(10,5)).ravel()
    ax[0].yaxis.set_major_locator(plt_autotick())
    ax[0].set_xlabel('sample value')
    ax[0].set_ylabel('frequency')
    ax[0].set_title('')
    ax[0].grid(axis='both')
    ax[1].set_xlabel('iteration')
    ax[1].set_ylabel('sample value')
    ax[1].grid(axis='both')
    ax[1].set_title('')
    fig = ax[0].figure
    fig.suptitle(c_name)
    fig.savefig(fig_dir + out_fname + '_stan_traceplot_' + c_name +
                    '_arviz' + '.png')
```

Listing 6.8. Post-processing section of type-1 NGMM STAN regression file. Ground motion summary.

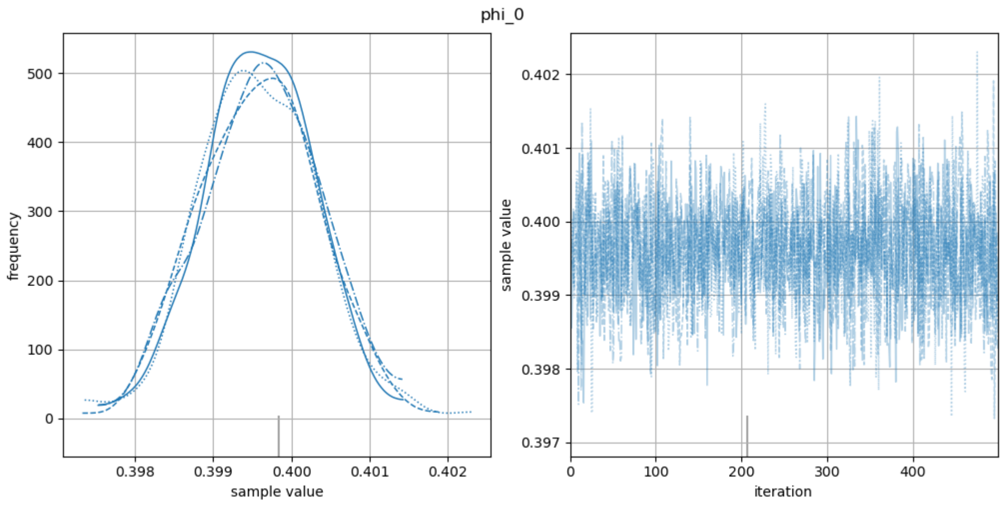


Figure 6.1. Posterior distribution and trace plot of ϕ_0 for NGAWest3* CA synthetic dataset Y1.

6.2.2 Python wrapper function for Type-2 NGMM

This section presents the modifications to Section 6.2.1 to run the NGMM type-2 STAN regression. The python wrapper functions for this regression are implemented in `regression_cmdstan_model2_uncorr_cells_sparse_unbounded_hyp.py` for the uncorrelated attenuation cells, and in `regression_cmdstan_model2_corr_cells_sparse_unbounded_hyp.py` for the partially correlated attenuation cells cases. Again, the implemented code has three main sections to be discussed next.

Pre-processing:

In addition to the input arguments specified in Section 6.2.1, mandatory input arguments for the NGMM tyre-2 regression are as follows.

- `df_cellinfo` contains the information about the cell IDs and coordinates (Section 4.2),
- `df_celldist` contains the information about the path segment length over each cell for

every ground motion in the regression flatfile, and

- c_a_erg is the ergodic value of the anelastic attenuation coefficient.

The additional pre-processing steps in type-2 NGMM as compared to type-1 NGMM is related to processing the anelastic attenuation cells as shown in Listing 6.9. First, if they are not set already, the RSN column is set as the index column in df_celldist and the cell ID column is set as an index column in df_cellinfo. Then, the cell-path distance dataframe is reduced to the columns that correspond to the ground-motions in df_flatfile. The IDs and names of all cells are stored in cell_ids_all and cell_names_all, respectively, and the cell-path distance matrix for all cells is stored in celldist_all. Only the cells with at least one crossing are used in the regression to reduce the computational cost, and are identified in i_cells_valid = np.where(celldist_all.sum(axis=0) > 0)[0]) line. The IDs and names of these cells are contained in cell_ids_valid and cell_names_valid, while celldist_valid contains the reduced cell-path distance segment matrix. celldist_valid is converted into a sparse CSR format in celldist_valid_sp = sparse.csr_matrix(celldist_valid). The mid-point coordinates of the cells with at least one cell crossing are stored in X_cells_valid.

```
if not df_celldist.index.name == 'rsn':
    df_celldist.set_index('rsn', drop=True, inplace=True)
#set cellid column as dataframe index, skip if cellid already the index
if not df_cellinfo.index.name == 'cellid':
    df_cellinfo.set_index('cellid', drop=True, inplace=True)

#cell data
#reorder and only keep records included in the flatfile
df_celldist = df_celldist.reindex(df_flatfile.index)
#cell info
cell_ids_all   = df_cellinfo.index
cell_names_all = df_cellinfo.cellname
#cell distance matrix
celldist_all   = df_celldist[cell_names_all]
#find cell with more than one paths
i_cells_valid = np.where(celldist_all.sum(axis=0) > 0)[0]
cell_ids_valid    = cell_ids_all[i_cells_valid]
cell_names_valid = cell_names_all[i_cells_valid]
celldist_valid    = celldist_all.loc[:,cell_names_valid].to_numpy()
celldist_valid_sp = sparse.csr_matrix(celldist_valid)
#number of cells
n_cell = celldist_all.shape[1]
n_cell_valid = celldist_valid.shape[1]
#cell coordinates
X_cells_valid = df_cellinfo.loc[i_cells_valid,['mptX','mptY']].values
```

Listing 6.9. Preprocessing section of type-2 NGMM STAN regression file. Cell-specific anelastic attenuation.

The Listing 6.10 summarizes the STAN input data for the NGMM type-2 regression. The additional input variables are:

- NCELL: number of cells.
- NCELL_SP: number of non-zero elements in the cell-path distance matrix.

- RC_val: array with the non-zero elements of the cell-path distance matrix.
- RC_w: array of the column indices of the non-zero cell-path distance matrix elements.
- RC_u: array of indices of the RC_val elements, which starts each new row.
- c_a_erg: the ergodic value of anelastic attenuation.

```
stan_data = {'N':          n_data,
             'NEQ':        n_eq,
             'NSTAT':      n_sta,
             'NCELL':      n_cell_valid,
             'NCELL_SP':   len(celldist_valid_sp.data),
             'eq':         eq_id,
             'stat':       sta_id,
             'X_e':        X_eq,
             'X_s':        X_sta,
             'X_c':        X_cells_valid,
             'rec_mu':     np.zeros(y_data.shape),
             'RC_val':     celldist_valid_sp.data,
             'RC_w':       celldist_valid_sp.indices+1,
             'RC_u':       celldist_valid_sp.indptr+1,
             'c_a_erg':    c_a_erg,
             'Y':          y_data,
            }
```

Listing 6.10. Preprocessing section of type-2 NGMM STAN regression file. STAN Input.

Regression:

The execution of the STAN regression is identical to the execution of the STAN regression for NGMM type-1.

Post-processing:

The anelastic attenuation coefficients for the NGMM type-2 regression are summarized in Listing 6.11. Mean, median, and standard deviation of the posterior distributions of the anelastic attenuation coefficients are stored in cells_ca_mu, cells_ca_med, and cells_ca_sig, respectively. Similarly, the mean and median impact as well as the epistemic uncertainty on the ground motion due to the cell-specific anelastic attenuation is computed in cells_LcA_mu, cells_LcA_med, and cells_LcA_sig; the mean and the median impact are calculated by multiplying the cell-path distance matrix (celldist_valid_sp) with cells_ca_mu and cells_ca_med. The ground-motion epistemic uncertainty due to the cell-specific anelastic attenuation is computed as the square root of the celldist_valid_sp elements squared multiplied by the squared elements of cells_ca_sig.

The posterior distribution statistics of the anelastic attenuation cells are summarized in df_catten_summary together with the cell ID, name, and coordinate information. df_catten_summary is saved as a CSV file with the '_stan_catten.csv' suffix.

```
#GMM anelastic attenuation
#---   ---   ---   ---   ---   ---   ---   ---
cells_ca_mu   = np.array([df_stan_posterior_raw.loc[:,'c_cap.%i'%(k)].mean()
                          for k in cell_ids_valid])
```

```
cells_ca_med = np.array([df_stan_posterior_raw.loc[:,'c_cap.%i'%(k)].median()
                         for k in cell_ids_valid])
cells_ca_sig = np.array([df_stan_posterior_raw.loc[:,'c_cap.%i'%(k)].std()
                         for k in cell_ids_valid])

#effect of anelastic attenuation in GM
cells_LcA_mu  = celldist_valid_sp @ cells_ca_mu
cells_LcA_med = celldist_valid_sp @ cells_ca_med
cells_LcA_sig = np.sqrt(celldist_valid_sp.power(2) @ cells_ca_sig**2)

#summary attenuation cells
catten_summary = np.vstack((np.tile(c_a_erg,  n_cell_valid),
                            cells_ca_mu,
                            cells_ca_med,
                            cells_ca_sig)).T
columns_names = ['c_a_erg','c_cap_mean','c_cap_med','c_cap_sig']
df_catten_summary = pd.DataFrame(catten_summary, columns = columns_names,
                                 index=df_cellinfo.index[i_cells_valid])
#create dataframe with summary attenuation cells
df_catten_summary = pd.merge(df_cellinfo[['cellname','mptLat','mptLon',
                                          'mptX','mptY','mptZ','UTMzone']],
                             df_catten_summary,
                             how='right', left_index=True, right_index=True)
df_catten_summary.to_csv(out_dir + out_fname + '_stan_catten' + '.csv',
                         index=True)
```

Listing 6.11. Post-processing section of type-2 NGMM STAN regression file. Cell-specific anelastic attenuation coefficient summary.

The summary of the non-ergodic ground motion for the NGMM type-2 (Listing 6.12) is organized similarly to the summary of non-ergodic ground motion for the NGMM type-1. The main difference is in the calculation of the median non-ergodic adjustment (y_mu) that includes the effect of the anelastic attenuation.

```
# GMM prediction
#--- --- --- --- --- --- --- ---
#mean prediction
y_mu = (coeff_0_mu + coeff_1e_mu + coeff_1as_mu + coeff_1bs_mu + cells_LcA_mu)

#compute residuals
res_tot     = y_data - y_mu
#residuals computed directly from stan regression
res_between = [df_stan_posterior_raw.loc[:,f'dB.{k}'].mean()
               for k in range(n_eq)]
res_between = np.array([res_between[k] for k in (eq_inv).astype(int)])
res_within  = res_tot - res_between

#summary predictions and residuals
predict_summary = np.vstack((y_mu, res_tot, res_between, res_within)).T
columns_names = ['nerg_mu','res_tot','res_between','res_within']
df_predict_summary = pd.DataFrame(predict_summary, columns = columns_names,
                                  index=df_flatfile.index)
#create dataframe with predictions and residuals
df_predict_summary = pd.merge(df_flatinfo, df_predict_summary, how='right',
                                   left_index=True,  right_index=True)
df_predict_summary[['eqid','ssn']] =
                                df_predict_summary[['eqid','ssn']].astype(int)
df_predict_summary.to_csv(out_dir  + out_fname + '_stan_residuals' + '.csv',
                              index=True)
```

Listing 6.12. Post-processing section of type-2 NGMM STAN regression file. Ground motion summary.

6.2.3 Python wrapper function for Type-3 NGMM

This subsection presents the modification of the STAN code for the NGMM type-3 regression. The python wrapper functions are implemented in `regression_cmdstan_model3_uncorr_cells_sparse_unbounded_hyp.py` for the uncorrelated attenuation cells, and in `regression_cmdstan_model3_corr_cells_sparse_unbounded_hyp.py` for the partially correlated attenuation cells, with the following main sections.

Pre-processing:

The extra optional input arguments for NGMM-type3 are:

- `c_2_erg`: The ergodic value of the geometrical spreading coefficient (default value is 0).
- `c_3_erg`: The ergodic value of the V_{S30} scaling (default value is 0).

Furthermore, the ground-motion regression dataframe must include the columns `x_2` and `x_3` for the geometrical spreading and V_{S30} scaling.

The additional preprocessing steps are shown in Listing 6.13. The `x_2` stores the geometrical spreading scaling for all ground motions and the `x_3` stores the V_{S30} scaling for all unique stations.

```
#geometrical spreading covariates
x_2 = df_flatfile['x_2'].values
#vs30 covariates
x_3 = df_flatfile['x_3'].values[sta_idx]
```

Listing 6.13. Preprocessing section of type-3 NGMM STAN regression file. Geometrical spreading and V_{S30} scaling.

The input STAN variables are presented in Listing 6.14. The extra input variables for the NGMM type-3 regression are listed below.

- `x_2`: array of size N containing the geometrical spreading terms.
- `x_3`: array of size NSTAT containing the V_{S30} scaling terms.
- `c_2_erg`: the ergodic value for geometrical spreading.
- `c_3_erg`: the ergodic value for V_{S30} scaling.

```
stan_data = {'N':        n_data,
             'NEQ':      n_eq,
             'NSTAT':    n_sta,
             'NCELL':    n_cell_valid,
             'NCELL_SP': len(celldist_valid_sp.data),
             'eq':       eq_id,
             'stat':     sta_id,
             'rec_mu':   np.zeros(y_data.shape),
             'Y':        y_data,
             'x_2':      x_2,
             'x_3':      x_3,
             'c_2_erg':  c_2_erg,
             'c_3_erg':  c_3_erg,
             'c_a_erg':  c_a_erg,
             'X_e':      X_eq,
             'X_s':      X_sta,
             'X_c':      X_cells_valid,
             'RC_val':     celldist_valid_sp.data,
             'RC_w':       celldist_valid_sp.indices+1,
             'RC_u':       celldist_valid_sp.indptr+1,
           }
```

Listing 6.14. Preprocessing section of type-3 NGMM STAN regression file. STAN Input.

Regression:

The STAN regression section is the same as those for NGMM type-1 and NGMM type-2.

Post-processing:

The posterior distribution statistics of the the non-ergodic coefficients are summarized in Listing 6.15. The mean, median, and standard deviation of $c_{2,P}$ are described in coeff_2p_mu, coeff_2p_med, and coeff_2p_sig, while the mean, median, and standard deviation of $c_{3,S}$ are described in coeff_2s_mu, coeff_2s_med, and coeff_2s_sig. All non-ergodic coefficients for the NGMM type-3 regression are summarized in df_coeffs_summary dataframe which is saved as a CSV file.

```
# GMM coefficients
#---   ---   ---   ---   ---   ---   ---   ---
#constant shift coefficient
coeff_0_mu  = df_stan_posterior_raw.loc[:,'dc_0'].mean()   * np.ones(n_data)
coeff_0_med = df_stan_posterior_raw.loc[:,'dc_0'].median() * np.ones(n_data)
coeff_0_sig = df_stan_posterior_raw.loc[:,'dc_0'].std()    * np.ones(n_data)

#spatially varying earthquake constant coefficient
coeff_1e_mu  = np.array([df_stan_posterior_raw.loc[:,f'dc_1e.{k}'].mean()
                         for k in range(n_eq)])
coeff_1e_mu  = coeff_1e_mu[eq_inv]
coeff_1e_med = np.array([df_stan_posterior_raw.loc[:,f'dc_1e.{k}'].median()
                         for k in range(n_eq)])
coeff_1e_med = coeff_1e_med[eq_inv]
coeff_1e_sig = np.array([df_stan_posterior_raw.loc[:,f'dc_1e.{k}'].std()
                         for k in range(n_eq)])
coeff_1e_sig = coeff_1e_sig[eq_inv]

#site term constant covariance
coeff_1as_mu  = np.array([df_stan_posterior_raw.loc[:,f'dc_1as.{k}'].mean()
                          for k in range(n_sta)])
coeff_1as_mu  = coeff_1as_mu[sta_inv]
```

```
coeff_1as_med = np.array([df_stan_posterior_raw.loc[:,f'dc_1as.{k}'].median()
                          for k in range(n_sta)])
coeff_1as_med = coeff_1as_med[sta_inv]
coeff_1as_sig = np.array([df_stan_posterior_raw.loc[:,f'dc_1as.{k}'].std()
                          for k in range(n_sta)])
coeff_1as_sig = coeff_1as_sig[sta_inv]

#spatially varying station constant covariance
coeff_1bs_mu  = np.array([df_stan_posterior_raw.loc[:,f'dc_1bs.{k}'].mean()
                          for k in range(n_sta)])
coeff_1bs_mu  = coeff_1bs_mu[sta_inv]
coeff_1bs_med = np.array([df_stan_posterior_raw.loc[:,f'dc_1bs.{k}'].median()
                          for k in range(n_sta)])
coeff_1bs_med = coeff_1bs_med[sta_inv]
coeff_1bs_sig = np.array([df_stan_posterior_raw.loc[:,f'dc_1bs.{k}'].std()
                          for k in range(n_sta)])
coeff_1bs_sig = coeff_1bs_sig[sta_inv]

#spatially varying geometrical spreading coefficient
coeff_2p_mu  = np.array([df_stan_posterior_raw.loc[:,f'c_2p.{k}'].mean()
                         for k in range(n_eq)])
coeff_2p_mu  = coeff_2p_mu[eq_inv]
coeff_2p_med = np.array([df_stan_posterior_raw.loc[:,f'c_2p.{k}'].median()
                         for k in range(n_eq)])
coeff_2p_med = coeff_2p_med[eq_inv]
coeff_2p_sig = np.array([df_stan_posterior_raw.loc[:,f'c_2p.{k}'].std()
                         for k in range(n_eq)])
coeff_2p_sig = coeff_2p_sig[eq_inv]

#spatially varying Vs30 coefficient
coeff_3s_mu  = np.array([df_stan_posterior_raw.loc[:,f'c_3s.{k}'].mean()
                         for k in range(n_sta)])
coeff_3s_mu  = coeff_3s_mu[sta_inv]
coeff_3s_med = np.array([df_stan_posterior_raw.loc[:,f'c_3s.{k}'].median()
                         for k in range(n_sta)])
coeff_3s_med = coeff_3s_med[sta_inv]
coeff_3s_sig = np.array([df_stan_posterior_raw.loc[:,f'c_3s.{k}'].std()
                         for k in range(n_sta)])
coeff_3s_sig = coeff_3s_sig[sta_inv]

# aleatory variability
phi_0_array = np.array([df_stan_posterior_raw.phi_0.mean()]*X_sta_all.shape[0])
tau_0_array = np.array([df_stan_posterior_raw.tau_0.mean()]*X_sta_all.shape[0])

#initiaize flatfile for sumamry of non-erg coefficinets and residuals
df_flatinfo = df_flatfile[['eqid','ssn','eqLat','eqLon','staLat','staLon',
                           'eqX','eqY','staX','staY','UTMzone']]

#summary coefficients
coeffs_summary = np.vstack((coeff_0_mu,
                            coeff_1e_mu,
                            coeff_1as_mu,
                            coeff_1bs_mu,
                            coeff_2p_mu,
                            coeff_3s_mu,
                            cells_LcA_mu,
                            coeff_0_med,
                            coeff_1e_med,
                            coeff_1as_med,
                            coeff_1bs_med,
                            coeff_2p_med,
                            coeff_3s_med,
                            cells_LcA_med,
                            coeff_0_sig,
                            coeff_1e_sig,
                            coeff_1as_sig,
                            coeff_1bs_sig,
                            coeff_2p_sig,
                            coeff_3s_sig,
                            cells_LcA_sig)).T
columns_names = ['dc_0_mean','dc_1e_mean','dc_1as_mean','dc_1bs_mean',
                 'c_2p_mean','c_3s_mean','Lc_ca_mean',
                 'dc_0_med', 'dc_1e_med', 'dc_1as_med', 'dc_1bs_med',
                 'c_2p_med', 'c_3s_med', 'Lc_ca_med',
                 'dc_0_sig', 'dc_1e_sig', 'dc_1as_sig', 'dc_1bs_sig',
                 'c_2p_sig', 'c_3s_sig', 'Lc_ca_sig']
df_coeffs_summary = pd.DataFrame(coeffs_summary, columns = columns_names,
                                 index=df_flatfile.index)
#create dataframe with summary coefficients
df_coeffs_summary = pd.merge(df_flatinfo, df_coeffs_summary, how='right',
                             left_index=True, right_index=True)
df_coeffs_summary[['eqid','ssn']] =
                            df_coeffs_summary[['eqid','ssn']].astype(int)
df_coeffs_summary.to_csv(out_dir + out_fname + '_stan_coefficients' + '.csv',
                         index=True)
```

Listing 6.15. Post-processing section of type-3 NGMM STAN regression file. Non-ergodic coefficients summary.

The non-ergodic ground motion summary is shown in Listing 6.16. The mean non-ergodic adjustment includes the effects of the non-ergodic geometrical spreading and non-ergodic V_{S30} scaling with `coeff_2p_mu*x_2` and `coeff_3s_mu*x_3[sta_inv]`.

