Evaluation of numerical integration methods for kernel averaged predictors

by

Congxing Zhu

B. S., Yangzhou University, 2012

A REPORT

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Statistics
College of Arts and Sciences

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2019

Approved by:

Major Professor
Dr. Trevor Hefley

Copyright

Abstract

In spatial applications, kernel averaged predictors have been used in disciplines such as entomology and ecology. Most of the approaches entomologists and ecologists use are ad-hoc implementations of kernel averaged predictors. In this report, I discuss a general way to compute the kernel averaged predictors. I evaluate two numerical integration methods to approximate kernel averaged predictors. Using a simulation study, I evaluate the approximation of kernel averaged predictors with a combination of three factors. The combinations consist of Gaussian and uniform kernel functions, quadrature rule and Monte Carlo numerical integration, and various numbers of numerical integration points. I illustrate the approximation of the kernel averaged predictor using field data on Hessian fly abundance. The results of the approximations are evaluated by comparing the reliability of the estimated regression coefficients and the run time under each setting. My simulation experiment and data illustration show that the rate of convergence using quadrature rule is faster than using Monte Carlo integration. In addition, my results demonstrate that a small number of numerical integration points can achieve a reasonable approximation for the kernel averaged predictors, which result in reliable statistical inference.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would especially like to thank my adviser Dr. Trevor Hefley. Without his help and support, this work would not be possible. I would like to thank my masters committee members, Dr. Weixing Song and Dr. Vahl, for their helpful comments and suggestions to improve this project. I would like to thank Dr. Ryan Schmid and Dr. Brian McCornack for sharing the Hessian fly abundance data for this project. Finally, I acknowledge and thank Dr. Lolafaye Coyne for her generous financial support through the Lolafaye Coyne Graduate Research Scholarship, which made this research possible during the summer months.

Chapter 1

Introduction

In spatial applications, traditional regression models assume that the expected value of the response can be explained by a linear combination of predictors measured at the same location. For example, a linear combination of the predictors specifies the expected value such that $\mathbb{E}(y(\mathbf{s})|\beta_0, \boldsymbol{\beta}) = \beta_0 + \mathbf{x}'(\mathbf{s})\boldsymbol{\beta}$, where $y(\mathbf{s})$ is the univariate response at location $\mathbf{s} \equiv (s_1, s_2)'$, $\mathbf{x}(\mathbf{s}) \equiv (x_1(\mathbf{s}), x_2(\mathbf{s}), ..., x_p(\mathbf{s}))'$ are the predictors measured at location $\mathbf{s}$, and $\boldsymbol{\beta} = (\beta_1, \beta_2, ..., \beta_p)'$ are the regression coefficients. However, predictor variables $\mathbf{x}(\mathbf{u})$ at another location $\mathbf{u}$, could influence $\mathbb{E}(y(\mathbf{s}))$ if $\mathbf{u}$ is close to $\mathbf{s}$. Therefore, using $\mathbf{x}(\mathbf{s})$ only may not correctly specify the expected value of the response, $\mathbb{E}(y(\mathbf{s}))$ (Heaton and Gelfand, 2011).

Heaton and Gelfand (2011) define a new predictor that incorporates the influence of surrounding predictors within an unknown neighborhood over the domain $\mathfrak{D}$. Specifically, the new kernel averaged predictor, $\widetilde{x}_j(\mathbf{s})$, can be written as

$$\widetilde{x}_j(\mathbf{s}) = \frac{1}{K(\mathbf{s}|\phi_j)} \int_{\mathfrak{D}} K(\mathbf{s}, \mathbf{u}|\phi_j) x_j(\mathbf{u}) d\mathbf{u} \,, \tag{1.1}$$

where j are the indices of the predictors at location $\mathbf{s}$ and j=1,2,3,...,p, $K(\mathbf{s}, \mathbf{u}|\phi_j)$ is a kernel defining a weight that is a function of the distance between $\mathbf{s}$ and $\mathbf{u}$ with parameter ϕ_j and $0 < K(\mathbf{s}|\phi_j) = \int_{\mathfrak{D}} K(\mathbf{s}, \mathbf{u}|\phi_j) d\mathbf{u} < \infty$ (Heaton and Gelfand, 2011).

The kernel averaged predictor method developed by Heaton and Gelfand (2011) has seen use in fields such as ecology and entomology (Chandler and Hepinstall-Cymerman, 2016; Goedhart et al., 2018), however many researchers rely on ad-hoc implementations such as choosing fixed values for the parameters ϕ_j (e.g., Belaire et al., 2014; Schmid et al., 2019), using model selection techniques to "estimate" parameters associated with the kernel (e.g., Bradter et al., 2013; Hefley et al., 2015; Stuber et al., 2017), or consider only a single functional form of the kernel (e.g., a uniform kernel). In my experience working with researchers in other disciplines, the failure to appropriately implement the kernel averaged predictor method developed by Heaton and Gelfand (2011) is due to difficulty performing the integration in 1.1. In most practical situations the integration in 1.1 cannot be performed analytically, and the numerical integration techniques required to approximate 1.1 can be challenging to implement.

In application, 1.1 can be approximated using

$$\widetilde{x}_j(\mathbf{s}) \approx \frac{1}{K(\mathbf{s}|\phi)} \sum_{q=1}^{Q} \Delta\mathbf{s}_q K(\mathbf{s}, \mathbf{s}_q|\phi) x_j(\mathbf{s}_q) \tag{1.2}$$

where Q is the number of integration points, $\mathbf{s}_q$ are all the spatial locations inside $\mathfrak{D}$, where $q = 1, 2, 3..., Q$, and $\Delta\mathbf{s}_q = \frac{1}{Q}|\mathfrak{D}|$. In spatial application, Q has the potential to be very large. For example, the National Land Cover Database (NLCD) provides nationwide data on land cover in the United States at a 30-m spatial resolution (Homer et al., 2015), which results in over 20 billion individual grid cells with unique values of $x_j(\mathbf{s})$. In practice, however, using over 20 billions integration points for 1.2 is an extremely time-consuming process. In this report, I evaluate quadrature rule and Monte Carlo integration method to approximate the kernel averaged predictors. Several different numbers of numerical integration points are used for the approximation.

In chapter 2, I review two numerical integration methods, quadrature rule and Monte Carlo integration. In chapter 3, I conduct the simulation study. In chapter 4, I provide an example that uses field data on Hessian fly abundance in agricultural fields to illustrate the approximation of the kernel averaged predictor of winter wheat. In chapter 5, I present the conclusions and future work.

Chapter 2

Review of numerical integration methods

Below I review two commonly used numerical integration methods in spatial applications: quadrature rule and Monte Carlo integration. I discuss using these two approaches to approximate the integral $\int_{\mathfrak{D}} f(\mathbf{s})d\mathbf{s}$, $f(\mathbf{s})$ is a two-dimensional function over the domain $\mathfrak{D}$.

2.1 Quadrature rule

To approximate $\int_{\mathfrak{D}} f(\mathbf{s})d\mathbf{s}$ using the quadrature rule, I divide the two-dimensional domain $\mathfrak{D}$ into Q subdomains of equal area, such that $\mathfrak{D} = \cup_{q=1}^{Q}\mathcal{A}_q$ and $|\mathcal{A}_q| = Q^{-1}|\mathfrak{D}|$. For each subdomain, let $\mathbf{s}_q{}^*$ be the centroid of $\mathcal{A}_q$. I approximate the integral with

$$\int_{\mathfrak{D}} f(\mathbf{s})d\mathbf{s} \approx \sum_{q=1}^{Q} |\mathcal{A}_q| f(\mathbf{s}_q{}^*). \tag{2.1}$$

Quadrature-rule is related to the commonly known Riemann sum integral approximation technique. In this report I use the term quadrature rule and Riemann sum interchangeably.

