12-2017

A First-Order Autoregressive Hurdle Poisson Model

Gadir Abdulaziz Alomair

UNIVERSITY OF NORTHERN COLORADO

Greeley, Colorado

The Graduate School

A FIRST-ORDER AUTOREGRESSIVE HURDLE POISSON MODEL

A Dissertation Submitted in Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy

Gadir Abdulaziz Alomair

College of Education and Behavioral Sciences
Department of Applied Statistics and Research Methods

December 2017

This dissertation by: Gadir Abdulaziz Alomair

Entitled: *A First-Order Autoregressive Hurdle Poisson Model*

has been approved as meeting the requirement for the Degree of Doctor of Philosophy in College of Education and Behavioral Sciences in Department of Applied Statistics and Research Methods

Accepted by the Doctoral Committee

Trent Lalonde, Ph.D., Research Advisor

Khalil Shafie Holighi, Ph.D., Co-Research Advisor

Jay R. Schaffer, Ph.D., Committee Member

Heng-Yu Ku, Ph.D., Faculty Representative

Date of Dissertation Defense

Accepted by the Graduate School

Linda L. Black, Ed.D.
Associate Provost and Dean
Graduate School and International Admissions

ABSTRACT

Alomair, Gadir Abdulaziz. *A First-Order Autoregressive Hurdle Poisson Model.* Published Doctor of Philosophy dissertation, University of Northern Colorado, 2017.

Count regression models are used when the response variable takes count or non-negative values. Poisson and negative binomial distributions are commonly used to model count data. A frequent matter with the count data is to have an excess number of zeros that can result in overdispersed data when using Poisson or negative binomial distributions. Appropriate approaches to use when modeling excess-zero data is to use either a hurdle or a zero-inflated Poisson (ZIP) distribution. Recently, the hurdle models are commonly used in fields such as medicine, epidemiology, genetics, and marketing. Excess-zero data occur frequently as a series of data that are repeatedly measured over time as well. In this dissertation, the hurdle distribution is used to model time series data that are counts with a high frequency of zeros. Particularly, a first order autoregressive hurdle process is formulated to model excess-zero time series data. Comparisons with two existing zero-inflated time series models are presented and the models are evaluated based on their prediction capabilities. It is concluded that the developed hurdle autoregressive model provides better prediction of future observations compared to the other zero-inflated Poisson models. The three models are used to

analyze the crime data and the results show that the three models do not provide good prediction of future observations.

ACKNOWLEDGEMENTS

In the name of Allah the Merciful, there are no words to express my deepest thanks to Allah for his blessings that could never be counted. His mercy was sent to me to relieve all the stress and guide me to accomplish this work. I would like to thank my advisor Dr. Trent Lalonde for believing in me to finish up this dissertation. His support, guidance, and patience were my motivations to finish. I will always remember him when he said "you are so close", it was an honor to be one of his students. My big thanks to my co-advisor Dr. Khalil Shafie for his significant endless help during my dissertation. He was like a father who wanted to guide his child to the right path. I will never forget how kind he was when asking about my progress. Additionally, I am thankful to Dr. Jay Schaffer for his advice during my graduate experience. I am also grateful to Dr. Ku for his comments that helped improved this dissertation.

I am thankful to all my friends in the states. A big thank to my friend Dr. Hasni for being patient with me and listening to my containments all the time. I would like to also thank my friends Dr. Amani for giving me the courage to finish and Dr. Samy, her words were such a stress relief to me.

I am forever thankful for my parents, Abdulaziz and Samiah, for their prayers during my entire life. I would not be able to be where I am without their love and kindness. My thanks are sent to my siblings, Ghadah, Faisal, Abdullah,

Fahad, Sarah, and Khaled for being in my life to support me. My deep appreciation are sent to my father in-law Abdullatif and my mother in-law Lateefa for their prayers during my graduate study.

Finally, My huge thanks go to my family. I want to thank my husband, classmate, and best friend Dr. Mohammad for his help, cooperation, and great patience during my graduate study. It was such a great experience to study together, sharing the same concerns, studying together, and helping each other. I will never forget those days when we studied together in the same classrooms. I would like to thank all my kids for being such a great children during my study. My oldest son Ammar thanks for taking the responsibility to take care of his siblings. My daughter Joud thanks for being such a mother of mine and asking about my progress in studying all the time. My son Abdullatif thanks for acting like a big kid and taking care of your little sister. My little daughter Reem, thanks for being in my life to relief all the pain by seeing you playing and singing. I am so proud and grateful to have all my wonderful kids.

TABLE OF CONTENTS

CHAPTER		Page
I	INTRODUCTION	1
	Count Data	1
	Excess Zero Counts	2
	Time Series	4
	Study Objectives	7
	Research Questions	7
	Study Limitations	8
II	LITERATURE REVIEW	10
	Count Data	11
	Excess-Zero Data	11
	Reasons for Zero-Inflation	12
	Consequences of Ignoring Zero-Inflation	12
	Generalized Linear Models	13
	Zero-Inflated Poisson Model	15
	Parameter Estimation and Hypothesis Testing for Zero-Inflated Poisson	17
	Hurdle Model	17
	Parameter Estimation and Hypothesis Testing for the Hurdle Model	19
	Time Series Data	20
	Types of Time Series	21
	Autocorrelation	22
	Probability Models of Time Series	23
	Autoregressive Process	24

Parameter Estimation 26
Count Time Series Data 28
Poisson Autoregressive Time Series 28
Parameter Estimation of Time Series with Count Regression Models 32
Hypothesis Testing of Time Series with Count Regression Models 33
AR Excess-Zero Models 34
Study Rationale 40
III METHODS 42
Research Questions 43
Hurdle Autoregressive Model 43
Likelihood Based Inference 45
Parameter Estimation 46
Parameter Hypothesis Testing 48
IV RESULTS 56
Simulation Steps 57
Simulation Results for the Hurdle Autoregressive Model, the Zero-Inflated Poisson Autoregressive Model, and the Zero-Inflated Poisson INGARCH Model 60
Standard Errors with Different Sample Sizes 79
Real Data Example 80
V CONCLUSIONS AND FUTURE RESEARCH 86
Conclusions 86
Future Research and Limitations 89
REFERENCES 92
APPENDIX A R CODE FOR SIMULATIONS 98
APPENDIX B R CODE FOR THE APPLICATION 140

LIST OF TABLES

1 Parameter Values Used for Simulations 54

2 Standard Errors, Relative Bias, and MADE for Sample Size 20 62

3 Standard Errors, Relative Bias, and MADE for Sample Size 50 63

4 Standard Errors, Relative Bias, and MADE for Sample Size 100 . . . 64

5 Standard Errors, Relative Bias, and MADE for Sample Size 200 . . . 65

6 Standard Errors, Relative Bias, and MADE for Sample Size 500 . . . 66

7 Predicted Mean Squared Error for h steps-ahead prediction of size 5 . 68

8 Predicted Mean Squared Error for h steps-ahead prediction of size 20 69

9 Predicted Mean Squared Error for h steps-ahead prediction of size 25 70

10 Predicted Mean Squared Error for h steps-ahead prediction of size 50 71

11 Predicted Mean Squared Error for h steps-ahead prediction of size 100 72

12 Percentages of Complete Positive Standard Errors for Each Sample Size 80

13 Parameter Estimates and Standard Errors for Hurdle AR Model, ZIP AR Model, and ZIP INGARCH Models 83

14 Predicted Mean Squared Error for h Steps-Ahead Prediction for Arson Data 84

LIST OF FIGURES

1 Hurdle Time Series Data of size 100 60

2 Predicted Time Series Data of Size 5 74

3 Predicted Time Series Mean of Size 5 74

4 Predicted Time Series Data of Size 20 75

5 Predicted Time Series Mean of Size 20 75

6 Predicted Time Series Data of Size 25 76

7 Predicted Time Series Mean of Size 25 76

8 Predicted Time Series Data of Size 50 77

9 Predicted Time Series Mean of Size 50 77

10 Predicted Time Series Data of Size 100 78

11 Predicted Time Series Mean of Size 100 78

12 Time Series Plot for Monthly Arson Counts in Pittsburgh, Pennsylvania from 1990 to 2001 82

13 Predicted Time Series Arson Data 85

14 Predicted Time Series Arson Mean 85

CHAPTER I

INTRODUCTION

Count regression models are used when a response variable only has integer values that go from zero to infinity. These count data commonly occur in several fields. Examples of count data in the health area include the number of doctor visits per year and the number of born people in a hospital per week. In traffic movement, counts include the number of weekly car accidents in a town and the number of cars passing through red lights per day. In such situations, the response has integer values only. The early developments of count models have been used in demography, biostatistics, and actuarial science. Recently, these models have been widely used in sociology, economics, and political science. Given the importance of using the proper model for data analysis, count regression models have become extensively used in practice.

Count Data

The count response variable in either experimental or observational studies is usually assumed to follow Poisson or negative binomial distribution. However, it is not guaranteed that using one of these distributions will lead to accurate results in the presence of some features of the count data that may give more reasonable results using other types of distributions (Miller, 2007). A common feature of count data is that they are collected over a fixed period or specific area. The analysis of

count data using linear regressions can lead to problems, since they can only be nonnegative numbers. Therefore, a Poisson distribution is a more appropriate distribution when dealing with count data. The regression model that can be applied to count data is the Poisson log-linear regression model given that the log function is used to link the response to the covariates and the parameters in the model. The Poisson log-linear regression model is derived from the Poisson distribution, which assumes equal mean and variance.

Excess Zero Counts

Count data sometimes have issues of excess zeros relative to what is allowed by standard models of Poisson or negative binomial distribution. There are many more zeros in the data than expected under Poisson distribution. In such cases, the data are called excess-zero counts and should be modeled appropriately. Examples of zero-inflated data include airplane crashes, daily labor deaths, rare disease diagnosis, and bank robberies in a specific period (Miller, 2007). These examples are of events that are unlikely to happen in the given period. For more illustration, the count for rare disease diagnosis in a hospital each day for a month will likely result in data with excess-zeros because this phenomenon does not occur most days of the month by virtue of being a "rare" disease. Therefore, the data will mostly be recorded as zeros. Suitable models when the count data have excess zeros are the hurdle model, zero-inflated model, and some negative binomial variations of these models (Min and Agresti, 2005). The choice of which model is most appropriate depends on several factors. For the present study, the hurdle model is used on

account of its simplicity of interpretation, consistency of results under zero-inflated/zero-deflated data, and ease of fit relative to other excess-zero models.

Modeling excess-zero data using the Poisson distribution can lead to inefficient results. This can be attributed to the violation of the equal mean and variance assumption in Poisson distribution when there is zero-inflation in the data (Zorn, 1996). The variance of the data with zero-inflation is usually higher than the mean and causes data overdispersion. This means that the variance is underestimated, which usually leads to biased parameter estimates, underestimated standard errors, and invalid inferences. Therefore, it is crucial to model zero-inflated data properly using one of the zero-inflated models to obtain satisfying results.

The hurdle model is a popular modified model for count data, and it is used when dealing with excess-zero data. One of the salient characteristics of the hurdle model is that "it provides a natural means for modeling overdispersion and underdispersion of the data" (Mullahy, 1986, p. 54). Overdispersion and underdispersion can occur as a result of misspecification in the data generating process, for the data do not support the probabilities of zero and positive realization in the parent distribution. Mullahy (1986) stated that by allowing more flexible specification of the probabilities of zero and positive realization, the modified distributions can handle overdispersion and underdispersion in the data. The idea that underlies the hurdle model is the assumption that zeros and positive observations have two different distributions, so the model is a mixture of two distributions. The zeros in the data are associated with the first distribution, which

is a binomial distribution. The positive observations are associated with the second distribution, which is a truncated Poisson or negative binomial distribution.

Time Series

A "time series" refers to measurements of the same variables(s) taken sequentially over a given period. Time series analysis is used when data are collected over time, as the time series is a sequence of data points taken at successive points in time. In other words, the time series tracks changes in the selected data points, which underscores its utility when analyzing data. It is used in statistics, finance, weather forecasting, earthquake prediction, engineering, census analysis, stock market analysis, and astronomy. For example, in marketing people count the number of products sold daily over a period or the number of weekly items returned at a store over a period. The counts, in this case the number of products sold or items returned, are repeatedly measured over a time interval. Time series analysis considers data points collected over time; thus, such analysis may reveal an internal structure such as autocorrelation or seasonality.

The major difference between time series and standard linear regression is that observations in the former are not usually independent. Because of dependency in the time series data, ordering is one of the features of time series, and changing ordering could change the data's meaning. There are many models that describe time series data depending on the pattern of the series. Time series models are meant to describe the features of the series, explaining how past values affect future values, and forecasting. The first-order autoregressive (AR) model, AR (1), is used in this study. This model is useful when there is an assumption that the current

observation depends on the value of the past observation. In models of this type, the observation from a time series is regressed on the past observation from the same time series.

The distribution of time series data is often assumed to be continuous. However, time series for count data frequently occur in many applications. For example, in the medical field, time series involving count data can be seen in situations such as recording the monthly occurrence of rare infections over 10-year periods. In this case, the counts are usually low because of the rarity of the infection/disease which causes a high frequency of zeros in the data. Such excess-zero time series cannot be properly accommodated as regular counts using Poisson or negative binomial models (Yang, 2012). Treating zero-inflated data as Poisson or negative binomial distributed data may negatively affect the validity of the results. The equality between the mean and the variance in the Poisson distribution no longer exists when there is inflation of zeros in the data. The problem of assuming this equality is that the variance is underestimated, causing overdispersion in the data that may give inconsistent parameter estimates and inflates type I error rates.

Another issue that merits consideration is that data collected over time usually exhibit some correlation between successive observations. This correlation should not be ignored to obtain precise results. Ignoring the correlation between observations may result in inefficient estimates and underestimated residual variance, which means that standard errors will be smaller than they should be.

Together, these factors may lead to bias variance in the estimates for the coefficients and inflates type I error rates.

Over the last three decades, interest in modelling excess-zero time series data has been steadily increasing. This can be explained by the fact that time series data with low counts are encountered in practical areas. Thus, studies of AR time series with excess-zero data have been modeled. Different methods have been used to model these data with different distribution types. Previous studies have used a binomial thinning operator method, such as the study of a first-order integer valued AR process with the zero-inflated Poisson (ZIP) innovation, ZINAR (1), which was developed by Jazi, Jones, and Lai (2012). This method, which has some flaws, is explained in detail in Chapter II. It is very restricted in its construction as the generalization of multivariate processes is not obvious and the interpretations of the results are difficult (Tjøstheim, 2012a).

An alternative approach for modeling zero-inflated time series data is to write models in the context of generalized linear models. In this case, the counts are modeled indirectly by the regression models' canonical link functions. The first ZIP mixed AR model using this method was presented by Yau, Lee, and Carrivick (2004). Yet, the model was too restrictive to accurately approximate the correlation between the observations. Yang (2012) then developed both observation-driven and parameter-driven time series for ZIP and ZINB models, which served as extensions of the model proposed by Yau et al. (2004). Even though these studies used the analogue of generalized linear models, none used the hurdle model.

Most of the previous research has focused on ZIP and ZINB models. Few papers have examined time series using the hurdle model; however, a Bayesian approach has been used in the studies done by Ver Hoef and Jansen (2007) and Neelon, Ghosh, and Loebs (2013).

Study Objectives

The hurdle model was used in this study because the excess-zero data occur in real-life applications. The model itself has the advantage of modeling excess-zero data as well as zero-deflated data, which was the case when there were fewer zeros in the data than expected under the Poisson or the negative binomial distributions (Min and Agresti, 2005). It also has a simpler structure than the ZIP model, making the likelihood function easier to maximize. The AR models were also useful in time series given that they were flexible at handling a broad dimension of time series patterns. The purpose of this study was to construct a first-order hurdle autoregressive model. Moreover, estimation of the parameters of this model using MLE was performed. Another objective of this study was to compare the performance of the proposed model to that of existing time series models of counts by applying the proposed model to a real data example.

Research Questions

The following research questions are addressed in this study:

Q1 How can we write a hurdle model for excess-zero time series data?

Q2 How are the parameters estimated for the hurdle autoregressive model?

Q3 How can the proposed model be implemented using R?

Q4 How does the proposed model perform specifically using standard errors, mean absolute deviation error, and relative bias?

Q5 How can the prediction of future observations be obtained and evaluated for the proposed model compared to existing times series models of counts?

Study Limitations

In this study, only the AR time series model is addressed; however, in some cases, it may be more useful to incorporate the moving average process to account for different correlation structures in the data. Moreover, this study applies to univariate data, but, in practice, excess-zero times series data can be also multivariate. In this study, the model's application in different geographic areas was not considered. Future research should utilize spatiotemporal models to examine the data at different locations. Finally, the count portion of the hurdle model can be modeled using either the truncated Poisson distribution or the negative binomial distribution, though, in this study, only the former (truncated Poisson distribution) is applied.

This dissertation consists of five chapters. Chapter I introduces the model used in this study as well as provides a general outline of this dissertation. The chapter begins with an explanation of the basic terms used in the study. From there, various studies related to the current study are discussed, as is the proposed model. Finally, the objective of the study and the research questions are introduced.

Chapter II offers an in-depth discussion of the components used in the study, including count data, the hurdle regression model, and the concept of time series, and then connects these three components. A review of the literature is provided in

this chapter to demonstrate the gap in the body of knowledge filled by this dissertation. At the end of this chapter, the study's rationale is discussed.

Chapter III clearly formulates the proposed HAR(1) model. The process of parameter estimation is provided in detail. Finally, the simulation structure is given.