```
# GMM prediction
#---  ---  ---  ---  ---  ---  ---  ---
#mean prediction
y_mu  = (coeff_0_mu + coeff_1e_mu + coeff_1as_mu + coeff_1bs_mu +
         coeff_2p_mu*x_2 + coeff_3s_mu*x_3[sta_inv] + cells_LcA_mu)

#compute residuals
res_tot     = y_data - y_mu
#residuals computed directly from stan regression
res_between = [df_stan_posterior_raw.loc[:,f'dB.{k}'].mean()
               for k in range(n_eq)]
res_between = np.array([res_between[k] for k in (eq_inv).astype(int)])
res_within  = res_tot - res_between

#summary predictions and residuals
```

```
predict_summary = np.vstack((y_mu, res_tot, res_between, res_within)).T
columns_names = ['nerg_mu','res_tot','res_between','res_within']
df_predict_summary = pd.DataFrame(predict_summary, columns = columns_names,
                                  index=df_flatfile.index)
#create dataframe with predictions and residuals
df_predict_summary = pd.merge(df_flatinfo, df_predict_summary, how='right',
                              left_index=True, right_index=True)
df_predict_summary[['eqid','ssn']] =
                            df_predict_summary[['eqid','ssn']].astype(int)
df_predict_summary.to_csv(out_dir  + out_fname + '_stan_residuals' + '.csv',
                          index=True)
```

Listing 6.16. Post-processing section of type-3 NGMM STAN regression file. Ground motion summary.

7 DEVELOPMENT OF NON-ERGODIC GROUND MOTION MODELS WITH R-INLA COMPUTER PLATFORM

The Integrated Nested Laplace Approximation (INLA) is an efficient method for approximating posterior distributions in Bayesian inference, providing an alternative to other methods such as MCMC. R-INLA (Rue et al., 2009) is a package that has implemented this method for Latent Gaussian Models (LGMs) in the open-source statistical software platform R (R Core Team, 2021). Non-ergodic GMMs belong to this family of models as the total non-ergodic effects can be expressed as the sum of linear additive terms ($\delta L2L$, $\delta S2S$, and $\delta P2P$), which follow Gaussian distributions. Additionally, to efficiently approximate the Gaussian random fields of the spatially varying terms, INLA expresses them as stochastic partial differential equations (SPDEs, Lindgren et al., 2011) and evaluates them using the finite element method.

For more details on spatial modeling in R-INLA, see also Lindgren and Rue (2015) and Lindgren et al. (2021), as well as Franco-Villoria et al. (2019). For more information on INLA, see also https://www.r-inla.org/, which provides general information, installation instructions, links to papers/tutorials, as well as to a google group.

7.1 GENERAL OVERVIEW OF R-INLA FUNCTIONS

This section is intended to provide a general overview of the INLA functionality and commands before presenting the regression codes for the development of the non-ergodic GMMs type-1 to type-3 in Section 7.2.

7.1.1 Regression Formula

The syntax for fitting regression models with INLA is similar to the syntax for fitting regression models with the `lm()` function. A regression formula (`formula`) is used to define the structure of the statistical model. The response variable is on the left-hand side of $\sim$, and the fixed effects, random effects, and spatially varying coefficients are on the right-hand side. Similar to the formula in `lm()`, adding `0` in the right-hand side removes the intercept from the model. The regression formula for the type-1 regression is:

```
form_inla_spatial <- y ~ 0 + intercept  +
                           f(eq, model="iid", hyper=prior_tau_0) +
                           f(sta, model="iid", hyper=prior_omega_1bs) +
                           f(idx.eq, model = spde_eq) +
                           f(idx.sta, model = spde_sta)
```

In this example, the intercept is first removed with `0` and then added back with `intercept`. By doing so, the default priors for the intercept will be the same as those for the other fixed effects. Subsections 7.1.2 to 7.1.7 provide further details and examples regarding the specification of prior distributions, random effects, spatially varying effects, cell-specific anelastic attenuation, organization of the regression data, and regression function.

The interested reader is also referred to Gómez-Rubio (2020); Krainski et al. (2019, 2021); Moraga (2019) for additional information regarding GP regression with INLA and examples of spatial models.

7.1.2 Specification of Prior Distributions

In setting the prior distributions in INLA, it is important to know how the hyper-parameters are internally represented. For instance, the standard deviations of Normal distributions are represented internally by the precision, the reciprocal of the squared standard deviation, in log-scale ($log(1/\sigma^2)$). Information on the internal representation of the hyper-parameters of the different likelihood functions and distributions can be accessed with `inla.doc('name')` where `'name'` is a string with the name of the likelihood function of the distribution.

The list of all implemented prior distributions can be seen by typing `names(inla.models()$prior)`. By default, the intercept of the model is assigned a Gaussian prior distribution with zero mean and zero precision, and the remaining fixed effects are assigned Gaussian prior distributions with zero mean and 0.01 precision. The values of these priors can be changed through the `control.fixed` argument in `inla()`.

```
  prior_fixed <- list(mean.intercept = inter_mean,
                      prec.intercept = inter_prec,
                      mean = (list(x1=x1_mean, default=def_mean)),
                      prec = (list(x1=x1_prec, default=def_prec)) )
  inla_fit <- inla(y~x1+x2+x3, data = data,
                   control.fixed = prior_fixed))
```

The list `prior_fixed` contains the information regarding the prior distributions of the fixed effects which is passed to `inla()`. `inter_mean` and `inter_prec` are the mean and log precision values of the Gaussian prior distribution for the intercept, `x1_mean` and `x1_prec` are the mean and log precision values of the Gaussian prior distribution for `x_1`, and `def_mean` and `def_prec` update the mean and log precision default values for the Gaussian prior distribution of the remaining covariates not assigned a prior distribution individually.

Multi-level prior distributions can be defined by specifying the higher-level distribution in the `control.fixed` arguments. In this case, the parameters of the lower-level distribution are not fixed but vary based on the upper-level distribution. In the following case, the log precision of the intercept is assigned a log-gamma distribution with 0.8 and 0.5 parameters.

```
prior_inter_prec <- list(prec = list(prior = "loggamma",
                                     param = c(0.8, 0.5)))

prior_fixed <- list(prec.intercept = prior_inter_prec)

inla_fit <- inla(y~x1+x2+x3, data = data,
                 control.fixed = prior_fixed))
```

Commonly used prior distributions in INLA for the development of NGMMS are the log-gamma distribution, the penalized-complexity prior for precision, and the penalized-complexity joint prior for correlation length and scale of a Matérn kernel function.

Log-gamma distribution:

The gamma distribution is often used as the prior distribution for the precision parameter of a normal distribution in Bayesian statistics. The Gamma distribution is the conjugate prior, meaning that the posterior distribution also follows a Gamma distribution. Assuming a Gamma prior for the precision means an inverse-Gamma distribution as the prior for the variance. The inverse-Gamma distribution has the majority of its mass away from zero making it a suitable prior when the existence of the model effects and the range of their standard deviation are known form previous studies. Random effects that fall into this group are the between-event and within-event aleatory terms and the site-independent constant site term. In all these cases, there is an abundance of previous literature demonstrating the existence of these effects in ground motion and reporting the range of typical values. Internally, INLA uses the log precision; thus, assigning a log-gamma prior distribution implies a prior gamma distribution for the precision (τ). The probability density function of a gamma distribution is given by:

$$\pi(\tau) = \frac{b^{\alpha}}{\Gamma(\alpha)} \tau^{\alpha-1} exp(-b\tau) \tag{7.1}$$

where $\alpha > 0$ is the shape parameter and $b > 0$ is the inverse-scale parameter. The mean of the distribution is equal to α/b, and the variance is equal to α/b^2.

Penalized-complexity prior:

The penalized-complexity prior for the precision is used to model a random effect that is unknown whether it exists if it has an effect the median ground motion. In this case, if the data do not support the existence of the random effect, the penalized-complexity prior promotes a posterior distribution that has most of its mass near zero when looking at the standard deviation or at very large values, when looking at the precision, to avoid overfitting. When there is significant evidence in the data, the mass of the posterior distribution deviates from zero, when looking at standard deviation, to allow accepting a wide range of random effect values.

The probability density function for the penalized complexity prior is:

$$\pi(\tau) = \frac{\lambda}{2} \tau^{-3/2} \exp\left(\lambda \tau^{-1/2}\right), \quad \tau > 0 \tag{7.2}$$

where $\lambda > 0$ is the distribution parameter. Internally, the precision penalized-complexity prior is transformed for log-precision to be conistent with the INLA internal representation. This distribution has more mass at large values of precision τ, and small values of standard deviation $\sigma = 1/\sqrt{\tau}$, for penalizing the unnecessary model complexity. For a given α and u, the parameter λ is internally calculated by:

$$\lambda = -\frac{\ln(\alpha)}{u} \tag{7.3}$$

such that

$$P(\sigma > u) = \alpha \, . \tag{7.4}$$

For example, the prior distribution for the precision of cell-specific anelastic attenuation is defined as:

```
prior_omega_cap <- list(prec=list(prior='pc.prec', param=c(0.01, 0.02)))
```

where $P(\omega_{ca,p} > 0.01) = 0.02$.

Penalized-complexity joint prior:

For spatially varying coefficients, A Matérn kernel function with a joint penalized-complexity prior for the range and scale is assigned to avoid overfitting. Model complexity is penalized by shrinking the scale (σ) towards zero and pushing the range (ρ) towards infinity which implies very small spatially varying effects (Fuglstad et al., 2019). Further information on the parameters of the Matérn kernel functions is provided in Section 7.1.4. The joint penalized-complexity prior is given by:

$$\pi(\rho, \sigma) = \frac{d}{2} \left(R\rho^{-1-d/2} \exp(-R\rho^{-d/2}) \right) \ (S \exp(-S\sigma)) \tag{7.5}$$

where R and S are the distribution parameters. Internally, R is defined by the probability (p_ρ) that ρ is smaller than the value ρ_0:

$$P(\rho < \rho_0) = p_\rho \tag{7.6}$$

and S is defined by the probability (p_σ) that σ is greater than the value σ_0:

$$P(\sigma > \sigma_0) = p_\sigma \tag{7.7}$$

7.1.3 Specification of Random Effects

Random effects are specified in INLA with the `f(i, model, hyper)` function. The first argument (`i`) is an index vector that is used to assign the observations into groups, the second argument (`model`) is the name of the model distribution, and the third optional argument (`hyper`) is for defining the prior distribution for the parameters of the model. Additional information on the options and arguments can be found with the `?f` command. For GMM development, the random effects can be modeled with `model="iid"` which creates independent and identically distributed random variables assigned to the observations based on the group vector. For example, the between-event residuals (δB^0_{es}) can be included in the model formula with:

```
  prior_tau_0 <- list(prec = list(prior = "loggamma", param = c(4.0, 0.5)))

  form_inla <- ... + f(eq, model="iid", hyper=prior_tau_0) + ...
```

where `eq` is the column in `data` that contains the event IDs, and `prior_tau_0` is a list with the prior distribution information for δB^0_{es}. The `...` denote the other fixed and random effects of the GMM functional form, which not important for this example, are omitted.

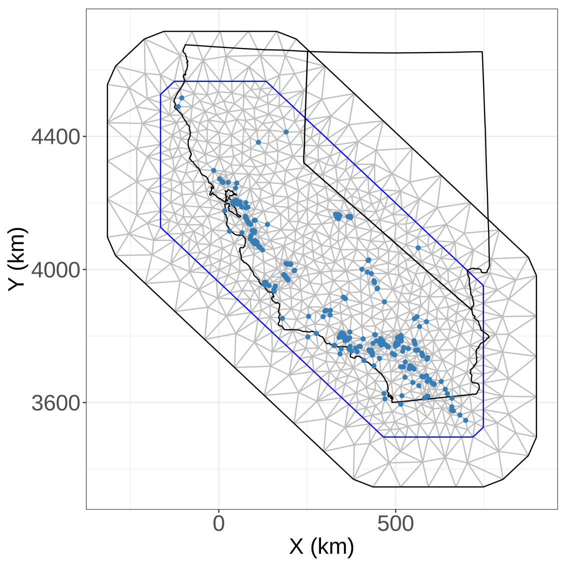


Figure 7.1. SPDE mesh for spatially varying earthquake term

7.1.4 Specification of Spatially Varying Coefficients

A Gaussian random field (GRF) can be used to model the spatially varying effects, such as the spatially varying earthquake constant, site constant, and V_{S30} scaling. In INLA, GRF are modeled as stochastic partial differential equations (SPDE) and are solved using the finite element method. For that, the first step in this approach is to create a 2D triangulated mesh of the region of study using the function:

```
    inla.mesh.2d(loc, max.edge, cutoff, offset)
```

where `loc` is a 2D array with the coordinates of interest, `cutoff` is the minimum allowed distance between triangle vertices, `offset` is the extension distance in the inner and outer regions, and `max.edge` is the maximum edge size for the inner and outer triangles. For example, the following command creates the mesh shown in Figure 7.1 for the events in NGAWest 2.

```
  #create earthquake mesh
  mesh_eq <- inla.mesh.2d(loc=as.matrix(X_eq),
                          max.edge = c(50,250),
                          cutoff = 3, offset = c(50, 150))

  #plotting mesh
  pl_eq_mesh  <- ggplot() + theme_bw() + gg(mesh_eq) +
                 geom_path(data=map_ca, aes(x=X,y=Y), color='black') +
                 geom_path(data=map_nv, aes(x=X,y=Y), color='black') +
                 geom_point(data=X_eq, aes(x=eqX,y=eqY), color=set1[2]) +
                 labs(x="X (km)", y="Y (km)")
  print(pl_eq_mesh)
```

The spatial correlation of the non-ergodic coefficients is modeled with the aid of a kernel function. In INLA, the SPDE model for a Matérn kernel function (Equation 7.8) is created with the `inla.spde2.pcmatern(mesh, alpha)`, where `mesh` is a triangulated mesh for the domain of the spatially varying coefficient, and `alpha` is a parameter controlling the smoothness of the process (α). The spatial length scale and size of the kernel function are controlled by κ and σ, respectively. Internaly, the correlation lenght is calculated as $r = \sqrt{2\nu}/\kappa$.

$$\boldsymbol{\kappa}(\vec{t}, \vec{t}') = \frac{\sigma^2}{2^{\nu-1}\Gamma(\nu)} \left(\kappa||\vec{t} - \vec{t}'||\right)^{\nu} K(\kappa||\vec{t} - \vec{t}'||) \tag{7.8}$$

The smoothness parameter ν of the Matérn kernel function is related to α through:

$$\nu = \alpha - \frac{d}{2} \tag{7.9}$$

where d is the dimension of the spatial domain.

Optional arguments of `inla.spde2.pcmatern()` include the definition of prior distributions for the correlation length (r) and scale (σ), which are defined by joint penalized-complexity prior to avoid overfitting. The prior distribution of the correlation length is defined as `prior.range = c(range_0, p_range_0)` such that $P(r < r_0) = p_{r_0}$, and the prior distribution for the sigma is defined as `prior.sigma =c(sigma_0, p_sigma_0)` such that $P(\sigma > \sigma_0) = p_{\sigma_0}$. For example, the following code excerpt shows the construction of the kernel function for the earthquake constant.

```
spde_eq <- inla.spde2.pcmatern(mesh = mesh_eq, alpha = 2,
                                prior.range = c(100, 0.95),
                                prior.sigma = c(.30, 0.1))
```

Once the kernel function is defined, the indices for the mesh nodes and the projection matrix between the mesh nodes and data coordinates can be generated. The index structure of the mesh nodes is obtained by `inla.spde.make.index(name, spde$n.spde)` where `name` is the name of the spatially varying effect, and `spde$n.spde` is the number of nodes. The projection matrix is used to approximate the kernel function at the data location based on the kernel function at the mesh modes. It is created by `inla.spde.make.A(mesh, loc, weights=NULL)` where `mesh` is the INLA mesh for spatially varying effect of interest, and `loc` are the coordinates of the data points. The optional argument `weights` is used to model the non-ergodic coefficients with non-constant covariance values, such as the geometrical spreading and V_{S30} coefficients whose effects scale with $f_{gs}(R_{rup}, M)$ and $f_{V_{S30}}(V_{S30})$, respectively. The commands to generate indices and projection matrix for the earthquake constant are:

```
A_eq    <- inla.spde.make.A(mesh_eq, loc = as.matrix(X_eq_all))
idx.eq  <- inla.spde.make.index("idx.eq",spde_eq$n.spde)
```

The effects of the spatially varying coefficients are included in the regression model by adding `f(idx, model)` in the model formula (`forumla`) argument of `inla()`. `idx` is the index structure of the mesh, and `model` is the SPDE object of the spatially varying coefficient. As an example, the next line of code shows how the spatially varying earthquake constant is included in the GMM functional form:

```
form_inla <- ... + f(idx.eq, model = spde_eq) + ...
```

where `idx.eq` are the indices of the earthquake mesh and `spde_eq` is the SDPE for the earthquake constant kernel function.

The previous discussion covers the modeling of the spatially varying coefficients with zero mean. That is typically sufficient for the earthquake and site constants where in the absence of data revert to zero, implying ergodic scaling. However, when modeling the geometric spreading or the V_{S30} coefficient as a spatially varying term, a non-zero mean should be allowed. In INLA, this is achieved by adding an extra fixed effect term in the regression formula to capture the average scaling. For instance, the next piece of code shows the modeling of the V_{S30} constant as a spatially varying term:

```
spde_sta_vs30   <- inla.spde2.pcmatern(mesh = mesh, alpha = alpha,
                                       prior.range = c(100, 0.95),
                                       prior.sigma = c(.40, 0.1))

A_sta_vs30     <- inla.spde.make.A(mesh,
                                  loc=as.matrix(X_sta_all), weights=x_3)
idx.sta_vs30   <- inla.spde.make.index("idx.sta_vs30",spde_sta_vs30$n.spde)
 form_inla <-  ... + x3 + f(idx.sta_vs30,  model = spde_sta_vs30) ...
```

where `spde_sta_vs30` is the SPDE for the spatially varying component of the V_{S30} coefficient, `A_sta_vs30` creates the projection matrix for the V_{S30} kernel function approximations at the station locations, `weights = x_3` specifies the scaling for the V_{S30} coefficient, `x_3` is the value of the V_{S30} scaling ($f_{V_{S30}}(V_{S30})$) at the station locations, and `idx.sta_vs30` are the indices of the V_{S30} SPDE mesh. The non-zero mean scaling is modeled by adding x_3 in the functional form (`form_inla`), and the spatially varying effect is modeled by adding `f(idx.sta_vs30, model = spde_sta_vs30)`.

7.1.5 Specification of Anelastic Attenuation

The effects of the cell-specific anelastic attenuation can be incorporated through the `z` latent model. The `z` model is used to define random effects of the form $\boldsymbol{Z}\ u$ with $\boldsymbol{Z}$ being a design matrix and u a vector of iid random variables with zero mean and τ precision. It is defined as `f(id, model='z', Z=Z)`, where `id` is the group index vector, and `Z` is the design matrix. The optional argument `hyper` is used to set the prior distributions for the mean and precision. In the case of the cell-specific anelastic attenuation, u is the cell-specific anelastic adjustment to the mean anelastic attenuation, and $\boldsymbol{Z}$ corresponds to the N_{gm} by N_{cell} cell-path-distance matrix (Section 4.2). N_{gm} is the number of ground-motion observation, and N_{cell} is the number of cells. The mean effect of the anelastic attenuation is captured by adding `R`, the linear distance scaling, in `form_inla`. The modeling of the cell-specific anelastic attenuation is shown below:

```
df_inla_covar$idx_cell <- 1:nrow(df_inla_covar)

RC_sparse <- as(RC ,"dgCMatrix") #sparse  matrix
prior_prec_cell <- list(prec = list(prior = 'pc.prec',
                                   param = c(0.01, 0.01)))

form_inla <- ... + R + f(idx_cell, model = "z",
                          Z = RC_sparse, hyper = prior_prec_cell) + ...
```

`idx_cell` is the column name in the input data frame corresponding to the cell indices. It is an integer vector ranging from 1 to N_{gm} but only the first N_{cell} elements are used. `RC` is the cell path distance matrix. Since the majority of the elements are equal to zero, it is transformed to a compressed sparse form to reduce the matrix-vector multiplication computational cost. A penalized prior distribution (`list(prior = 'pc.prec', param = c(0.01, 0.01))`) is used for the precision of anelastic attenuation adjustments ($P(\omega_{ca,p} > 0.01) = 0.01$); see Section 7.1.2 for further information on the definition of the priors.

7.1.6 Organization of Input Data

The `inla.stack(tag, data, effects, A)` function is used to organize all input data, including indices, covariates for the fixed and random effects, and projection matrices for the spatially varying terms.

- `tag` is a character string identifying the data
- `data` is a list that contains a vector with the observations
- `effects` is a list with the data-frame of the covariates for the fixed effects and mesh indices for the spatially varying terms
- `A` is a list with all the projection matrices provided in the same order as `effects`

The example below shows the regression formula and the stack for the type-1 regression.

```
#covariates
df_inla_covar <- data.frame(intercept = 1, eq = eq_id, sta = sta_id)

#regression formula
form_inla_spatial <- y ~ 0 + intercept  +
                         f(eq, model="iid", hyper=prior_tau_0) +
                         f(sta, model="iid", hyper=prior_omega_1bs) +
                         f(idx.eq, model = spde_eq) +
                         f(idx.sta, model = spde_sta)

#build stack
stk_inla_spatial <- inla.stack(tag = 'model_inla_spatial',
                              data = list(y = y_data),
                              effects = list(df_inla_covar,
                                             idx.eq = idx.eq,
                                             idx.sta = idx.sta),
                              A = list(1, A_eq, A_sta))
```

In the `data` list, `y` corresponds to the name of the prediction variable in the regression formula, and `y_data` is a vector containing the total regression residuals (Section 3.1). In the `effects` list, the data-frame `df_inla_covar` contains the covariates for the fixed effects and the group indices for the events and stations for the random effects. In this example, `df_inla_covar` has three columns, `intercept` is a column of ones, that is the covariate for the regression intercept, `eq` is a column with the event ids, ground-motions with the save event id correspond to the same earthquake, and `sta` is a column with the station ids, ground-motions with same station id were recorded at the same station. `idx.eq` are the mesh node indices for the spatially varying earthquake term, and `idx.sta` are the mesh node indices for the spatially varying station term. It should be noted that the column names in the covariates data frame and list names for the mesh

Table 7.1. Available assessment criteria for control.compute

Criterion	Option
Marginal likelihood	cpo
Conditional predictive ordinate	cpo
Predictive integral transform	pit
Deviance information criterion	dic
Widely applicable Bayesian information criterion	waic

node indices must be the same as the names of the coefficient and effects in form_inla_spatial. The first element of A is 1 as the fixed and random effects are already in the correct scale – scale factor is unity. The second and third elements of A are the projection matrices for the spatially varying earthquake and site terms which are A_eq and A_sta, respectively.

7.1.7 Inla Regression Function

The function inla() is used to fit the model. The main input arguments are:

- formula: the functional of the regression model similar to lm(). The prediction variable is on the left-hand side of $\sim$, while the fixed and the random effects are defined on the right hand side of $\sim$
- data: the list data that will be used for fitting the model (Section 7.1.6)
- family: the distribution family for the likelihood function. Implemented likelihoods include "gaussian", "poisson", and "binomial". The "gaussian" likelihood is typically used for GMMs
- control.family: list of specifications for the prior distributions for the hyper-parameters of the likelihood function (Section 7.1.2)
- control.fixed: list of specifications for the prior distributions of the hyper-parameters of the fixed terms in formula
- control.predictor: list of specifications for computing variables such as the projection matrices
- control.compute: list of specifications for enabling various assessment criteria. Table 7.1 summarize the available assessment criteria.
- control.inla: list of specifications used by INLA to approximate the posterior distributions and hyper-parameters. The most commonly used options are: strategy which controls the approximation method and int.strategy which controls the integration method. Table 7.2 summarizes the available approximation options, and Table 7.3 summarizes the available integration options.

As an example, the following piece of code is showing the model fit for the type-1 NGMM.