2.2 Monte Carlo integration

Monte Carlo integration uses random sampling of a function to compute an estimate of its integral numerically. Given a set of Q random sample locations over the domain $\mathfrak{D}$,

$q = 1, 2, 3, ..., Q$, and ${\mathbf{s}_q}^*$ is the sample location. The Monte Carlo approximation is

$$\int_{\mathfrak{D}} f(\mathbf{s})d\mathbf{s} \approx \frac{1}{Q}\sum_{q=1}^{Q} |\mathfrak{D}| f({\mathbf{s}_q}^*). \tag{2.2}$$

Intuitively, the Monte Carlo approximation in 2.2 computes the mean of the product of $f({\mathbf{s}_q}^*)$ and $|\mathfrak{D}|$.

Different types of approximations can cause different error terms. The error term is the difference between the approximated value and the exact value of $\int_{\mathfrak{D}} f(\mathbf{s})d\mathbf{s}$. The error bound is the theoretical largest error term, denoted by O. The quadrature rule results in an error bound $O(Q^{-2/r})$ where Q is the number of integration points, r is the dimension of $\mathfrak{D}$, which equals two in my study. The error bound then can be written as $O(Q^{-1})$. For Monte Carlo numerical integration, the error bound is $O(Q^{-1/2})$, regardless of the dimensionality of $f(\mathbf{s})$ (Liu, 2008). Theoretically, quadrature rule should converge faster (when Q increase) than Monte Carlo.

Chapter 3

Simulation study

3.1 Introduction

I conduct a simulation experiment to evaluate two numerical integration methods for the kernel averaged predictor. The results of the approximation are evaluated by comparing the reliability of the estimated regression coefficient and the run time under each computational setting.

3.2 Generating data

For my simulation, I use a unit square domain $\mathfrak{D} = [0, 1] \times [0, 1]$. I divide $\mathfrak{D}$ into 40,000 equal area square grid cells. I define one location-specific spatially correlated point-level predictor $x(\mathbf{s})$, which was generated using a low-rank Gaussian predictive process (Banerjee et al., 2008). I then randomly select the sample locations $\mathbf{s}_i$ uniformly from $\mathfrak{D}$, where $i = 1, 2, 3, ..., 1000$. For each spatial location $\mathbf{s}_i$, I use 1.2 to generate the kernel averaged predictor $\widetilde{x}(\mathbf{s}_i)$ with Q equals to the total number of the grid cells 40,000. The kernel functions I use are Gaussian kernel and uniform kernel.

After I generate $\widetilde{x}(\mathbf{s}_i)$, I then use $\widetilde{x}(\mathbf{s}_i)$ to generate $y(\mathbf{s}_i)$. I generate $y(\mathbf{s}_i)$ using four different generalized linear models with canonical link functions, which includes: Poisson, normal, binomial, and Bernoulli regression models. In my simulation study, I choose the intercept $\beta_0 = 0$ and the regression coefficient $\beta_1 = 1$ for all the regression models. The link functions I use are log, identity, and logit, for Poisson, normal, binomial and Bernoulli, respectively. The variance I choose for the normal regression model is 1. I set the total number of trials for binomial regression model as 100.

3.3 Computational settings

I use different computational settings to approximate the kernel averaged predictor $\widetilde{x}_j(\mathbf{s})$. The computational settings I evaluate consist of a combination of two factors (Table 3.1).

Table 3.1: Computational settings for the simulation study. For each kernel averaged predictor scenario, I use two types of numerical integration methods and 11 different number of numerical integration points for a total of 22 different settings.

Integration Method	**Integration Points (Q)**
quadrature rule	$50, 100, 150, 200, 250, 300$ $350, 400, 450, 500, 550$
Monte Carlo	

3.4 Simulation settings

I conduct 100 simulations for each computational setting under each generalized linear model. In each simulation of one individual computational setting, I generate a new data set, I then use 1.2 to approximate the kernel averaged predictor $\widetilde{x}(\mathbf{s}_i)$ with that computational setting. The kernel function I use for 1.2 is the same with what I use for generating the data. I then use this new predictor $\widetilde{x}(\mathbf{s}_i)$ and the response from the generated data for the generalized linear model. The regression coefficient is estimated using maximum likelihood estimation. I can then extract the 95% Wald confidence interval from the covariance matrix for $\hat{\beta}_1$. As a result, I will obtain 100 different regression coefficient $\hat{\beta}_1$ along with 95% Wald confidence interval for β_1 for each computational setting. I then compute whether the 95% Wald confidence interval cover the true value of β_1 we assumed and get the coverage probability. Using this coverage probability from 100 simulations, the 95% confidence interval of population proportion of the coverage probabilities for β_1 can be obtained. I also report the run time for each simulation setting. The run time is the time in seconds required to maximize the likelihood function using the Nelder-Mead algorithm (Nelder and Mead, 1965).

I expect that for quadrature rule, after the number of the integration points in 1.2 arrives to a specific value Q_1 and $Q_1 \ll 40{,}000$, the 95% coverage probability of β_1 will stay stable and close to 95%. Meanwhile, for Monte Carlo integration, I expect the specific value is Q_2 and Q_2 should be larger than Q_1. In addition, $\hat{\beta}_1$ will become unbiased with the number of numerical integration points increases for both methods.

3.5 Results

From my simulation study results, I observed that using quadrature rule and Monte Carlo integration had different performances on approximating the kernel average predictor. To avoid duplication, I reported the worse simulation results (Figure 3.1). Under this scenario, I used the Gaussian kernel function to approximate $\widetilde{x}(\mathbf{s}_i)$ and performed Poisson regression model to estimate β_0 and β_1. I also provided the simulation results from the other seven scenarios in appendix A.

In this scenario, the coverage probabilities of β_1 using Monte Carlo integration were much lower than using quadrature rule. The rate of convergence for using Monte Carlo integration was also slower than using quadrature rule. However, with the number of numerical integration points increased, the coverage probability increased for both numerical integration methods. Meanwhile, the trend of the distribution of $\hat{\beta}_1$ estimated from the 100 simulated data sets also highlighted the result. The variances of $\hat{\beta}_1$ were larger when using Monte Carlo integration than using quadrature rule. In addition, the quadrature rule needed smaller number of integration points than Monte Carlo integration to obtain unbiased estimates. Moreover, the run time for each computational setting had an approximately linear relationship with the number of numerical integration points for both methods. There are no big differences in the run time between the two numerical integration methods.