Chapter IV describes the proposed model, and an estimation method is evaluated by conducting a simulation study. A real data example is provided to demonstrate the performance of the proposed model's compared to two previous models.

In Chapter V, the results are discussed, and conclusions are provided. Limitations of the study and future work are discussed as well.

CHAPTER II

LITERATURE REVIEW

Analyzing data requires knowledge of the type of data. For many analyses, distributions of the data may follow something other than the basic assumption of normal distribution. Count data are a prime example. Count data do not follow a normal distribution, for they are integer values that cannot be negative, unlike normally distributed data. Take the example of the number of tourists per month over a 10-year period in Cyprus, or the daily number of visits to a website, or the monthly number of doctor visits. These examples are not only count data; they are also measured repeatedly over a period. In particular, these are count time series data that have special features because of the repeated measurements of the observations. It is crucial to choose the appropriate type of analysis, and the concomitant proper distribution assumptions, for the observed data to ensure accurate results (Miller, 2007).

In this chapter, count data are first reviewed. Then, there is a discussion of the special case of count data having excess-zeros in the data and how to model them using ZIP and hurdle models, in addition to a discussion of the method of estimation of model parameters and hypothesis testing. After that, time series, as well as time series characteristics and models, are explained, as are estimation of parameters, hypothesis testing, and modeling time series with count data. Previous

studies of modeling excess-zero time series data using different methods are explained. Finally, the rationale for using the proposed model is discussed.

Count Data

Count data are defined by Cameron and Trivedi (1998) as nonnegative integer value random variables; therefore, these data should follow a discrete distribution. Responses can have any integer value starting from zero, and there is no upper limit. Moreover, one feature of count data is that the mean tends to increase as the count increases, then the variance increases because of the mean-variance relationship in count data. For example, in Poisson distribution with the density function of

$$p(Y_i = y_i) = \frac{e^{\lambda_i} {\lambda_i}^{y_i}}{y_i!},$$

where $y_i = 0, 1, 2, ...,$ The mean and variance are as follows:

$$E(y_i) = \lambda_i$$

$$var(y_i) = \lambda_i,$$

where λ_i is the rate of occurrence. This shows equality between the mean and variance.

Excess-Zero Data

Count data are usually distributed as Poisson or negative binomial distributions. However, there are some situations where the assumptions of these distributions are not applied to the data. One case is when the data have excess

zeros. Having many zeros in the data is referred to as zero-inflation. Zero-inflation means having more zeros in the data than expected under Poisson distribution or any of its variations, that is, there are more zeros than any other value in the data (Zorn, 1996). This issue was recognized by Weiler (1964) ; who proposed a method for mixing two different distributions when there are many zeros in the data.

Reasons for Zero-Inflation

According to Miller (2007), there can be two sources of excess-zeros in data. The first one is called true zeros and the second is sampling zeroes. For example, when counting the number of people who have a rare disease, zeros can occur from two different sources. The first one is that the collected sample might not be close the area where the disease is prevalent; hence, true zeros. The second source is because of the disease's rarity, the sample has many zero responses even though the sampled area is near where the disease is prevalent; hence, it is called sampling zeros.

Consequences of Ignoring Zero-Inflation

One assumption of Poisson distribution is the equality between the mean and the variance. Excess-zero data can violate this assumption. Variance tends to be greater than the mean when there are many zeros in the data. Consequently, variance is underestimated, leading to overdispersion (McCullagh and Nelder, 1989). Overdispersion has unwelcome results, such as inconsistent parameter estimates and underestimated standard errors, leading to the inflation of type I error rate (Jang, 2005; Martin et al., 2005).

There are simple solutions for count data with many zeros. One is to delete all the zero responses. This may result in deleting most of the data, thereby

negatively affecting the validity of the results and resulting in biased parameter estimates given that there will not be enough data (Tooze, Grunwald, and Jones, 2002). The second solution is to assume normality of the response and use OLS to estimate the parameters. This solution can result in inconsistent estimates of the coefficients, for the excess-zero data are nonlinearly related to parameters. Another solution is to transform the response to the normal distribution. Given that count data are usually right-skewed, the appropriate transformation relies on the natural logarithm. Yet, this solution is difficult to apply to the excess-zero count data given that most of the counts are zeros and the natural logarithm of zero is not a real number (Xiao-Hua and Tu, 1999; King, 1989). Occasionally, adding small values such as 0.001 to the zeros can help to get the log transformed zeros. This approach, however, can result in inflation of the adjusted value that has been transformed. For example, if 70% of the data are zeros, the transformed distribution will have the same percent abundance of the value transformed (Delucchi and Bostrom, 2004; Miller, 2007). Moreover, it has been shown that this approach leads to a biased estimate of the parameters (King, 1989).

Generalized Linear Models

The more reasonable solution when dealing with excess-zero count data consists of specifying the data's distribution. According to Miller (2007), these nonlinear types of data can be modeled similarly to the generalized linear models (GLM). GLM allow the mean to be any possible function of the parameters and predictors instead of a linear function (McCullagh and Nelder, 1989). For all distributions under GLM, there are three components, the random component, the

systematic component, and the link function. The random component describes the distribution of the response, which must come from the exponential family. The systematic component is the linear combination of the predictors and the parameters. The link function is the function that connects the random component to the systematic components; hence it connects the outcome variable to the independent variables. The probability density function (pdf) of the exponential family has the following form:

$$f(Y;\theta,\phi) = exp(\frac{Y\theta - b(\theta)}{a(\phi)} + c(Y,\phi)),$$

where Y is the outcome variable, θ is the canonical parameter that is associated with the mean, ϕ is a scale parameter that is associated with the variance, and a, b, c are scalar functions.

In GLM, the parameters are estimated by maximizing the log likelihood of the distribution given the observed data (McCullagh and Nelder, 1989). The mean and the variance can be obtained using the following two functions:

$$E(Y) = b'(\theta),$$

$$var(Y) = b''(\theta)a(\phi).$$

One of the advantages of using GLM is that several link functions can be used to suit the different distributions (Miller, 2007). For instance, the binomial distribution has the logit of the response as the link function and the Poisson distribution has the log of the response as the link function.

In the case of excess-zero data, there are two sources of inequality between the mean and the variance that due to the heterogeneity of the outcomes and that due to zero-inflation in the data (Miller, 2007). Previous studies have shown that it is appropriate to use a mixture of two different distributions when dealing with these types of data. Zorn (1996) argued that dual regime models should be used when there is zero-inflation in the count data. The dual regime consists of what is called a transition stage, which is associated with the binomial distribution, and another stage associated with having an event that has a Poisson distribution.Delucchi and Bostrom (2004) justified the use of the two distributions. Their approach was based on determining the difference, and significance, between the proportion of subjects with true zeros and the proportion of subjects with sampled zeros. There are other approaches that assume one source of zeros and the other distribution is associated with the positive observations only. In sum, there are several ways to model zero-inflated count data, and these ways are discussed in the following sections.

Zero-Inflated Poisson Model

One approach to modeling excess-zero data is to use the zero-inflated Poisson (ZIP) model. This model was proposed by Lambert (1992). In the data used by Lambert, around 50% of the data were zero. Formally, there is a method to test whether the ZIP model should be used instead of Poisson distribution, the score test proposed by Van den Broek (1995). The assumption of ZIP, according to Lambert (1992), means two sources for zeros and thus two distributions. The first source considers zero as a regular count for a Poisson distribution. The second source is associated with only producing zeros; hence, it has a different distribution. For

example, the counts of the situation of having syphilis in a hospital in a fixed time: Zeros can occur in the data for two reasons. The first reason is that the observed subjects did not get infected because they were not surrounded by people who had this rare disease, resulting in all zeros in the data. The second reason is that subjects were exposed, but this disease is uncommon, which results in zeros. The second source is associated with the Poisson-distributed part. Consequently, the ZIP model is applicable when the dataset is assumed to have two reasons for zero inflation (Min and Agresti, 2005) and zero can occur in the two different stages (Cameron and Trivedi, 1998). Thus, ZIP has two systematic components for the two distributions. The first one has a binomial distribution, and the second one has a Poisson distribution. The pdf of the ZIP model can be written as:

$$f(Y = y; \pi, \lambda) = \begin{cases} \pi + (1 - \pi) f_{Poisson}(0), & y = 0 \\ (1 - \pi) f_{Poisson}(y), & y > 0 \end{cases}$$

where $f_{Poisson}$ is the pdf for Poisson distribution and π is the probability of getting zeros in the data. The first part of the ZIP data generating process has a binomial distribution and results in only zeros. The second part has a Poisson distribution, which results in counts that start from zero and are not bounded by any positive number (Cameron and Trivedi, 1998). The theory behind the mixture distributions of ZIP is that the data have two different stages, the transition stage that addresses the inflation of the zeros and the event stage that represents the unobserved heterogeneity of the outcomes with the zero as part of the counts (Jang, 2005).

There are three components associated with the ZIP model. First, the random component, a mixed distribution previously mentioned $Y \sim ZIP(\pi, \lambda)$. Second, the systematic components of ZIP model are $\eta_{1i} = X_1\beta_1$ and $\eta_{2i} = X_2\beta_2$, where X_1 and β_1 are the design matrix and the parameter vector corresponding to the logistic part of the model, X_2 and β_2 are the design matrix and the parameter vector corresponding to the Poisson part of the model. Third, the link functions are $\eta_{1i} = logit(\pi)$ for the binary part and $\eta_{2i} = ln(\lambda)$ for the Poisson part.

Parameter Estimation and Hypothesis Testing for Zero-Inflated Poisson

The MLE is used to estimate the parameters of ZIP. Lambert (1992) derived the log likelihood function of ZIP:

$$\ell(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2) = \sum_{y_i=0} log(e^{\boldsymbol{X}_{1i}\boldsymbol{\beta}_1} + exp(-e^{\boldsymbol{X}_{2i}\boldsymbol{\beta}_2})) + \sum_{y_i>0}(y_i\boldsymbol{X}_{2i}\boldsymbol{\beta}_2 - e^{\boldsymbol{X}_{2i}\boldsymbol{\beta}_2}) - \sum_{i=1}^{n} log(1 + e^{\boldsymbol{X}_{1i}\boldsymbol{\beta}_1}) - \sum_{y_i>0} log(y_i!),$$

where $\boldsymbol{X}_1$ and $\boldsymbol{\beta}_1$ are the matrix and the vector of parameters corresponding to the event stage, and $\boldsymbol{X}_2$ and $\boldsymbol{\beta}_2$ are the matrix and vector of the parameters of the values for the transition stage Lambert (1992). The estimation of the coefficients can be obtained by taking the partial derivatives with respect to each parameter and setting it equal to zero. In terms of hypotheses testing for the ZIP model, the hypothesis that $\boldsymbol{\beta} = B$ could be tested using Wald tests, $Z_i = \frac{(\hat{\beta}_i - B)}{SE_{\hat{\beta}_i}}$.

Hurdle Model

Another method that handles excess-zero data and gives consistent parameter estimates is what is called a hurdle model. The hurdle model is popular

in econometrics (Thomas, 2010). This model was proposed by Mullahy (1986). The concept underlying the hurdle approach is that there is a transition stage that is associated with having only zeros in the data and an event stage that is associated with having only positive values in the data (Mullahy, 1986). The event stage with only positive values is, for example, a truncated negative binomial or left-truncated Poisson distribution that is cut out of zero values, which is the distribution assumed in this study (Min and Agresti, 2005) . Specifically, the hurdle model has a mixture of distributions, as does the ZIP model. However, in the hurdle model, there is a binary distribution that is related to only zero values and another truncated count distribution that is related to only positive values. Mullahy (1986) treated zeros as a separate case from the positive values because he assumed that one process generates zeros and one generates positive values. The hurdle model has a pdf of,

$$f(Y = y; \pi, \lambda) = \begin{cases} \pi, & y = 0 \\ (1 - \pi)\frac{\lambda^y}{y!(e^\lambda - 1)}, & y > 0. \end{cases}$$

Although the first part of the data generating process has a binary distribution, as does the ZIP distribution, the second part is obviously different. It follows a left-truncated Poisson distribution, which is a conditional Poisson distribution, for all the values in this part are strictly positive. The part that corresponds to the count is conditioned on giving positive outcomes, that is, when the hurdle is crossed. There are three components for hurdle model, as there are for the GLM. The random component is $y \sim hurdle(\pi, \lambda)$. There are two systematic components

which are $\eta_{1i} = \boldsymbol{X}'_1\boldsymbol{\beta}_1$ and $\eta_{2i} = \boldsymbol{X}'_2\boldsymbol{\beta}_2$, where $\boldsymbol{X}_1$ and $\boldsymbol{\beta}_1$ are the design matrix and the parameter vector corresponding to the logistic part, $\boldsymbol{X}_2$ and $\boldsymbol{\beta}_2$ is the design matrix and the parameter vector corresponding to the truncated Poisson part of the model. Finally, the link functions are $\eta_{1i} = logit(\pi)$ for the binary part and $\eta_{2i} = ln(\lambda)$ for the truncated Poisson part.

Examples to demonstrate the idea underlying the hurdle model include the number of people caught drunk while driving in a month. In this model, the assumption is different than it is for the ZIP model. In this case, it is assumed that there is a reason why no people caught drunk (i.e., only zeros). In contrast to the ZIP, it is assumed that there is another process that generates only positive values. Therefore, all the zeros have a binary distribution and the positive counts that are truncated from zero have a left-truncated Poisson distribution.

Parameter Estimation and Hypothesis Testing for the Hurdle Model

The hurdle model's parameters can be estimated using MLE, yet, unlike the ZIP model, the likelihood function can be maximized separately for each component, giving the hurdle model the advantage of an orthogonal parameterization, which makes it easier to fit and interpret than the ZIP model (Miller, 2007; Welsh, Cunningham, Donnelly, and Lindenmayer, 1996). Mullahy (1986) developed the log likelihood function as follows:

$$\ell(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2) = \sum_{y_i=0} \boldsymbol{X}'_{1i}\boldsymbol{\beta}_1 - \sum_{i=0}^{n} log(1+e^{\boldsymbol{X}'_{1i}\boldsymbol{\beta}_1}) + \sum_{y_i>0} [\boldsymbol{X}'_{2i}\boldsymbol{\beta}_2 y_i - log(y_i!) - log(exp(e^{\boldsymbol{X}'_{2i}\boldsymbol{\beta}_2} - 1))].$$

Similar to the ZIP model, after parameter estimation, hypotheses can be tested using the Wald test. The Wald z-test of the hypothesis that $\boldsymbol{\beta}_i = B$ is given by

$$Z_i = \frac{(\hat{\beta}_i - B)}{SE_{\hat{\beta}_i}}$$

Wald test is based on the parameter estimates and their standard errors derived using maximum likelihood in the case of the hurdle model. For a test with two tails, χ^2 with one degree of freedom can be used, which is equivalent to the squared Wald test (Agresti, 2007). Wald statistics are very similar to the partial t-statistics in the classical regression.

Time Series Data

A time series is a collection of observations made sequentially over time (Chatfield, 2003). Time series examples can be seen in many fields, such as economics, the physical sciences, engineering, and medicine. For example, in economics, researchers may record a company's average income in successive months, profits in successive months, or the average price of a product in different seasons. In the physical sciences, researchers may record average amount of rainfall in successive weeks, air temperature on successive days, and humidity in successive months. In marketing, time series analysis can be conducted to track the sale of a

specific product in successive months over the course of several years. Additionally, time series can be used when the observations measured are binary (i.e., they only can be 0 or 1). An example of such binary measurements is the recording of "on" and "off" of a light over successive hours. There are several reasons for using time series analysis. First, it can be used for descriptive purposes. For example, when plotting data collected over time, it can easily be noticed if there is a seasonal effect on prices, turning points, or outliers. The second objective of time series analysis is explanatory. This can be used when observations are taken over two or more variables; in such cases, one can use a time series to explain the variation in another time series. Furthermore, time series analysis can be used to predict future observations. Finally, time series is very useful for process control. Time series can measure the quality of manufacturing in an effort to control the manufacturing process (Chatfield, 2003).

Types of Time Series

Times series can be discrete or continuous depending on the measurements. A continuous time series refers to observations taken continuously over time even if the measured variable is discrete. In contrast, a time series is called discrete when the observations are measured at set time points even if the variable being measured is continuous. One of the features of time series is that the successive observations are not independent of each other because the same observations are measured repeatedly. As a consequence of this feature, future observations in time series can be predicted from past observations. Time series can be completely predictable, which is called deterministic, or partly predictable, which is called stochastic

(Chatfield, 2003). There are also two well-known models in time series, namely parameter-driven models and observation-driven models. The general difference between these types of models comes from the way that they account for autocorrelation. For parameter-driven models, if we let Y_t be the observation at time t and x_t the explanatory variable at time t, then a Poisson parameter-driven model has the mean at time t as $\mu_t = exp[\boldsymbol{x}_t'\boldsymbol{\beta} + \alpha_t]$, where the vector $\boldsymbol{\beta}$ has the coefficients and α_t is the stationary Gaussian process. In this model, the correlation between the observations is generated by a latent process (Y. Wang, 2002). The observations are assumed to be independently distributed conditioning on the latent process. The coefficients of the regression are interpretable on the scale of log mean. The estimation of the parameters, however, is difficult given that the likelihood function involves the fold integral of the sample size (Liu, 2012). Recently, Davis and Yau (2011) constructed a pairwise likelihood method to estimate parameters.

The other type is observation-driven models. In contrast to parameter-driven models, the conditional mean in observation-driven models relies directly on the previous observations, as the name suggests (Liu, 2012). Because the likelihood can be calculated directly, the estimation of the parameters can be obtained easily. A complete demonstration of the difference between the two models is shown by Nelson and Leroux (2006). In this dissertation, the focus is on observation-driven models.