```
fit_inla_spatial <- inla(form_inla_spatial,
                    data = inla.stack.data(stk_inla_spatial),
```

Table 7.2. Approximation options for strategy

Approximation Method	Option
Simplified Laplace Approximation, default	simplified.laplace
Adaptive approximation	adaptive
Gaussian approximation	gaussian
Laplace approximation	laplace

Table 7.3. Integration options for int.strategy

Integration Method	Option
Central composite design, default (Box and Draper, 2007)	ccd
Numerical integration	grid
Empirical Bayes integration (Carlin and Louis, 2008)	eb

```
                family = "gaussian",
                control.family = list(hyper=list(prec=prior_phi_0)),
                control.fixed = prior_fixed,
                control.predictor = list(A=inla.stack.A(stk_inla_spatial)),
                control.compute = list(dic = TRUE, cpo = TRUE, waic = TRUE),
                control.inla = list(int.strategy='eb', strategy="gaussian"),
                verbose=TRUE, num.threads=n_threads )
```

form_inla_spatial is the regression functional from. The function inla.stack.data() extracts the data from the INLA stack (defined in Section 7.1.6). A Gaussian distribution is assigned to the likelihood function. For modeling the covariance in INLA, the between-event and within-event residuals are modeled separately as opposed to Abrahamson and Youngs (1992) where they are both included in the covariance of the likelihood function. Here, the between-event residual is included in form_inla_spatial as a random effect that is grouped by the event id, and within-event residual is modeled by the likelihood function as a Gaussian iid random variable with ϕ_0 standard deviation. The control.family defines the prior distribution for ϕ_0 in terms of the log precision. The prior distributions of the GMM fixed effects are defined in control.fixed; prior_fixed is a list with the GMM fixed effects prior distributions as defined in Section 7.1.2. The function inla.stack.A() extracts the projection matrices from the INLA stack. The list assigned to control.compute defines the various assessment criteria to be calculated; in this example, it includes the Deviance information criterion, the Conditional predictive ordinate, and the widely applicable Bayesian information criterion. The verbose option outputs the current state of the optimization function as it progresses, and num.threads specifies the maximum number of CPU threads to be utilized by INLA during the parallel processing steps.

7.2 R-INLA REGRESSION FOR NON-ERGODIC GMMS

Similar to the review of the STAN regression files in Chapter 6, the R-INLA regression code for the type-1, type-2 and type 3 NGMMs are presented in Section 7.2.1), Section 7.2.2, and Section 7.2.3, respectively.

All INLA regression files are divided into three sections:

- pre-processing summarizes the input arguments for regression,
- regression defines the form of the problem, specifies the posterior distributions, and performs the INLA regression; and
- post-processing extracts and saves the posterior distributions of the non-ergodic coefficients and hyperparameters.

7.2.1 R-INLA regression function for Type-1 NGMM

The RunINLA function implemented in regression_inla_model1_unbounded_hyp.R located in the ngmm_tools/Analyses/R_lib/regression/inla subdirectory of the Github repository is used to run the GP regression of the type-1 NGMM (Section 3.1).

Pre-processing:

The mandatory input arguments for RunINLA are as follows.

- df_flatfile is the ground-motion regression data frame that contains the source information (event ID and earthquake coordinates), site information (station ID and site coordinates), and total regression residuals for every ground motion record. Details on the structure of the ground-motion flatfile are provided in Section 4.1.
- out_fname defines the names of the output files.
- out_dir specifies the location of the output directory.

The optional input arguments for RunINLA are as follows.

- res_name defines the column name in df_flatfile for the total regression residuals, the default column name is 'tot'.
- alpha controls the smoothness of the Matérn kernel function for the spatially varying coefficients (Section 7.1.4), the default value is 2.
- mesh_edge_max sets the maximum length of the triangle sides composing the mesh for the spatially varying coefficients. The maximum edge length for the extension zone is five times the maximum edge length of the inner zone. The default value is 15.
- mesh_inner_offset defines the extent of the inner zone around the events and stations. The default value is 15.
- mesh_outer_offset defines the size of the extension zone around the inner zone. The default value is 50.
- flag_gp_approx dictates whether the more general Laplace approximation (flag_gp_approx=FALSE) or the computationally faster Gaussian approximation (flag_gp_approx=TRUE) is used. The default option is TRUE.

- `n_threads` is the maximum number of CPU cores used in the parallel processing steps in INLA. Consider reducing `n_threads` if INLA crashes due to an out-of-memory error. The default option is the total number of CPU cores.
- `runinla_flag` is the flag for running the INLA regression (`runinla_flag=TRUE`) or loading and processing the raw output from a previous regression (`runinla_flag=FALSE`). The default option is TRUE.

Listing 7.1 summarizes the source, site, and total residual information. `n_data` is the number of the ground-motion data. `data_eq_all` contains the source information for all records; the first column is the event id, the second column is the magnitude, and the third and fourth columns are the hypo-center coordinates in UTM. `eq_idx` contains the row indices of `data_eq_all` corresponding to unique event ids, and `eq_inv` contains the indices to reconstruct the original array from the unique values. `data_eq` contains the unique rows of `data_eq_all`, constructed based on `eq_idx`. `X_eq` includes the hypo-center UTM coordinates for all unique events, and `X_eq_all` includes the hypo-center UTM coordinates for all records. `n_eq` is the number of events.

Equivalently, `data_sta_all` contains the site information for all stations; the first column is the station id, the second column is the V_{S30}, and the third and fourth columns are the station UTM coordinates. `sta_idx` contains the row indices of `data_sta_all` corresponding to unique station ids, and `sta_inv` contains the indices to reconstruct the original array from the unique values. `data_sta` contains the unique site information for all stations. `X_sta` holds the UTM coordinates for all unique stations, and `X_sta_all` holds the station UTM coordinates for all records. `n_sta` is the number of stations.

`y_data` contains the total ergodic residuals for all ground motions. `utm_zone` is the UTM zone used for the transformation between UTM coordinates and Latitude and Longitude coordinates for the events and stations, and `utm_no` is the UTM zone number.

```r
# Preprocess Input Data
# ---------------------------
n_data <- nrow(df_flatfile)
#earthquake data
data_eq_all <- df_flatfile[,c('eqid','mag','eqX', 'eqY')]
out_unq  <- UniqueIdxInv(df_flatfile[,'eqid'])
eq_idx   <- out_unq$idx
eq_inv   <- out_unq$inv
data_eq  <- data_eq_all[eq_idx,]
X_eq     <- data_eq[,c(3,4)]
X_eq_all <- data_eq_all[,c(3,4)]
#create earthquake ids for all records (1 to n_eq)
eq_id <- eq_inv
n_eq  <- nrow(data_eq)

#station data
data_sta_all <- df_flatfile[,c('ssn','Vs30','staX','staY')]
out_unq   <- UniqueIdxInv(df_flatfile[,'ssn'])
sta_idx   <- out_unq$idx
sta_inv   <- out_unq$inv
data_sta  <- data_sta_all[sta_idx,]
```

```
X_sta      <- data_sta[,c(3,4)]
X_sta_all  <- data_sta_all[,c(3,4)]
#create station indices for all records (1 to n_sta)
sta_id <- sta_inv
n_sta  <- nrow(data_sta)

#ground-motion observations
y_data <- df_flatfile[,res_name]

#UTM zone
utm_zone <- unique(df_flatfile$UTMzone)
utm_no   <- as.numeric(gsub("([0-9]+).*$", "\\1", utm_zone))
```

Listing 7.1. Preprocessing section of type-1 NGMM INLA regression file.

Regression:

The regression section includes the definition of the prior distributions for the fixed effects (Listing 7.2), the development of the mesh and definition of the kernel functions for the spatially varying coefficients (Listing 7.3), the definition of the prior distributions for the aleatory terms (Listing 7.4), the formulation of the NGMM functional form (Listing 7.5), and the INLA regression (Listing 7.6).

In Listing 7.2, prior_fixed specifies the prior distributions for all fixed effects. In particular, it assigns a normal prior distribution to the model intercept (intcp) with a zero mean and 0.2 variance (5 precision). df_inla_covar contains all the group indices and coefficient covariates used in the regression model; intrcp is the intercept covariate which is a vector of ones, eq contains the event ids for all ground motions, and sta contains the station ids for all ground motions.

```
# Run INLA, fit model
# ---------------------------
#fixed effects
#---   ---   ---   ---   ---   ---
#prior on the fixed effects
prior_fixed <- list(mean.intercept = 0, prec.intercept = 5,
                    mean = (list(intcp=0.0, default=0)),
                    prec = (list(intcp=5.0, default=0.01)))

#covariates
df_inla_covar <- data.frame(intcp = 1, eq = eq_id, sta = sta_id)
```

Listing 7.2. Regression Section of type-1 NGMM INLA regression file. Definition of fixed-effects.

Next, Listing 7.3 develops the mesh for the spatially varying event and station terms, defines the kernel functions for these terms, and sets prior distributions for the hyper-parameters of the kernel functions. inla.mesh.2d() creates a mesh, which encompasses all events and stations (X_eq and X_sta), and stores it in mesh. The SPDE for the $\delta c_{1,E}$ kernel function is stored in spde_eq. The correlation length ($\ell_{1,E}$) and scale ($\omega_{1,E}$) are assigned a joint penalized-complexity prior distribution such that there is a 95% probability that the correlation length of $\delta c_{1,E}$ will be less than 100 km ($P(\ell_{1,E} < 100) = 0.95$) and 10% probability that the scale of $\delta c_{1,E}$ will be greater than 0.3 ($P(\omega_{1,E} > 0.3) = 0.1$). Similarly, the SPDE for the $\delta c_{1a,S}$

kernel function is defined in spde_sta. The correlation length ($\ell_{1a,S}$) and scale ($\omega_{1a,S}$) are assigned a joint penalized-complexity prior distribution with $P(\ell_{1a,S} < 100) = 0.95$ and $P(\omega_{1a,S} > 0.4) = 0.1$. inla.spde.make.A(mesh, loc = as.matrix(X_eq_all)) generates and stores in A_eq the projection matrix for $\delta c_{1,E}$ for the event locations of all ground motions, and inla.spde.make.index("idx.eq",spde_eq$n.spde) generates and stores in idx.eq the indices of the mesh for the SPDE for the $\delta c_{1,E}$ kernel function. Correspondingly, A_sta holds the projection matrix for $\delta c_{1a,S}$ for all stations, and idx.sta holds the indices of the mesh for the SPDE for the $\delta c_{1a,S}$ kernel function.

The spatially independent site term ($\delta_{1b,S}$) is modeled with a normal distribution with zero mean and $\omega_{1b,S}$ standard deviation. The prior of $\omega_{1b,S}$ is parameterized through the log-precision ($\log(1/\omega_{1b,S}^2)$) which is assigned a log-gamma distribution with a 0.8 shape and 0.5 rate parameter in prior_omega_1bs.

```
#spatial model
#---   ---   ---   ---   ---   ---
#input arguments
edge_max     <- mesh_edge_max
inner_offset <- mesh_inner_offset
outer_offset <- mesh_outer_offset

#domain mesh
mesh <- inla.mesh.2d(loc=rbind(as.matrix(X_eq),as.matrix(X_sta)) ,
                     max.edge = c(1,5)*edge_max,
                     cutoff = 3, offset = c(inner_offset , outer_offset))

#prior distributions
#spde earthquake prior
spde_eq <- inla.spde2.pcmatern(mesh = mesh, alpha = alpha,
                               prior.range = c(100, 0.95),
                               prior.sigma = c(.30, 0.1))
#spde station prior
spde_sta <- inla.spde2.pcmatern(mesh = mesh, alpha = alpha,
                                prior.range = c(100, 0.95),
                                prior.sigma = c(.40, 0.1))

A_eq    <- inla.spde.make.A(mesh, loc = as.matrix(X_eq_all))
idx.eq  <- inla.spde.make.index("idx.eq",spde_eq$n.spde)
A_sta   <- inla.spde.make.A(mesh, loc = as.matrix(X_sta_all))
idx.sta <- inla.spde.make.index("idx.sta",spde_sta$n.spde)

#site independent term
prior_omega_1bs <- list(prec = list(prior = "loggamma",
                                    param = c(0.8, 0.5)))
```

Listing 7.3. Regression section of type-1 NGMM INLA regression file. Definition of spatially varying terms.

In Listing 7.4 the standard deviation of the within-event residuals (ϕ_0) is parameterized through the log-precision ($\log(1/\phi_0^2)$) and is assigned a log-gamma distribution with a 5.0 shape and 0.5 rate parameter in prior_phi_0. Similarly, the log-precision of the between-event residual ($\log(1/\tau_0^2)$) is assigned a log-gamma distribution with a 4.0 shape and 0.5 rate parameter in prior_tau_0.

```
#aleatory terms
#---  ---  ---  ---  ---  ---
#prior distributions
prior_phi_0 <- list(prec = list(prior = "loggamma", param = c(5.0, 0.5)))
prior_tau_0 <- list(prec = list(prior = "loggamma", param = c(4.0, 0.5)))
```

Listing 7.4. Regression section of type-1 NGMM INLA regression. Definition of aleatory prior distributions.

The `form_inla_spatial` spells out functional form for the mean for the type-1 NGMM regression. That is, the expected value of the regression residual (y) is equal to the sum of the intercept (`intcp`), spatially varying earthquake term (`f(idx.eq, model = spde_eq)`), spatially varying site term (`f(idx.sta, model = spde_sta)`), spatially independent site term (`f(sta, model="iid", hyper=prior_omega_1bs)`), and between-event residual (`f(eq, model="iid", hyper=prior_tau_0)`). As mentioned in Section 7.1.7, it is for modeling convenience that the between-event residual is included in the mean function and the within-event residual is modeled in the standard deviation of the likelihood function.

The spatially varying coefficients are defined based on their mesh indices and the SPDE of their kernel function. For example, the spatially varying earthquake term is defined by the event mesh indices `idx.ed` and the SPDE for the $\delta_{1,E}$ kernel function `spde_eq`. Similarly, the random effects, such as $\delta_{1b,S}$ and δB, are specified based on the column name in `df_inla_covar` (Listing 7.2) for their group indices and the `iid` designation for the `model` argument. The `hyper` argument sets the prior distribution which is specified in Listing 7.3 for $\delta c_{1b,S}$ and in Listing 7.4 for δB.

The `stk_inla_spatial` encapsulates all data used in INLA regression. The `data` argument specifies the name of regression residuals. The name of the list argument should correspond to the name of the prediction variable in `form_inla_spatial`. The argument `A` is given a list with the projection matrices for the spatially varying earthquake and site constants and the unit scale for the random effects. The argument `effects` includes the mesh indices for the spatially varying terms and group indices for the random effects in the same order specified in `A`.

```
#inla model
#---  ---  ---  ---  ---  ---
#functional form (with spatial var)
form_inla_spatial <- y ~ 0 + intcp +
                          f(idx.eq, model = spde_eq) +
                          f(idx.sta, model = spde_sta) +
                          f(sta, model="iid", hyper=prior_omega_1bs) +
                          f(eq, model="iid", hyper=prior_tau_0)

#build stack
stk_inla_spatial <- inla.stack(data = list(y = y_data),
                               A = list(A_eq, A_sta, 1),
                               effects = list(idx.eq = idx.eq,
                                              idx.sta = idx.sta,
                                              df_inla_covar),
                               tag = 'model_inla_spatial')
```

Listing 7.5. Regression section of type-1 NGMM INLA regression file. Functional form formulation.

The INLA regression is summarized in Listing 7.6. If `flag_gp_approx` is TRUE the empirical Bayes approach and the Gaussian approximation is used in the regression (`control.inla = list(int.strategy='eb', strategy="gaussian")`); while if `flag_gp_approx` is FALSE the full Laplace approximation is used instead. The distribution family of the within-event residuals is specified in `family="gaussian"`, and the prior distribution for the precision of δW is given in `control.family = list(hyper = list(prec = prior_phi_0))`. The regression data are specified in `data = inla.stack.data(stk_inla_spatial)`. The `control.fixed = prior_fixed` assigns the prior distribution for the fixed effects, which, for NGMM-type1, corresponds to the intercept of the model. The projection matrices are passed to the regression through the `control.predictor = list(A = inla.stack.A(stk_inla_spatial))` argument. Once the regression is complete, the regression model is saved in the output directory.

If `runinla_flag` is TRUE, the INLA regression is performed as described in the preceding paragraph, while if `runinla_flag` is FALSE, a previous regression model saved in the output directory is loaded. The second option is useful when new post-processing steps are added because it avoids the computational cost associated with fitting the regression model.

```
#fit inla model
#---   ---   ---   ---   ---   ---
if(runinla_flag){
  #run model (spatial)
  if(flag_gp_approx == TRUE){
    fit_inla_spatial <- inla(form_inla_spatial,
                             data = inla.stack.data(stk_inla_spatial),
                             family = "gaussian",
                             control.family =
                                   list(hyper = list(prec = prior_phi_0)),
                             control.fixed = prior_fixed,
                             control.predictor =
                                   list(A = inla.stack.A(stk_inla_spatial)),
                             control.compute = list(dic = TRUE,
                                                    cpo = TRUE,
                                                    waic = TRUE),
                             control.inla = list(int.strategy='eb',
                                                 strategy="gaussian"),
                             verbose=TRUE, num.threads=n_threads)
  }else{
    fit_inla_spatial <- inla(form_inla_spatial,
                             data = inla.stack.data(stk_inla_spatial),
                             family = "gaussian",
                             control.family =
                                   list(hyper = list(prec = prior_phi_0)),
                             control.fixed = prior_fixed,
                             control.predictor =
                                   list(A = inla.stack.A(stk_inla_spatial)),
                             control.compute = list(dic = TRUE,
                                                    cpo = TRUE,
                                                    waic = TRUE),
                             verbose=TRUE, num.threads=n_threads)
  }
  #save results
  dir.create(out_dir, showWarnings=FALSE, recursive=TRUE)
  save(fit_inla_spatial,
       file=file.path(out_dir,paste0(out_fname,'_inla_fit','.Rdata')) )
}else{
  #load results
  load(file.path(out_dir,paste0(out_fname,'_inla_fit','.Rdata')) )
}
```

Listing 7.6. Regression section of type-1 NGMM INLA regression file. Model Fit.

Post-processing:

The post-processing section summarizes in tables and plots the hyper-parameters' posterior distributions, the non-ergodic coefficients, and the non-ergodic residuals.

Listing 7.7 summarizes in hyp_param the main percentiles (2.5^{th}, 50^{th}, 97.5^{th} percentile), and the mean and standard deviation of the NGMM hyper-parameters. The $\omega_{1b,S}$, ϕ_0, and τ_0, which internally are represented by the precision, are transformed into standard-deviation units.