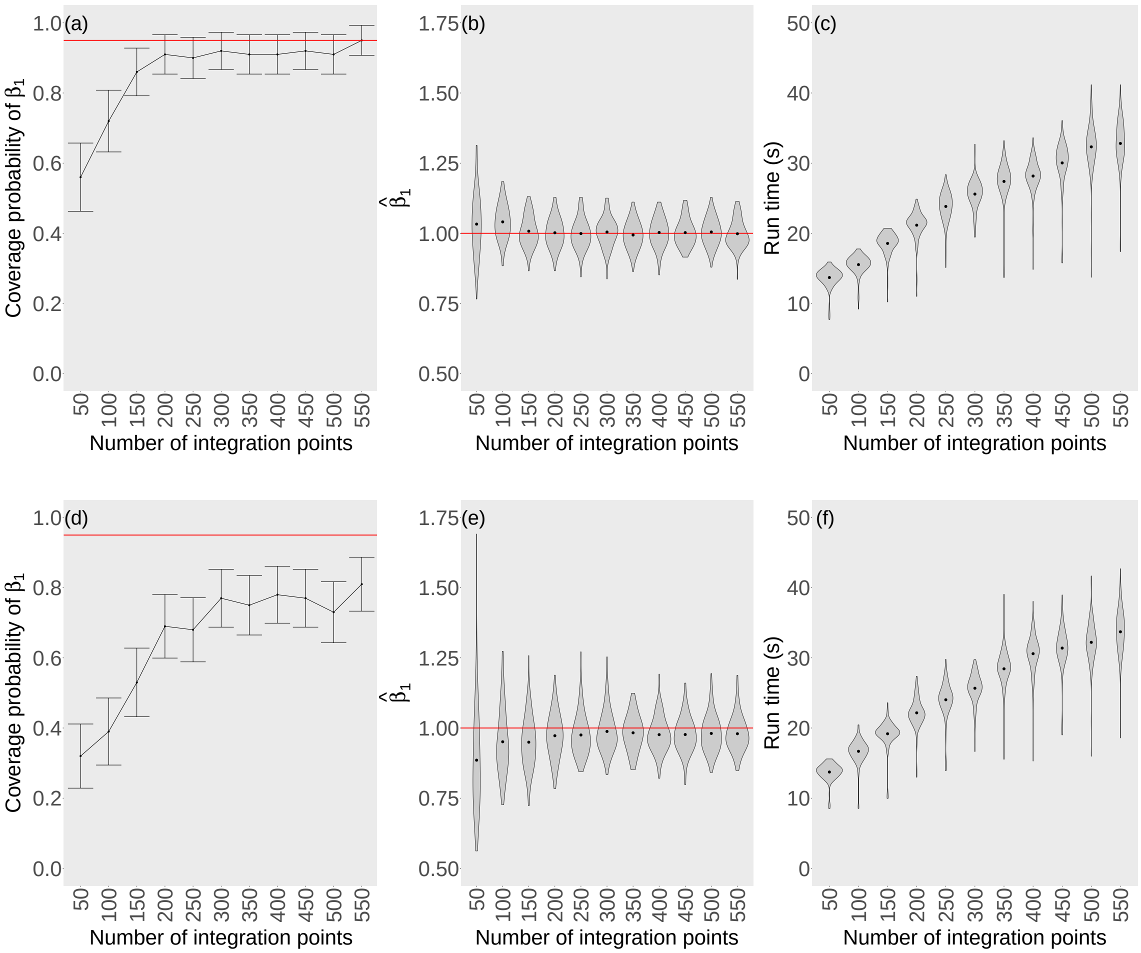

Figure 3.1: Results from the simulation experiment obtained using Gaussian kernel function and Poisson regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% *coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.*

Chapter 4

Hessian fly data example

4.1 Data

The Hessian fly, *Mayetiola destructor*, is a harmful pest and can reduce wheat yields dramatically (Schmid et al., 2018). I use the Hessian fly abundance data to illustrate how to approximate the kernel averaged predictor of winter wheat. The Hessian fly abundance data include the count of Hessian fly pupae at multiple locations within six different fields from Kansas (Figure 4.1). The sample locations chosen for counting the number of Hessian fly are laid out evenly inside each field. Because wheat is the first choice host for the Hessian fly (Harris et al., 2001; Chen et al., 2009), I expect that the value of kernel averaged predictor of winter wheat can influence the number of Hessian fly in the following spring.

4.2 Statistical analysis

The simulation study shows me that the quadrature rule performs better than Monte Carlo integration for approximating the kernel average predictor. Therefore, I use quadrature rule to approximate the kernel averaged predictor of winter wheat $\widetilde{x}_w(\mathbf{s}_i)$. The response $y(s)$ is the count of Hessian flies and $\widetilde{x}_w(\mathbf{s}_i)$ is the predictor. The kernel functions I use are the uniform kernel and truncated Gaussian kernel. I assume that the number of Hessian fly in one sample location can be influenced by at most 5 kilometers range of winter wheat away from that location. For each field, I use the same domain for all the sample locations in that field to compute $\widetilde{x}_w(\mathbf{s}_i)$. Hence, the whole domain $\mathfrak{D} = \cup_{k=1}^{6}\mathfrak{D}_k$. The total number of grid

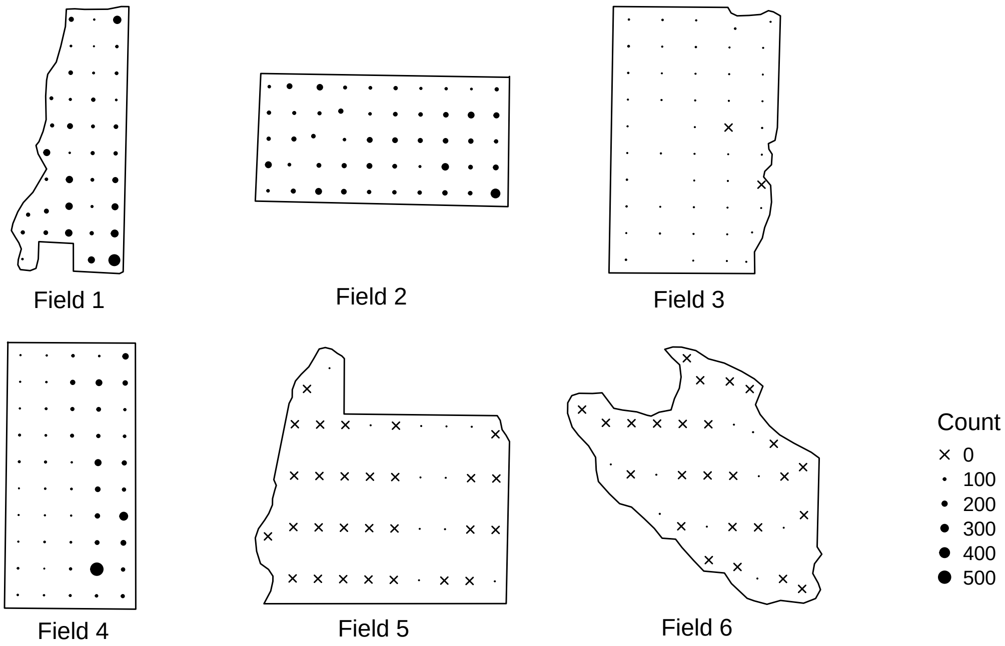

Figure 4.1: Locations and numbers of Hessian fly pupae from six fields in Kansas, USA in 2016.

cells inside the domains then results in approximate 340,000. The predictor $x_w(\mathbf{s})$ inside each grid cell is a binary predictor. I set $x_w(\mathbf{s})$ as one if the value of $x_w(\mathbf{s})$ is winter wheat, otherwise zero. I compare the estimated results using 170,000 numerical integration points with the results obtained using 340,000 points.

I assume that the number of the Hessian fly pupae has a Poisson distribution because the counts can only have non-negative integer values. Practitioners care about the relationship between the number of Hessian fly and the value of kernel averaged predictor of winter wheat. To infer this relationship, I use quadrature rule and Poisson regression model to estimate the parameters. The estimators I am interested are the kernel parameter $\hat{\phi}$, the intercept $\hat{\beta}_0$, and the regression coefficient $\hat{\beta}_1$. I report the 95% confidence interval of ϕ, β_0, and β_1.