Autocorrelation

Given that measurements of the observations in time series analysis are collected over time, the correlation between different observations at different times is an important measure. This correlation can be measured using the sample

autocorrelation coefficients, alternatively called the serial correlation coefficients. The idea here is similar to that of the regular correlation coefficient between two variables. In time series, any two different observations, no matter how much they are apart, can be considered two different variables; thus, the correlation between them can be measured. The correlation between two observations that are k times apart is called the autocorrelation coefficient at lag k and can be calculated by the following formula:

$$r_k = \frac{\sum_{t=1}^{N-k}(Y_t - \bar{Y})(Y_{t+k} - \bar{Y})}{\sum_{t=1}^{N}(Y_t - \bar{Y})^2}.$$

where N is the total number of observations, Y_t is the observation at time t, Y_{t+k} is the observation at time $t + k$, and $\bar{Y}$ is the overall mean. The autocorrelation coefficients can be interpreted using the correlogram, which is a graph of r_k against k.

Probability Models of Time Series

Various types of time series models are considered under what is called the stochastic process. A stochastic process is "a statistical phenomenon that evolves in time according to probabilistic laws" (Chatfield, 2003, p.27). Mathematically, it is an assortment of ordered random variables presented at a set of time points. The random variable Y at time t can be denoted as Y_t. Examples of stochastic processes include the number of car accidents in successive days in a specific town, the number of deaths in a city in successive weeks, and the air temperature in an area in successive days.

One important class of stochastic processes is the stationary time series. A time series is said to be stationary if there is no systematic change in the mean and the variance and periodic variations have been removed (Chatfield, 2003). In other words, it is a time series whose properties such as mean, variance, and autocorrelation do not change over time. Time series can be strictly stationary when the distribution of $Y(t_1), Y(t_2), ..., Y(t_k)$ is the same as the distribution of $Y(t_1 + \tau), Y(t_2 + \tau), ..., Y(t_k + \tau)$, where t_i is the time andτ is a constant. That means shifting the time t_i by τ does not affect the distribution.

Autoregressive Process

There are many time series models but the focus in this study is the AR model. A process Y_t is said to be an AR process of order 1 (i.e., AR(1)) if for a purely random process (or white noise), Z_t with mean zero and variance ${\sigma_z}^2$,

$$Y_t = \alpha Y_{t-1} + Z_t.$$

This is called a first-order process because Y_t is regressed on only the past value Y_{t-1} , meaning that the value of the current observation Y_t depends on the value of the previous observation Y_{t-1}. First-order AR is intensively used in time series because of its simple structure and its wide range of applications (Cui, 2009). A process of order p can be written as

$$Y_t = \alpha_1 Y_{t-1} + ... + \alpha_p Y_{t-p} + Z_t.$$

This looks like a multiple linear regression where the final Y_t is regressed on the previous p values. Z_t is a sequence of uncorrelated, identically distributed random variables that called "white noise" process (Thombs, 1986). From the equation of AR(p), it appears that the values of the variable Y_t are dependent on past values of the process. The behavior of the autocorrelation function, however, suggests that the values of Y_t depend only on the previous value Y_{t-1} , which is the main characteristic of the AR process. AR(1) can be written as

$$Y_t = \alpha(\alpha Y_{t-2} + Z_{t-1}) + Z_t$$

$$= \alpha^2(\alpha Y_{t-3} + Z_{t-2}) + \alpha Z_{t-1} + Z_t$$

The pth order AR model can be written as

$$Y_t - \mu = \alpha_1(Y_{t-1} - \mu) + \alpha_2(Y_{t-2} - \mu) + ... + \alpha_p(Y_{t-p} - \mu) + Z_t. \quad (1)$$

The pth order AR model generates correlated successive random variables. Given that correlation, the autocovariance between the observations is nonzero. Multiplying Equation (1) by $Y_{t-k} - \mu$, where $k = 0, \pm1, \pm2, ...$; and taking the expectation, the autocovariance function is given by,

$$\gamma(k) = E[(Y_t - \mu)(Y_{t-1} - \mu)]$$

$$= \alpha_1 E[(Y_{t-1} - \mu)(Y_{t-k} - \mu)] + ... + \alpha_p E[(Y_{t-p} - \mu)(Y_{t-k} - \mu)] + E[Z_t(Y_{t-k} - \mu)]$$

$$= \alpha_1\gamma(k-1) + ... + \alpha_p\gamma(k-p)$$

The autocorrelation function is defined as

$$\rho(k) = corr(Y_t, Y_{t-k}) = \frac{\gamma(k)}{\gamma(0)}, k = ..., -2, -1, 0, 1, 2, ...,$$

where $\gamma(0)$ is given by

$$\gamma(0) = \frac{\sigma^2}{1 - \rho(1)\alpha_1 - ... - \rho(p)\alpha_p}.$$

As k becomes larger, the autocorrelation function goes toward zero for stationary AR processes. In the case of AR(1), set $\rho(k) = \alpha_1^k$, where the absolute value of α_1 is less than 1 and $\rho(k)$ decreases toward zero (Thombs, 1986). The stationarity of the covariance means that the covariance function does not depend on the time origin; rather it depends on the time differences.

Parameter Estimation

In AR(p), $\alpha_1, ..., \alpha_p$ are parameters that need to be estimated. Some estimates are based on a single observation of fixed length of AR(p), such as Yule-Walker, least squares estimator, and MLE. Yule-Walker is named after G.U. Yule and Sir Gilbert Walker (Chatfield, 2003). The autocorrelation function $\rho(k)$ satisfies the equation.

$$\rho(k) = \alpha_1\rho(k-1) + ... + \alpha_p\rho(k-p).$$

The Yule-Walker equations can be obtained by substituting $k = 1, 2, ..., p$ in the previous equation. In so doing, the parameters can be estimated by replacing the autocorrelations with the estimates.

Another estimator for the parameters are the least squares estimates. The idea of the least squares estimates is based on minimizing the squared error distance

given by

$$Q = \sum_{j=p+1}^{n} Z_j^2 = \sum_{j=p+1}^{n} (Y_j - \alpha_1 Y_{j-1} - ... - \alpha_p Y_{j-p})^2.$$

Taking the derivative of this equation and setting it equal to zero gives the least squares estimate. It is asymptotically identical to the Yule-Walker estimates. One of the crucial assumptions of the least squares estimates is the linearity of the response. Therefore, it is not appropriate to use with all types of data.

MLE is another method that can be used to estimate parameters. MLE is a good approach insofar as it does not require the linearity of the response, which means it can be used for various distributions. MLE is a method of estimating the parameters given the observations by obtaining values of the parameters that can maximize the likelihood function. To obtain MLE, the density function of all the observations should be specified as

$$f(y_1, y_2, ..., y_N|\mathbf{\Theta}).$$

Then, by considering the observations $y_1, ..., y_N$ fixed parameters and the vector of parameters $\mathbf{\Theta}$ to be variable, the likelihood function can be written as follows,

$$L(\mathbf{\Theta}|y_1, ..., y_N).$$

Then, in practice, the natural logarithm is usually obtained,

$$\ell(\mathbf{\Theta}|y_1, ..., y_N).$$

Finally, the MLE is obtained by finding the values that maximize the log-likelihood function. The choice between least squares estimates, Yule-Walker, and MLE depends on several factors, including the type of distribution of the response, whether it is possible to know the distribution of the error terms, and whether the observations are independent of each other.

Count Time Series Data

Poisson Autoregressive Time Series

The AR model is a very critical process in time series analysis as it is one of the most used models (Tjøstheim, 2012b). Regularly, in time series, the response at time t, Y_t, is assumed to be continuous and can be any value. Non-Gaussian time series data has recently shed light on common occurrences of many applications for time series in many fields (Ferland, Latour, and Oraichi, 2006). Examples of time series for counts occur in many fields, especially in epidemiology, such as the example mentioned by Ferland et al. (2006) about the number of cases of campylobacteriosis infection for ten successive years, for which the series was recorded every 28 days. In this example, instead of the unrealistic assumption of homogeneous variance, the variance was assumed to change with level, that is, more realistically. Other examples in daily life include the number of claims reported each month by insurance companies for a year or the daily number of clients that connect to computer server (Kedem and Fokianos, 2002).

Previously, much work has been done on extending the univariate case in time series to a multivariate with stationary, nonstationary, linear, and nonlinear cases. That being said, there was not much work when the observations are counts.

The reason research is behind on time series analysis for count data is that problems appear when dealing with count data. Take, for instance, the AR model of order p,

$$Y_t = d + \sum_{i=1}^{p} a_i Y_{t-i} + e_t.$$

Obviously, Y_t must be integer values, but, if we take the $a_i's$ and e_t to be real values, then the values of Y_t will not be integers, whereas taking $a_i's$ and e_t to be integer values will result in nonstationary models (Tjøstheim, 2012b).

One of the solutions to the problem of securing the model is using what is called the thinning operator method, first developed by McKenzie (1985, 1988) and Al-Osh and Alzaid (1987) in research on the first-order nonnegative integer valued AR (INAR(1)) process. Recently, the literature has included uses of these types of models, such as the papers by McKenzie (2003),Drost, Van den Akker, and Werker (2008a),Drost, Van den Akker, and Werker (2008b), Weiß (2008), and Jung, Liesenfeld, and Richard (2011). Thinning operators use a combination of integers and real values of $a_i's$ and e_t . Let Y be the random variable of integer observations, Z_t be a sequence of Bernoulli random variables, and α be the probability of success, such that $p(Z_t = 1) = 1 - p(Z_t = 0) = \alpha$ Let $\circ$ be defined using the following operation,

$$\alpha \circ Y = \sum_{t=1}^{Y} Z_t,$$

where on the Y being given, the variable $\alpha \circ Y$ has a binomial distribution. Then, the first-order AR model using the thinning operator approach is

$$Y_t = \alpha_1 \circ Y_{t-1} + e_t,$$

where e_t is a vector of nonnegative integer valued random variables that are independent and identically distributed and independent of all the counting series $Z_i's$. This model can be used for birth-death models and can potentially model the count of stock processes because it has the advantage of having an integer valued innovation process (Tjøstheim, 2012b). Yet, this model is somewhat limited in use. First, it is a very restricted approach, for the autocorrelation must be positive. Moreover, the generalization of this approach to multivariate outcomes is not clear. Furthermore, in the second order model, the Bernoulli variables $Z_i's$ used to realize α_1 must be independent of those $Z_i's$ used in $\alpha_2 \circ Y_{t-2}$ and so forth (Tjøstheim, 2012b).

Alternatively, there is another type of model to deal with count time series analysis. These models are analogues of GLMs. Unlike the thinning operator models, the analogues of GLMs can be constructed to hold nonlinear models, negative covariates, and negative correlations. Moreover, although these models are appropriate to use for the first-order Poisson, they can be generalized to order p and to the multivariate Poisson models. Several examples used this method, such as counting the total number of transactions for a stock every specific period, counting

the number of times that an asthma situation occurred daily, and counting the monthly number of special infections.

The idea of using the GLM is that in the AR models, the log of the conditional mean at time t depends on the previous mean value, the past observation, current or past covariate value, or all of them. The AR model can be written as

$$log(\lambda_t) = \alpha X_t + \gamma Y_{t-1},$$

where λ_t is the current expected value, X_t is the current covariate value, Y_{t-1} is the past observation, α and γ are the regression parameters. Specifically, this model means that the λ_t is regressed on the past covariate value and the past value of the observed process. The AR model using the analogue of the GLMs with link functions is the approach considered in this study because regular counts, for example, are not modeled directly; rather, they are modeled indirectly using the appropriate link function such as log intensity for the Poisson case or the logit function for the binary case.

Count time series data, in the context of GLMs, have been discussed in previous studies. For example, Zeger and Qaqish (1988) reviewed observation-driven count AR models using a quasi-likelihood approach in which the conditional means and variances, given past values, are functions of the past response value. The mean at time t (i.e., current time) is written exponentially in terms of the current value of the covariate, previous value of the covariate, and previous value of the response as

follows,

$$\mu_t(\boldsymbol{\beta}) = exp[\boldsymbol{X}_t'\gamma + \sum_{i=1}^{p} \theta_i(log\widetilde{Y}_{t-i} - \boldsymbol{X}_{t-i}'\gamma)],$$

where $t = 1, 2, ..., N$, $\boldsymbol{\beta} = (\gamma', \theta_1, ..., \theta_p)'$ is a vector that contains the covariates, $\boldsymbol{X}_t'$ contains the covariate values at time t, $\boldsymbol{X}_{t-i}'$ contains covariate values at time $(t-i)$, and $\widetilde{Y}_{t-i} = Y_{t-i} + c, c > 0$ so that $Y_{t-i} = 0$ is not an absorbing state. Different studies of the Poisson AR were done including observation-driven models and parameter-driven models with linear and nonlinear models. Davis, Dunsmuir, and Streett (2003) considered an important situation of log-linear model where they introduced the previous observations of the time series into the mean of the current observation using a linear filter that has been applied to martingale differences. The model is given by,

$$log(\mu_t) = \boldsymbol{X}_t'\boldsymbol{\beta} + \sum_{i=1}^{\infty} \gamma_i e_{t-i},$$

where

$$e_t = (Y_t - e^{log(\mu_t)})e^{-\alpha log(\mu_t)}, \alpha \in (0, 1].$$

Parameter Estimation of Time Series with Count Regression Models

As the data for the count regression models are positively skewed, the appropriate method of estimation of the parameters is to use MLE. As previously discussed, this method is based on taking the log of the likelihood function and maximizing it. To do that, we need to solve what is called the score function to estimate each parameter. Score functions are actually partial derivatives with respect to each parameter in the model. Solving the score functions gives the

parameter estimations as follows

$$S_N(\boldsymbol{\Theta}) = \frac{\partial \ell(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} = \mathbf{0}, \tag{2}$$

where $\boldsymbol{\Theta}$ is a vector of all the parameters that exist in the model and ℓ is the log likelihood function.

Hypothesis Testing of Time Series with Count Regression Models

One of the methods for testing the significance of the parameters is to use the Wald test. For instance, to test the hypothesis $H_0 : \mathbf{C}\boldsymbol{\Theta} = \boldsymbol{A}$ versus $H_1 : \mathbf{C}\boldsymbol{\Theta} \neq \boldsymbol{A}$, where $\mathbf{C}$ is an appropriate $r \times n$ matrix, $\boldsymbol{\Theta}$ is $n \times 1$ parameter vector, and $\boldsymbol{A}$ is $r \times 1$ vector of the outcomes for the hypotheses. The Wald test is given by

$$W_N = \{\mathbf{C}\hat{\boldsymbol{\Theta}} - \boldsymbol{A}\}'\{\mathbf{C}\boldsymbol{G}^{-1}(\hat{\boldsymbol{\Theta}})\mathbf{C}'\}^{-1}\{\mathbf{C}\hat{\boldsymbol{\Theta}} - \boldsymbol{A}\}, \tag{3}$$

where $\boldsymbol{G}$ is the information matrix. The observed information matrix which is called Hessian matrix can be written as

$$H_N(\boldsymbol{\Theta}) = -\frac{\partial^2 \ell(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta} \partial \boldsymbol{\Theta}'}. \tag{4}$$

The elements of the matrix are the second partial derivatives for each parameter with respect to all parameters in the model and the diagonal elements of the inverse of the Hessian matrix are the estimated squared standard errors.

AR Excess-Zero Models

As the time series of counts has not studied as diligently as continuous observed processes, time series of excess-zero models have received less attention than the general counts. Several authors have analyzed the time series analysis of the ZIP model, though there is a lack of studies that consider the hurdle model. Despite the lack of studies on time series analysis with zero-inflated models in general, the notable frequency of the type of data that need to be analyzed as time series with zero-inflation cannot be ignored.

There are many examples of excess-zero data in life that merit analysis using a time series model. One example is the prevalence of a very rare infection over time. In this example, given the disease's rarity, there is a high frequency of zeros in the data. Moreover, the data are collected over time, so the correlation between successive observations must be considered. Ignoring the inflation of zeros in the data as well as the failing to account for autocorrelation in the data will have undesirable consequences that may affect the results of the analysis by giving inconsistent parameter estimates, biased parameter estimates, and underestimated standard errors.

Time series models with excess-zeros were introduced using different methods, such as the thinning operator method, the analogue of the GLM, and a Bayesian approach with posterior mean. In this study, the AR model written in the context of the GLM is used. A ZIP AR model was first proposed by P. Wang (2001). P. Wang (2001) introduced a Markov ZIP model in which a two-state

discrete time Markov chain with transition probabilities associated with the covariates is employed to account for the serial correlation between the observations. The ZIP model is a two-component model, so P. Wang (2001) modeled the correlation between the observations such that the two components in ZIP model were associated with the two-state discrete time Markov chain. The two components of the ZIP model are the binary state, which only gives zero outcomes, and the Poisson distributed state, which results in zero and positive counts.