```
## Post-processing Results
# --------------------------
#hyper-parameters
hyp_param <- data.frame(matrix(ncol = 6, nrow = 0))
colnames(hyp_param) <- colnames(fit_inla_spatial$summary.hyperpar)

hyp_param['dc_0',]     <- fit_inla_spatial$summary.fixed['intcp',]
#correlation lengths of spatial terms
hyp_param['ell_1e',]   <- fit_inla_spatial$summary.hyperpar
                                                    ['Range for idx.eq',]
hyp_param['ell_1as',]  <- fit_inla_spatial$summary.hyperpar
                                                    ['Range for idx.sta',]
#standard deviations of spatial terms
hyp_param['omega_1e',]   <- fit_inla_spatial$summary.hyperpar
                                                    ['Stdev for idx.eq',]
hyp_param['omega_1as',]  <- fit_inla_spatial$summary.hyperpar
                                                    ['Stdev for idx.sta',]
hyp_param['omega_1bs',]  <- 1/sqrt(fit_inla_spatial$summary.hyperpar
                                                    ['Precision for sta',] )
#aleatory terms
hyp_param['phi_0',]  <- 1/sqrt( fit_inla_spatial$summary.hyperpar
                                ['Precision for the Gaussian observations',] )
hyp_param['tau_0',]  <- 1/sqrt( fit_inla_spatial$summary.hyperpar
                                                    ['Precision for eq',] )
#unavailable sd for transformed variables
hyp_param[c('omega_1bs','phi_0','tau_0'),'sd'] <- NA
```

Listing 7.7. Post-processing section of type-1 NGMM INLA regression file. Hyper-parameter summary.

Listing 7.8 includes the calculation of the mean and standard deviation of the non-ergodic coefficients, and the total, within-event, and between-event non-ergodic residuals.

The posterior distributions of the spatially varying terms are contained in the summary.random field of fit_inla_spatial. For example, $\delta\vec{c}_{1,E}$ is extracted at the mesh nodes with coeff_1e <- fit_inla_spatial$summary.random$idx.eq where idx.eq is the mesh index name used for $\delta c_{1,E}$. `coeff_1e$mean` contains the posterior mean values of $\delta\vec{c}_{1,E}$ while `coeff_1e$sd` contains the posterior standard deviation of $\delta\vec{c}_{1,E}$. The projection matrix from the mesh nodes to the event locations is created with `prjct_grid_eq <- inla.mesh.projector(mesh, loc = as.matrix(X_eq));` the `loc` argument specifies the target location which in this case is event coordinates (`X_eq`). The mean and epistemic uncertainty of $\delta\vec{c}_{1,E}$ ($\mu(\delta\vec{c}_{1,E})$ and $\psi(\delta\vec{c}_{1,E})$) is estimated at the event locations in the `coeff_1e_mu <- inla.mesh.project(prjct_grid_eq, coeff_1e$mean)` and `coeff_1e_sig <- inla.mesh.project(prjct_grid_eq, coeff_1e$sd)` lines, respectively. The same process is used for the calculation of $\mu(\delta\vec{c}_{1a,S})$ and $\psi(\delta\vec{c}_{1a,S})$ stored in `coeff_1as_mu` and `coeff_1as_sig`.

The posterior distribution of $\delta c_{1b,S}$ is obtained from the INLA model in `coeff_1bs <- fit_inla_spatial$summary.random$sta`, where `sta` is the name of the random effect in `form_inla_spatial` used for modeling $\delta\vec{c}_{1b,S}$. Since $\delta c_{1b,S}$ is modeled as a random effect group by the station indices, no interpolation is required. The mean and epistemic uncertainty of $\delta\vec{c}_{1b,S}$ ($\mu(\delta\vec{c}_{1b,S})$ and $\psi(\delta\vec{c}_{1b,S})$) is obtained in lines `coeff_1bs_mu <- coeff_1bs$mean` `coeff_1bs_sig <- coeff_1bs$sd`.

The mean adjustment of the NGMM from the ergodic base model is computed in `y_new_mu`. That is the part of the total ergodic residuals treated as a systematic effect in NGMM. `y_new_mu` is equal to the sum of the mean values of δc_0, $\delta\vec{c}_{1a,S}$ and $\delta\vec{c}_{1b,S}$.

The total non-ergodic residuals (`res_tot_mu`) are calculated by subtracting the NGMM mean adjustment from the total ergodic residuals. The between-event non-ergodic residuals are obtained directly form the INLA model as they were modeled as random effect grouped by the event id (`fit_inla_spatial$summary.random$eq$mean[eq_inv]`), `eq` is the name of the δB random term in `form_inla_spatial`. The within-event non-ergodic residuals are computed by subtracting the between-event residuals from the total non-ergodic residuals.

The `df_coeff` dataframe summarizes all information about the non-ergodic coefficients described in the previous paragraphs. The `df_predict_summary` summarizes the mean non-ergodic adjustment and non-ergodic residuals.

```
## Summarize coefficients and residuals
# ---------------------------
#projections
prjct_grid_eq  <- inla.mesh.projector(mesh, loc = as.matrix(X_eq))
prjct_grid_sta <- inla.mesh.projector(mesh, loc = as.matrix(X_sta))

#coefficients
coeff_1e  <- fit_inla_spatial$summary.random$idx.eq
coeff_1as <- fit_inla_spatial$summary.random$idx.sta
coeff_1bs <- fit_inla_spatial$summary.random$sta
#coeff mean and std
coeff_1e_mu   <- inla.mesh.project(prjct_grid_eq,  coeff_1e$mean)
coeff_1e_sig  <- inla.mesh.project(prjct_grid_eq,  coeff_1e$sd)
coeff_1as_mu  <- inla.mesh.project(prjct_grid_sta, coeff_1as$mean)
coeff_1as_sig <- inla.mesh.project(prjct_grid_sta, coeff_1as$sd)
coeff_1bs_mu  <- coeff_1bs$mean
coeff_1bs_sig <- coeff_1bs$sd

#mean prediction
```

```
y_new_mu <- hyp_param['dc_0','mean'] + coeff_1e_mu[eq_inv] +
            coeff_1as_mu[sta_inv] + coeff_1bs_mu[sta_inv]

#residuals
res_tot_mu <- y_data - y_new_mu
res_dB_mu  <- fit_inla_spatial$summary.random$eq$mean[eq_inv]
res_dWS_mu <- res_tot_mu - res_dB_mu

#summary dataframes
df_flatinfo  <- df_flatfile[,c('rsn','eqid','ssn','eqLat','eqLon',
                               'staLat','staLon','eqX','eqY','staX','staY')]

#summary coefficients
df_coeff <- data.frame(rsn=df_flatinfo$rsn,
                       dc_0_mean=hyp_param['dc_0','mean'],
                       dc_1e_mean=coeff_1e_mu[eq_inv],
                       dc_1as_mean=coeff_1as_mu[sta_inv],
                       dc_1bs_mean=coeff_1bs_mu[sta_inv],
                       dc_0_sig=hyp_param['dc_0','sd'],
                       dc_1e_sig=coeff_1e_sig[eq_inv],
                       dc_1as_sig=coeff_1as_sig[sta_inv],
                       dc_1bs_sig=coeff_1bs_sig[sta_inv])
df_coeff <- merge(df_flatinfo, df_coeff, by=c('rsn'))

#summary predictions and residuals
df_predict_summary <- data.frame(rsn=df_flatinfo$rsn, nerg_mu=y_new_mu,
                                 res_tot=res_tot_mu, res_between=res_dB_mu,
                                 res_within=res_dWS_mu)
df_predict_summary <- merge(df_flatinfo, df_predict_summary, by=c('rsn'))
```

Listing 7.8. Post-processing section of type-1 NGMM INLA regression file. Summary of non-ergodic coefficients and residuals.

The code in Listing 7.9 computes the probability density function (PDF) and cumulative density function (CDF) of the marginal posterior distributions of all the hyper-parameters. This step involves first the transformation of the posterior distributions from the internal scale to the linear scale of each hyper-parameter and then the interpolation of the marginal posterior over a regular quantile interval. All marginal distributions, in the internal scale, are stored in `fit_inla_spatial$internal.marginals.hyperpar`.

The calculation of the posterior distribution for the standard deviation of δW^0 (.i.e ϕ_0) is presented as a representative example, but the same procedure applies to all the other hyper-parameters. Internally, the posterior distribution is calculated for the log-precision of δW^0, which is stored in `fit_inla_spatial$internal.marginals.hyperpar [['Log precision for the Gaussian observations']]`. The `inla.tmarginal(fun, marginal)` function is used to transform a marginal posterior distribution from the internal to the desired scale. The `fun` argument defines the transformation function, which for the log-precision to standard deviation transformation is: `function(x) exp(-x/2)`. The PDF for the marginal posterior distribution of ϕ_0 is stored in `post_phi_0`. The `trapz(post_phi_0$x, post_phi_0$y)` normalization ensures that the area under the PDF is unity. The CDF for the ϕ_0 posterior distribution is stored in the `y_int` field of `post_phi_0`. Lastly, the posterior of ϕ_0 is evaluated over a regular quantile interval so that it can be summarized together with the other posterior distributions in the `hyp_posterior` data frame.

```
## Posterior distributions
# ---------------------------
#intercept
post_dc_0 <- as.data.frame(fit_inla_spatial$marginals.fixed$intcp)
#aleatory parameters
post_phi_0 <- as.data.frame(inla.tmarginal(function(x) exp(-x/2),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['Log precision for the Gaussian observations']]))
post_tau_0 <- as.data.frame(inla.tmarginal(function(x) exp(-x/2),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['Log precision for eq']]))
#non-ergodic scales
post_omega_1e  <- as.data.frame(inla.tmarginal(function(x) exp( x),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['log(Stdev) for idx.eq']]))
post_omega_1as <- as.data.frame(inla.tmarginal(function(x) exp( x),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['log(Stdev) for idx.sta']]))
post_omega_1bs <- as.data.frame(inla.tmarginal(function(x) exp(-x/2),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['Log precision for sta']]))
#correlation length
post_ell_1e   <- as.data.frame(inla.tmarginal(function(x) exp( x),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['log(Range) for idx.eq']]))
post_ell_1as  <- as.data.frame(inla.tmarginal(function(x) exp( x),
                   fit_inla_spatial$internal.marginals.hyperpar
                       [['log(Range) for idx.sta']]))

#normalization
post_dc_0$y      <- post_dc_0$y      / trapz(post_dc_0$x, post_dc_0$y)
post_phi_0$y     <- post_phi_0$y     / trapz(post_phi_0$x, post_phi_0$y)
post_tau_0$y     <- post_tau_0$y     / trapz(post_tau_0$x, post_tau_0$y)
post_omega_1e$y  <- post_omega_1e$y  / trapz(post_omega_1e$x,
                                             post_omega_1e$y)
post_omega_1as$y <- post_omega_1as$y / trapz(post_omega_1as$x,
                                             post_omega_1as$y)
post_omega_1bs$y <- post_omega_1bs$y / trapz(post_omega_1bs$x,
                                             post_omega_1bs$y)
post_ell_1e$y    <- post_ell_1e$y    / trapz(post_ell_1e$x,  post_ell_1e$y)
post_ell_1as$y   <- post_ell_1as$y   / trapz(post_ell_1as$x, post_ell_1as$y)

#compute posterior cdfs
post_dc_0$y_int      <- cumtrapz(post_dc_0$x, post_dc_0$y)
post_phi_0$y_int     <- cumtrapz(post_phi_0$x, post_phi_0$y)
post_tau_0$y_int     <- cumtrapz(post_tau_0$x, post_tau_0$y)
post_omega_1e$y_int  <- cumtrapz(post_omega_1e$x, post_omega_1e$y)
post_omega_1as$y_int <- cumtrapz(post_omega_1as$x, post_omega_1as$y)
post_omega_1bs$y_int <- cumtrapz(post_omega_1bs$x, post_omega_1bs$y)
post_ell_1e$y_int    <- cumtrapz(post_ell_1e$x, post_ell_1e$y)
post_ell_1as$y_int   <- cumtrapz(post_ell_1as$x, post_ell_1as$y)
```

```
#posterior distributions
#define quantiles
hyp_posterior <- data.frame(quant=seq(0.0,1.0,0.01))
#compute pdf and cdf
if (! all(is.na(post_dc_0$y_int))){
  hyp_posterior$dc_0          <- approx(post_dc_0$y_int,       post_dc_0$x,
                                        hyp_posterior$quant)$y
  hyp_posterior$dc_0_pdf      <- approx(post_dc_0$y_int,       post_dc_0$y,
                                        hyp_posterior$quant)$y
} else {
  hyp_posterior$dc_0          <- NaN
  hyp_posterior$dc_0_pdf      <- NaN
}
if (! all(is.na(post_ell_1e$y_int))){
  hyp_posterior$ell_1e        <- approx(post_ell_1e$y_int,     post_ell_1e$x,
                                        hyp_posterior$quant)$y
  hyp_posterior$ell_1e_pdf    <- approx(post_ell_1e$y_int,     post_ell_1e$y,
                                        hyp_posterior$quant)$y
} else {
  hyp_posterior$ell_1e        <- NaN
  hyp_posterior$ell_1e_pdf    <- NaN
}
if (! all(is.na(post_ell_1as$y_int))){
  hyp_posterior$ell_1as       <- approx(post_ell_1as$y_int, post_ell_1as$x,
                                        hyp_posterior$quant)$y
  hyp_posterior$ell_1as_pdf   <- approx(post_ell_1as$y_int, post_ell_1as$y,
                                        hyp_posterior$quant)$y
} else {
  hyp_posterior$ell_1as       <- NaN
  hyp_posterior$ell_1as_pdf   <- NaN
}
if (! all(is.na(post_omega_1e$y_int))){
  hyp_posterior$omega_1e      <- approx(post_omega_1e$y_int, post_omega_1e$x,
                                        hyp_posterior$quant)$y
  hyp_posterior$omega_1e_pdf  <- approx(post_omega_1e$y_int, post_omega_1e$y,
                                        hyp_posterior$quant)$y
} else {
  hyp_posterior$omega_1e      <- NaN
  hyp_posterior$omega_1e_pdf  <- NaN
}
if (! all(is.na(post_omega_1as$y_int))){
  hyp_posterior$omega_1as     <- approx(post_omega_1as$y_int,
                                  post_omega_1as$x, hyp_posterior$quant)$y
  hyp_posterior$omega_1as_pdf <- approx(post_omega_1as$y_int,
                                  post_omega_1as$y, hyp_posterior$quant)$y
} else {
  hyp_posterior$omega_1as     <- NaN
  hyp_posterior$omega_1as_pdf <- NaN
}
if (! all(is.na(post_omega_1bs$y_int))){
  hyp_posterior$omega_1bs     <- approx(post_omega_1bs$y_int,
                                  post_omega_1bs$x, hyp_posterior$quant)$y
  hyp_posterior$omega_1bs_pdf <- approx(post_omega_1bs$y_int,
                                  post_omega_1bs$y, hyp_posterior$quant)$y
  } else {
    hyp_posterior$omega_1bs      <- NaN
    hyp_posterior$omega_1bs_pdf  <- NaN
  }
  if  (! all(is.na(post_phi_0$y_int))){
    hyp_posterior$phi_0          <- approx(post_phi_0$y_int,      post_phi_0$x,
                                    hyp_posterior$quant)$y
    hyp_posterior$phi_0_pdf      <- approx(post_phi_0$y_int,      post_phi_0$y,
                                    hyp_posterior$quant)$y
  } else {
    hyp_posterior$phi_0          <- NaN
    hyp_posterior$phi_0_pdf      <- NaN
  }
  if  (! all(is.na(post_tau_0$y_int))){
    hyp_posterior$tau_0          <- approx(post_tau_0$y_int,      post_tau_0$x,
                                    hyp_posterior$quant)$y
    hyp_posterior$tau_0_pdf      <- approx(post_tau_0$y_int,      post_tau_0$y,
                                    hyp_posterior$quant)$y
  } else {
    hyp_posterior$tau_0          <- NaN
    hyp_posterior$tau_0_pdf      <- NaN
  }
```

Listing 7.9. Post-processing section of type-1 NGMM INLA regression file. Full Posterior distributions of hyper-parameter.

The code in Listing 7.10 produces the figures showing the spatial distribution of the SPDE mesh, the mean values, epistemic uncertainty of the spatially varying terms, and the posterior distributions of the hyper-parameters. In particular:

- pl_mesh produces the figure for the mesh of the spatially varying coefficients.
- pl_dc_1e_mu_map produces the figure for the spatial variability of the mean values of $\delta\vec{c}_{1,E}$.
- pl_dc_1e_sd_map produces the figure for the spatial variability of the epistemic uncertainty of $\delta\vec{c}_{1,E}$.
- pl_dc_1as_mu_map produces the figure for the spatial variability of the mean values of $\delta\vec{c}_{1a,S}$.
- pl_dc_1as_sd_map produces the figure for the spatial variability of the epistemic uncertainty of $\delta\vec{c}_{1a,S}$.
- pl_dc_0_post creates the figure with the posterior distribution of δc_0.
- pl_omega_1e_post creates the figure of the posterior distribution of $\omega_{1,E}$.
- pl_omega_1as_post creates the figure of the posterior distribution of $\omega_{1a,S}$.
- pl_omega_1bs_post creates the figure of the posterior distribution of $\omega_{1b,S}$.
- pl_ell_1e_post creates the figure of the posterior distribution of $\ell_{1,E}$.
- pl_ell_1as_post creates the figure of the posterior distribution of $\ell_{1a,S}$.

```
# Plotting
# ---------------------------
#plotting info
set1    <- RColorBrewer::brewer.pal(7, "Set1") #color map
#California
map_ca <- subset( map_data("state"), region %in% c("california"))
map_ca_utm <- LongLatToUTM(lat=map_ca$lat, lon=map_ca$long, utm_no)
map_ca[,c('X','Y')] <- map_ca_utm[,c('X','Y')]/1000
#Nevada
map_nv <- subset( map_data("state"), region %in% c("nevada"))
map_nv_utm <- LongLatToUTM(lat=map_nv$lat, lon=map_nv$long, utm_no)
map_nv[,c('X','Y')] <- map_nv_utm[,c('X','Y')]/1000

#Earthquake - Station Mesh
pl_mesh  <- ggplot() + theme_bw() + gg(mesh) +
            geom_path(data=map_ca, aes(x=X,y=Y), color='black') +
            geom_path(data=map_nv, aes(x=X,y=Y), color='black')+
            geom_point(data=X_eq, aes(x=eqX,y=eqY, size=as.factor('EQ'),
                                   color=as.factor('EQ'))) +
            geom_point(data=X_sta, aes(x=staX,y=staY, size=as.factor('STA'),
                                    color=as.factor('STA'))) +
            scale_size_manual(values=c(2.0,0.5),
                              labels = c('Earthquakes','Stations'),
                              name=element_blank()) +
            scale_color_manual(values=c(set1[1],set1[2]),
                               labels = c('Earthquakes','Stations'),
                               name=element_blank()) +
            labs(x="X (km)", y="Y (km)") +
            theme(plot.title=element_text(size=20),
                  axis.title=element_text(size=20),
                  axis.text.y=element_text(size=20),
                  axis.text.x=element_text(size=20),
                  legend.key.size = unit(1, 'cm'),
                  legend.text=element_text(size=20),
                  legend.position = c(0.20, 0.10))

# plot of non-ergodic terms mean and sd of spatially varying event terms
#dc_1e map mean
pl_dc_1e_mu_map <- ggplot() + theme_bw()
pl_dc_1e_mu_map <- plot_field(coeff_1e$mean, mesh,
                              xrange=c(-200,800), yrange=c(3400,4750),
                              pl=pl_dc_1e_mu_map)
pl_dc_1e_mu_map <- pl_dc_1e_mu_map + geom_path(data=map_ca, aes(x=X,y=Y),
                                                color='black') +
                   geom_path(data=map_nv, aes(x=X,y=Y), color='black') +
                   geom_point(data=X_eq, aes(x=eqX,y=eqY),
                              color=I("black"), size=0.4) +
                   labs(x="X (km)", y="Y (km)") +
                   theme(axis.title = element_text(size=20),
                         axis.text.y = element_text(size=20),
                         axis.text.x = element_text(size=20))
#dc_1e map sigma
pl_dc_1e_sd_map <- ggplot() + theme_bw()
pl_dc_1e_sd_map <- plot_field(coeff_1e$sd, mesh,
                              xrange=c(-200,800), yrange=c(3400,4750),
                                   pl=pl_dc_1e_sd_map)
pl_dc_1e_sd_map <- pl_dc_1e_sd_map + geom_path(data=map_ca, aes(x=X,y=Y),
                                                 color='black') +
                    geom_path(data=map_nv, aes(x=X,y=Y), color='black') +
                    geom_point(data=X_eq, aes(x=eqX,y=eqY),
                               color=I("black"), size=0.4) +
                    labs(x="X (km)", y="Y (km)") +
                    theme(axis.title = element_text(size=20),
                          axis.text.y = element_text(size=20),
                          axis.text.x = element_text(size=20))
#dc_1as map mean
pl_dc_1as_mu_map <- ggplot() + theme_bw()
pl_dc_1as_mu_map <- plot_field(coeff_1as$mean, mesh,
                               xrange=c(-200,800), yrange=c(3400,4750),
                               pl=pl_dc_1as_mu_map)
pl_dc_1as_mu_map <- pl_dc_1as_mu_map + geom_path(data=map_ca, aes(x=X,y=Y),
                                                   color='black') +
                      geom_path(data=map_nv, aes(x=X,y=Y), color='black') +
                      geom_point(data=X_sta, aes(x=staX,y=staY),
                                 color=I("black"), size=0.2) +
                      labs(x="X (km)", y="Y (km)") +
                      theme(axis.title = element_text(size=20),
                            axis.text.y = element_text(size=20),
                            axis.text.x = element_text(size=20))
#dc_1as map sigma
pl_dc_1as_sd_map <- ggplot() + theme_bw()
pl_dc_1as_sd_map <- plot_field(coeff_1as$sd, mesh,
                               xrange=c(-200,800), yrange=c(3400,4750),
                               pl=pl_dc_1as_sd_map)
pl_dc_1as_sd_map <- pl_dc_1as_sd_map + geom_path(data=map_ca, aes(x=X,y=Y),
                                                   color='black') +
                      geom_path(data=map_nv, aes(x=X,y=Y), color='black') +
                      geom_point(data=X_sta, aes(x=staX,y=staY),
                                 color=I("black"), size=0.2) +
                      labs(x="X (km)", y="Y (km)") +
                      theme(axis.title = element_text(size=20),
                            axis.text.y = element_text(size=20),
                            axis.text.x = element_text(size=20))

#posterior distributions
#dc_0
pl_dc_0_post <- ggplot(post_dc_0, aes(x,y)) + theme_bw() + geom_line() +
                geom_vline(xintercept = hyp_param['dc_0','mean'],
                           colour = "red") +
                labs(x = 'dc_0', y = 'posterior', title='Posterior dc_0') +
                theme(plot.title=element_text(size=20),
                      axis.title=element_text(size=20),
                      axis.text.y=element_text(size=20),
                      axis.text.x=element_text(size=20))
#omega_1e
pl_omega_1e_post <- ggplot(post_omega_1e, aes(x,y)) + theme_bw() +
                    geom_line() +
                    geom_vline(xintercept = hyp_param['omega_1e','mean'],
                               colour = "red") +
                        labs(x = 'omega_1e', y = 'posterior',
                             title='Posterior omega_1e') +
                        theme(plot.title=element_text(size=20),
                              axis.title=element_text(size=20),
                              axis.text.y=element_text(size=20),
                              axis.text.x=element_text(size=20))
#omega_1as
pl_omega_1as_post <- ggplot(post_omega_1as, aes(x,y)) + theme_bw() +
                        geom_line() +
                        geom_vline(xintercept = hyp_param['omega_1as','mean'],
                                   colour = "red") +
                        labs(x = 'omega_1as', y = 'posterior',
                             title='Posterior omega_1as') +
                        theme(plot.title=element_text(size=20),
                              axis.title=element_text(size=20),
                              axis.text.y=element_text(size=20),
                              axis.text.x=element_text(size=20))
#omega_1bs
pl_omega_1bs_post <- ggplot(post_omega_1bs, aes(x,y)) + theme_bw() +
                        geom_line() +
                        geom_vline(xintercept = hyp_param['omega_1bs','mean'],
                                   colour = "red") +
                        labs(x = 'omega_1bs', y = 'posterior',
                             title='Posterior omega_1bs') +
                        theme(plot.title=element_text(size=20),
                              axis.title=element_text(size=20),
                              axis.text.y=element_text(size=20),
                              axis.text.x=element_text(size=20))
#ell_1e
pl_ell_1e_post <- ggplot(post_ell_1e, aes(x,y)) + theme_bw() +
                     geom_line() +
                     geom_vline(xintercept = hyp_param['ell_1e','mean'],
                                colour = "red") +
                     labs(x = 'ell_1e', y = 'posterior',
                          title='Posterior ell_1e') +
                     theme(plot.title=element_text(size=20),
                           axis.title=element_text(size=20),
                           axis.text.y=element_text(size=20),
                           axis.text.x=element_text(size=20))
#ell_1as
pl_ell_1as_post <- ggplot(post_ell_1as, aes(x,y)) + theme_bw() +
                      geom_line() +
                      geom_vline(xintercept = hyp_param['ell_1as','mean'],
                                 colour = "red") +
                      labs(x = 'ell_1as', y = 'posterior',
                           title='Posterior ell_1as') +
                      theme(plot.title=element_text(size=20),
                            axis.title=element_text(size=20),
                            axis.text.y=element_text(size=20),
                            axis.text.x=element_text(size=20))
```

Listing 7.10. Post-processing section of type-1 NGMM INLA regression file. Plotting of spatially varying coefficients and posterior distributions.

The Listing 7.11 creates the output folder `out_dir` to save the data frames generated in the post-processing section. More specifically, the exported data frames include the followings.

- `*_inla_hyperparameters.csv` contains the mean, standard deviation, and main quantiles of the NGMM hyper-parameters.
- `*_inla_residuals.csv` contains the mean non-ergodic adjustment and total, within-event, and between-event non-ergodic residuals.
- `*_inla_coefficients.csv` includes the mean estimate and epistemic uncertainty of the non-ergodic terms.
- `*_inla_hyperposterior.csv` contains the full marginal posterior distributions of the NGMM hyper-parameters.

Additionally, the figures presented in Listing 7.10 are saved in the `figures` subdirectory.