4.3 Results

The results show that using 170,000 numerical integration points can produce close estimates with using 340,000 integration points (Table 4.1). Additionally, the results show that the expected number of Hessian flies estimated using uniform kernel is lower than using the truncated Gaussian kernel. The values of kernel averaged predictor of winter wheat is slightly larger when using a uniform kernel than using a truncated Gaussian kernel (Figure 4.2). The

Table 4.1: Maximum likelihood estimates and 95% CIs for kernel parameter and the regression coefficients: the intercept and the coefficient for kernel averaged predictor of winter wheat. The kernel averaged predictor of winter wheat is approximated using quadrature rule along with two different numbers of numerical integration points. Two different kernels are compared for the approximations. The values in the parentheses give the 95% CI for the parameters.

Kernels	Points	$\hat{\phi}$ (m)	$\hat{\beta}_0$	$\hat{\beta}_1$	AIC
uniform	170,000	2295 (2293,2298)	6.37 (6.33,6.41)	-10.89 (-11.11,-10.67)	12077
uniform	340,000	2295 (2292,2298)	6.40 (6.36,6.44)	-11.16 (-11.38,-10.93)	11762
Gaussian	170,000	1815 (1774,1857)	6.48 (6.43,6.53)	-11.09 (-11.34,-10.83)	13154
Gaussian	340,000	1813 (1770,1857)	6.44 (6.38,6.48)	-10.73 (-10.98,-10.48)	13248

Akaike information criterion (AIC) value is smaller when using a uniform kernel than using a truncated Gaussian kernel.

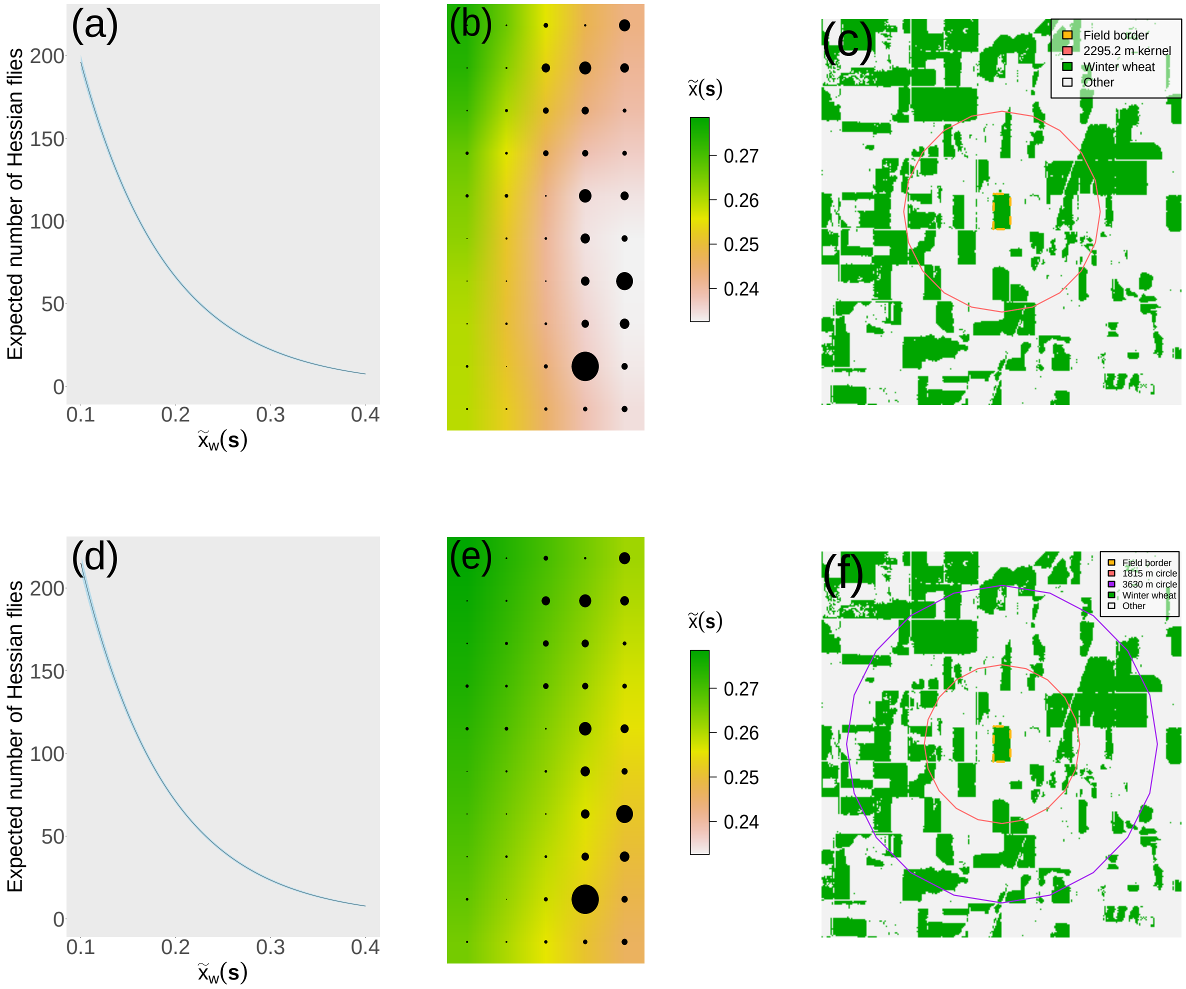

Figure 4.2: Results from approximating the kernel averaged predictor of winter wheat using 170,000 numerical integration points for uniform kernel and truncated Gaussian kernel functions with Poisson regression model. Panel a and d show the expected number of Hessian flies (black curve) with the 95% confidence interval (light blue shaded area). Panel b and e show the approximated kernel averaged predictor of winter wheat for field 4 and the number of Hessian flies. The black dots in panel b and e indicate the number of Hessian flies. Panel c and f present how the winter wheat was distributed around field 4. The red circle in panel c denotes the estimated area that contains 100% of the uniform weights. The red circle and purple circle in panel f show the estimated areas that contain 68% and 95% of the Gaussian weights when approximate the kernel averaged predictor of winter wheat using truncated Gaussian kernel, respectively.

Chapter 5

Discussion

The simulation study show that the rate of convergence for the quadrature rule is faster than the Monte Carlo integration in two dimensions, which matches the theory that quadrature rule performs better than Monte Carlo integration in low dimension (Liu, 2008). The quadrature rule requires a smaller number of integration points to obtain unbiased estimations for the parameters than the Monte Carlo integration highlight the result. Those results show that using full number of grid cells as the total number of integration points is not necessary, as using smaller number of integration points can also produce reliable and unbiased estimates. The run time of using quadrature rule is similar with using Monte Carlo numerical method, but the run time of using uniform kernel are mostly longer than using Gaussian kernel. Overall, from my simulation study, I find that quadrature rule performs better than Monte Carlo integration. Hence, I recommend using quadrature rule to approximate the kernel averaged predictors in spatial applications.