The ZIP mixed AR model was first introduced by Yau et al. (2004) to evaluate participatory ergonomics intervention in occupational health. In their study, the dependence between successive observations is characterized via an unobservable process. They considered parameter-driven models where the estimation of parameters was computationally burdensome. Moreover, the structure of the AR(1) they used was too restrictive to approximate the correlation in many time series data. They introduced random effects into the linear predictor to account for the correlation between the observations. The model is given by

$$log(\lambda_t) = logW_t + \boldsymbol{X}_t'\boldsymbol{\beta} + u_t,$$

where λ_t is the mean of Poisson model at time t, W_t is exposure at time t that is associated with the response, $\boldsymbol{X}_t$ is a vector of covariates, and u_t is a latent component and may be assumed to follow a first-order AR process. Zhu (2012) extended the integer-valued generalized AR conditional heteroscedasticity (GARCH) model to handle excess-zeros and overdispersion in the data. The model

was constructed as follows

$$\begin{aligned} Y_t|\mathscr{F}_{t-1} &: ZIP(\lambda_t, \pi), \\ \lambda_t &= \delta + \theta\lambda_{t-j} + \psi Y_{t-i}. \end{aligned} \tag{5}$$

where λ_t is the parameter corresponds to the Poisson part of the model, $\delta > 0, \theta \geq 0, \psi \geq 0, i = 1, ..., p, j = 1, ..., q, \mathscr{F}_{t-1}$ is the σ field generated by $Y_{t-1}, Y_{t-2}, ...$ which represents all information known about the outcome for times up to and including $t-1$. In this model, λ_t is written in terms of the past value λ_{t-1} as well as the past observation Y_{t-1}. This model, however, is not written in the context of the GLM, since it has linear structure rather than an exponential function which made some restriction to to parameters to be positive. Moreover, the parameter associated with the binary part π_t is not included in the AR model.

The idea of separating the ZIP AR model into two parts corresponds to the different components in the ZIP model, such that each parameter is modeled separately (i.e., the binary and the Poisson parts), was first introduced by Yang (2012) and Yang, Zamba, and Cavanaugh (2013). These authors proposed a class of ZIP AR models using the method of maximum partial likelihood estimator (MPLE). In their study, they had, $logit(\pi_t) = Z_{t-1}\gamma$ and $log(\lambda_t) = X_{t-1}\beta$, where λ_t is the intensity parameter of the Poisson part of the model, π_t is the zero inflation parameter, X_{t-1} and Z_{t-1} are the covariates at the time $t-1$ associated with the baseline Poisson distribution and binomial part of the model, respectively, and β and γ are the regression parameters associated with the Poisson and binomial parts

of the model, respectively. Then the ZIP AR model is given by

$$\begin{aligned} logit(\pi_t) &= \gamma_0 + \gamma_1 \mathcal{I}_{(Y_{t-1}>0)}, \\ log(\lambda_t) &= \beta_0 + \beta_1 \mathcal{I}_{(Y_{t-1}>0)}. \end{aligned} \tag{6}$$

As can be seen, the current value of λ depends on the indicator variable of Y at time $t-1$ being greater than 0 and some dispersion parameter, and the current value of π depends on the indicator at time $t-1$ being greater than 0 as well. Even though the ZIP model is usually preferred when there is an assumption of the two type of zeros (i.e., structural or true zeros and sampling zeros), in practice, there is often a lack of evidence regarding the source of the zeros in the data. Yet, the hurdle model is more general in the sense of handling both the situation of zero-inflation and zero-deflation in the data, and the hurdle does not have to be set at 0. Moreover, the estimation of the parameters in the ZIP model is done simultaneously for the zero and the nonzero parts of the model, which makes the interpretations more difficult than the hurdle model that separates these two parts. Furthermore, most previous studies of count time series were based on the partial likelihood approach. The reason for using partial maximum likelihood instead of the full maximum likelihood is to simplify the computations of the full likelihood by dropping some components that are regarded as unimportant or have minimal effect on the estimated parameters when the sample is large enough. In this study, though, the full maximum likelihood is applied.

Despite a number of papers that have studied time series with ZIP model, there is a dearth of studies on time series with the hurdle model and AR process in particular. Ver Hoef and Jansen (2007) elaborated a Bayesian approach. In their study, they developed Bayesian hierarchical models of ZIP and hurdle models with space-time errors to examine haul-out patterns of harbor seals on glacial ice. The ZIP and the hurdle models were constructed by using first AR models as a random effect in both ZIP and hurdle models. To illustrate, they used a modified hurdle model as follows,

$$Y_{ij}|Z_{ij} = \begin{cases} 0, & Z_{ij} = 0 \\ Poi(\lambda_{ij}) + 1, & Z_{ij} = 1, \end{cases}$$

where $Poi(\lambda_{ij})$ is Poisson random variable with mean λ_{ij} and Z_{ij} is a Bernoulli random variable with mean π_{ij}, assuming

$$log(\lambda_{ij}) = \upsilon_i + x_{ij}\beta + \epsilon_{ij}$$

$$logit(\pi_{ij}) = \mu_i + x_{ij}\alpha + \delta_{ij}.$$

where υ_i and μ_i are the means for each part, $x'_{ij}s$ are the spatio-temporal covariates, ϵ_{ij} and δ_{ij} are random errors and assumed to be spatially autocorrelated, and α and β are the regression parameters. They treated υ and μ as linear models for temporal covariates,

$$\upsilon_i = \upsilon_0 + \boldsymbol{t}'_i\boldsymbol{\eta} + \theta_i,$$

$$\mu_i = \mu_0 + \boldsymbol{t}'_i\boldsymbol{\gamma} + \psi_i,$$

where $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ are regression parameters and $\boldsymbol{t}_i$ are the temporal covariates. θ_i and ψ_i are temporally autocorrelated errors defined as,

$$\theta_i = \phi_\theta \theta_{i-1} + \sigma_\theta W_{\theta i}, \quad i > 1,$$

$$\psi_i = \phi_\psi \psi_{i-1} + \sigma_\psi W_{\psi i}, \quad i > 1,$$

where $W_{\theta i}$ and $W_{\psi i}$ are independent random errors. The assumption of random errors led to biased inferences. That bias was solved from a very similar approach that allowed correlation in the hurdle AR model (Neelon et al., 2013). The problem with using a Bayesian approach is that the requirement for choosing the prior distributions for all parameters can be time-consuming. Morover, the choice of prior distributions needs to be justified. Finally, this approach is computationally difficult, especially for complex models such as a hurdle model.

As can be seen, past studies do not offer much information about time series analysis with the hurdle model. Specifically, to date, no studies have introduced the AR(1) process for such a model using the analogue of GLM, in which the present mean is a function of the past value of itself as well as the past observation value. In other words, in Poisson or binomial count regression models, the counts are modeled using link functions to characterize the counts. These link functions are different, depending on different distributions. Therefore, AR models for count data are defined by canonical parameters that characterize count distribution. In this study, the ideas of the ZIP model constructed by Zhu (2012) and the ZIP model

constructed by Yang (2012) and Yang et al. (2013) are combined to construct the first-order AR hurdle model.

Study Rationale

Excess-zero data occur frequently in practice. Ignoring the inflation of zeros in the data and using the Poisson distribution does not give precise results, for the assumption of equality between the mean and the variance for the Poisson distribution is violated. Some negative consequences of ignoring zero-inflation include inconsistent parameter estimates, underestimated standard errors, and invalid inferences (Miller, 2007; Mullahy, 1986). In this study, the focus is on examining the hurdle model for several reasons. First, while the ZIP model is appropriate for modeling zero-inflation only, the hurdle model is also appropriate for modeling zero-deflation in the data (Min and Agresti, 2005). Min and Agresti (2005) found that if there is zero-deflation in the data, the estimate of the parameters in the first components of ZIP model is ∞, meaning that the fit has no zero-inflation at that level. Second, estimates are sometimes unstable when using the ZIP even if there is significant zero-inflation in the data. Min and Agresti (2005) ran a simulation to study this problem; they assumed the hurdle model with zero-deflation at one setting of the predictor, but the data were zero inflated. They found that the estimates of the predictors under the ZIP model were unstable. They used another simulation assuming the ZIP model, and they found that the hurdle model gave parameter estimates in the second part of the model similar to those given by the ZIP. Finally, the ZIP model is more complex to fit than the hurdle model because the model's components must be simultaneously fitted on account of

the fact that zeros can appear in both components, which makes them dependent (Min and Agresti, 2005).

Although simple, the AR process is very important in time series analysis because of its wide application in practice. When there are excess zeros in data, coupled with measurements taken for successive time periods, it is important to use time series analysis to obtain accurate results. Most of the previous studies mentioned dealt with the ZIP model, which is not suited for some cases. Moreover, the few studies that looked at the hurdle model used Bayesian approaches which can be challenging to apply. There are no studies that examined a first-order AR hurdle model using the analogue of GLMs. In ordinary linear regression where data are often normal and independent, the mean response of a variable of interest is related to the predictors and the parameters through a linear equation. In non-Gaussian data, however, such as count and binary data, using an ordinary linear regression leads to inconsistent parameter estimates. This can be attributed to count data's distribution, which is frequently skewed, and the variance's increase as the counts increase, which is not the case for normally distributed data, as in regular cross-sectional data that can be resolved successfully by GLMs (Kedem and Fokianos, 2002). A first-order AR model, in the context of GLMs, is used in this study because of the ease of application, interpretability, and the possibility for predicting observations (Tjøstheim, 2012b).

CHAPTER III

METHODS

The hurdle model is an approach that effectively models count data with more zeros than expected under the Poisson distribution. This is explained by the hurdle model's structure of separating zeros and positive observations, in contrast to the ZIP model; the hurdle model makes interpretations easier than the ZIP, can handle zero-deflation in the data, and often results in consistent parameter estimates, unlike the ZIP model.

AR models in time series are frequently used because of their simplicity. In previous studies, AR processes have been used widely for a Gaussian distribution. Recently, the use of AR processes with count regression models has gained traction because of their practical applications. In terms of count data that have excess-zeros, ZIP AR models have been examined using different methods. One method used is the analogue of GLMs, which have been shown to be generalized to non-Poisson models and multivariate models (Tjøstheim, 2012b). In this study, GLMs are imported to time series. Even though the hurdle model does not belong to the exponential family, each model component does belong to a distribution from the exponential family (i.e., Poisson and binomial distribution) that has random components, systematic components, and link functions. Moreover, time series that follow GLMs has been studied for another excess-zero model, namely the ZIP

model, by Yau et al. (2004),Yang (2012), and Yang et al. (2013). In this study, the hurdle autoregressive (hurdle AR) in the context of GLMs is constructed.

Research Questions

Q1 How can we write a hurdle model for excess-zero time series data?

Q2 How are the parameters estimated for the hurdle autoregressive model?

Q3 How can the proposed model be implemented using R?

Q4 How does the proposed model perform specifically using standard errors, mean absolute deviation error, and relative bias?

Q5 How can the prediction of future observations be obtained and evaluated for the proposed model compared to existing times series models of counts?

This chapter is organized as follows. First, construction of the hurdle AR model is explained and model notations are clarified. Second, the likelihood function is constructed to estimate model parameters. Third, score functions and the diagonal Hessian matrix are derived to obtain the estimation of the parameters and the standard errors of the parameters. Finally, an evaluation of the proposed model is described.

Hurdle Autoregressive Model

A first-order AR hurdle model is appropriate for time series involving counts that have a high frequency of zeros. Examples of such data are counts of rare disease over a given period, counts of rainy days in a city for a year, or counts of daily sex offense crimes. Previous studies have shown that these data can be

modeled using the ZIP AR models in the context of the GLMs. This study is an extension of previous work on ZIP models and includes use of the hurdle model.

The first-order AR hurdle model is a two-part model, with each part corresponding to one of two different distributions (Yang, 2012). The first part is associated with the binomial distribution, where the mean response is linked to the parameters using the logit function. In this part, the value of the current success probability, π_t, is written with respect to the past probability, π_{t-1} , as well as the past observation, Y_{t-1} . In other words, the current probability depends on the preceding probability as well as the value of the past observation. The second part of the model is associated with the truncated Poisson distribution. In this part, the value of the current expectation, λ_t, is written with respect to the past expectation, λ_{t-1}, as well as the past observation, Y_{t-1}. That is, the current expectation depends on the preceding expectation and the past observation. Let Y_t be the observation at time t, where $t = 1, ..., N$. π_t is the probability of getting zero at time t, and λ_t is the intensity parameter of the Poisson baseline at time t, which means $Y_t|Y_{t-1} \sim hurdle(\pi_t, \lambda_t)$. The first research question is answered by writing the hurdle AR model components as,

$$\begin{aligned} \pi_t &= \frac{exp(\alpha + \beta\ logit(\pi_{t-1}) + \gamma Y_{t-1})}{1 + exp(\alpha + \beta\ logit(\pi_{t-1}) + \gamma Y_{t-1})} \\ (\lambda_t|Y_t > 0) &= exp(\delta + \theta\ ln(\lambda_{t-1}) + \psi Y_{t-1}) \end{aligned} \tag{7}$$

where

α is the intercept of the linear systematic component of the binary part of the model.

β represents the effect of the prior probability.

γ represents the effect of the prior outcome.

δ is the intercept of the linear systematic component of the truncated Poisson part.

θ represents the effect of the prior expectation.

ψ represents the effect of the prior outcome in the truncated Poisson part.

Likelihood Based Inference

The likelihood is a function of the model's parameters in light of the data. Constructing the likelihood function is the first step to estimating all parameters. In previous studies of time series analysis for count data, partial likelihood was used instead of the full likelihood (Kedem and Fokianos, 2002)and (Yau et al., 2004; Yang, 2012; Yang et al., 2013). The reason for using partial likelihood is to simplify the complicated likelihood function by omitting some unimportant parts, such as auxiliary information, which minimally affect large sample properties of the estimated parameters (Yang, 2012). Nevertheless, in the current study, full likelihood is used. Even though the data are not independent in the AR processes, full likelihood can be used because the data's dependence is limited (Kedem and Fokianos, 2002). In the first-order AR model, each observation is only dependent on the past observation. The general form of the likelihood function for the hurdle

AR(1) is given by

$$\begin{aligned} L(\boldsymbol{\Theta}|y_1,...,y_N) &= f(Y_1)\prod_{t=2}^{N} f(y_t|y_{t-1},\boldsymbol{\Theta}) \\ &= f(Y_1)\prod_{t=2}^{N}(\pi_t)^{I(Y_t=0)}[(1-\pi_t)\frac{\lambda_t^{Y_t}}{Y_t!(e^{\lambda_t}-1)}]^{(1-I(Y_t=0))}, \end{aligned} \tag{8}$$

where $f(Y_1)$ is the first density function assumed to be independent of any parameters; therefore, it can be dropped for simpler calculations.

Parameter Estimation

in order to estimate the parameters, the likelihood function must be maximized by solving the score functions $S_N(\boldsymbol{\Theta}) = \mathbf{0}$. Score equations are partial derivatives of the log-likelihood function that are defined as in Equation (2). To obtain the score functions, the log of the likelihood is first obtained as follows,

$$\begin{aligned} \ell(\boldsymbol{\Theta}|y_1,...,y_N) &= \sum_{t=2}^{N} log f(y_t|y_{t-1},\boldsymbol{\Theta}) \\ &= \sum_{t=2}^{N} I(Y_t=0)log(\pi_t) + \sum_{t=2}^{N}(1-I(Y_t=0))log(1-\pi_t) \\ &+ \sum_{t=2}^{N}(1-I(Y_t=0))[Y_t log(\lambda_t) - logY_t! - log(e^{\lambda_t}-1)]. \end{aligned} \tag{9}$$

Then, the score functions for all the parameters are given by

$$\begin{aligned}\frac{\partial \ell}{\partial \alpha} &= \frac{\partial \ell}{\partial \pi_t}\frac{\partial \pi_t}{\partial \alpha} \\ &= \sum_{t=2}^{N}[I(Y_t=0)\frac{1}{\pi_t} + (1-I(Y_t=0))\frac{-1}{1-\pi_t}]\times \\ &\quad \pi_t(1-\pi_t)[1+\frac{\beta \partial logit(\pi_{t-1})}{\partial \alpha}],\end{aligned} \tag{10}$$

$$\text{where } \frac{\beta \partial logit(\pi_{t-1})}{\partial \alpha} = \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})}\frac{\partial \pi_{t-1}}{\partial \alpha}.$$

$$\begin{aligned}\frac{\partial \ell}{\partial \beta} &= \frac{\partial \ell}{\partial \pi_t}\frac{\partial \pi_t}{\partial \beta} \\ &= \sum_{t=2}^{N}[I(Y_t=0)\frac{1}{\pi_t} + (1-I(Y_t=0))\frac{-1}{1-\pi_t}]\times \\ &\quad \pi_t(1-\pi_t)[logit(\pi_{t-1})+\frac{\beta \partial logit(\pi_{t-1})}{\partial \beta}],\end{aligned} \tag{11}$$

$$\text{where } \frac{\beta \partial logit(\pi_{t-1})}{\partial \beta} = \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})}\frac{\partial \pi_{t-1}}{\partial \beta}.$$

$$\begin{aligned}\frac{\partial \ell}{\partial \gamma} &= \frac{\partial \ell}{\partial \pi_t}\frac{\partial \pi_t}{\partial \gamma} \\ &= \sum_{t=2}^{N}[I(Y_t=0)\frac{1}{\pi_t} + (1-I(Y_t=0))\frac{-1}{1-\pi_t}]\times \\ &\quad \pi_t(1-\pi_t)[Y_{t-1}+\frac{\beta \partial logit(\pi_{t-1})}{\partial \gamma}],\end{aligned} \tag{12}$$

$$\text{where } \frac{\beta \partial logit(\pi_{t-1})}{\partial \gamma} = \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})}\frac{\partial \pi_{t-1}}{\partial \gamma}.$$

$$\begin{aligned}\frac{\partial \ell}{\partial \delta} &= \frac{\partial \ell}{\partial \lambda_t}\frac{\partial \lambda_t}{\partial \delta} \\ &= \sum_{t=2}^{N}[(1 - I(Y_t = 0))(\frac{Y_t}{\lambda_t} - \frac{e^{\lambda_t}}{e^{\lambda_t} - 1})]\lambda_t[1 + \frac{\theta \partial ln(\lambda_{t-1})}{\partial \delta}],\end{aligned} \tag{13}$$

$$\text{where } \frac{\theta \partial ln(\lambda_{t-1})}{\partial \delta} = \frac{1}{\lambda_{t-1}}\frac{\theta \partial \lambda_{t-1}}{\partial \delta}.$$

$$\begin{aligned}\frac{\partial \ell}{\partial \theta} &= \frac{\partial \ell}{\partial \lambda_t}\frac{\partial \lambda_t}{\partial \theta} \\ &= \sum_{t=2}^{N}[(1 - I(Y_t = 0))(\frac{Y_t}{\lambda_t} - \frac{e^{\lambda_t}}{e^{\lambda_t} - 1})]\lambda_t[ln(\lambda_{t-1}) + \frac{\theta \partial ln(\lambda_{t-1})}{\partial \theta}],\end{aligned} \tag{14}$$

$$\text{where } \frac{\theta \partial ln(\lambda_{t-1})}{\partial \theta} = \frac{1}{\lambda_{t-1}}\frac{\theta \partial \lambda_{t-1}}{\partial \theta}.$$

$$\begin{aligned}\frac{\partial \ell}{\partial \psi} &= \frac{\partial \ell}{\partial \lambda_t}\frac{\partial \lambda_t}{\partial \psi} \\ &= \sum_{t=2}^{N}[(1 - I(Y_t = 0))(\frac{Y_t}{\lambda_t} - \frac{e^{\lambda_t}}{e^{\lambda_t} - 1})]\lambda_t[Y_{t-1} + \frac{\theta \partial ln(\lambda_{t-1})}{\partial \psi}],\end{aligned} \tag{15}$$

$$\text{where } \frac{\theta \partial ln(\lambda_{t-1})}{\partial \psi} = \frac{1}{\lambda_{t-1}}\frac{\theta \partial \lambda_{t-1}}{\partial \psi}.$$

The parameter estimates can be obtained by simultaneously setting the Equations (10) to (15) to zero then solving them.