```
# Write Out
# ----------------------------
fig_dir <- file.path(out_dir ,'figures')
#create output directories
dir.create(out_dir, showWarnings = FALSE)
dir.create(fig_dir, showWarnings = FALSE)
#data files
# ---   ---   ---   ---   ---
write.csv(as.data.frame(t(hyp_param)),
          file=file.path(out_dir,
                    paste0(out_fname,'_inla_hyperparameters','.csv')) )
write.csv(df_predict_summary,
          file=file.path(out_dir,
                    paste0(out_fname,'_inla_residuals','.csv')),
          row.names = FALSE )
write.csv(df_coeff,
          file=file.path(out_dir,
                    paste0(out_fname,'_inla_coefficients','.csv')),
          row.names = FALSE )
write.csv(hyp_posterior,
          file=file.path(out_dir,
                    paste0(out_fname,'_inla_hyperposterior','.csv')),
          row.names = FALSE )

#figures
# ---   ---   ---   ---   ---
#mesh
ggsave(file.path(fig_dir,paste0(out_fname,'_mesh','.png')),
       plot=pl_mesh,  device='png')
#spatial distribution of coefficients
ggsave(file.path(fig_dir,paste0(out_fname,'_map_dc_1e_mu','.png')),
       plot=pl_dc_1e_mu_map,  device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_map_dc_1e_sd','.png')),
       plot=pl_dc_1e_sd_map,  device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_map_dc_1as_mu','.png')),
       plot=pl_dc_1as_mu_map, device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_map_dc_1as_sd','.png')),
       plot=pl_dc_1as_sd_map, device='png')
```

```
#posterior distribution
ggsave(file.path(fig_dir,paste0(out_fname,'_post_dc_0','.png')),
       plot=pl_dc_0_post,        device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_post_omega_1e','.png')),
       plot=pl_omega_1e_post,    device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_post_omega_1as','.png')),
       plot=pl_omega_1as_post,   device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_post_omega_1bs','.png')),
       plot=pl_omega_1bs_post,   device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_post_ell_1e','.png')),
       plot=pl_ell_1e_post,      device='png')
ggsave(file.path(fig_dir,paste0(out_fname,'_post_ell_1as','.png')),
       plot=pl_ell_1as_post,     device='png')

rm(fit_inla_spatial)
return(NA)
```

Listing 7.11. Post-processing section of type-1 NGMM INLA regression file. Exporting Results.

7.2.2 R-INLA regression function for Type-2 dataset

This subsection presents the modifications to Section 7.2.1 to run the INLA regression for the NGMM type-2. The R code for this regression is implemented in `regression_inla_model2_uncorr_cells_unbounded_hyp.R`, which can be found in the `ngmm_tools/Analyses/R_lib/regression/inla` subdirectory of the Github repository.

Pre-processing:

In addition to the input arguments required for NGMM type-1, mandatory arguments for the NGMM type-2 regression are as follows.

- `df_cellinfo` contains the information about the cell IDs and coordinates (Section 4.2).
- `df_cellmat` contains the information about the lengths of the cell-path segments for every ground motion in the regression flatfile.
- `c_a_erg` is the ergodic value of the anelastic attenuation coefficient.

The main additional pre-processing steps in NGMM-type2 are associated with the processing of `df_cellinfo` (Listing 7.12). In `df_cellmat <- df_cellmat[match(df_flatfile$rsn, df_cellmat$rsn), ]`, the cell-path lengths are reorganized to be in the same order as their corresponding ground motions. To reduce the computational cost, only the cell with one or more paths crossing them are used in the regression (`cell_valid <- colSums(df_cellmat[, cell_names_all]) > 0`). Lastly, the cell-path distance matrix is converted to a sparse format (`RC_sparse <- as(RC, "dgCMatrix")`) to improve the efficiency of the matrix multiplication operations.

```
#cell data
#keep only cell distance for records in df_flatfile
df_cellmat <- df_cellmat[match(df_flatfile$rsn, df_cellmat$rsn), ]
assert_that(nrow(df_flatfile) == nrow(df_cellmat))
#cell info
```

```
 cell_names_all <- colnames(df_cellmat)
 cell_names_all <- cell_names_all[str_detect(cell_names_all,'c.')]
 cell_ids_all   <- as.integer( str_extract(cell_names_all,'\\d+') )
#cells with crossing paths
 cell_valid     <- colSums(df_cellmat[,cell_names_all])  > 0
 cell_names     <- cell_names_all[cell_valid]
 cell_ids       <- cell_ids_all[cell_valid]

#distance matrix
RC <- as.matrix(df_cellmat[,cell_names])
RC_sparse <- as(RC ,"dgCMatrix") #sparse matrix
 print( paste('max R_rup misfit', max(abs(rowSums(RC) - df_flatfile$Rrup))) )
```

Listing 7.12. Preprocessing section of type-2 NGMM INLA regression file.

Regression:

The fixed effects (Listing 7.13) include an additional term (R), which corresponds to the mean of the cell-specific anelastic attenuation coefficient. It is assigned a normal prior distribution with the ergodic value of the anelastic coefficient as its mean and 1000 as its precision. In df_inla_covar, the closest-point-on-rupture to site distance (R_{rup}) is assigned as a covariate for the cell-specific anelastic attenuation mean (R=df_flatfile$Rrup).

```
# Run INLA, fit model
# ---------------------------
#fixed effects
#---   ---   ---   ---   ---   ---
#prior on the fixed effects
 prior_fixed <- list(mean.intercept = 0, prec.intercept = 5,
                     mean = (list(intcp=0.0, R=c_a_erg, default=0)),
                     prec = (list(intcp=5.0, R=10000,   default=0.01)))

#covariates
 df_inla_covar <- data.frame(intcp=1, R=df_flatfile$Rrup,
                             eq=eq_id, sta=sta_id)
```

Listing 7.13. Regression Section of type-2 NGMM INLA regression file. Definition of fixed-effects.

The variation of the cell-specific anelastic attenuation coefficients is modeled with a penalized complexity prior distribution such that there is less than 10% probability $\omega_{ca,P}$ will be greater than 0.01 ($P(\omega_{ca,P} > 0.01) = 0.1$). The cell indices are stored in idx_cell of the df_inla_covar dataframe.

```
#cell-specific anelastic attenuation
#---   ---   ---   ---   ---   ---
 prior_omega_ca <- list(prec = list(prior = 'pc.prec', param = c(0.01, 0.1)))

#cell ids
 df_inla_covar$idx_cell <- 1:nrow(df_inla_covar)
```

Listing 7.14. Regression section of type-2 NGMM INLA regression file. Definition of aleatory prior distributions

The additional terms in form_inla_spatial for the type-2 NGMM are the mean effect of the cell-specific anelastic attenuation introduced by R, and the variation cell-specific anelastic attenuation introduced by the z model (f(idx_cell, model = "z", Z = RC_sparse, hyper= prior_omega_ca)) where idx_cell is the column name for the attenuation cell indices in df_inla_covar. The stk_inla_spatial is built in the same way as for NGMM type-1.

```
#inla model
#---   ---   ---   ---   ---   ---
#functional form (with spatial var)
form_inla_spatial <- y ~ 0 + intcp + R +
                                      f(eq, model="iid", hyper=prior_tau_0) +
                                      f(sta, model="iid", hyper=prior_omega_1bs) +
                                      f(idx.eq, model = spde_eq) +
                                      f(idx.sta, model = spde_sta) +
                                      f(idx_cell, model = "z",
                                        Z = RC_sparse, hyper=prior_omega_ca)

#build stack
stk_inla_spatial <- inla.stack(data = list(y = y_data),
                               A = list(A_eq, A_sta, 1),
                               effects = list(idx.eq = idx.eq,
                                              idx.sta = idx.sta,
                                              df_inla_covar),
                               tag = 'model_inla_spatial')
```

Listing 7.15. Regression section of type-2 NGMM INLA regression file. Functional form formulation.

Post-processing:

The post-processing section for NGMM type-2 follows a similar structure to the post-processing section for NGMM type-1. The main addition is the development of the data-frame with the estimated non-ergodic anelastic attenuation coefficients (Listing 7.16). The estimated variation of the attenuation cell coefficients can be found in idx_cell from fit_inla_spatial$summary.random. The mean and standard deviation for the variation of the attenuation cells is extracted from the regression model in cell_atten <- fit_inla_spatial$summary.random$idx_cell[-(1:n_data),], where n_data is the number of ground motions. That is because the first n_data rows correspond to the posterior distributions of the product of the cell-path segment length matrix and the variation of the attenuation cells, and the next n_cell rows correspond to the posterior distributions of variation of the attenuation cells, directly. The first n_{gm} rows correspond to the posterior distributions of the product of cell-path segment length matrix and the variation of the estimated cell attenuation coefficients, and the next n_c rows correspond to the posterior distributions of variation of the estimated cell attenuation coefficients, directly. n_{gm} is the number of ground motions, and n_c is the number of cells.

```
#cell specific anelastic attenuation
cell_atten <- fit_inla_spatial$summary.random$idx_cell[-(1:n_data),]
#cell mean and std
cap_mu  <- cell_atten$mean      + hyp_param['mu_cap','mean']
cap_sig <- sqrt(cell_atten$sd^2 + hyp_param['mu_cap','sd']^2)

#summary attenuation cells
```

```
df_catten_summary <- data.frame(cellid=cell_ids, c_cap_mean=cap_mu,
                                  c_cap_sig=cap_sig)
df_catten_summary <- merge(df_cellinfo[c('cellid','cellname',
                                         'mptLat','mptLon',
                                         'mptX','mptY','mptZ','UTMzone')],
                           df_catten_summary, by=c('cellid'))
```

Listing 7.16. Post-processing section of type-2 NGMM INLA regression file. Summary of non-ergodic anelastic attenuation coefficients. and residuals.

7.2.3 R-INLA regression function for Type-3 dataset

The changes for the modeling of the NGMM type-3 are presented in this section. The R code for this regression is implemented in `regression_inla_model2_uncorr_cells_unbounded_hyp.R`.

Pre-processing:

The additional input arguments for this regression are as follows.

- `c_2_erg` is the ergodic value for the geometrical spreading coefficient.
- `c_3_erg` is the ergodic value for the V_{S30} scaling coefficient.

Furthermore, `df_flatfile` requires two additional columns; `x_2` includes the geometrical spreading scaling factors of $c_{2,P}(t_E)$ for each ground motion, and `x_3` includes the V_{S30} scaling factors of $c_{3,S}(t_S)$ for each ground motion.

The extra variables which are defined in the pre-processing section are `x_2`, a column vector with the geometrical spreading scaling factors, and `x_3`, a column vector with the V_{S30} scaling factors.

```
#covariates
x_2 <- df_flatfile[,'x_2']
x_3 <- df_flatfile[,'x_3']
```

Listing 7.17. Preprocessing section of type-3 NGMM INLA regression file.

Regression:

In addition to the fixed effects described in the previous sections, the functional form of this NGMM includes `x2` and `x3`. `x2` and `x3` correspond to the mean of spatially varying geometrical spreading and V_{S30} scaling, respectively. They are assigned normal prior distributions with their ergodic values as their mean and 25 precision.

```
# Run INLA, fit model
# --------------------------
#fixed effects
#---    ---    ---    ---    ---    ---
#prior on the fixed effects
prior_fixed <- list(mean.intercept = 0, prec.intercept = 5,
                    mean = (list(intcp=0.0, R=-0.005,
                                 x2=c_2_erg, x3=c_3_erg,
                                 default=0)),
                    prec = (list(intcp=5.0, R=10000,
                                        x2=25.0,      x3=25.0,
                                        default=0.01)))

#covariates
df_inla_covar <- data.frame(intcp=1, R=df_flatfile$Rrup,
                                    x2=x_2,   x3=x_3,
                                    eq=eq_id,  sta=sta_id)
```
Listing 7.18. Regression Section of type-3 NGMM INLA regression file. Definition of fixed-effects for geometrical spreading and V_{S30} scaling.

The spatially varying component of the geometrical spreading and V_{S30} scaling are defined in Listing 7.19. The SPDE object and prior distribution of $c_{2,P}(t_E)$ is contained in `spde_eq_gs`, while the SPDE object and prior distribution of $c_{3,S}(t_S)$ is contained in `spde_sta_vs30`. The correlation length and scale of $c_{2,P}(t_E)$ ($\ell_{2,P}$ and $\omega_{2,P}$) are assigned a joint penalized complexity prior such that $P(\ell_{2,P} < 100) = 0.95$ and $P(\omega_{2,P} > 0.3) = 0.1$. Similarly, $\ell_{3,S}$ and $\omega_{3,S}$) are assigned a joint penalized complexity prior such that $P(\ell_{3,S} < 100) = 0.95$ and $P(\omega_{3,S} > 0.4) = 0.1$. Since $c_{2,P}(t_E)$ varies as a function of the earthquake coordinates, its projection matrix for all event location is created in `inla.spde.make.A(mesh, loc=as.matrix(X_eq_all), weights=x_2)`. The argument `weights = x_2` specifies the scaling factors for the geometrical spreading. Equivalently, the projection matrix for the V_{S30} scaling is created in `inla.spde.make.A(mesh, loc=as.matrix(X_sta_all), weights=x_3)` for all site locations. `idx.eq_gs` and `idx.sta_vs30` contain the mesh indices for the geometrical spreading and V_{S30} scaling terms, respectively.

```
#spatial model
#---   ---   ---   ---   ---   ---
#spde geom spreading term prior
spde_eq_gs      <- inla.spde2.pcmatern(mesh = mesh, alpha = alpha,
                                        prior.range = c(100, 0.95),
                                        prior.sigma = c(.30, 0.1))
#spde Vs30 term prior
spde_sta_vs30   <- inla.spde2.pcmatern(mesh = mesh, alpha = alpha,
                                        prior.range = c(100, 0.95),
                                        prior.sigma = c(.40, 0.1))

A_eq_gs        <- inla.spde.make.A(mesh, loc = as.matrix(X_eq_all),
                                   weights = x_2)
idx.eq_gs      <- inla.spde.make.index("idx.eq_gs",spde_eq_gs$n.spde)
A_sta_vs30     <- inla.spde.make.A(mesh, loc = as.matrix(X_sta_all),
                                   weights = x_3)
idx.sta_vs30   <- inla.spde.make.index("idx.sta_vs30",spde_sta_vs30$n.spde)
```
Listing 7.19. Regression section of type-3 NGMM INLA regression file. Definition of spatially varying geometrical spreading and V_{S30} scaling.

The functional form for type-3 NGMM, in addition to the term of the previous functional forms, includes the mean effect of the geometrical spreading (`x2`), the mean effect of the V_{S30} scaling (`x3`), the spatially varying effect of geometrical spreading (`f(idx.eq_gs, model=spde_eq_gs)`), and the spatially varying effect of the V_{S30} scaling `f(idx.sta_vs30, model=spde_sta_vs30)`. The `stk_inla_spatial` is built in the same way as for NGMM types.

```
#inla model
```

```
#---    ---    ---    ---    ---    ---
#functional form (with spatial var)
form_inla_spatial <- y ~ 0 + intcp + x2 + x3 + R +
                          f(eq, model="iid", hyper=prior_tau_0)  +
                          f(sta, model="iid", hyper=prior_omega_1bs) +
                          f(idx.eq_const,  model = spde_eq_const) +
                          f(idx.sta_const, model = spde_sta_const) +
                          f(idx.eq_gs,     model = spde_eq_gs) +
                          f(idx.sta_vs30,  model = spde_sta_vs30) +
                          f(idx_cell, model = "z", Z = RC_sparse,
                            hyper=prior_omega_ca)


#build stack
stk_inla_spatial <- inla.stack(data = list(y = y_data),
                               A = list(A_eq_const, A_sta_const,
                                        A_eq_gs, A_sta_vs30, 1),
                               effects = list(idx.eq_const = idx.eq_const,
                                              idx.sta_const = idx.sta_const,
                                              idx.eq_gs = idx.eq_gs,
                                              idx.sta_vs30 = idx.sta_vs30,
                                              df_inla_covar),
                               tag = 'model_inla_spatial')
```

Listing 7.20. Regression section of type-3 NGMM INLA regression file. Functional form formulation.

Post-processing:

The post-processing section of NGMM type-3 creates the same output files as the post-processing section for NGMM-type2. The main additions are the summary of the posterior distributions of the mean, scale, and correlation length of the spatially varying geometrical spreading and V_{S30} in hyp_param and hyp_posterior, as well, as the posterior mean, mode, and standard-deviation for $c_{2,P}$ and $c_{3,S}$ at all event and station locations in df_coeff.