In the previous research, researchers used a one kilometer uniform kernel to approximate the kernel averaged predictor of winter wheat (Schmid et al., 2019). However, using 2295 meters (m) radius to implement the approximation is more supported by the Hessian fly abundance data. Besides, the Poisson regression model using quadrature rule along with uniform kernel to approximate the kernel averaged predictor of winter wheat has a smaller AIC value when compared to the same regression model, but using a Gaussian kernel. The smaller AIC value means that using uniform kernel is preferred to Gaussian kernel in this scenario. The negative regression coefficient show that the winter wheat proportion around one location has negative effect on the number of Hessian flies in the coming spring. Those findings help me better understand the outbreak of Hessian flies, and then implement a suitable management for the field plantings. For example, if I know the proportion of winter wheat two kilometers around one location, I can obtain an estimated number of Hessian flies at that location in the coming spring, and then I can better maintain the arrangements of field plantings. My study involves the estimation of the kernel parameter, which can provide researchers some ideas for their future research. In conclusion, the results from Hessian fly abundance data show me that two kilometers area of wheat proportion is needed, to gain a better estimation on the relationship of the number of Hessian fly and kernel averaged predictor of winter wheat. Meanwhile, a smaller number of integration points can capture adequate information for approximating the kernel averaged predictor of winter wheat.

For the future work, I can try using some other quadrature rules, such as trapezoidal, Simpson's, and Gaussian quadrature rules to approximate the kernel average predictor. Theoretically, these quadrature rules should perform better than the Riemann sum rule used in this study. In addition, using the Bayesian approach would enable me to show uncertainty in derived quality quickly. The Bayesian method has prior distribution for the parameters, which can also make the model structure more flexible.

Bibliography

Banerjee, S., Gelfand, A. E., Finley, A. O., and Sang, H. (2008). Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):825–848.

Belaire, J. A., Kreakie, B. J., Keitt, T., and Minor, E. (2014). Predicting and mapping potential whooping crane stopover habitat to guide site selection for wind energy projects. *Conservation Biology*, 28(2):541–550.

Bradter, U., Kunin, W. E., Altringham, J. D., Thom, T. J., and Benton, T. G. (2013). Identifying appropriate spatial scales of predictors in species distribution models with the random forest algorithm. *Methods in Ecology and Evolution*, 4(2):167–174.

Chandler, R. and Hepinstall-Cymerman, J. (2016). Estimating the spatial scales of landscape effects on abundance. *Landscape Ecology*, 31(6):1383–1394.

Chen, M.-S., Liu, X., Wang, H., and El-Bouhssini, M. (2009). Hessian fly (diptera: Cecidomyiidae) interactions with barley, rice, and wheat seedlings. *Journal of Economic Entomology*, 102(4):1663–1672.

Goedhart, P. W., Lof, M. E., Bianchi, F. J., Baveco, H. J. M., and van der Werf, W. (2018). Modelling mobile agent-based ecosystem services using kernel-weighted predictors. *Methods in Ecology and Evolution*, 9(5):1241–1249.

Harris, M., Sandanayaka, M., and Griffin, W. (2001). Oviposition preferences of the Hessian fly and their consequences for the survival and reproductive potential of offspring. *Ecological Entomology*, 26(5):473–486.

Heaton, M. J. and Gelfand, A. E. (2011). Spatial regression using kernel averaged predictors. *Journal of Agricultural, Biological, and Environmental Statistics*, 16(2):233–252.

Hefley, T. J., Baasch, D. M., Tyre, A. J., and Blankenship, E. E. (2015). Use of opportunistic sightings and expert knowledge to predict and compare whooping crane stopover habitat. *Conservation Biology*, 29(5):1337–1346.

Homer, C., Dewitz, J., Yang, L., Jin, S., Danielson, P., Xian, G., Coulston, J., Herold, N., Wickham, J., and Megown, K. (2015). Completion of the 2011 national land cover database for the conterminous United States–representing a decade of land cover change information. *Photogrammetric Engineering & Remote Sensing*, 81(5):345–354.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing.* Springer Science & Business Media.

Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *The Computer Journal*, 7(4):308–313.

Schmid, R. B., Hefley, T., Lollato, R., and McCornack, B. P. (2019). Landscape effects on Hessian fly, mayetiola destructor (diptera: Cecidomyiidae), distribution within six kansas commercial wheat fields. *Agriculture, Ecosystems & Environment*, 274:52–61.

Schmid, R. B., Knutson, A., Giles, K. L., and McCornack, B. P. (2018). Hessian fly (diptera: Cecidomyiidae) biology and management in wheat. *Journal of Integrated Pest Management*, 9(1):14.

Stuber, E. F., Gruber, L. F., and Fontaine, J. J. (2017). A Bayesian method for assessing multi-scale species-habitat relationships. *Landscape Ecology*, 32:2365–2381.

Appendix A

Results from other scenarios in the simulation study

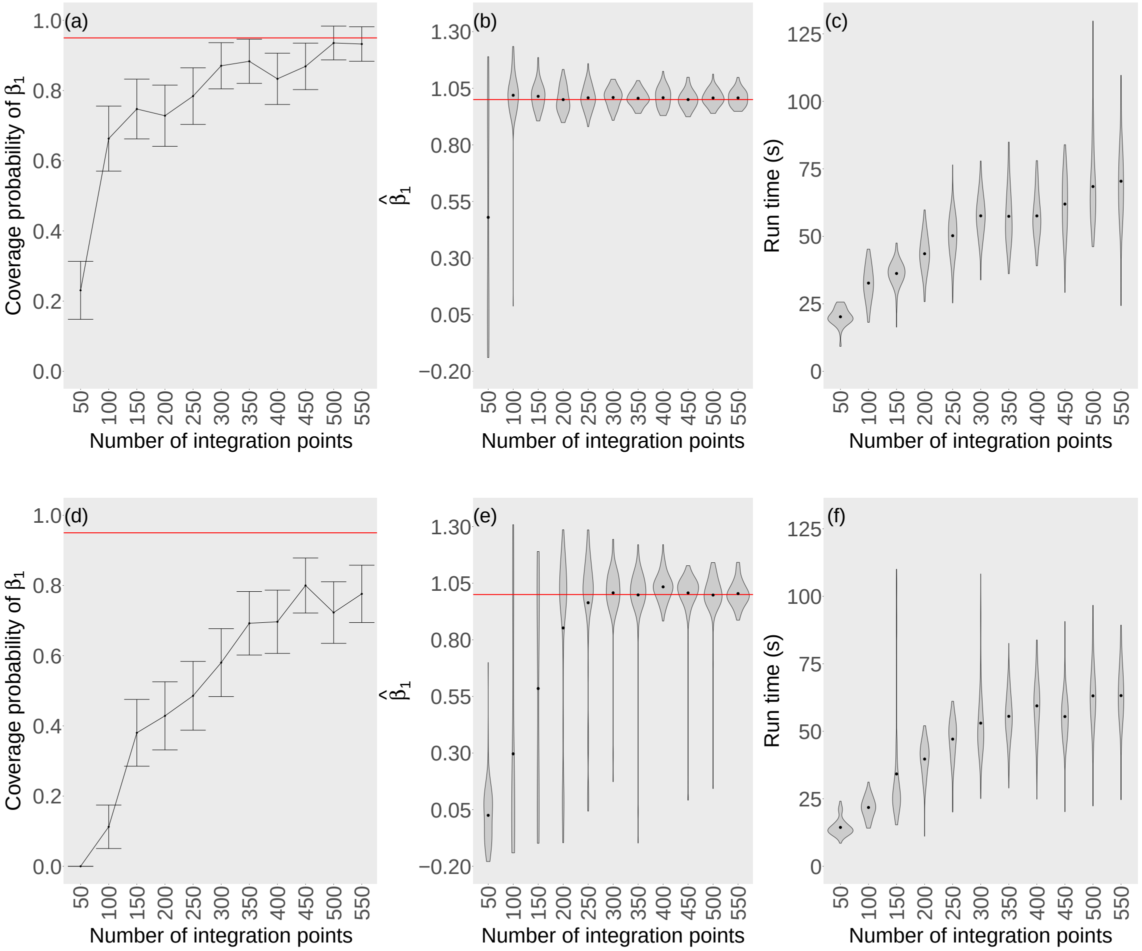

Figure A.1: Results from the simulation experiment obtained using uniform kernel function and Poisson regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% *coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.*