Parameter Hypothesis Testing

The observed information matrix, alternatively referred to as the negative Hessian matrix, can be used to calculate the standard errors of the maximum

likelihood. The negative Hessians are the second partial derivatives defined by Equation (4). Then, the MLE with the standard errors obtained from the negative Hessians can be employed to construct a Wald test for each parameter in the likelihood setting. The diagonal elements of the observed information matrix are given by

$$\frac{\partial^2 \ell}{\partial \alpha \partial \alpha'} = \sum_{t=2}^{N} \{ [I(Y_t = 0) \frac{-1}{\pi_t^2} + (1 - I(Y_t = 0)) \frac{-1}{(1-\pi_t)^2}]$$
$$\pi_t(1-\pi_t)[1 + \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial \pi_{t-1}}{\partial \alpha}]$$
$$+ [I(Y_t = 0) \frac{1}{\pi_t} + (1 - I(Y_t = 0)) \frac{-1}{1-\pi_t}] + (1 - 2\pi_t) + (1 - 2\pi_t)$$
$$\frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial \pi_{t-1}}{\partial \alpha} + \pi_t(1-\pi_t) \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial^2 \pi_{t-1}}{\partial \alpha \partial \alpha'} \}$$

$$\frac{\partial^2 \ell}{\partial \beta \partial \beta'} = \sum_{t=2}^{N} \{ [I(Y_t = 0) \frac{-1}{\pi_t^2} + 1 - I(Y_t = 0) \frac{-1}{(1-\pi_t)^2}]$$
$$\pi_t(1-\pi_t)[logit(\pi_{t-1}) + \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial \pi_{t-1}}{\partial \beta}]$$
$$+ [I(Y_t = 0) \frac{1}{\pi_t} + (1 - I(Y_t = 0)) \frac{-1}{1-\pi_t}] + (1 - 2\pi_t) logit(\pi_{t-1}) + \pi_t(1-\pi_t)$$
$$\frac{1}{\pi_{t-1}(1-\pi_{t-1})} + (1 - 2\pi_t) \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial \pi_{t-1}}{\partial \beta}$$
$$+ \pi_t(1-\pi_t)(\frac{1}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial \pi_{t-1}}{\partial \beta} + \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial^2 \pi_{t-1}}{\partial \beta \partial \beta'}) \}$$

$$\frac{\partial^2 \ell}{\partial \gamma \partial \gamma'} = \sum_{t=2}^{N} \{ [I(Y_t = 0) \frac{-1}{\pi_t^2} + (1 - I(Y_t = 0)) \frac{-1}{(1-\pi_t)^2}]$$
$$\pi_t(1-\pi_t)[Y_{t-1} + \frac{\beta}{\pi_{t-1}(1-\pi_{t-1})} \frac{\partial \pi_{t-1}}{\partial \gamma}] +$$

$$[I(Y_t=0)\frac{1}{\pi_t}+(1-I(Y_t=0))\frac{-1}{1-\pi_t}]+(1-2\pi_t)Y_{t-1}$$
$$+\pi_t(1-\pi_t)+(1-2\pi_t)\frac{\beta}{\pi_{t-1}(1-\pi_{t-1})}\frac{\partial\pi_{t-1}}{\partial\gamma}$$
$$+\pi_t(1-\pi_t)\frac{\beta}{\pi_{t-1}(1-\pi_{t-1})}\frac{\partial^2\pi_{t-1}}{\partial\gamma\partial\gamma'}\}$$

$$\frac{\partial^2\ell}{\partial\delta\partial\delta'}=\sum_{t=2}^{N}\{[(1-I(Y_t=0))\frac{-{Y_t}^2}{\lambda_t}+e^{\lambda_t}]\lambda_t[1+\frac{1}{\lambda_{t-1}}\frac{\theta\partial\lambda_{t-1}}{\partial\delta}]$$
$$+[(1-I(Y_t=0))\frac{Y_t}{\lambda_t}-\frac{e^{\lambda_t}}{e^{\lambda_t}-1}][1+\frac{\lambda_t}{\lambda_{t-1}}+\frac{\lambda_{t-1}}{\lambda_t\lambda_{t-1}^2}\frac{\theta\partial\lambda_{t-1}}{\partial\delta}+\frac{\lambda_t}{\lambda_{t-1}}\frac{\partial^2\lambda_{t-1}}{\partial\delta\partial\delta'}]\}$$

$$\frac{\partial^2\ell}{\partial\theta\partial\theta'}=\sum_{t=2}^{N}\{[(1-I(Y_t=0))\frac{-{Y_t}^2}{\lambda_t}+e^{\lambda_t}]\lambda_t[ln(\lambda_{t-1})+\frac{1}{\lambda_{t-1}}\frac{\theta\partial\lambda_{t-1}}{\partial\theta}]$$
$$+[(1-I(Y_t=0))\frac{Y_t}{\lambda_t}-\frac{e^{\lambda_t}}{e^{\lambda_t}-1}][ln(\lambda_t)+\frac{\lambda_t}{\lambda_{t-1}}+\frac{\lambda_{t-1}}{\lambda_t\lambda_{t-1}^2}\frac{\theta\partial\lambda_{t-1}}{\partial\theta}+\frac{\lambda_t}{\lambda_{t-1}}\frac{\partial^2\lambda_{t-1}}{\partial\theta\partial\theta'}]\}$$

$$\frac{\partial^2\ell}{\partial\psi\partial\psi'}=\sum_{t=2}^{N}\{[(1-I(Y_t=0))\frac{-{Y_t}^2}{\lambda_t}+e^{\lambda_t}]\lambda_t[Y_t+\frac{1}{\lambda_{t-1}}\frac{\theta\partial\lambda_{t-1}}{\partial\psi}]$$
$$+[(1-I(Y_t=0))\frac{Y_t}{\lambda_t}-\frac{e^{\lambda_t}}{e^{\lambda_t}-1}][Y_t+\frac{\lambda_t}{\lambda_{t-1}}+\frac{\lambda_{t-1}}{\lambda_t\lambda_{t-1}^2}\frac{\theta\partial\lambda_{t-1}}{\partial\psi}+\frac{\lambda_t}{\lambda_{t-1}}\frac{\partial^2\lambda_{t-1}}{\partial\psi\partial\psi'}]\}$$

For a sufficient sample size, maximum likelihood is usually followed by Wald tests for significance. Wald tests can be constructed to test the individual parameters. To test the hypotheses $H_0 : \mathbf{C\Theta} = \boldsymbol{A}$ versus $H_1 : \mathbf{C\Theta} \neq \boldsymbol{A}$, where $\mathbf{C}$ is an appropriate $r \times 6$ matrix, $\mathbf{\Theta}$ is 6×1 parameter vector, and $\boldsymbol{A}$ is $r \times 1$ vector of the outcomes for the hypotheses. The Wald test is given by Equation(3).

In the next chapter, a simulation study is conducted and an example of a time series (http://www.forecastingprinciples.com) is analyzed. The data are crime data from Pittsburgh, Pennsylvania. The data were collected each month for 12 years, covering the period of 1990-2001. The crime data, collectively titled *Car Beats Plus*, are the frequency of different crimes over 15 geographic areas in Pittsburgh. Crimes investigated in this data set include alcohol-related violations, drug offenses, robbery, rape, gambling, public drunkenness, fraud, burglaries, and arson. The variable chosen in this study is the crime of arson because arson has a very high frequency of zeros. Given that this study is not focused on spatial time series, only one of the areas is used in the analysis, specifically, the 13th precincts in Pittsburgh. Data were collected monthly from January 1990 to December 2001, totaling 144 observations for each police station. The proposed model's performance is compared to the previous ZIP AR model developed by Yang (2012) and the ZIP INGARCH model developed by Zhu (2012).

The first model to be compared is the ZIP AR model (Yang, 2012), which is given in Equation(6). In this model, the current value of λ depends on the indicator variable of Yat time $t-1$ being greater than 0 and some dispersion parameter, and the current value of π depends on the indicator at time $t-1$ being greater than 0 as well. Neither past outcomes nor past means are considered in this model. The second model is the ZIP INGARCH (Zhu, 2012), which is given in Equation(5). The model is similar to the proposed model in this study, except it is linear. For that, the parameters are restricted to be positive in order to obtain a positive λ.

Moreover, the parameter associated with the logistic part (π) is not considered to be dependent on time.

The standard errors of the MLE are calculated for the estimated parameters (Liu, 2012). Moreover, to answer the fifth research question, the prediction of future observations is obtained. The predictive probability is used in the proposed model for forecasting future observations. The h-step ahead prediction, where $h = (1, 2, 3, 4, 5)$, is performed such that the parameters of the model are updated sequentially when there is a new observation (Yang, 2012). The probability of the next count being zero is computed first. If it is greater than the predetermined cutoff value (the mean of the logistic parameter pi), then the predicted value, is assumed to be a zero (Yang, 2012). If the probability is less than the predetermined value then it is assumed to be positive value. The prediction is then compared for the three models. To measure the accuracy of the predictions, the predicted root mean squared error (PRMSE) is computed (Maiti, Biswas, Guha, and Ong, 2014). To calculate the PRMSE, the dataset is divided into two parts. The first part is used to fit the model and obtain parameter estimation, and the second part is used for prediction. The PRMSE for h-step ahead forecasting is given by

$$PRMSE1(h) = \sqrt{\frac{1}{M-h+1}\sum_{t=M}^{M+N-h}(Y_{t+h} - \hat{Y}_{t+h})}, \tag{16}$$

$$PRMSE2(h) = \sqrt{\frac{1}{M-h+1}\sum_{t=M}^{M+N-h}(Y_{t+h} - \hat{\mu}_{t+h})}, \tag{17}$$

where $\hat{Y}_{t+h}$ is the predicted observation at time t+h and $\hat{\mu}_{t+h}$ is the predicted mean of the h-step-ahead forecasting distribution of Y_{t+h} given the past observations $(Y_1, .., Y_t)$.

A simulation study is conducted to evaluate the finite sample performance of the estimators. The sample sizes (i.e., time series length) used throughout the simulation are N=20, N=50, N = 100, N = 200, and N = 500 (Yang, 2012). All simulation procedures are performed using R version 3.3.1.. The ConstrOptim function is used to solve the likelihood function based on the Nelder-Mead method. The model used to generate the data is given by,

$$\pi_t = \frac{exp(\widetilde{\alpha} + \widetilde{\boldsymbol{\beta}}\ logit(\pi_{t-1}) + \widetilde{\boldsymbol{\gamma}} Y_{t-1})}{1 + exp(\widetilde{\alpha} + \widetilde{\boldsymbol{\beta}}\ logit(\pi_{t-1}) + \widetilde{\boldsymbol{\gamma}} Y_{t-1})}$$

$$(\lambda_t | Y_t > 0) = exp(\widetilde{\delta} + \widetilde{\boldsymbol{\theta}}\ ln(\lambda_{t-1}) + \widetilde{\boldsymbol{\psi}} Y_{t-1}),$$

where $\widetilde{\alpha}$, $\widetilde{\boldsymbol{\beta}}, \widetilde{\boldsymbol{\gamma}}, \widetilde{\delta}, \widetilde{\boldsymbol{\theta}}$, and $\widetilde{\boldsymbol{\psi}}$ are the parameter values that approximate the values obtained by the analyzed real data example. Each vector is 3×1. The first value in each vector is the value of the parameter obtained from fitting the data. The second value is to decrease the magnitude of that estimate obtained from fitting the data by 0.05. The third value is to increase the magnitude of that estimate obtained from fitting the data by 0.05. The reason of having small range of true values is because three of the six parameters need to have restrictions on them in order to obtain reasonable outcome values. The parameter associated with the previous success probability in the logistic part ,β, is constrained between -1 and 1 because larger values of β can easily inflate the observations when generating the data. In the log

linear model part, constraints are set on the parameter associated with the prior mean ,θ, and the parameter associated with the prior observation ,ψ. The constraints are set so that $|\theta| < 1$, $|\psi| < 1$, and $|\theta + \psi| < 1$ (Liboschik, Fokianos, and Fried, 2015).

Table 1

Parameter Values Used for Simulations

α	β	γ	δ	θ	ψ
0.47	-0.80	-0.08	-0.20	0.06	0.13
0.42	-0.85	-0.13	-0.25	0.01	0.08
0.52	-0.75	-0.03	-0.15	0.11	0.18

Table (1) shows the true parameter values that are used in the simulation. π_1 and λ_1 are given randomly chosen initial values. After that, the first outcome value is generated from the hurdle distribution using the initial values of π_1 and λ_1. The data are then generated from the hurdle distribution (i.e., the binomial distribution for the zeros and the truncated Poisson distribution for the positive counts), where each outcome value is calculated based on the previous outcome value. Finally, the model is fitted and the parameters are estimated using MLE. The previous steps are then repeated independently from the true model K = 1000 times; then the average standard errors, average relative bias, and the average mean absolute deviation error (MADE) for the proposed model are computed and average. The MADE is given by The mean absolute deviation error (MADE) is the evaluation criterion (Zhu, 2012). MADE is a method for measuring forecast error. However, in the study by Zhu

(2012), it was used for error estimation, so it is used to estimate the errors here as well. It is the averaged absolute value of the difference between actual parameter values and their estimated values. In MADE, the absolute value of the error is obtained. It can be calculated as follows,

$$MADE = \frac{1}{K}\sum_{i=1}^{K}|\hat{\boldsymbol{\Theta}}_{\boldsymbol{i}} - \boldsymbol{\Theta}_{\boldsymbol{i}}|, \tag{18}$$

where $\boldsymbol{\Theta} = (\alpha, \beta, \gamma, \delta, \theta, \psi)$ is a vector of all the parameters in the model. PRMSE are calculated and compared for the three models(Christou, 2013). In order to calculate the PRMSE, extra observations are generated for the prediction, so the actual generated sample sizes are N+M=(25, 70, 125, 250, 600). The samples N=(20, 50, 100, 200, 500) are used for estimation and the rest M=(5,20,25,50,100) are used for prediction (Maiti et al., 2014). The models by Yang (2012) and Zhu (2012) are also fitted to the generated hurdle data and the results based on the three models are compared.

CHAPTER IV

RESULTS

This chapter presents the simulation study to evaluate the model that is presented in chapter III. The performance of the hurdle AR model with different sample sizes was examined. The performance of the model was examined based on its standard errors, relative bias, and mean absolute deviation error (MADE). The main purpose of the simulation was to examine the predictive capabilities of the hurdle AR model compared to the ZIP AR model and the ZIP INGARCH model. Finally, a numerical example was presented to demonstrate the model.

The chapter is organized as follows: In the first section, general steps of how the simulation is implemented are explained. In the second section, simulation results for the hurdle first-order autoregressive hurdle AR model in terms of the standard errors, relative bias, and mean absolute deviation error is provided for the three models are presented. A comparison between the three models to evaluate the prediction is presented in the third section. Finally, the fourth section presents the performance of the models on a real data set.

The first three research questions are answered as follows:

Q1 How can we write a hurdle model for excess-zero time series data?

This research question is answered in chapter III by writing Equation (7).

Q2 How are the parameters estimated for the hurdle autoregressive model?

This research question is answered in chapter III as well using maximum likelihood estimation.

Q3 How can the proposed model be implemented using R?

This question is answered in appendix A by writing the R codes.