```
## Post-processing Results
# ---------------------------
#hyper-parameters
hyp_param <- data.frame(matrix(ncol = 6, nrow = 0))
colnames(hyp_param) <- colnames(fit_inla_spatial$summary.hyperpar)

hyp_param['dc_0',]    <- fit_inla_spatial$summary.fixed['intcp',]
#means of spatial terms
hyp_param['mu_2p',]    <- fit_inla_spatial$summary.fixed['x2',]
hyp_param['mu_3s',]    <- fit_inla_spatial$summary.fixed['x3',]
#correlation lengths of spatial terms
hyp_param['ell_1e',]  <- fit_inla_spatial$summary.hyperpar
                                   ['Range for idx.eq_const',]
hyp_param['ell_1as',] <- fit_inla_spatial$summary.hyperpar
                                   ['Range for idx.sta_const',]
hyp_param['ell_2p',]  <- fit_inla_spatial$summary.hyperpar
                                   ['Range for idx.eq_gs',]
hyp_param['ell_3s',]  <- fit_inla_spatial$summary.hyperpar
                                   ['Range for idx.sta_vs30',]
```

```
#standard deviations of spatial terms
hyp_param['omega_1e',]   <- fit_inla_spatial$summary.hyperpar
                                              ['Stdev for idx.eq_const',]
hyp_param['omega_1as',] <- fit_inla_spatial$summary.hyperpar
                                              ['Stdev for idx.sta_const',]
hyp_param['omega_1bs',] <- 1/sqrt(fit_inla_spatial$summary.hyperpar
                                              ['Precision for sta',] )
hyp_param['omega_2p',]   <- fit_inla_spatial$summary.hyperpar
                                              ['Stdev for idx.eq_gs',]
hyp_param['omega_3s',]   <- fit_inla_spatial$summary.hyperpar
                                              ['Stdev for idx.sta_vs30',]
#anelastic attenuation
hyp_param['mu_cap',]     <- fit_inla_spatial$summary.fixed['R',]
hyp_param['omega_cap',] <- 1/sqrt(fit_inla_spatial$summary.hyperpar
                                              ['Precision for idx_cell',] )
#aleatory terms
hyp_param['phi_0',] <- 1/sqrt( fit_inla_spatial$summary.hyperpar
                                 ['Precision for the Gaussian observations',] )
hyp_param['tau_0',] <- 1/sqrt( fit_inla_spatial$summary.hyperpar
                                 ['Precision for eq',] )
#unavailable sd for transformed variables
hyp_param[c('omega_1bs','omega_cap','phi_0','tau_0'),'sd'] <- NA

## Summarize coefficients and residuals
# ---------------------------
df_flatinfo  <- df_flatfile[,c('rsn','eqid','ssn',
                               'eqLat','eqLon','staLat','staLon',
                               'eqX','eqY','staX','staY')]