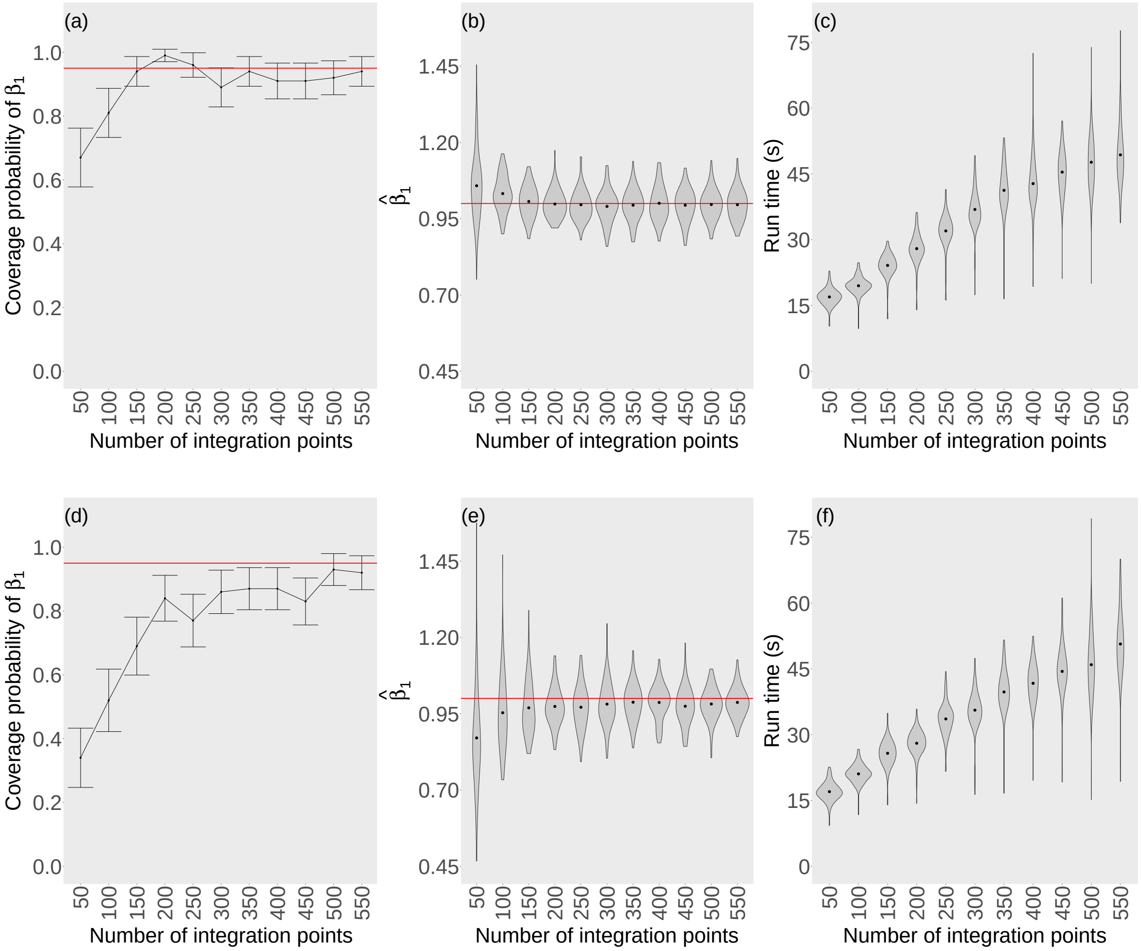

Figure A.2: Results from the simulation experiment obtained using Gaussian kernel function and normal regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.

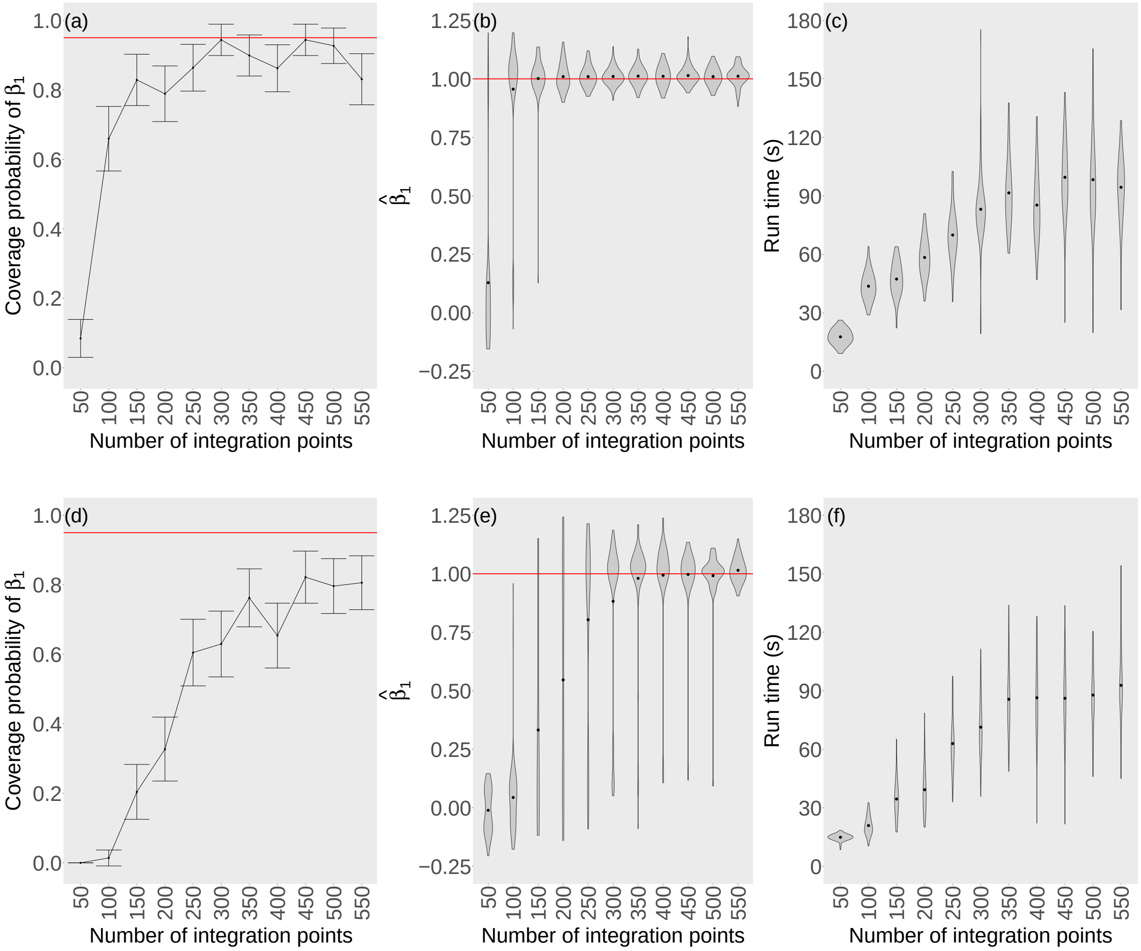

Figure A.3: Results from the simulation experiment obtained using uniform kernel function and normal regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.