Simulation Steps

Consider the model

$$\pi_t = \frac{exp(\alpha + \beta\ logit(\pi_{t-1}) + \gamma Y_{t-1})}{1 + exp(\alpha + \beta\ logit(\pi_{t-1}) + \gamma Y_{t-1})}$$

$$(\lambda_t | Y_t > 0) = exp(\delta + \theta\ ln(\lambda_{t-1}) + \psi Y_{t-1})$$

A simulation study is conducted using the following steps:

Step 1: The three sets of the true regression parameters are chosen so that the first one is based on the estimates obtained from running the real data set. The second set is chosen so that the values are .05 less than the values obtained by the data. Finally, the third set of parameters is chosen to be .05 more than the values obtained by the data, as shown in chapter III. It turns out that some of the parameters have to be constrained. Looking at the hurdle AR model, the parameter associated with the previous success probability in the logistic part ,β, is set to be between -1 and 1 because larger values of β can easily inflate the observations when generating the data. In the log linear model part, constraints are set on the parameter associated with the prior mean ,θ, and the parameter associated with the prior observation ,ψ. The constraints are set so that $|\theta| < 1$, $|\psi| < 1$, and

$|\theta + \psi| < 1$ (Liboschik et al., 2015). The reason for having the constraints is that, as can be seen, the current rate is written exponentially in terms of the other parameters. Therefore, the value of the current rate gets larger as the values of the parameters get larger.

Step 2: Choose initial values for λ and π.

Step 3: Generate the first response value using the hurdle Poisson distribution.

Step 4: Define $(\lambda_t|Y_t > 0) = exp(\delta + \theta\ ln(\lambda_{t-1}) + \psi Y_{t-1})$ and $\pi_t = \frac{exp(\alpha+\beta\ \ logit(\pi_{t-1})+\gamma Y_{t-1})}{1+exp(\alpha+\beta\ \ logit(\pi_{t-1})+\gamma Y_{t-1})}$ and then generate the response variable Y_t from the hurdle Poisson distribution using the functions *rbinom* and *trpois* in R. Here, the current expectation λ_t depends on the past observation value Y_{t-1} as well as the past expectation λ_{t-1} , and the current probability π_t depends on the past observation value Y_{t-1} as well as the past probability π_{t-1}.

Step 5: Using the maximum likelihood estimation method, fit the first-order hurdle autoregressive, hurdle AR, model.

Step 6: Get the parameter estimates of the model, and then get the values of the two series of $\hat{\lambda}$ and $\hat{\pi}$ based on these estimates and the initial values of λ_1 and π_1.

Step 7: Repeat the previous steps 1,000 times, then calculate the mean of each parameter estimate and the average of the standard errors as well as the relative bias for the regression coefficients. The average of the predicted root mean squared error, (PRMSE1), and (PRMSE2), to evaluate the h-steps ahead prediction and the mean absolute deviation error (MADE) are calculated using Equations (16), (17), and (18), respectively, where smaller values of MADE, PRMSE1, and

PRMSE2 are better. The same generated data are fitted using the ZIP AR model developed by Yang (2012) given in Equation (6). The developed model is also compared to the ZIP INGARCH model developed by Zhu (2012) given in Equation (5). In these two models, the excess-zero data are generated and the ZIP AR and ZIP INGARCH are used instead of the hurdle AR. The purpose of the comparison to the two existing models with the ZIP AR is to examine how adding the past means can impact the prediction ability of the model. The other comparison of the ZIP INGARCH model is to study how the predictions is affected when changing versus fixing the probability parameter over time. Both of these models are ZIP models, not hurdle models, but these are the only available existing models for excess-zero data and thus are used for comparison to the developed hurdle model.

One of the issues encountered when generating the data is that the true parameters initially were chosen without any constraints which caused some of the counts to have very large values. Controlling the parameters by setting constraints on some of them, as mentioned before, helped keep the values of the data and the values of the rate λ from getting inflated. One of the simulation conditions that was planned but not applied was to control the proportion of zeros in the generated data as 20% and 50%. It is found to be challenging to control the proportion of zeros in the generated data. Moreover, it is not the focus of the current study to see how the models perform with different proportions of zeros in the data. Even though this condition was not considered when generating the hurdle data, the generated data have approximately 50% zeros. Figure (1) shows the actual generated hurdle time

series data. As can be seen from the plot, the values are mostly zeros, with some small numbers the way excess-zero data usually look.

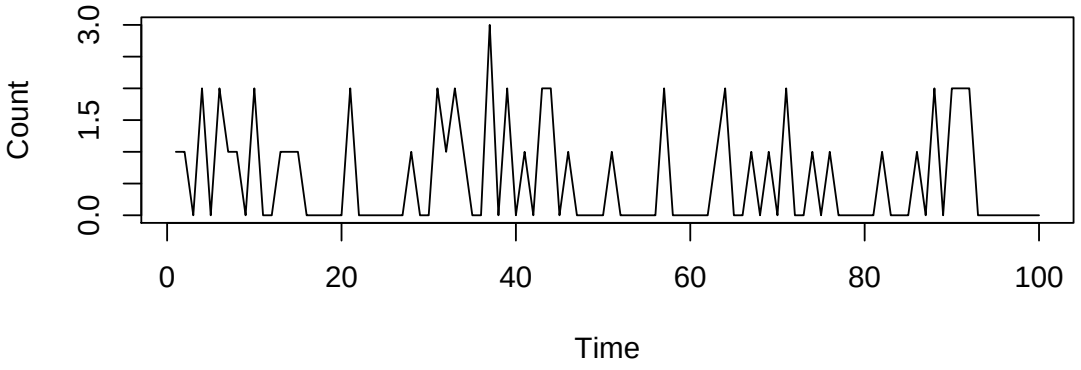

Figure 1. Hurdle Time Series Data of size 100

Simulation Results for the Hurdle Autoregressive Model, the Zero-Inflated Poisson Autoregressive Model, and the Zero-Inflated Poisson INGARCH Model

Q4 How does the proposed model perform specifically using standard errors, mean absolute deviation error, and relative bias?

To answer this research question, tables (2) to (6) summarize the simulation results for the hurdle AR model with different sample sizes. Each sample size is presented in a separate table, with the three sets of parameters in order as listed in table (1). The purpose of this simulation as mentioned before, is to examine how the standard errors, relative bias, and MADE for the developed hurdle AR model change with different sample sizes. From the tables, it can be noticed that standard errors of the parameters, on average, have considerably small values. They also

decrease as the sample size increases. There is not noticeable difference between the standard errors for the parameters associated with the logistic part and the parameters associated with the count part of the model. For the ZIP AR and the ZIP INGARCH models, the standard errors get smaller as the sample size gets larger as well.

In terms of the relative bias that is obtained by taking the scaled difference between the estimated values and the true value , as the sample size increases, the estimates seem to get closer to the true parameters. The difference is actually more noticeable with the larger sample sizes, 200 and 500. The parameters represent the effect of the prior observation in the logistic part, and the prior λ in the count part, seem to have the largest bias in the hurdle model (i.e. γ and θ). Their bias get smaller for larger sample sizes but they still have considerably large bias.

The MADE measures how far the estimates are from the true parameter values. Generally, the MADE values are similar for all the parameters except for the parameter associated with the prior probability, β, has considerably large MADE value. As with relative bias, the values of MADE seem to get smaller as the sample sizes increase. The hurdle AR model shows generally better results for larger sample sizes as expected.

Table 2

Standard Errors, Relative Bias, and MADE for Sample Size 20

Sample Size	Parameter Scheme	Hurdle AR Model					ZIP AR Model			ZIP INGARCH Model		
		para.	Est.	(S.E.)	Rel.Bias	MADE	para.	Est.	(S.E.)	para.	Est.	(S.E.)
20	1	α	-0.7891	0.8189	-0.7689	0.3758	γ_0	-3.2848	570.720	π	0.4582	-
		β	0.4689	0.3682	-0.4552	0.3642						
		γ	1.3896	0.9363	-5.2723	0.4578	γ_1	-3.6829	2146.575			
		δ	-0.2875	0.8118	0.1256	0.2580	β_0	0.0759	0.5219	δ	0.8356	1.8105
		θ	0.4559	0.4536	1.8938	0.1582				θ	0.3341	1.1438
		ψ	-0.3556	0.9770	-1.0722	0.1918	β_1	-0.5362	0.8695	ψ	0.2089	0.9766
20	2	α	-1.114	1.2676	-0.0840	0.0352	γ_0	-1.114	1.2676	π	0.4405	-
		β	0.3306	0.4430	-0.0319	0.0271						
		γ	1.7138	1.6890	-0.3262	0.0424	γ_1	0.3306	0.4430			
		δ	0.2622	0.7777	-0.0471	0.0124	β_0	1.7138	1.6890	δ	0.8379	1.9465
		θ	0.6563	0.4568	1.4866	0.0152				θ	0.3217	1.1178
		ψ	-0.3942	1.2681	-0.1363	0.0142	β_1	0.2622	0.7777	ψ	0.2017	1.0795
20	3	α	-0.6159	0.8046	-1.1293	0.6214	γ_0	-3.0016	533.843	π	0.4685	-
		β	0.5331	0.3562	-0.8844	0.6638						
		γ	1.2701	1.1599	-22.4055	0.7638	γ_1	-4.1636	2210.748			
		δ	-0.2636	0.8604	0.3916	0.3991	β_0	0.0806	0.5366	δ	0.8366	1.8671
		θ	0.4422	0.4712	1.5616	0.2660				θ	0.3427	1.1843
		ψ	-0.2889	0.9778	-1.3468	0.3400	β_1	-0.5225	0.8673	ψ	0.2148	0.9755

Table 3

Standard Errors, Relative Bias, and MADE for Sample Size 50

Sample Size	Parameter Scheme	Hurdle AR Model					ZIP AR Model			ZIP INGARCH Model		
		para.	Est.	(S.E.)	Rel.Bias	MADE	para.	Est.	(S.E.)	para.	Est.	(S.E.)
50	1	α	-0.0496	0.3391	-1.1056	0.5497	γ_0	-4.0772	489.759	π	0.4672	-
		β	0.3874	0.3432	-1.4843	1.1874						
		γ	0.2933	0.3733	-4.6673	0.4886	γ_1	-0.1678	1020.33			
		δ	-0.0586	0.3522	-0.7067	0.3337	β_0	-0.0758	0.3365	δ	0.9189	1.8730
		θ	0.4240	0.3456	6.0683	0.4954				θ	0.2517	1.2176
		ψ	-0.1624	0.4106	-2.2499	0.4068	β_1	-0.1334	0.5522	ψ	0.2264	0.5927
50	2	α	-0.1557	0.3662	-1.3708	0.5972	γ_0	-5.6025	716.95	π	0.3577	-
		β	0.3124	0.3613	-1.3675	1.1624						
		γ	0.4102	0.3978	-4.1560	0.5993	γ_1	1.1878	1265.38			
		δ	-0.1265	0.3905	-0.4937	0.3764	β_0	-0.1302	0.3256	δ	0.9201	1.7329
		θ	0.4306	0.3500	42.0640	0.5415				θ	0.2331	1.1472
		ψ	-0.1563	0.4363	-2.9539	0.3947	β_1	-0.1359	0.5616	ψ	0.2110	0.5960
50	3	α	0.01587	0.3294	-0.9684	0.5374	γ_0	-2.8792	318.413	π	0.4820	-
		β	0.4223	0.3350	-1.5615	1.1711						
		γ	0.2584	0.3637	-9.6067	0.4310	γ_1	-0.9039	798.929			
		δ	-0.0320	0.3346	-0.7853	0.3060	β_0	-0.0275	0.3454	δ	0.9260	1.5319
		θ	0.4418	0.3228	3.0139	0.4794				θ	0.2745	0.9127
		ψ	-0.1190	0.3957	-1.6599	0.3971	β_1	-0.1305	0.5577	ψ	0.2615	0.6524

Table 4

Standard Errors, Relative Bias, and MADE for Sample Size 100

Sample Size	Parameter Scheme	Hurdle AR Model					ZIP AR Model			ZIP INGARCH Model		
		para.	Est.	(S.E.)	Rel.Bias	MADE	para.	Est.	(S.E.)	para.	Est.	(S.E.)
		α	0.0645	0.2266	-0.8626	0.4156	γ_0	-3.1109	223.865	π	0.4715	-
		β	0.3513	0.3430	-1.4391	1.1513						
100	1	γ	0.1212	0.2306	-2.5161	0.2691	γ_1	-0.0108	457.314			
		δ	-0.0872	0.2189	-0.5637	0.2306	β_0	-0.1152	0.2507	δ	0.8464	1.5498
		θ	0.4184	0.3004	5.9736	0.4899				θ	0.3335	0.9749
		ψ	-0.0650	0.2229	-1.5006	0.2615	β_1	-0.0586	0.3952	ψ	0.2108	0.4084
		α	-0.0003	0.2377	-1.0007	0.4279	γ_0	-4.8146	396.418	π	0.4610	-
		β	0.2662	0.3408	-1.3132	1.11627						
100	2	γ	0.1829	0.2366	-2.4076	0.3548	γ_1	0.9813	706.486			
		δ	-0.1223	0.2346	-0.5106	0.2612	β_0	-0.1617	0.2410	δ	0.8287	1.3759
		θ	0.4292	0.3169	41.9206	0.5286				θ	0.3358	0.8648
		ψ	-0.0719	0.2387	-1.8989	0.2561	β_1	-0.0690	0.3899	ψ	0.18401	0.4260
		α	0.1164	0.2266	-0.7760	0.4190	γ_0	-1.8909	103.692	π	0.4842	-
		β	0.3897	0.3317	-1.5196	1.13971						
100	3	γ	0.0909	0.2220	-4.0306	0.2328	γ_1	-0.3927	264.319			
		δ	-0.0306	0.2030	-0.7955	0.2018	β_0	-0.040	0.2504	δ	0.8674	1.2692
		θ	0.3925	0.2943	2.5689	0.4542				θ	0.3356	0.7574
		ψ	-0.0643	0.2077	-1.3575	0.2821	β_1	-0.0556	0.3953	ψ	0.2314	0.3732

Table 5

Standard Errors, Relative Bias, and MADE for Sample Size 200

Sample Size	Parameter Scheme	Hurdle AR Model					ZIP AR Model			ZIP INGARCH Model		
		para.	Est.	(S.E.)	Rel.Bias	MADE	para.	Est.	(S.E.)	para.	Est.	(S.E.)
		α	0.1209	0.1638	-0.7427	0.3525	γ_0	-1.9872	2.5518	π	0.4753	-
		β	0.3530	0.3414	-1.4413	1.1530						
200	1	γ	0.0414	0.1536	-1.5179	0.1716	γ_1	-0.1604	144.544			
		δ	-0.0766	0.1455	-0.6168	0.1725	β_0	-0.1119	0.1834	δ	0.9171	1.2152
		θ	0.3991	0.2930	5.6519	0.4924				θ	0.2531	0.7790
		ψ	-0.0298	0.1393	-1.2296	0.1924	β_1	-0.0410	0.2826	ψ	0.2653	0.2552
		α	0.0731	0.1722	-0.8258	0.3501	γ_0	-3.7806	190.5029	π	0.4649	-
		β	0.2384	0.3357	1.2805	1.0884						
200	2	γ	0.09471	0.1571	-1.7286	0.2487	γ_1	0.9614	309.062			
		δ	-0.1256	0.1627	-0.4974	0.2058	β_0	-0.1970	0.1812	δ	0.9082	1.5664
		θ	0.4244	0.2941	41.4472	0.5328				θ	0.2399	1.0559
		ψ	-0.0363	0.1492	-1.4541	0.1782	β_1	-0.03531	0.2862	ψ	0.2447	0.2704
		α	0.1505	0.1676	-0.7104	0.3744	γ_0	-1.1426	17.2728	π	0.4851	-
		β	0.3785	0.3329	-1.5047	0.3329						
200	3	γ	0.0396	0.1471	-2.3223	0.1471	γ_1	-0.1321	48.8522			
		δ	-0.0387	0.1343	-0.7417	0.1343	β_0	-0.0288	0.1790	δ	0.9294	1.3302
		θ	0.3861	0.2677	2.5107	0.2677				θ	0.2674	0.8168
		ψ	-0.0130	0.2076	-1.0723	0.1285	β_1	-0.0258	0.2780	ψ	0.2855	0.2651

Table 6

Standard Errors, Relative Bias, and MADE for Sample Size 500

Sample Size	Parameter Scheme	Hurdle AR Model					ZIP AR Model			ZIP INGARCH Model		
		para.	Est.	(S.E.)	Rel.Bias	MADE	para.	Est.	(S.E.)	para.	Est.	(S.E.)
		α	0.1464	0.1191	-0.6884	0.3249	γ_0	-1.2463	2.6554	π	0.4745	-
		β	0.3144	0.3460	-1.3930	1.1144						
500	1	γ	0.0152	0.0939	-1.1900	0.1196	γ_1	-0.0274	9.1718			
		δ	-0.0819	0.0919	-0.5904	0.1357	β_0	-0.1094	0.1183	δ	0.8522	0.7335
		θ	0.3983	0.2608	5.6394	0.5146				θ	0.3405	0.4593
		ψ	0.0042	0.0817	-0.9675	0.1343	β_1	-0.0151	0.1798	ψ	0.1696	0.1601
		α	0.1056	0.1171	-0.7484	0.3151	γ_0	-2.1085	23.499	π	0.4630	-
		β	0.1739	0.3129	-1.2046	1.0239						
500	2	γ	0.0538	0.0935	-1.4145	0.1900	γ_1	0.2207	47.248			
		δ	-0.1112	0.1044	-0.5551	0.16458	β_0	-0.1894	0.1212	δ	0.8326	1.0449
		θ	0.4489	0.2770	43.8983	0.5507				θ	0.3427	0.6842
		ψ	-0.0089	0.0859	-1.1116	0.1141	β_1	-0.0232	0.1843	ψ	0.1457	0.2112
		α	0.1708	0.1262	-0.6714	0.3503	γ_0	-0.8355	0.3298	π	0.4862	-
		β	0.3628	0.3372	-1.4838	1.1128						
500	3	γ	0.0147	0.0907	-1.4915	0.0886	γ_1	0.0009	0.5412			
		δ	-0.0394	0.0803	-0.7367	0.1195	β_0	-0.0144	0.1124	δ	0.8820	0.7931
		θ	0.3879	0.2268	2.5269	0.5119				θ	0.3294	0.4878
		ψ	0.01300	0.0754	-0.9277	0.1686	β_1	-0.0140	0.1734	ψ	0.2218	0.1593

Q5 How can the prediction of future observations be obtained and evaluated for the proposed model compared to existing times series models of counts?