#summary coefficients
df_coeff <- data.frame(rsn=df_flatinfo$rsn,
                       dc_0_mean=hyp_param['dc_0','mean'],
                       dc_1e_mean=coeff_1e_mu[eq_inv],
                       dc_1as_mean=coeff_1as_mu[sta_inv],
                       dc_1bs_mean=coeff_1bs_mu[sta_inv],
                       c_2p_mean=coeff_2p_mu[eq_inv],
                       c_3s_mean=coeff_3s_mu[sta_inv],
                       dc_0_sig=hyp_param['dc_0','sd'],
                       dc_1e_sig=coeff_1e_sig[eq_inv],
                       dc_1as_sig=coeff_1as_sig[sta_inv],
                       dc_1bs_sig=coeff_1bs_sig[sta_inv],
                       c_2p_sig=coeff_2p_sig[eq_inv],
                       c_3s_sig=coeff_3s_sig[sta_inv])
df_coeff <- merge(df_flatinfo, df_coeff, by=c('rsn'))
```
Listing 7.21. Post-processing section of type-3 NGMM INLA regression file.

8 LESSONS LEARNED FROM APPLYING NON-ERGODIC VARYING COEFFICIENT MODEL TO CONTROLLED PHYSICS-BASED SIMULATION DATASETS

Physic-based ground motion simulation datasets present two major advantages. First, simulations can be obtained for any source regions, sites, and paths, all of which can be sampled many times. Hence, the source, site, and path effects can be more accurately estimated compared to the sparse empirical datasets. Second, since the source parameters of earthquakes, site conditions, and the underlying velocity model used in simulations are known, the overall accuracy of the variable coefficient model, VCM, outputs can be directly verified against the inputs.

In this section, we discuss the work on applying a modified VCM on one CyberShake simulation dataset in southern California (hereafter CS15.4, Meng and Goulet (2022)). CyberShake incorporates an ERF with a full 3D wave propagation to perform physics-based PSHA by computing ground motions at chosen sites (Graves et al., 2011; Jordan and Callaghan, 2018). RotD50 pseudo-spectral acceleration (PSA) at periods of 2, 3, 5, and 10s are calculated from each simulated seismogram. CS15.4 includes 336 sites, 360,472 events and 97,214,974 seismograms (Figure 8.1). Meng and Goulet (2022) randomly generated four reduced datasets of 100 (CS15.4-100), 200 (CS15.4-200), 600 (CS15.4-600) and 1000 (CS15.4-1000) events (Table 8.1 and Figure 8.2). Then, VCM is applied to the four datasets, and the outputs are compared among themselves to check the sensitivity on dataset sizes. Next, Meng and Goulet (2022) identified results consistent with the input components, which can be used to constrain better future Probabilistic Seismic Hazard Analysis (PSHA) applications. When discrepancies emerge between inputs and outputs, Meng and Goulet (2022) explored their causes by conducting more tests on controlled datasets and propose potential solutions for future VCM applications.

8.1 METHODOLOGY

Meng and Goulet (2022) defined a simple function form to develop the ergodic GMM, which includes magnitude (M) scaling, geometric spreading, magnitude-distance interaction and linear site response:

Table 8.1. SCEC Simulations

Dataset name	Number of events	Number of records
CS15.4-100	100	$21,269$
CS15.4-200	200	$56,252$
CS15.4-600	600	$159,659$
CS15.4-1000	1000	$265,252$

$$f_{erg} = c_0 + c_1 M + c_2 M^2 + (c_3 + c4M) \log\left(\sqrt{R_{rup}^2 + h^2}\right) + f(V_{S30}) \tag{8.1}$$

where h is set to $4.5km$; $f(V_{S30})$ is the linear site response term based on the slowness-averaged shear wave velocity in the top 30 m of the velocity model (V_{S30}):

$$f_{V_{S30}} = c_5 \log(\min(V_{S30}, 800)/800) \qquad V_{S30} \leq 1500m/sec \tag{8.2}$$

$$f_{V_{S30}} = c_5 \log(\min(V_{S30}, 800)/1500) \qquad V_{S30} > 1500m/sec \tag{8.3}$$

The total residuals are decomposed into three spatially varying components (source, site, path effects) and the remaining aleatory residuals:

$$y_{es} = f_{erg} + \delta L2L(\vec{x}_e) + \delta S2S(\vec{x}_s) + \delta P2P(\vec{x}_e, \vec{x}_s) + \delta B_e^0 + \delta WS_{es}^0 \tag{8.4}$$

where $\delta L2L(\vec{x}_e)$ are the source effects that spatially vary with event locations $\vec{x}_e$; $\delta S2S(\vec{x}_s)$ are the site effects that spatially vary with site locations $\vec{x}_s$; $\delta P2P(\vec{x}_e, \vec{x}_s)$ are the spatially varying path effects; δB_e^0 and δWS_{es}^0 are the remaining residuals after all spatial correlations are considered. For this study, Meng and Goulet (2022) experimented with three modeling approaches for $\delta P2P(\vec{x}_e, \vec{x}_s)$. The first approach is the two-dimensional (2D) cell-specific attenuation (hereafter 2D cell approach), similar to the one introduced by Dawood and Rodriguez-Marek (2013). The entire study region is divided into 2D cells of 0.25^o by 0.25^o. The path effects are the sum of anelastic attenuation from all cells along a horizontal straight line from the closest point to the site:

$$\delta P2P(\vec{x}_e, \vec{x}_s) = \Sigma_i \delta_i \Delta R_i \tag{8.5}$$

where δ_i is the attenuation per kilometer of travel in the i^{th} cell, which is intended to capture the anelastic attenuation in traditional GMM development. δ_i is computed as an independent random effect during regression (i.e., no spatial correlation). ΔR_i is the approximate travel distance within the i^{th} cell. The second approach is the three-dimensional (3D) cell-specific attenuation (hereafter 3D cell approach). The entire subsurface is divided vertically into $1km$ layers. In the top $10km$, each layer is further divided into cells of 0.25^o by 0.25^o. The wave propagation is assumed as a straight line from the hypocenter to the site. The third approach only considers the aggregate path effects along the entire path. Similar to the path term described in Al Atik et al. (2010), each

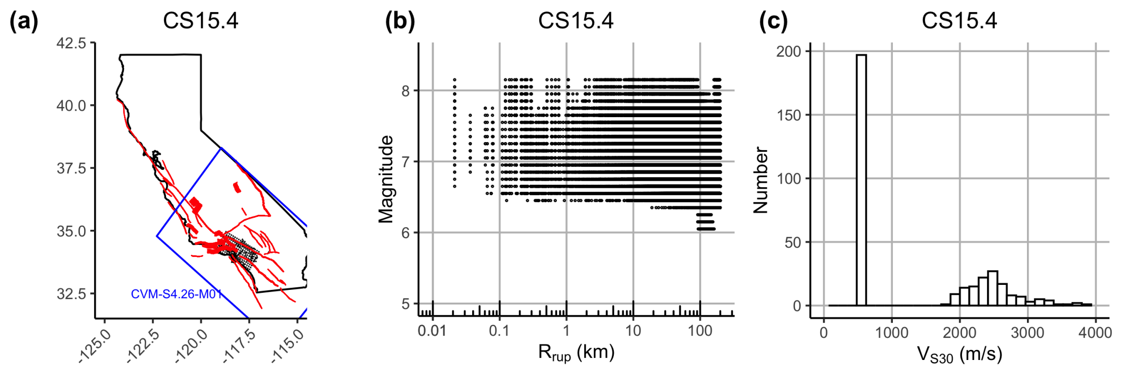


Figure 8.1. (a) Map of CS15.4; red dots and black triangles denote events' hypocenters and station locations (sites), respectively; the blue box outlines the extension of the 3D velocity model. (b) Magnitude-R_{rup} distribution of events in CS15.4. (c) V_{S30} distribution of sites in CS15.4. Modified from Meng and Goulet (2022).

path between a source region and a site is considered unique. Meng and Goulet (2022) used the midpoint t_p between the site and the closest point of the causative fault to represent the approximate location of a path:

$$\delta P2P(\vec{x}_e, \vec{x}_s) = \beta_1(\vec{x}_{mp}) R_{rup} \tag{8.6}$$

where $\beta_1(\vec{x}_{mp})$ is anelastic attenuation that spatially varies with midpoint locations $\vec{x}_{mp}$. Hereafter this approach will be referred to as the midpoint approach.

The next step is to determine the amount of correlation in $\delta L2L(\vec{x}_e)$, $\delta S2S(\vec{x}_s)$, and $\delta P2P(\vec{x}_{mp})$. This is achieved by imposing a prior distribution for the Gaussian Process (Rasmussen and Nickisch, 2010) on the three terms. After many trial runs with the reduced CyberShake datasets, Meng and Goulet (2022) found the most satisfying mesh triangle size for the balance of the precision and computational cost is 0.1^o.

8.2 RESULTS

Figure 8.3 and 8.4 show the maps of $\delta L2L(\vec{x}_e)$ and the epistemic predictive uncertainty $\omega_{L2L}(\vec{x}_e)$ for the four datasets, respectively (Meng and Goulet, 2022). For CS15.4-100, the variations of $\delta L2L(\vec{x}_e)$ are constrained to the event locations (Figure 8.3). At locations without events, $\delta L2L(\vec{x}_e)$ returns to the median value (i.e., zero) and the predictive uncertainty $\omega_{L2L}(\vec{x}_e)$ increases significantly, as intended (Figure 8.4 and 8.4). For the three larger datasets, the correlation lengths ℓ of $\delta L2L(\vec{x}_e)$ increase significantly and as a result, the variations of $\delta L2L(\vec{x}_e)$ extend to large areas without events (Figure 8.3). Moreover, there is no longer a clear distinction in $\omega_{L2L}(\vec{x}_e)$ between locations with and without data for the three larger datasets (Figure 8.4). In other words, the VCM fails to capture the genuine source effects when the number of events in CyberShake datasets increases.

The results of $\delta S2S(\vec{x}_s)$ and its predictive uncertainty $\omega_{S2S}(\vec{x}_S)$ are almost identical across four datasets (Figure 8.5 and 8.6) (Meng and Goulet, 2022). Within the Ventura and Los Angeles basins,

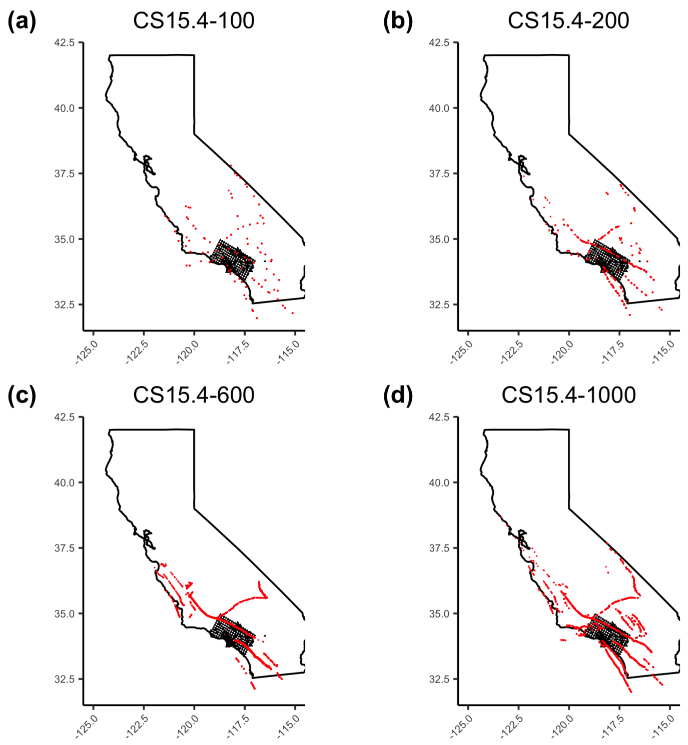


Figure 8.2. Map of four reduced CS15.4 datasets: red dots and black triangles denote events' hypocenters and station locations, respectively. Most of the sites are concentrated in the Los Angeles area, but a few are scattered farther away. Modified from Meng and Goulet (2022).

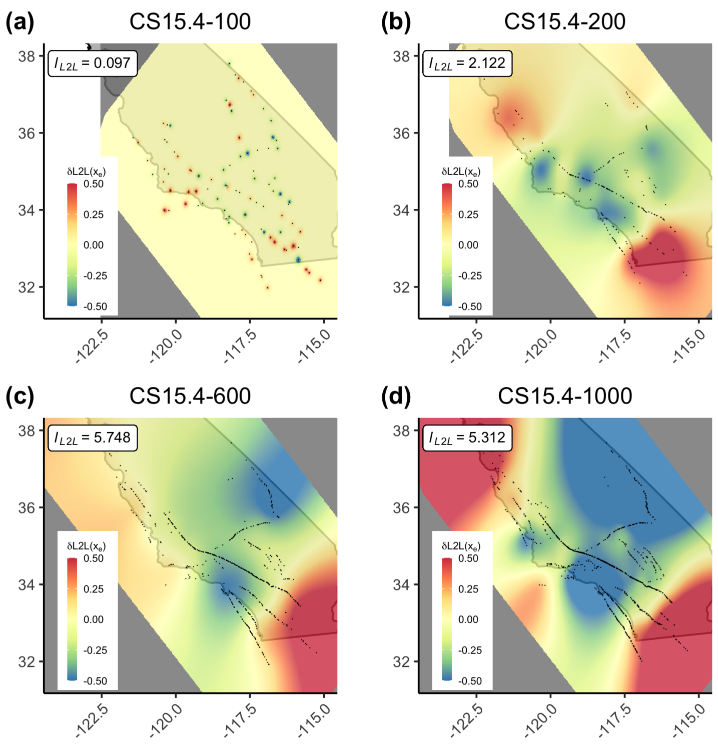


Figure 8.3. Results of $\delta L2L(\vec{x}_e)$ for the four reduced datasets. Modified from Meng and Goulet (2022).

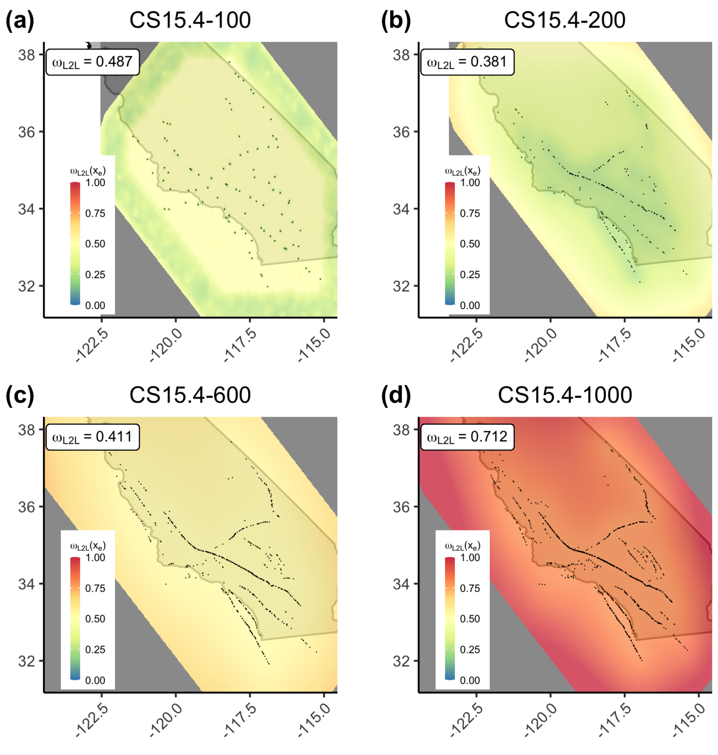


Figure 8.4. Results of $\omega_{L2L}(\vec{x}_e)$ for the four reduced datasets. Modified from Meng and Goulet (2022).

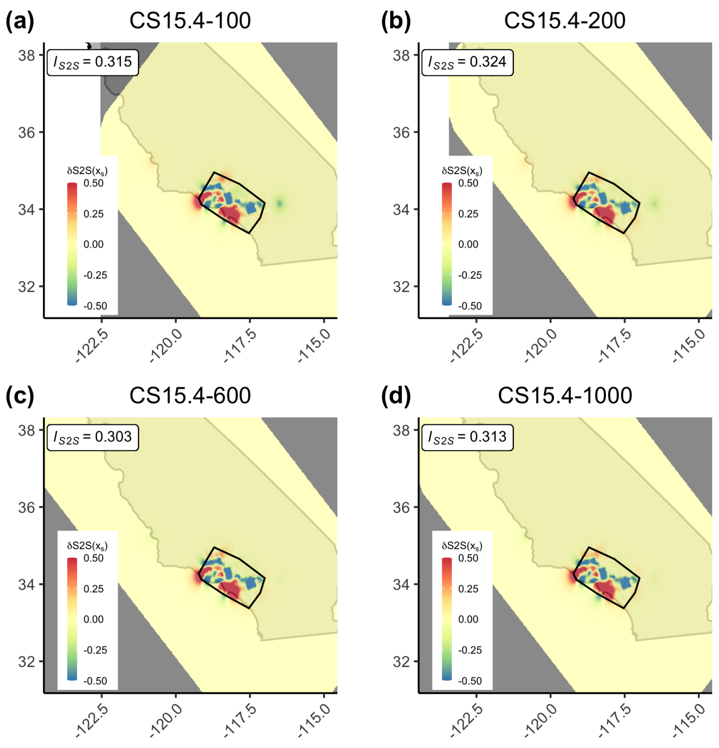


Figure 8.5. Results of $\delta S2S(\vec{x}_e)$ for the four reduced datasets. Modified from Meng and Goulet (2022).

the sites have the strongest site responses. In comparison, the hard rock sites at the San Gabriel Mountains have much weaker site responses. The site responses return to zero outside of the site coverage, where they show increased epistemic uncertainty. The few test sites located outside of the box show the same trends. The consistent pattern suggests that the genuine site effects of CyberShake datasets are accurately recovered by the VCM technique.

For the 2D cell approach, the variations of δ_i are constrained at locations with data (Figure 8.7), and the uncertainty of δ_i increases at locations without data. However, the approach does not capture the strong anelastic attenuation expected within the Ventura and Los Angeles basins (Figures 8.7). In all four datasets, cells within the basins and their immediate vicinity (hereafter inner cells) have δ_i close to zero (Figure 8.7), which represent the median attenuation of all the cells sampled by seismic waves. Outside the basins (hereafter outer cells), the δ_i patterns are dependent on the data distribution.

The δ_i results obtained by the 3D cell approach show a slight improvement over the 2D cell approach (Meng and Goulet, 2022). At shallow depths ($0 - 3km$), the outer cells have consistent weak attenuation, especially for the three larger datasets, which agree with the Q_S maps. However, δ_i of the inner cells at shallow depths are still close to zero for four datasets. (Figure 8.8). The number of inner cells with close-to-zero δ_i quickly decreases with depth. Below 5km, there is no longer a distinct area with close-to-zero δ_i for all the datasets. Cells with weak and strong attenuation are located randomly, with very little agreement with the Q_S maps at the corresponding depth.

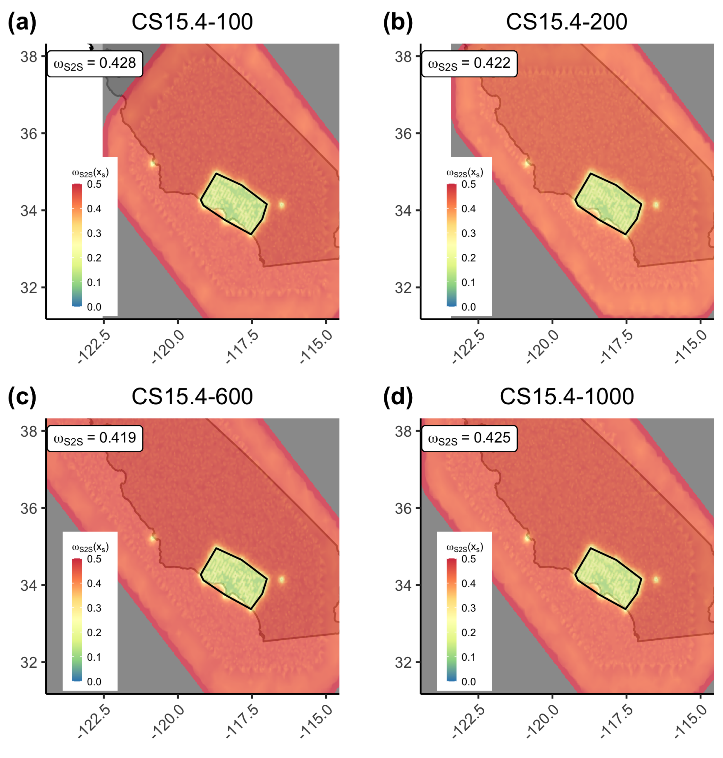


Figure 8.6. Results of $\omega_{S2S}(\vec{x}_e)$ for the four reduced datasets. Modified from Meng and Goulet (2022).

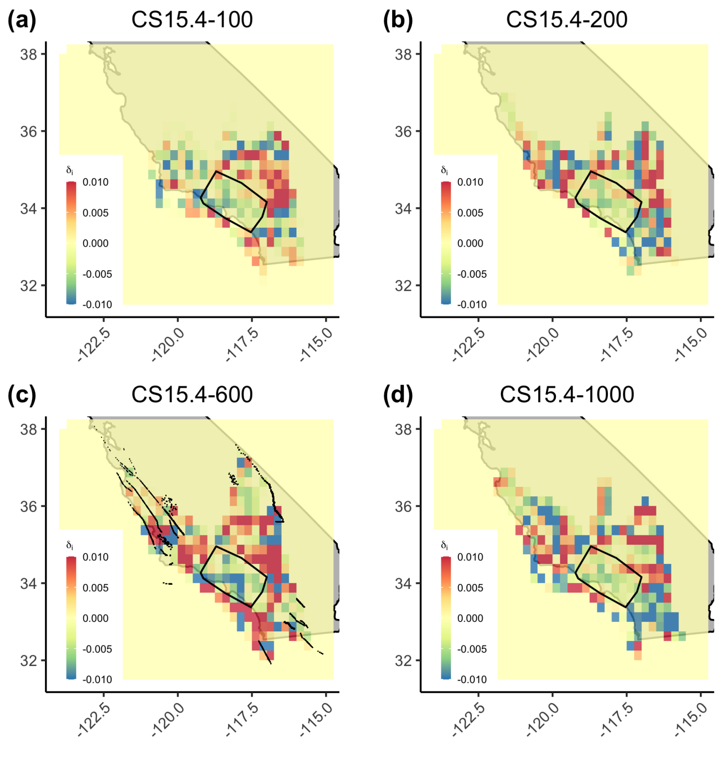


Figure 8.7. Results of δ with the 2D cell approach. Black box denotes site coverage.Modified from Meng and Goulet (2022).

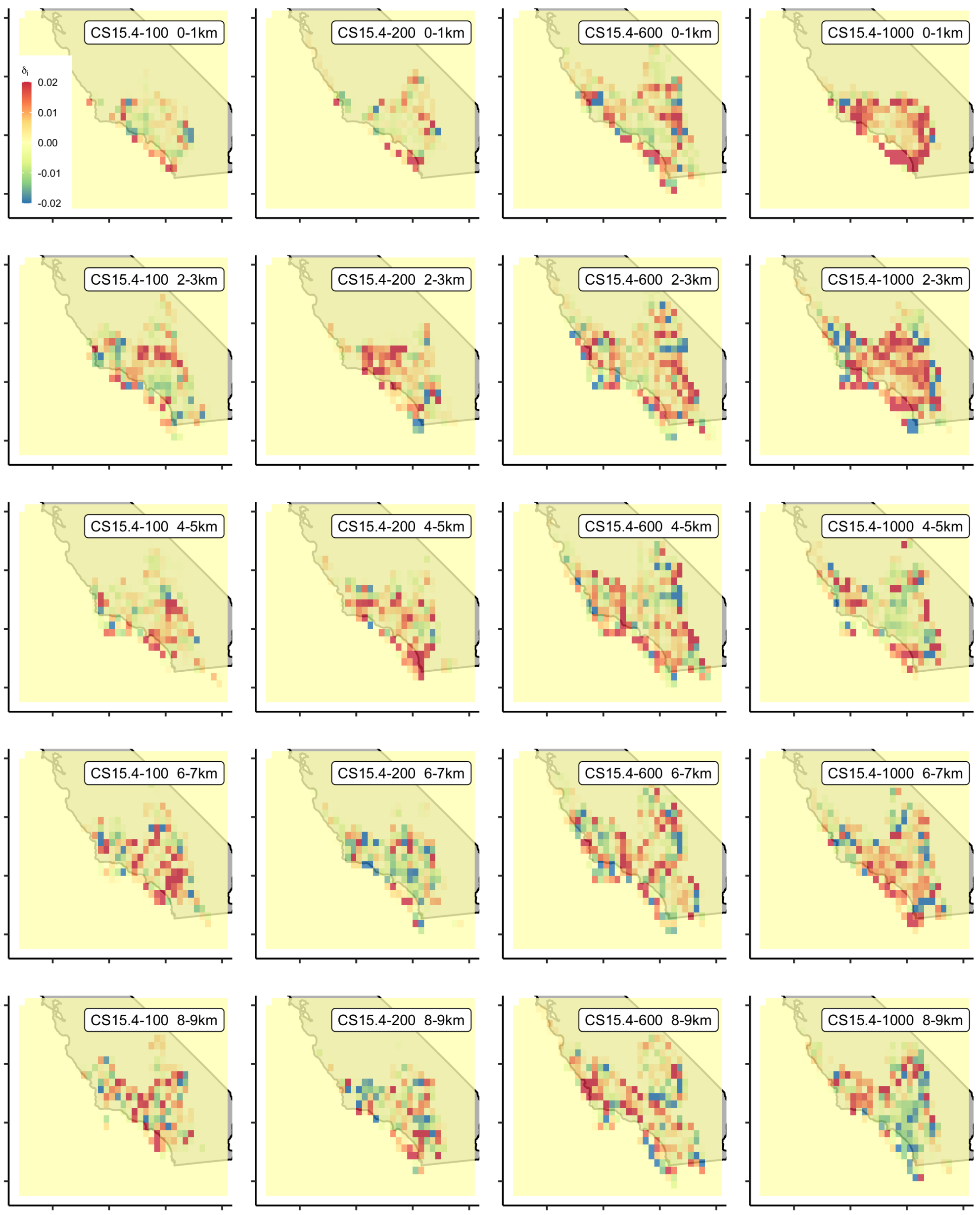


Figure 8.8. Results of δ with the 3D cell approach at selected depths. Each row denotes one reduced dataset. Modified from Meng and Goulet (2022).

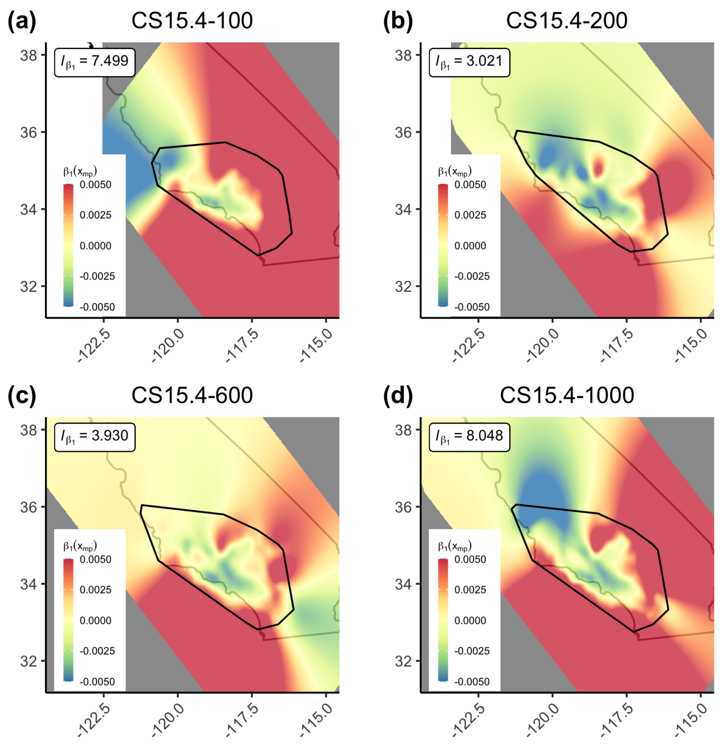


Figure 8.9. Results of $\beta(\vec{x}_{mp})$ for the four reduced datasets. The black line denotes the outline of all midpoints. Modified from Meng and Goulet (2022).

In comparison with the 2D and 3D cell approaches, the midpoint approach enables the recovery of strong attenuations within the basins for the four datasets (Figure 8.9), and the predictive uncertainties $\omega_{P2P}(\vec{x}_{mp})$ increase at locations without data (Meng and Goulet, 2022). However, similar to $\delta L2L(\vec{x}_e)$, the variations of $\beta_1(\vec{x}_{mp})$ extend to large areas without data due to the very large correlation lengths (Figure 8.9). Moreover, the correlation lengths are evidently different among the four datasets, which result in significantly different patterns outside the basins. Therefore, although the midpoint approach leads to results that are more consistent with the input velocity model, issues on the stability of the approach for different sizes remain.

8.3 DISCUSSION

The very large correlation lengths of $\delta L2L(\vec{x}_e)$ are likely caused by two unique properties of the CyberShake datasets (Meng and Goulet, 2022). First, the majority of events are self-similar, that is, they follow the same magnitude-rupture area relationship (i.e., same stress drop) (Somerville et al., 1999). Second, the majority of events in CyberShake datasets are associated with principal faults in California. When the number of events becomes large, they occur densely along those faults and form large linear features (Figure 8.2). The self-similar events along large linear features thus lead to large correlation lengths of $\delta L2L(\vec{x}_e)$. In R-INLA, the Gaussian Process is assumed isotropic, that is, the process only depends on the distances between two data points, but not the azimuth. As a result, when predicting at new locations, the large correlation length is projected onto all azimuths. To test this hypothesis, Meng and Goulet (2022) randomly selected another 600

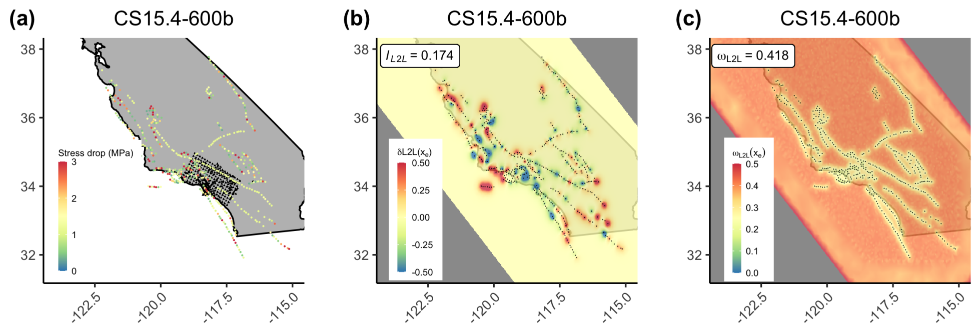


Figure 8.10. (a) Map of CS15.4-600b. Events are color-coded by stress drop; sites are in black. (b) and (c) Results of $\delta L2L(\vec{x}_e)$ and $\omega_{L2L}(\vec{x}_e)$for CS15.4-600b, respectively. Modified from Meng and Goulet (2022).

events in CS15.4 (hereafter, CS15.4-600b) under two restrictions: (1) the minimum hypocentral distance between any two events is $10km$; and (2) the stress drops are evenly distributed between 0 and $3MPa$ (Figure 8.10). After applying VCM to CS15.4-600b, Meng and Goulet (2022) found that the correlation length of $\delta L2L(\vec{x}_e)$ becomes much smaller (Figure 9.10b). The variations of $\delta L2L(\vec{x}_e)$ are constrained at event locations and correlate well with stress drops (Figure 8.3). $\delta L2L(\vec{x}_e)$ also increases at locations without data as intended (Figure 8.10). This test confirms that large correlation lengths of $\delta L2L(\vec{x}_e)$ for CyberShake datasets are indeed caused by the dense linear distribution and uniform stress drop.

Both the 2D and 3D cell approaches do not capture the strongest attenuation at shallow depths within the Ventura and Los Angeles basins for any reduced dataset. Although the cell-specific attenuation term is designed to recover anelastic attenuation, in reality it includes all the systematic effects that are not modeled in GMM. First of all, the complicated wavefields due to the smooth 3D velocity model (e.g., strong reflections and refractions at interfaces where seismic wave velocities change significantly, and generation of large surface waves). Second, due to the dense site distributions in CS15.4, the azimuths from the source location to all the sites are very similar for many events. Therefore, some source effects (e.g., radiation patterns) are likely to be incorrectly mapped into the path effects. Third, the epistemic uncertainty of the 3D velocity model itself is an important component of the path effects but not modeled yet. Another issue with the cell-specific attenuation approach is the single-line assumption of wave-propagation path. To study the sensitivity of the cell approaches, Meng and Goulet (2022) performed a test with the 3D controlled dataset. After computing the input $\delta P2P(\vec{x}_e, \vec{x}_s)$ from the hypocenter to site, we randomly shift the hypocenters by a small distance on the rupture plane $\mathcal{N}(0, \sigma_1^2)$. Now there is a subtle difference between the path used in computing the input path effects and the path used for regression in the VCM. Even with a very small shift ($\sigma_1 = 1km$), significant deviations between the output and input δ are observed (Figure 8.11). As a result, it can be concluded that the cell approaches are highly sensitive to the assumption of seismic wave propagation paths. All the events in CS15.4 are above magnitude 6. The rupture plane of a magnitude 6 earthquake typically has a length of $30km$ and a width of $12km$. For such large rupture planes, the point source assumption used in the VCM may not be appropriate. In the near future, we will continue to investigate the more accurate representation of wave-propagation paths in CyberShake simulations and better ways to decompose the systematic effects so that the genuine path effects can be captured. This issue has little impact on empirical regressions, as most events have small magnitudes and can be treated as point sources. An issue arises from empirical datasets for larger magnitude events however, as we do not know the correct answer from the simulations, of whether path effects are due to source, wave reflections/refractions, or anelastic attenuation; the empirical determination of path effects should probably include increased epistemic uncertainty.

Unlike the cell approaches, the midpoint approach only considers the aggregated attenuations from one event to one site. As a result, the strongest attenuations within the basins are recovered. However, the midpoint approach also smooths out small-scale features in attenuation patterns by aggregating them, which results in large correlation lengths of $\delta P2P(\vec{x}_e, \vec{x}_s)$ (Figure 8.9). Since R-INLA does not allow specifying the lower- or upper-bound of the correlation length, Meng and Goulet (2022) investigated finding alternative ways to constrain correlation length of $\delta P2P(\vec{x}_e, \vec{x}_s)$ during VCM regression. One way to quantify the spatial correlation of path effects is to apply the semivariogram model to path terms computed from mixed-effects regression (Walling and Abrahamson, 2012). The range of values in the best-fitting semivariogram function denotes the lag distance beyond which path terms are no longer correlated. Meng and Goulet (2022) first computed the path terms with the mixed-effects regression for CS15.4-1000 and perform a regression on the semivariogram function with a spherical model (Figure 8.12). The best-fitting range value, 1.209, was used as the fixed correlation length of $\delta P2P(\vec{x}_e, \vec{x}_s)$ in VCM regression. The updated pattern still captures the strongest attenuation within the basins and much weaker attenuation outside the basins (Figure 8.12). More importantly, the weak attenuations are constrained at locations with data. This approach is promising for broader application but would require additional investigation using recorded data.

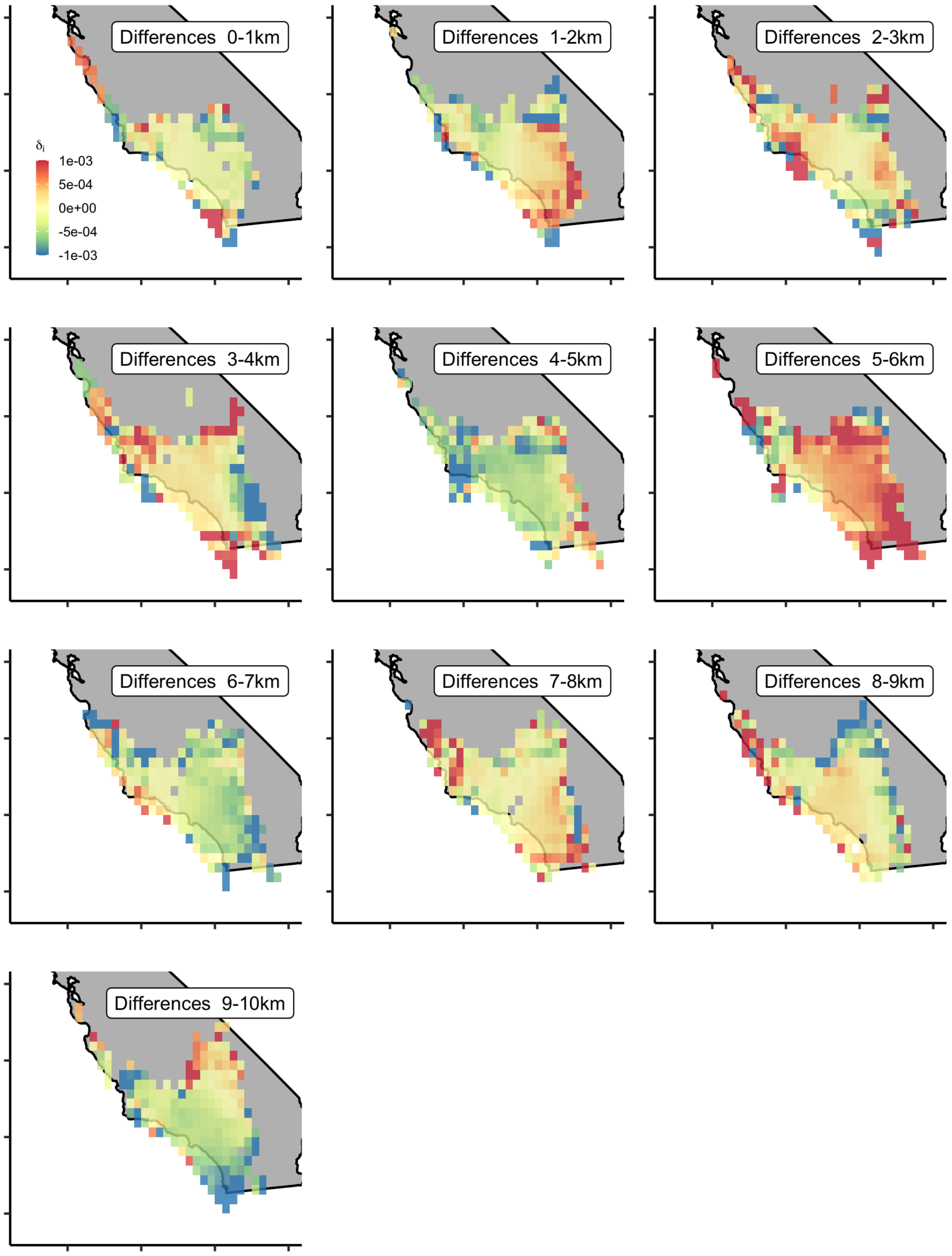


Figure 8.11. Maps of differences between input and output δ for the 3D controlled dataset with the 3D cell approach and shifted hypocenters. Modified from Meng and Goulet (2022).

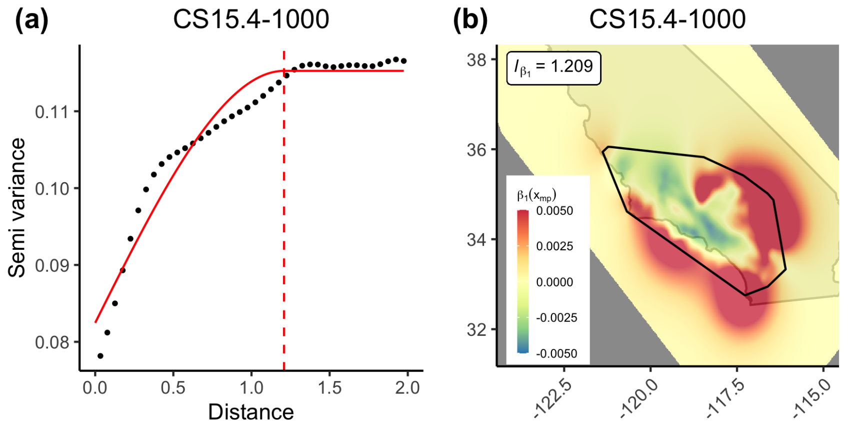


Figure 8.12. (a) Semivariogram model of path terms computed by mixed-effects regression for CS15.4-1000. The red line denotes the best-fitting spherical model. The red dashed line denotes the range value 1.209 of the best-fitting model. (b) Results of $\beta_1(\vec{x}_{mp})$ for CS15.4-1000 with correlation length fixed at 1.209. Modified from Meng and Goulet (2022).

9 A LIST OF NOTATIONS USED IN NON-ERGODIC GMMS

9.1 ACRONYMS

IP:	Intensity parameter
PSA:	Pseudo spectral acceleration
FAS:	Fourier amplitude spectra
EAS:	Effective amplitude spectra
GMM:	Ground motion model
VCM:	Varying coefficient model
MLE:	Maximum likelihood estimation
GP:	Gaussian Process
RVT:	Random vibration theory

9.2 GMM INPUT VARIABLES

M:	Moment magnitude
R_{rup}:	Closest point on rupture-to-site distance
R_x:	Horizontal distance from the top of the rupture measured perpendicular to the fault strike
R_{y0}:	Horizontal distance off the end of the rupture measured parallel to strike.
$\Delta\vec{R}$:	Cell-path segments lengths of the anelastic attenuation cells cells
V_{S30}	Time average shear wave velocity at the top $30m$
F_{RV}:	Reverse fault scaling factor
F_N:	Normal fault scaling factor
f_{NL}:	Non-linear site amplification
f_{HW}:	Hanging wall scaling

9.3 MODEL PARAMETERS

c_i:	Ergodic GMM coefficient
$c_{i,x}$:	Non-ergodic GMM coefficient where x can be: S for systematic site effects, P for systematic site effects, or E for systematic site source
$\delta c_{i,x}$:	Non-ergodic adjustment to GMM coefficient
$\vec{c}_{ca,p}$:	Cell specific anelastic attenuation coefficients
$\delta S2S$:	Total site-to-site non-ergodic term
$\delta P2P$:	Total path-to-path non-ergodic term
$\delta L2L$:	Total Source-to-source non-ergodic term
δB_e:	Between-event aleatory term
δW_{es}:	Within-event aleatory terms
δWS_{es}:	Within-event within-site term of a partially non-ergodic GMM
δB_e^0:	Between-event term of a non-ergodic GMM
δWS_{es}^0:	Within-event within-site term of a non-ergodic GMM

9.4 MODEL HYPERPARAMETERS

$\ell_{i,x}$:	Correlation length in the kernel function of $c_{i,x}$ or $\delta c_{i,x}$
$\omega_{i,x}$:	Scale/Standard deviation of the $c_{i,x}$ or $\delta c_{i,x}$ kernel function
ϕ_{S2S}:	Standard deviation of $\delta S2S$
ϕ_{P2P}:	Standard deviation of $\delta P2P$
τ_{L2L}:	Standard deviation of $\delta L2L$
τ:	Standard deviation of δB_{es}
ϕ:	Standard deviation of $\delta W_{e,s}$
τ_0:	Standard deviation of δB^0_e
ϕ_0:	Standard deviation of $\delta WS^0_{e,s}$

9.5 OTHER SYMBOLS

y:	Response variable of GMM
$\vec{x}$:	Array of GMM input variables (e.g. R_rup, V_{S30})
ρ:	Correlation coefficient
$\vec{\theta}$:	Array of all GMM parameters
$\vec{\theta}_{hyp}$:	Array of all GMM hyperparameters
$\kappa_i(\vec{t},\vec{t'})$:	Kernel function of $c_{i,x}$ or $\delta c_{i,x}$
t_E:	Earthquake coordinates
t_{Rup}:	Coordinates of the closest-point on the rupture to each site
t_S:	Site coordinates