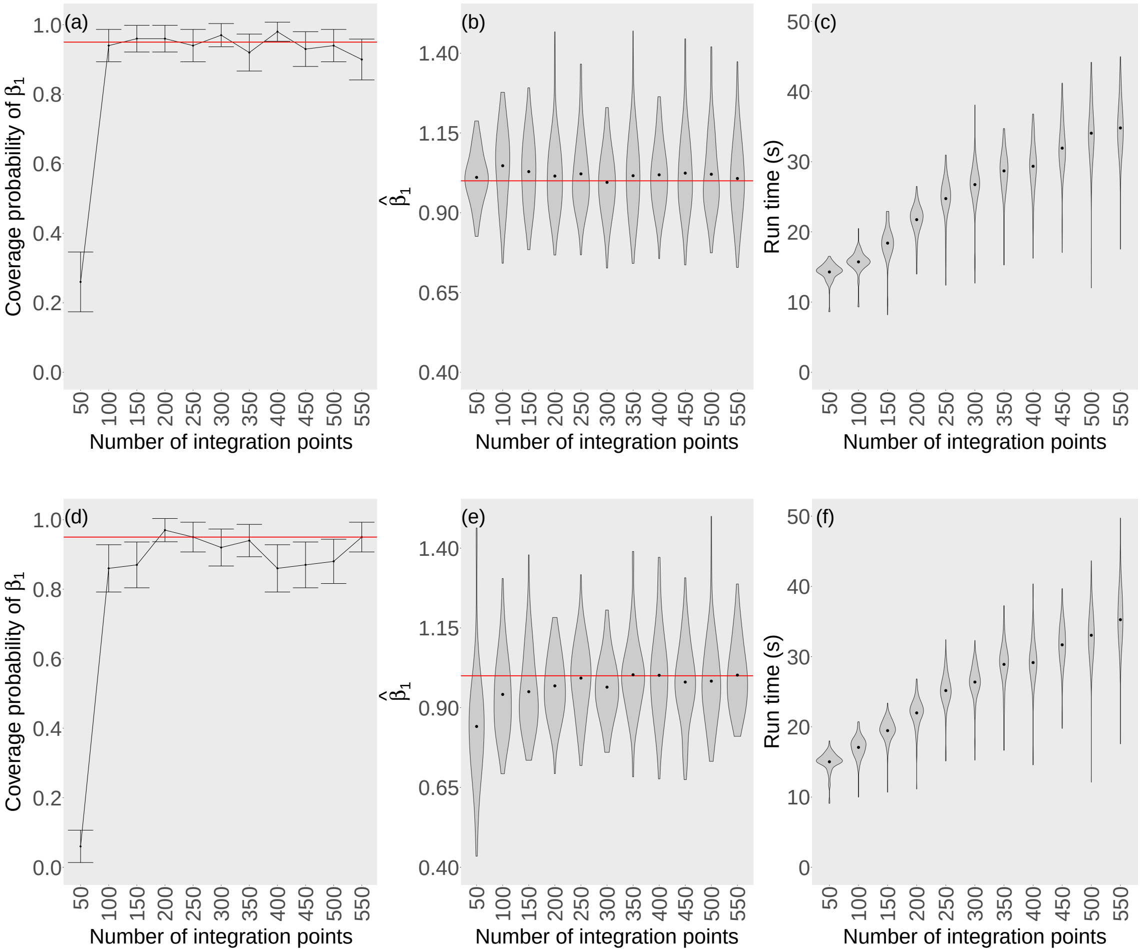

Figure A.4: Results from the simulation experiment obtained using Gaussian kernel function and binomial regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.

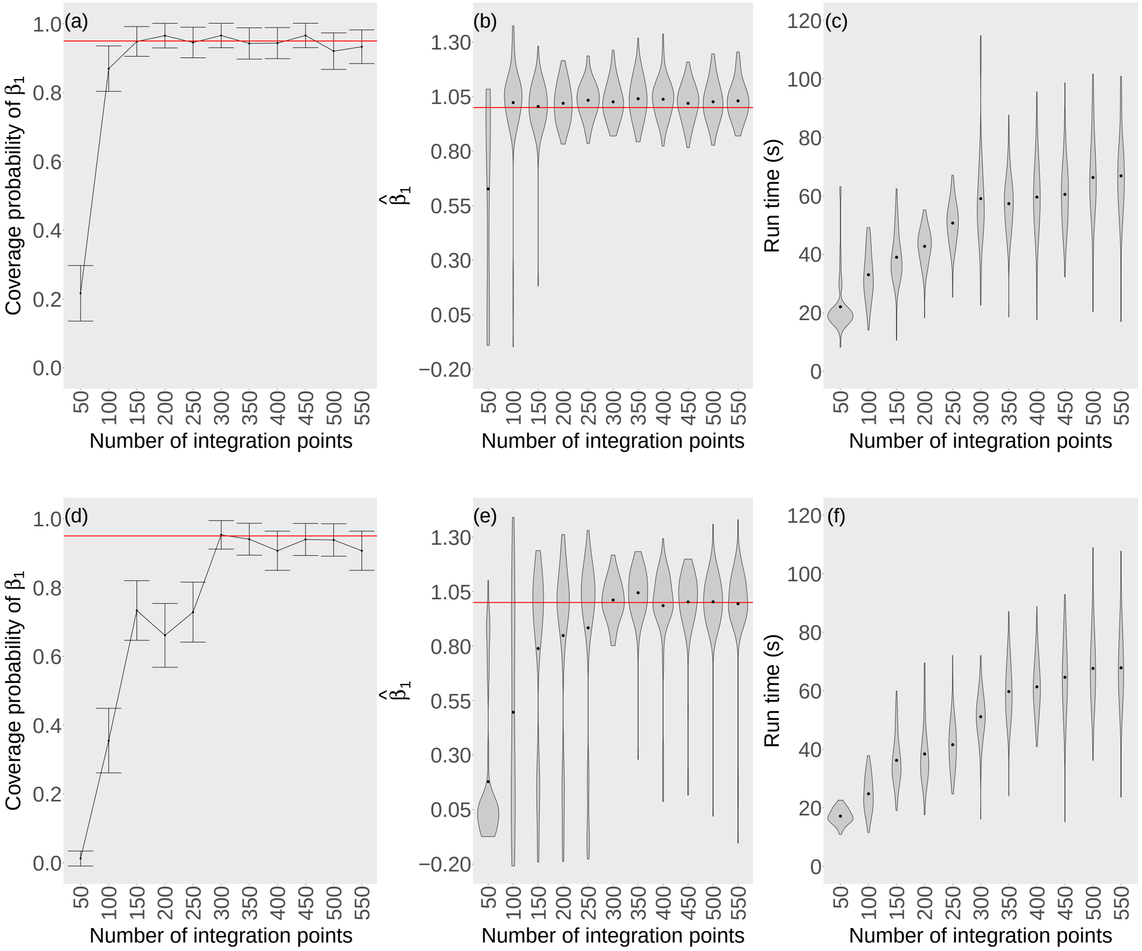

Figure A.5: Results from the simulation experiment obtained using uniform kernel function and binomial regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% *coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.*