To answer this research question, tables (7) to (11) show the calculation of the predicted root mean squared error (PRMSE) once using the predicted observations (PRMSE1) and the other calculation using the predicted mean (PRMSE2) for the three sets of parameters of each sample size, where smaller values of PRMSE1 and PRMSE2 indicate better predictions. Using the predicted observations to evaluate the accuracy of the prediction (PRMSE1) shows that the hurdle AR model and the ZIP AR model have smaller values compared to the ones produced when fitting the ZIP INGARCH model. This is expected since this model does not allow the parameter π to change over time and thus uses less information to predict future observations. The hurdle AR and the ZIP AR show similar results in PRMSE1. Using the predicted mean to evaluate the prediction (PRMSE2) shows that the hurdle AR and the ZIP AR models result in a slightly smaller values than ZIP INGARCH in most cases. The values increase for the three models across steps-ahead h=(1, 2, 3, 4, 5) as expected. Therefore one can expect that the chance of making a good prediction becomes lower as we get further from the present.

Table 7

Predicted Mean Squared Error for h steps-ahead prediction of size 5

Sample Size	Parameter Scheme	h-step	Hurdle AR Model		ZIP AR Model		ZIP INGARCH Model	
			PRMSE1	PRMSE2	PRMSE1	PRMSE2	PRMSE1	PRMSE2
5	1	1	1.0817	1.0294	0.9916	0.8655	1.3357	0.9448
		2	1.2094	1.1509	1.1086	0.9677	1.4934	1.0563
		3	1.3965	1.3290	1.28020	1.1174	1.7245	1.2198
		4	1.7103	1.6277	1.5679	1.3685	2.1120	1.4939
		5	2.4206	2.3019	2.2173	1.9354	2.9869	2.1127
5	2	1	1.2486	1.1284	0.9603	0.8572	1.2967	0.9248
		2	1.3960	1.2616	1.0736	0.9584	1.4498	1.0339
		3	1.6120	1.4568	1.2397	1.1067	1.6741	1.1939
		4	1.9743	1.7842	1.5184	1.3554	2.0503	1.4622
		5	2.7920	2.5233	2.1473	1.9169	2.8996	2.0679
5	3	1	1.3739	1.3891	1.0682	0.9284	1.4820	1.0358
		2	1.5361	1.5531	1.1943	1.0379	1.6569	1.1581
		3	1.7737	1.7934	1.3791	1.1985	1.9132	1.3373
		4	2.1723	2.1965	1.6891	1.4679	2.3432	1.6378
		5	3.0722	3.1063	2.3887	2.0759	3.3138	2.3162

Table 8

Predicted Mean Squared Error for h steps-ahead prediction of size 20

Sample Size	Parameter Scheme	h-step	Hurdle AR Model		ZIP AR Model		ZIP INGARCH Model	
			PRMSE1	PRMSE2	PRMSE1	PRMSE2	PRMSE1	PRMSE2
20	1	1	1.0702	0.9730	0.9982	0.9053	1.3755	0.9631
		2	1.0980	0.9800	1.0241	0.9288	1.4112	0.9881
		3	1.1281	1.0300	1.0522	0.9543	1.4499	1.0152
		4	1.1608	1.0600	1.0827	0.9819	1.4919	1.0446
		5	1.1965	1.0921	1.1160	1.0122	1.5378	1.0768
20	2	1	1.0398	0.9230	0.9587	0.8768	1.2832	0.9325
		2	1.0668	0.9470	0.9836	0.8996	1.3165	0.9567
		3	1.0961	0.9730	1.0106	0.9243	1.3526	0.9829
		4	1.1278	1.0001	1.0399	0.9511	1.3918	1.0114
		5	1.1625	1.0300	1.0719	0.9803	1.4346	1.0425
20	3	1	1.1397	1.1278	1.0372	0.9276	132.665	58.8697
		2	1.1693	1.1571	1.0641	0.9517	136.112	60.3990
		3	1.2014	1.1889	1.0933	0.9778	139.841	62.0541
		4	1.2362	1.2233	1.1250	1.0061	143.896	63.8532
		5	1.2742	1.2610	1.1596	1.0371	148.324	65.8183

Table 9

Predicted Mean Squared Error for h steps-ahead prediction of size 25

Sample Size	Parameter Scheme	h-step	Hurdle AR Model		ZIP AR Model		ZIP INGARCH Model	
			PRMSE1	PRMSE2	PRMSE1	PRMSE2	PRMSE1	PRMSE2
25	1	1	1.0343	0.9161	0.9746	0.8977	1.4009	0.9631
		2	1.0556	0.9350	0.9947	0.9162	1.4298	0.9830
		3	1.0783	0.9551	1.0161	0.9359	1.4606	1.0041
		4	1.1026	0.9765	1.0389	0.9569	1.4934	1.0267
		5	1.1285	0.9995	1.0634	0.9794	1.5286	1.0508
25	2	1	1.0150	0.9045	0.9456	0.8741	1.3048	0.9290
		2	1.0359	0.9232	0.9651	0.8921	1.3317	0.9482
		3	1.0582	0.9430	0.9859	0.9113	1.3604	0.9686
		4	1.0820	0.9642	1.0080	0.9318	1.3910	0.9903
		5	1.1075	0.9869	1.0317	0.9537	1.4237	1.0136
25	3	1	1.0819	0.9561	1.0196	0.9381	1.5504	1.0305
		2	1.1042	0.9758	1.0406	0.9575	1.5824	1.0518
		3	1.1279	0.9968	1.0630	0.9781	1.6164	1.0744
		4	1.1533	1.0192	1.0869	1.0001	1.6528	1.0986
		5	1.1804	1.0432	1.1125	1.0236	1.6917	1.1244

Table 10

Predicted Mean Squared Error for h steps-ahead prediction of size 50

Sample Size	Parameter Scheme	h-step	Hurdle AR Model		ZIP AR Model		ZIP INGARCH Model	
			PRMSE1	PRMSE2	PRMSE1	PRMSE2	PRMSE1	PRMSE2
50	1	1	1.0533	0.9278	0.9702	0.9119	1.4726	0.9670
		2	1.0640	0.9372	0.9800	0.9212	1.4875	0.9768
		3	1.0750	0.9469	0.9902	0.9307	1.5030	0.9869
		4	1.0864	0.9569	1.0007	0.9406	1.5189	0.9974
		5	1.0982	0.9673	1.0115	0.9507	1.5353	1.0081
50	2	1	1.0222	0.9750	0.9430	0.8829	1.3573	0.9270
		2	1.0326	0.9849	0.9525	0.8919	1.3711	0.9364
		3	1.0433	0.9951	0.9624	0.9011	1.3853	0.9461
		4	1.0543	1.0057	0.9726	0.9107	1.4000	0.9561
		5	1.0657	1.0165	0.9831	0.9205	1.4151	0.9665
50	3	1	1.0733	0.9630	1.0020	0.9397	1.6544	1.0445
		2	1.0842	0.9728	1.0122	0.9492	1.6711	1.0551
		3	1.0955	0.9829	1.0227	0.9590	1.6885	1.0661
		4	1.10711	0.9933	1.0335	0.9692	1.7063	1.0773
		5	1.1190	1.0040	1.0447	0.9797	1.7248	1.0890

Table 11

Predicted Mean Squared Error for h steps-ahead prediction of size 100

Sample Size	Parameter Scheme	h-step	Hurdle AR Model		ZIP AR Model		ZIP INGARCH Model	
			PRMSE1	PRMSE2	PRMSE1	PRMSE2	PRMSE1	PRMSE2
100	1	1	1.0525	0.9108	0.9673	0.9103	1.3747	0.9598
		2	1.0578	0.9154	0.9722	0.9148	1.3817	0.9647
		3	1.0632	0.9200	0.9772	0.9148	1.3887	0.9696
		4	1.0687	0.9247	0.9822	0.9242	1.3958	0.9746
		5	1.0742	0.9295	0.9873	0.9290	1.4031	0.9796
100	2	1	1.0232	0.8891	0.9373	0.8775	1.2309	0.9172
		2	1.0292	0.8930	0.9420	0.8819	1.2371	0.9218
		3	1.0300	0.8982	0.9468	0.8864	1.2434	0.9265
		4	1.0390	0.9022	0.9516	0.8910	1.2498	0.9313
		5	1.0411	0.9071	0.9566	0.8956	1.2563	0.9361
100	3	1	1.0808	0.9441	0.9969	0.9435	1.5601	1.0150
		2	1.0862	0.9489	1.0019	0.9482	1.5680	1.0201
		3	1.0918	0.9537	1.0070	0.9530	1.5760	1.0253
		4	1.0974	0.9586	1.0122	0.9579	1.5841	1.0306
		5	1.1032	0.9637	1.0174	0.9629	1.5923	1.0360

PRMSE1 and PRMSE2, however, are not enough to evaluate the quality of the prediction of each model. To demonstrate the actual prediction, figures (2) to (11) show a few randomly selected datasets from the 1000 replicates for each sample size. For each sample size, the generated data are plotted once with the predicted data for each model and once with the predicted means for each model. The red line represents the generated data that are used for prediction comparison purposes, the green dashed line represents the hurdle data and the hurdle mean, the blue long-dashed line represents the ZIP AR data and the ZIP AR mean, and the purple long-dashed line represents the ZIP INGARCH data and the ZIP INGARCH mean. It can be clearly seen from the plots that the predicted hurdle data track the actual data by moving upward and downward as the actual data do. The ZIP AR and the ZIP INGARCH, however, not surprisingly show straight lines, indicating a failure to track the data. Obviously, the same results are found using the means of the three data sets. The hurdle mean at each time point tracks the actual data, while the means of the other two data sets show just straight lines that clearly do not predict the data correctly. An obvious interpretation of the findings is that the additional parameters included in the hurdle model add more information about the time series data, and thus, the developed model contributes by adding significant information that helps predict future observations. From the plots, it can be seen that the interpretation of the similar results of PRMSE1 and PRMSE2 for the three models but much better prediction for the hurdle AR model by plotting the data is that the actual data generally have many zeros, which makes the values of the PRMSE1 and PRMSE2 for the ZIP AR and ZIP INGARCH models appear to be

small, even though they do not show good predictions of the future observations, as the hurdle AR model does.