t_{MP}:	Coordinate of mid-point between source and site
t_C:	Cell coordinates
$\mu(y)$:	Mean estimate of the y ground-motion parameter
$\psi(y)$:	Epistemic uncertainty of y ground-motion parameter
$\mu(c_i)$:	Mean estimate of c_i coefficient
$\psi(c_i)$:	Epistemic uncertainty of c_i coefficient
$\hat{}*$:	New scenarios in GP predictions (e.g. t^*_E corresponds to location of new earthquake)
f_{erg} :	Median ergodic ground motion function.
f_{nerg} :	Median non-ergodic ground motion function.

10 CONCLUDING REMARKS

10.1 SUMMARY

This report presents a summary of different methods, and the associated computer tools, for the development of non-ergodic GMMs. An emphasis is placed on methods that use GP as it offers a convenient framework for expressing spatially varying non-ergodic terms. The cell-specific anelastic attenuation can be combined with GP to model systematic effects related to the path. A simple example of the steps for developing a non-ergodic GMM using a synthetic dataset and making predictions at new locations is included in the electronic supplement of the report.

The use of non-ergodic GMMs in PSHA is a promising development, as the reduction in aleatory variability can significantly impact the seismic hazard at large return periods, and improve the accuracy of the site-specific hazard. In PSHA applications, the reduction of the aleatory variability should be combined with the consideration of epistemic uncertainty due to the uncertainty in the estimates of the non-ergodic terms in addition to the epistemic uncertainty in the extrapolation to large magnitudes and short distances of the underlying ergodic GMMs. A higher computational cost is associated with developing and applying non-ergodic GMMs. This limitation can be overcome by utilizing high-performance computers or efficient approximation methods. For example, INLA. Rue et al. (2009) provides an efficient method for estimating the non-ergodic terms and Lacour and Abrahamson (2019) provide an efficient approach for propagating non-ergodic terms in PSHA.

10.2 ROLE OF NUMERICAL SIMULATIONS

In the numerical simulations comparison, the proposed technique was applied to the CyberShake dataset. By definition, CyberShake is a non-ergodic model, and the various inputs used for the simulations are known, making it an ideal testbed for better understanding and testing the VCM capabilities. It was found that the obtained site effects are consistent with the simulation datasets. Furthermore, they are consistent in trends with empirical datasets, suggesting they could be extracted using the VCM and used to improve future non-ergodic GMMs for common site coverage. The VCM was not able to capture the simulated source effects because the dense linear distribution of self-similar events in the CyberShake dataset violates the isotropic assumption required by the Gaussian Process. In this case, one may compute the point-estimate of the event term instead. For the path effects, both the 2D and 3D cell approaches do not recover the anelastic attenuation from the 3D velocity model, mainly because other unmodeled effects like source radiation pattern and scattering lead to inaccurate estimation for longer periods. Moreover, CyberShake only has moderate to large magnitude events, the large rupture planes and surface wave generation lead to rather complicated wave propagation effects. A more sophisticated representation of wave propagation is necessary to recover the anelastic attenuation pattern. This finite-fault effect was one of the motivations for the development of large-scale simulations in the first place. However, the midpoint approach tends to better recover the attenuation patterns, especially when fixed correlation length is carefully selected, as it does not make any assumption on the wave propagation path.

This study provides important guidance for future applications of VCM to CyberShake datasets and other simulation datasets. Due to the large discrepancy in magnitude range and earthquake density between CyberShake and empirical datasets, the specific issues encountered in the computation of source and path effects are unlikely to appear for empirical datasets. However, we expect that the general lessons learned will encourage further verifications from modelers. Moreover, this study highlighted the shortcomings of GMMs based on empirical datasets with mostly point sources, which handle large rupture planes and complicated path effects poorly. It is crucial to develop GMMs based on simulations with large magnitude events, so that GMM can adequately predict source and path effects from future large earthquakes. Last but not least, lessons learnt here will help further development of the technique to capture the genuine path effects for non-ergodic modeling development, both from simulations and as part of GMMs.

10.3 FUTURE STEPS

As larger empirical datasets become available, new non-ergodic GMMs are anticipated to continue adding spatially varying non-ergodic terms to capture more systematic site, path, and source effects.

Numerical simulations can also be used to test the decisions and assumptions associated with non-ergodic GMM scaling. There is still uncertainty in the repeatability of source effects for a given region or a single fault. In particular, with the use of small magnitude events to constrain the non-ergodic terms, the scaling of the non-ergodic source terms from small magnitudes to larger magnitudes has not been fully validated. The variability due to fault physics complexity may inherently be irreducible at the time scales we are working with, even in consideration of fault maturity information, which is quite limited. Similarly, path effects constrained by small events are theoretically simpler than for large events (waves emitted from different points of the fault and traversing a large volume to a site where their effect is aggregated). Developments in three key areas can improve non-ergodic modeling: (a) continued collection of recorded ground motions to constrain repeatable effects over large areas, (b) numerical simulations to quantify the differences in path effects of large earthquakes with extended ruptures and small earthquakes with point source ruptures, and (c) earthquake physics to help with better prediction of source effects. Future studies should also evaluate the stability of hyperparameters between different areas to determine if a set of generic hyperparameters can be used. This will allow the development of non-ergodic GMM for regions with fewer recordings, as larger datasets are required to estimate the model hyperparameters than non-ergodic terms. Finally, even considering their current limitations, non-ergodic GMMs such as those described here have advantages over ergodic (or global) GMMs in increasing the accuracy of PSHA estimates and are expected to remain a useful tool to this end.

REFERENCES

Abrahamson, N. A., Al-Atik, L., Bayless, J., Dinsick, A., Dreger, D. S., Gregor, N., Kuehn, N., Walling, M., Watson-Lamprey, J., Wooddell, K., and Youngs, R. R. (2015). "Southwestern united states ground motion characterization sshac level 3." *Report no.*, GeoPentech. rev. 2.

Abrahamson, N. A., Kuehn, N. M., Walling, M., and Landwehr, N. (2019). "Probabilistic Seismic Hazard Analysis in California Using Nonergodic Ground Motion Models." *Bulletin of the Seismological Society of America*, 109(4), 1235–1249.

Abrahamson, N. A. and Silva, W. J. (2007). "Abrahamson & Silva NGA Ground Motion Relations for the Geometric Mean Horizontal Component of Peak and Spectral Ground Motion Parameters." *Report no.*, PEER, Berkeley, CA.

Abrahamson, N. A., Silva, W. J., and Kamai, R. (2014). "Summary of the ASK14 ground motion relation for active crustal regions." *Earthquake Spectra*, 30(3), 1025–1055.

Abrahamson, N. A. and 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.

Al Atik, L., Abrahamson, N. A., Bommer, J. J., Scherbaum, F., Cotton, F., and Kuehn, N. M. (2010). "The Variability of Ground-Motion Prediction Models and Its Components." *Seismological Research Letters*, 81(5), 794–801.

Ancheta, T. D., Darragh, R. B., Stewart, J. P., Seyhan, E., Silva, W. J., Chiou, B. S., Wooddell, K. E., Graves, R. W., Kottke, A. R., Boore, D. M., Kishida, T., and Donahue, J. L. (2014). "NGA-West2 database." *Earthquake Spectra*, 30(3), 989–1005.

Anderson, J. G. and Brune, J. N. (1999). "Probabilistic seismic hazard analysis without the ergodic assumption." *Seismological Research Letters*, 70(1), 19–28.

Arroyo, D. and Ordaz, M. (2010a). "Multivariate bayesian regression analysis applied to ground-motion prediction equations, part 1: Theory and synthetic example." *Bulletin of the Seismological Society of America*, 100(4), 1551–1567.

Arroyo, D. and Ordaz, M. (2010b). "Multivariate bayesian regression analysis applied to ground-motion prediction equations, part 2: Numerical example with actual data." *Bulletin of the Seismological Society of America*, 100(4), 1568–1577.

Bates, D., Mächler, M., Bolker, B., and Walker, S. (2015). "Fitting linear mixed-effects models using lme4." *Journal of Statistical Software*, 67(1), 1–48.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, New York, NY.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). "Variational Inference: A Review for Statisticians." *Journal of the American Statistical Association*, 112(518), 859–877.

Box, G. E. and Draper, N. R. (2007). *Response surfaces, mixtures, and ridge analyses*. John Wiley & Sons.

Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of Markov Chain Monte Carlo*, Vol. 148. Chapman and Hall/CRC, <https://www.taylorfrancis.com/books/9781420079425> (5).

Caramenti, L., Menafoglio, A., Sgobba, S., and Lanzano, G. (2020). "Multi-Source Geographically Weighted Regression for Regionalized Ground-Motion Models." *Report No. 67/2020*, MOX, Dipartimento di Matematica Politecnico di Milano, Milano, Italy.

Carlin, B. P. and Louis, T. A. (2008). *Bayesian methods for data analysis*. CRC Press.

Chen, Y., Bradley, B. A., and Baker, J. W. (2021). "Nonstationary spatial correlation in New Zealand strong ground-motion data." *Earthquake Engineering and Structural Dynamics*, (April), 1–20.

Chiou, B. S. and Youngs, R. R. (2014). "Update of the Chiou and Youngs NGA Model for the Average Horizontal Component of Peak Ground Motion and Response Spectra." *Earthquake Spectra*, 30(3), 1117–1153.

Collins, N., Graves, R., and Somerville, P. (2006). "Revised analysis of 1d rock simulations for the nga-e project." *Report no.*, PEER, Berkeley, CA.

Dawood, H. M. and Rodriguez-Marek, A. (2013). "A Method for including path effects in ground-motion prediction equations: An example using the Mw 9.0 Tohoku earthquake aftershocks." *Bulletin of the Seismological Society of America*, 103(2 B), 1360–1372.

Der Kiureghian, A. and Ditlevsen, O. (2009). "Aleatory or epistemic? Does it matter?." *Structural Safety*, 31(2), 105–112.

Derras, B., Bard, P. Y., and Cotton, F. (2014). "Towards fully data driven ground-motion prediction models for Europe." *Bulletin of Earthquake Engineering*, 12(1), 495–516.

Finley, A. O. (2011). "Comparing spatially-varying coefficients models for analysis of ecological data with non-stationary and anisotropic residual dependence." *Methods in Ecology and Evolution*, 2(2), 143–154.

Franco-Villoria, M., Ventrucci, M., and Rue, H. (2019). "A unified view on Bayesian varying coefficient models." *Electronic Journal of Statistics*, 13(2), 5334–5359.

Fuglstad, G. A., Simpson, D., Lindgren, F., and Rue, H. (2019). "Constructing Priors that Penalize the Complexity of Gaussian Random Fields." *Journal of the American Statistical Association*, 114(525), 445–452.

Gómez-Rubio, V. (2020). *Bayesian inference with INLA*. CRC Press.

Graves, R., Jordan, T. H., Callaghan, S., Deelman, E., Field, E., Juve, G., Kesselman, C., Maechling, P., Mehta, G., Milner, K., et al. (2011). "Cybershake: A physics-based seismic hazard model for southern california." *Pure and Applied Geophysics*, 168(3), 367–381.

Hermkes, M., Kuehn, N. M., and Riggelsen, C. (2014). "Simultaneous quantification of epistemic and aleatory uncertainty in GMPEs using Gaussian process regression." *Bulletin of Earthquake Engineering*, 12(1), 449–466.

Jordan, T. H. and Callaghan, S. (2018). "Cybershake models of seismic hazards in southern and central california." *Proceedings of the US national conference on earthquake engineering*.

Krainski, E., Gómez-Rubio, V., Bakka, H., Lenzi, A., Castro-Camilo, D., Simpson, D., Lindgren, F., and Rue, H. (2019). *Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and INLA*. Chapman and Hall/CRC, <https://www.taylorfrancis.com/books/9780429629853> (12).

Krainski, E., Gómez-Rubio, V., Bakka, H., Lenzi, A., Castro-Camilo, D., Simpson, D., Lindgren, F., and Rue, H. (2021). "Advanced spatial modeling with stochastic partial differential equations using r and inla, <https://becarioprecario.bitbucket.io/spde-gitbook/>.

Kuehn, N. (2021). "A primer for using inla to estimate ground-motion models." *Report no.*, <https://engrxiv.org/6ut3p/>.

Kuehn, N. (In press). "Comparison of bayesian varying coefficient models for the development of nonergodic ground-motion models." *Bulletin of Earthquake Engineering*.

Kuehn, N. M. and 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.

Kuehn, N. M. and Abrahamson, N. A. (2020). “Spatial correlations of ground motion for non-ergodic seismic hazard analysis.” *Earthquake Engineering and Structural Dynamics*, 49(1), 4–23.

Kuehn, N. M., Abrahamson, N. A., and Walling, M. A. (2019). “Incorporating Nonergodic Path Effects into the NGA�West2 Ground�Motion Prediction Equations.” *Bulletin of the Seismological Society of America*, 109(2), 575–585.

Kuehn, N. M., Kishida, T., AlHamaydeh, M., Lavrentiadis, G., and Bozorgnia, Y. (2020). “A Bayesian model for truncated regression for the estimation of empirical ground-motion models.” *Bulletin of Earthquake Engineering*, 18(14), 6149–6179.

Lacour, M. (2022). “Efficient non-ergodic ground-motion prediction for large datasets.” *Bulletin of Earthquake Engineering*, (0123456789).

Lacour, M. and Abrahamson, N. A. (2019). “Efficient Propagation of Epistemic Uncertainty in the Median Ground�Motion Model in Probabilistic Hazard Calculations.” *Bulletin of the Seismological Society of America*, 109(5), 2063–2072.

Landwehr, N., Kuehn, N. M., Scheffer, T., and Abrahamson, N. A. (2016). “A nonergodic ground-motion model for California with spatially varying coefficients.” *Bulletin of the Seismological Society of America*, 106(6), 2574–2583.

Lavrentiadis, G. (2021). *Non-ergodic ground-motion models for California, Ground-motion embedment factors for the Seattle Region, and Global fault displacement model*. University of California, Berkeley.

Lavrentiadis, G. and Abrahamson, N. A. (In revision, 2022). “A non-ergodic spectral acceleration ground motion model for california developed with random vibration theory.” *Bulletin of Earthquake Engineering*.

Lavrentiadis, G., Abrahamson, N. A., and Kuehn, N. M. (2021). “A non-ergodic effective amplitude ground-motion model for California.” *Bulletin of Earthquake Engineering*, (0123456789).

Lin, P. S., Chiou, B. S., Abrahamson, N. A., Walling, M., Lee, C. T., and Cheng, C. T. (2011). “Repeatable source, site, and path effects on the standard deviation for empirical ground-motion prediction models.” *Bulletin of the Seismological Society of America*, 101(5), 2281–2295.

Lindgren, F., Bolin, D., and Rue, H. (2021). “The SPDE approach for Gaussian and non-Gaussian fields: 10 years and still running.” *arXiv*, 1–33.

Lindgren, F. and Rue, H. (2015). “Bayesian Spatial Modelling with R - INLA.” *Journal of Statistical Software*, 63(19), 1–25.

Lindgren, F., Rue, H., and Lindström, J. (2011). “An explicit link between gaussian fields and gaussian markov random fields: The stochastic partial differential equation approach.” *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 73(4), 423–498.

Lunn, D., Spiegelhalter, D., Thomas, A., and Best, N. (2009). “The BUGS project: Evolution, critique and future directions.” *Statistics in Medicine*, 28(25), 3049–3067.

Meng, X. and Goulet, C. (In revision, 2022). “Lessons learned from applying varying coefficient model to controlled simulation datasets.” *Bulletin of Earthquake Engineering*.

Moraga, P. (2019). *Geospatial health data: Modeling and visualization with R-INLA and shiny*. CRC Press.

Okazaki, T., Morikawa, N., Iwaki, A., Fujiwara, H., Iwata, T., and Ueda, N. (2021). “Ground-motion prediction model based on neural networks to extract site properties from observational records.” *Bulletin of the Seismological Society of America*, 111(4), 1740–1753.

Ordaz, M., Singh, S. K., and Arciniega, A. (1994). “Bayesian Attenuation Regressions: an Application to Mexico City.” *Geophysical Journal International*, 117(2), 335–344.

Paciorek, C. J. and Schervish, M. J. (2006). “Spatial modelling using a new class of nonstationary covariance functions.” *Environmetrics*, 17(5), 483–506.

Plummer, M. (2003). “JAGS : A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling JAGS : Just Another Gibbs Sampler.” *DSC 2003 Working Papers*, Vienna, Austria, <http://www.ci.tuwien.ac.at/Conferences/DSC-2003/>.

Powell, M. J. (2009). “The bobyqa algorithm for bound constrained optimization without derivatives.” *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge*, 26.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, <https://www.R-project.org/>.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, <https://www.R-project.org/>.

Rasmussen, C. E. and Nickisch, H. (2010). “Gaussian Processes for Machine Learning (GPML) Toolbox.” *Journal of Machine Learning Research*, 11, 3011–3015.

Rasmussen, C. E. and Williams, C. K. I. (2006). “Gaussian processes for machine learning.” *The MIT Press, Cambridge, MA, USA*, 38, 715–719.

Riddell, A., Hartikainen, A., and Carter, M. (2021). “pystan (3.0. 0).” *PyPI, March*.

Rue, H., Martino, S., and Chopin, N. (2009). “Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations.” *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 71(2), 319–392.

Simpson, D., Rue, H., Riebler, A., Martins, T. G., and Sørbye, S. H. (2017). “Penalising model component complexity: A principled, practical approach to constructing priors.” *Statistical Science*, 32(1), 1–28.

Somerville, P., Irikura, K., Graves, R. W., Sawada, S., Wald, D., Abrahamson, N. A., Iwasaki, Y., Kagawa, T., Smith, N., and Kowada, A. (1999). “Characterizing Crustal Earthquake Slip Models for the Prediction of Strong Ground Motion.” *Seismological Research Letters*, 70(1), 59–80.

Stan Development Team (2022). “The Stan Core Library, <http://mc-stan.org/> (February). Version 2.29.0.

Stein, M. L. (1991). “A kernel approximation to the kriging predictor of a spatial process.” *Annals of the Institute of Statistical Mathematics*, 43(1), 61–75.

Stewart, J. P., Afshari, K., and Goulet, C. A. (2017). “Non-ergodic site response in seismic hazard analysis.” *Earthquake Spectra*, 33(4), 1385–1414.

Sung, C.-H. and Lee, C.-T. (2019). “Improvement of the quantification of epistemic uncertainty using single-station ground-motion prediction equations.” *Bulletin of the Seismological Society of America*, 109(4), 1358–1377.

Thompson, E., Wald, D. J., and Worden, C. (2014). “Avs 30 map for california with geologic and topographic constraints.” *Bulletin of the Seismological Society of America*, 104(5), 2313–2321.

Trugman, D. T. and Shearer, P. M. (2018). “Strong correlation between stress drop and peak ground acceleration for recent M 1–4 earthquakes in the San Francisco bay area.” *Bulletin of the Seismological Society of America*, 108(2), 929–945.

van der Wilk, M., Dutordoir, V., John, S., Artemev, A., Adam, V., and Hensman, J. (2020). “A framework for interdomain and multioutput Gaussian processes.” *arXiv:2003.01115*.

Ver Hoef, J. M., Peterson, E. E., Hooten, M. B., Hanks, E. M., and Fortin, M.-J. (2018). “Spatial autoregressive models for statistical inference from ecological data.” *Ecological Monographs*, 88(1), 36–59.

Wald, D. J. and Allen, T. I. (2007). “Topographic slope as a proxy for seismic site conditions and amplification.” *Bulletin of the Seismological Society of America*, 97(5), 1379–1395.

Walling, M. and Abrahamson, N. (2012). “Non-ergodic probabilistic seismic hazard analyses.” *Proc. 15th World Conf. Earthquake Eng., Lisbon, Portugal, Paper*, number 1627.

Wang, P. (2020). *Predictability and Repeatability of Non-Ergodic Site Response for Diverse Geological Conditions*. University of California, Los Angeles.

Wang, X., Ryan, Y. Y., and Faraway, J. J. (2018). *Bayesian Regression Modeling with INLA*. Chapman & Hall/CRC.

Wills, C. J., Gutierrez, C. I., Perez, F. G., and Branum, D. M. (2015). “A next generation Vs30 map for California based on geology and topography.” *Bulletin of the Seismological Society of America*, 105(6), 3083–3091.

Withers, K. B., Moschetti, M. P., and Thompson, E. M. (2020). “A Machine Learning Approach to Developing Ground Motion Models From Simulated Ground Motions.” *Geophysical Research Letters*, 47(6), 1–9.

Yong, A., Hough, S. E., Iwahashi, J., and Braverman, A. (2012). “A terrain-based site-conditions map of california with implications for the contiguous united states.” *Bulletin of the Seismological Society of America*, 102(1), 114–128.

A ELECTRONIC SUPPLEMENT

- The project home page is located at:
  https://www.risksciences.ucla.edu/nhr3/ngmm
- The web page for the first workshop can be found at:
  https://www.risksciences.ucla.edu/nhr3/ngmm-workshop-1
- The web page for the second workshop can be found at:
  https://www.risksciences.ucla.edu/nhr3/ngmm-workshop-2
- The software tools for the development of NGMMs can be found at:
  https://github.com/NHR3-UCLA/ngmm_tools
- The Docker image for the cross-platform installation of the NGMM tools can be found at:
  https://github.com/NHR3-UCLA/docker-ngmm_tools