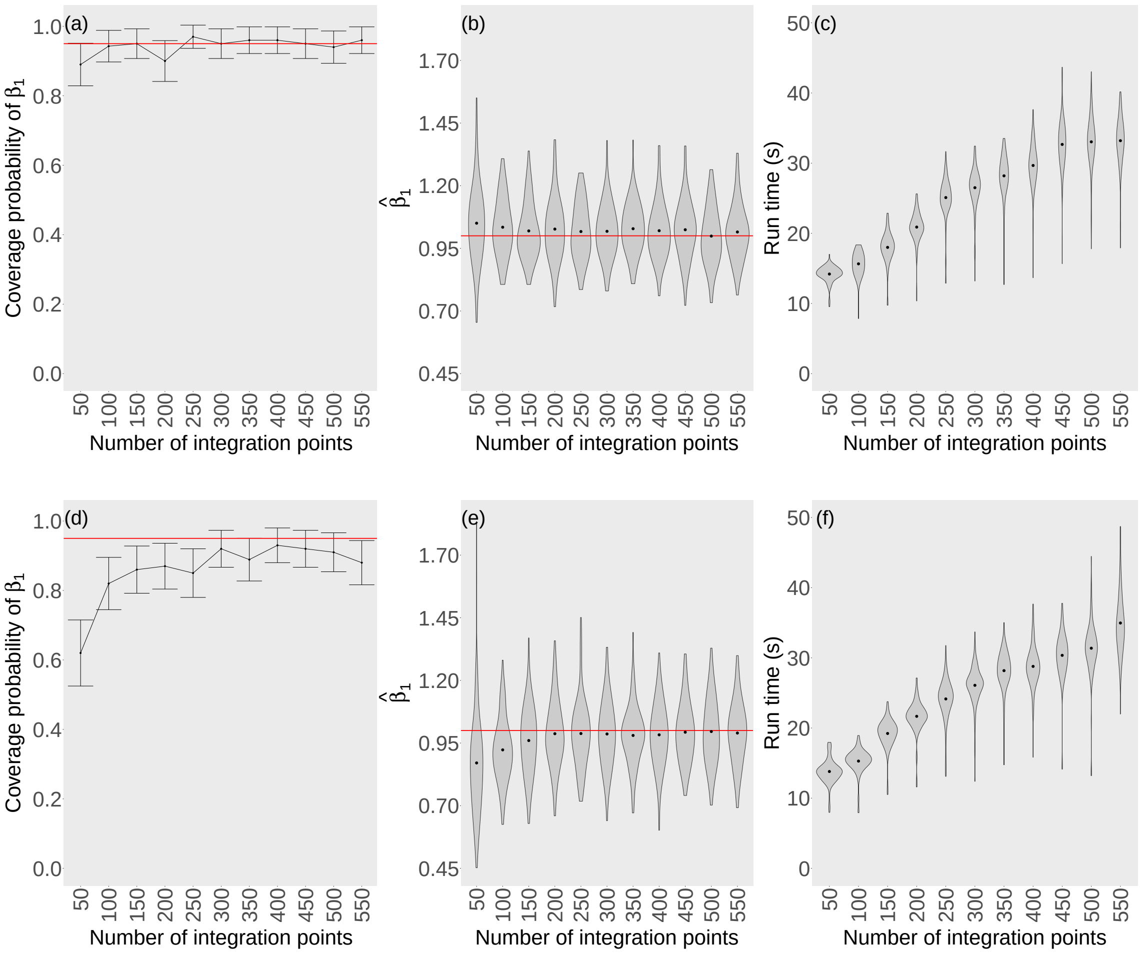

Figure A.6: Results from the simulation experiment obtained using Gaussian kernel function and Bernoulli regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.

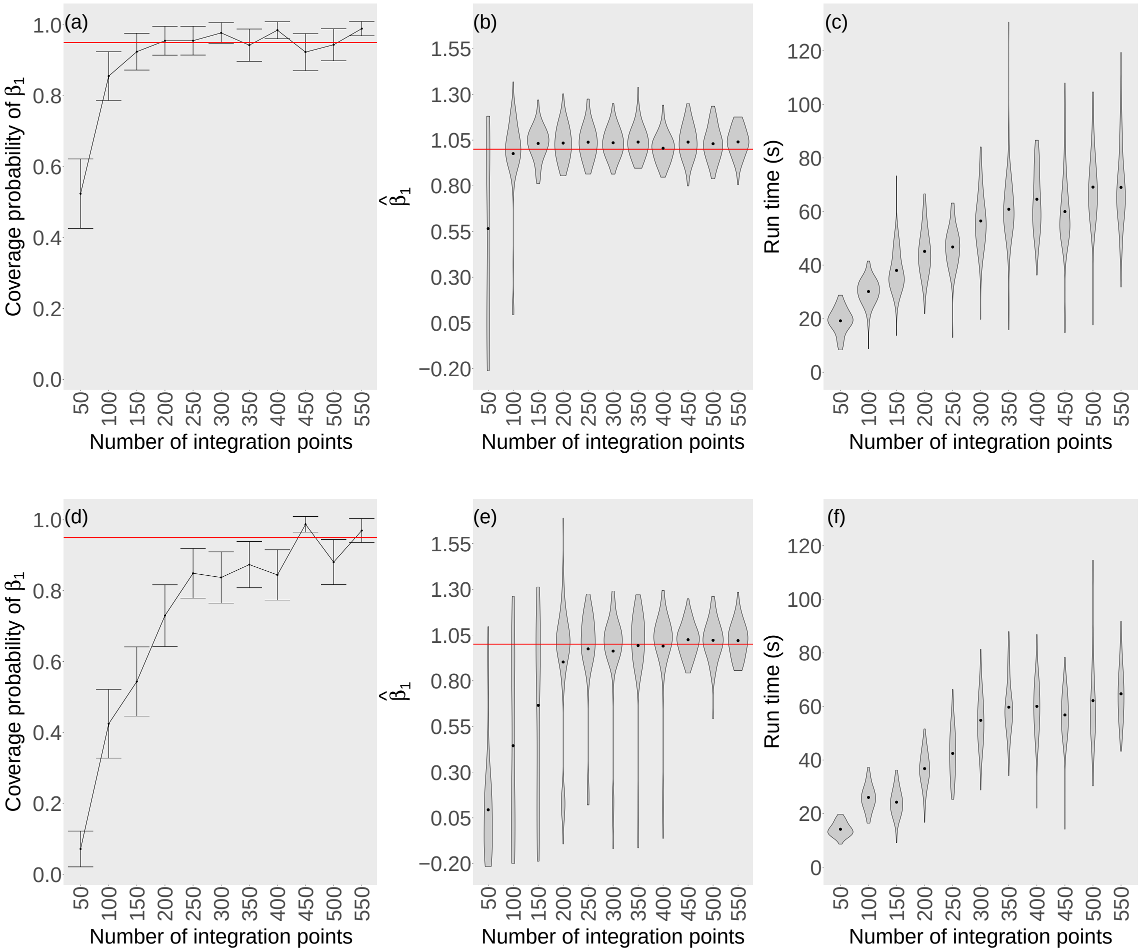

Figure A.7: Results from the simulation experiment obtained using uniform kernel function and Bernoulli regression model. Panel a and d show the 95% confidence interval of the coverage probability for the regression coefficient, β_1. The red lines in panel a and d represent 95% coverage probability. Panel b and e show violin plots for the $\hat{\beta}_1$ obtained from 100 simulated data sets. The red lines in panel b and e represent the true value which is $\beta_1 = 1$. The black dots in panel b and e represent the mean of $\hat{\beta}_1$ obtained from 100 simulated data sets for each computational setting. Panel c and f present the run time to obtain the MLE for a single data set for each simulation setting. The black dots in panel c and f represent the mean of 100 run time for each computational setting.