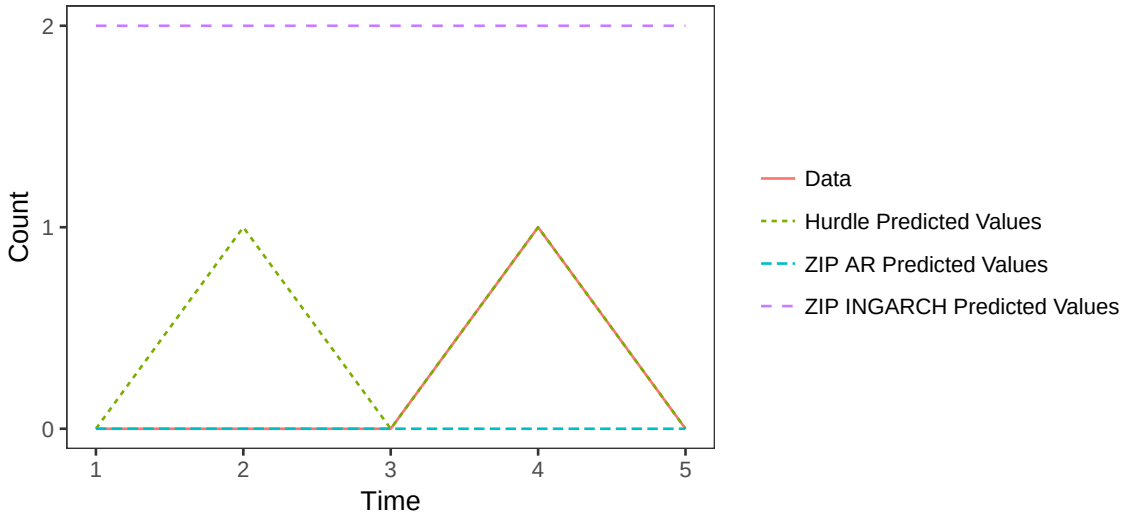

Figure 2. Predicted Time Series Data of Size 5

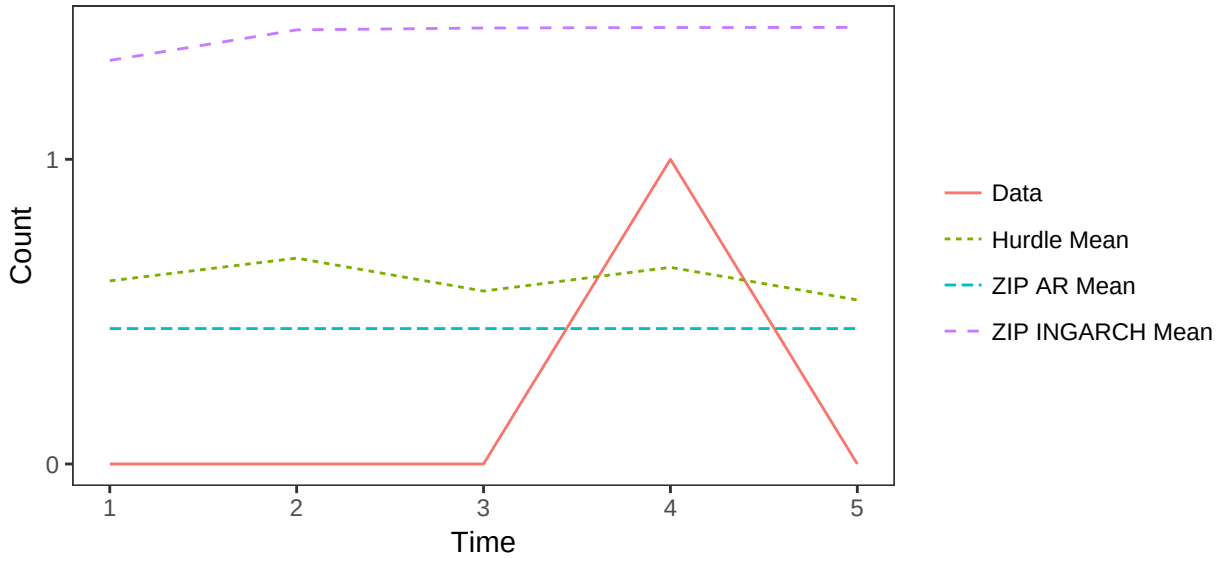

Figure 3. Predicted Time Series Mean of Size 5

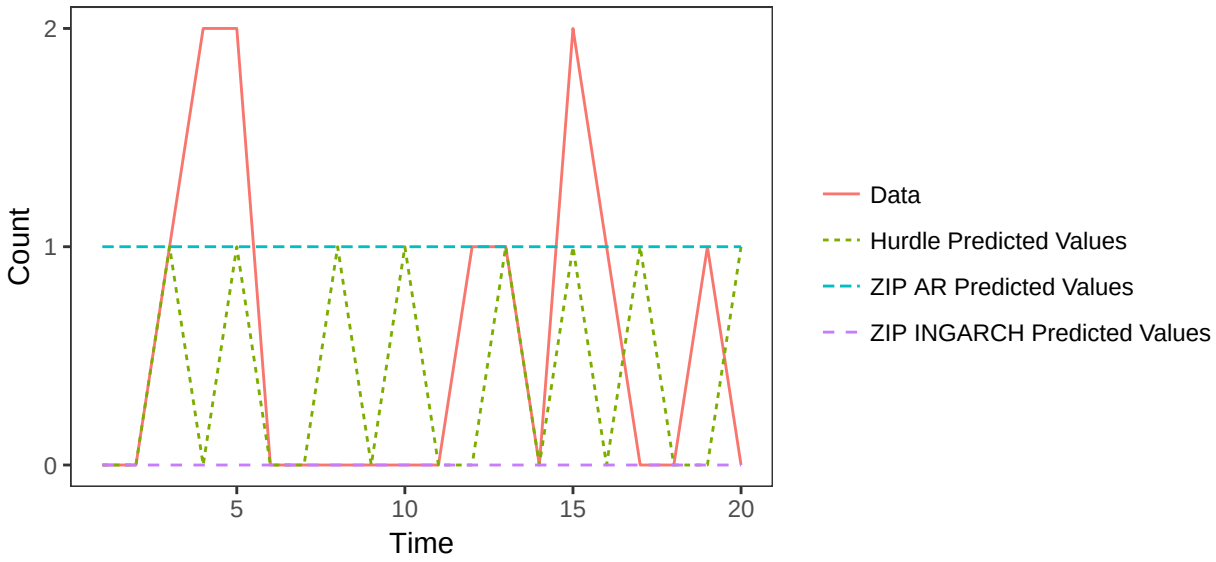

Figure 4. Predicted Time Series Data of Size 20

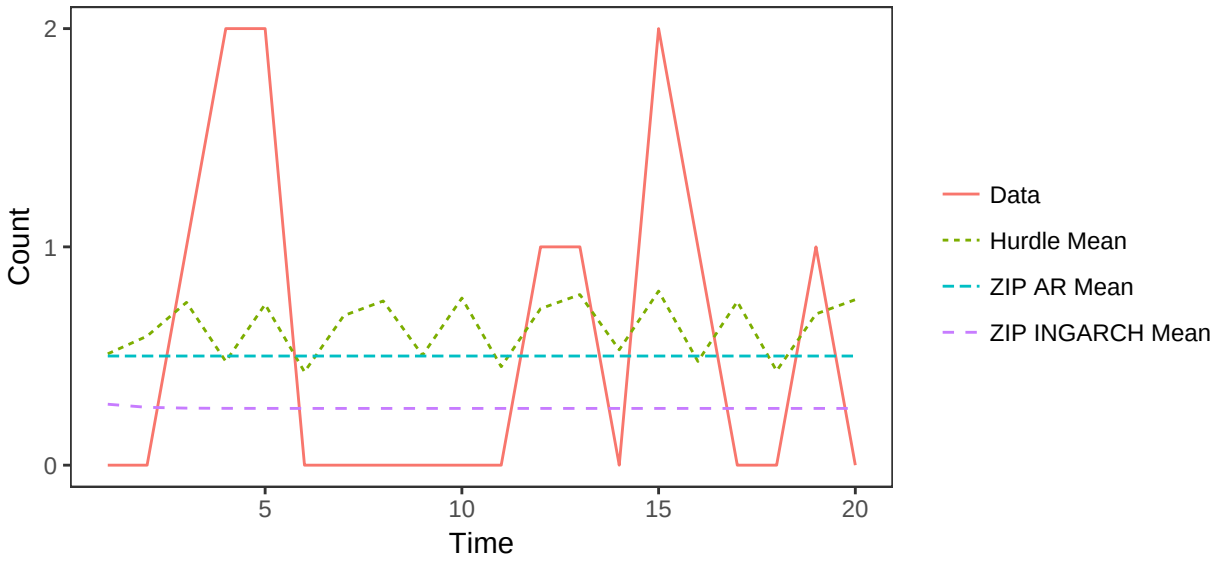

Figure 5. Predicted Time Series Mean of Size 20

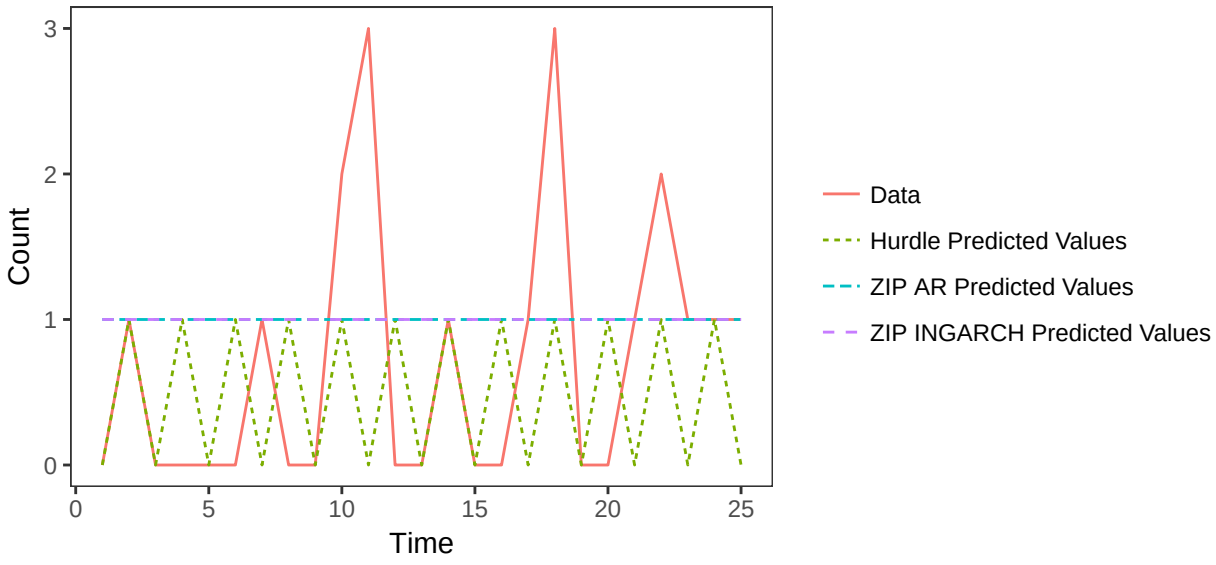

Figure 6. Predicted Time Series Data of Size 25

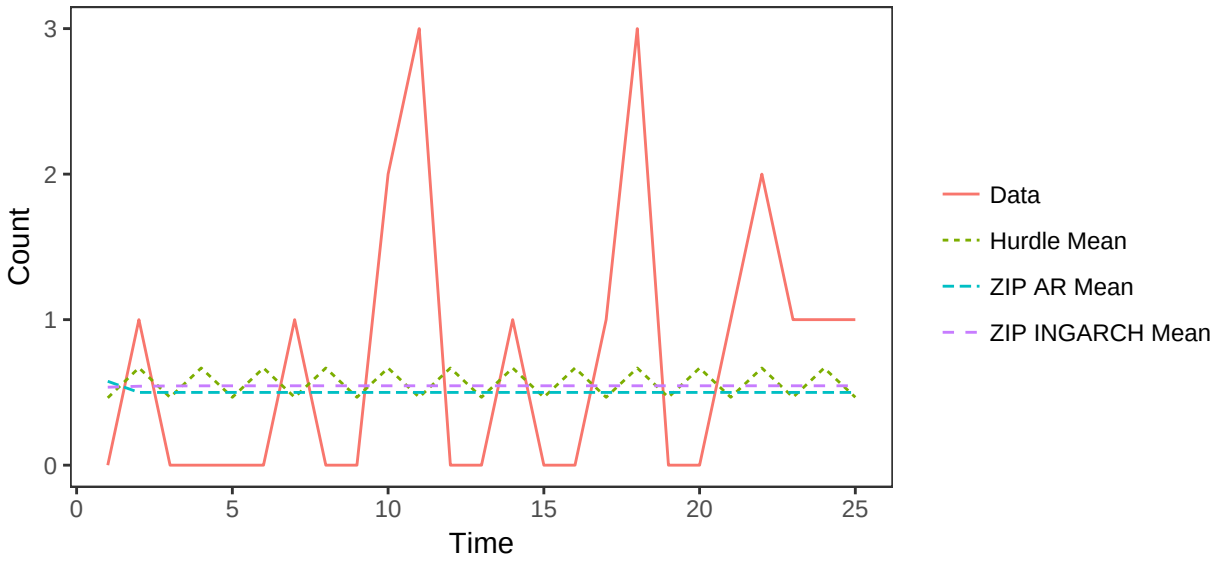

Figure 7. Predicted Time Series Mean of Size 25

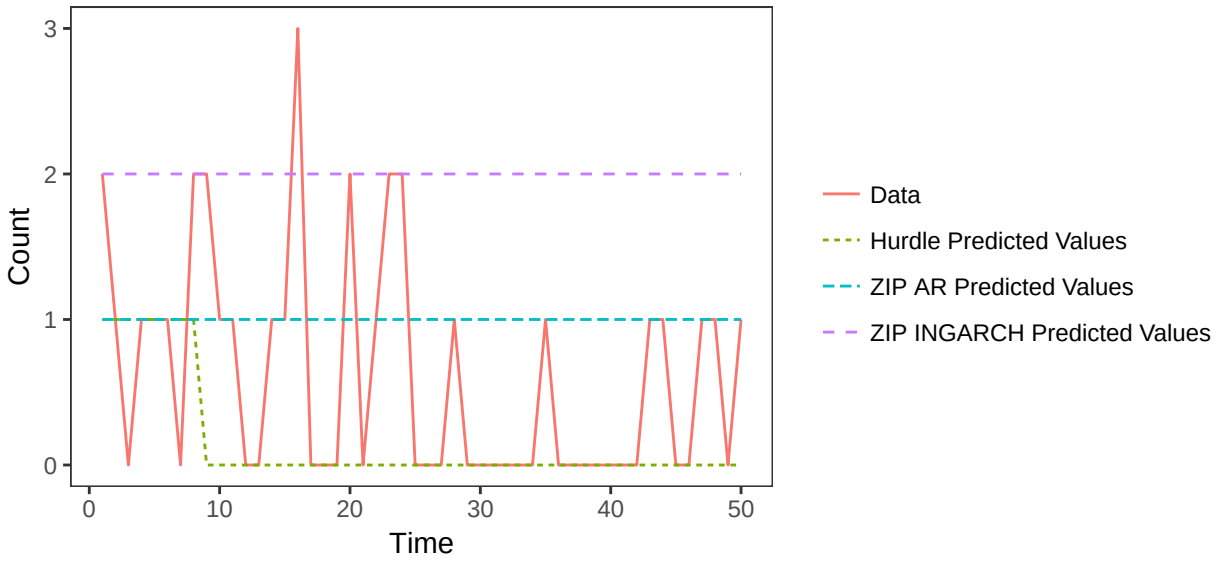

Figure 8. Predicted Time Series Data of Size 50

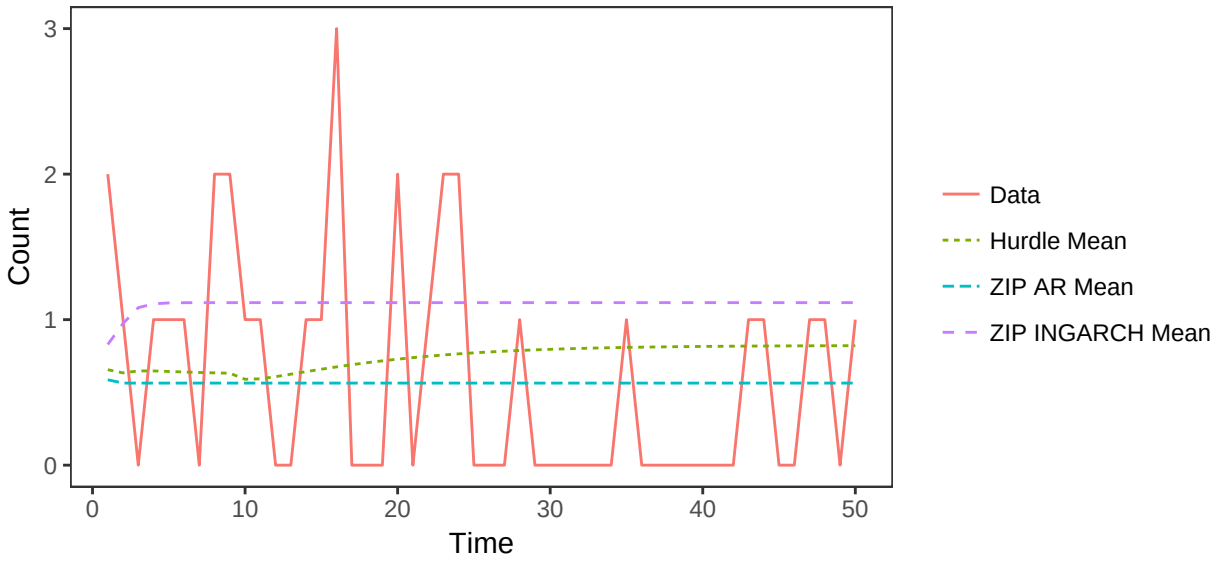

Figure 9. Predicted Time Series Mean of Size 50

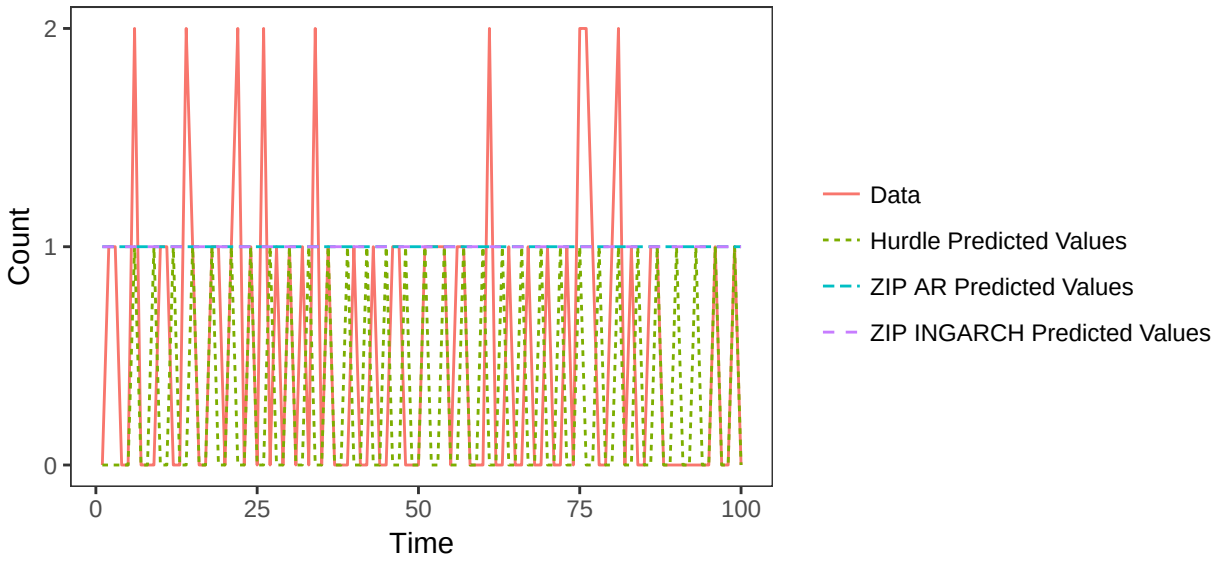

Figure 10. Predicted Time Series Data of Size 100

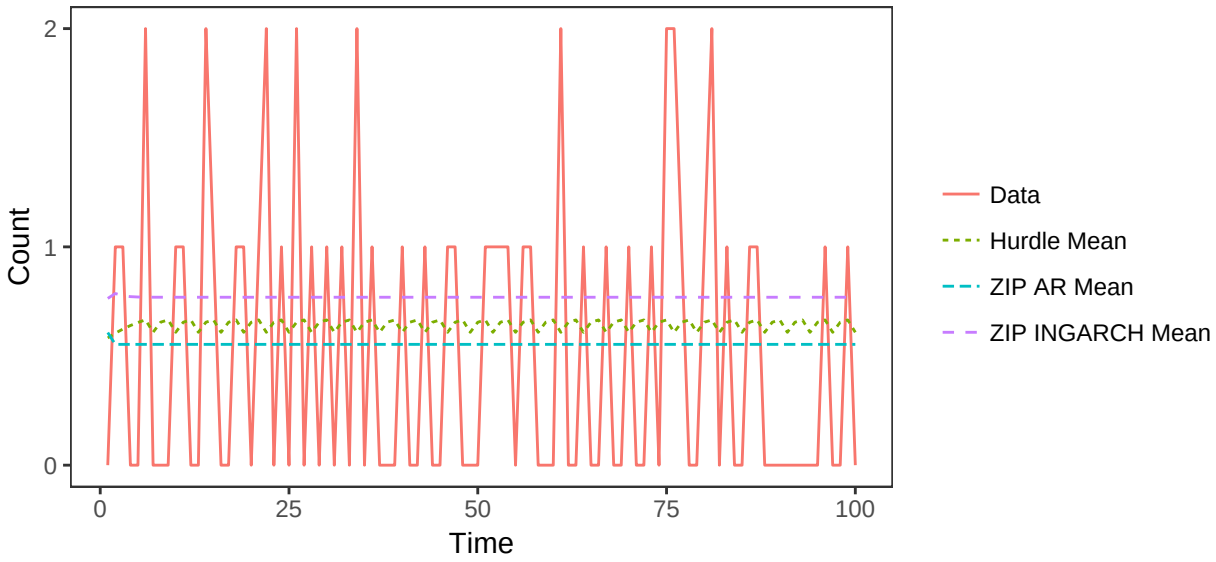

Figure 11. Predicted Time Series Mean of Size 100

Standard Errors with Different Sample Sizes

One of the issues that is encountered during the simulation is the calculation of the standard errors. The Hessian matrix was calculated in R for the proposed model using the function OptimHess, which gives an approximation of the Hessian matrix and probably not the exact values. Standard errors are obtained by taking the square roots of the diagonal of the inverse of the negative Hessian matrix. During the 1000 runs in the simulation, some of the estimated Hessian matrices are not positive definite, thus the inverse of them, which are the covariance matrices, are not positive definite either. That give some negative values of the diagonal line of the covariance matrices (i.e. negative variances), thus the standard errors are not real numbers. This issue is found only in the small sample sizes (20 and 50) and is not present in the larger sample sizes (200 and 500). Most of the previous studies of count time series ran their simulations based on large sample sizes (Christou, 2013; Davis, Dunsmuir, and Streett, 2003; Drost, Van den Akker, and Werker, 2008b; Fokianos, Rahbek, and Tjøstheim, 2009; Maiti, Biswas, Guha, and Ong, 2014; Yang, 2012). Therefore, this model performs better with larger sample sizes. Another possible reason for the negative variances is because the function used in R does not give the exact calculation of the Hessian. Therefore it is of interest in the future to compare an exact hand calculation of the Hessian to program-based calculation. Table (12) below shows the percentages of completed positive standard errors with different sample sizes for the 1000 replications. As can be seen, this issue is mostly noticed with the sample sizes of 20 and 50 for the hurdle AR and the ZIP AR

models. Those sample sizes were not considered in previous research in this area. It is found that even when there are negative standard errors in the hurdle or ZIP AR model, they are only for the parameters associated with the logistic part and not for the count part. In the ZIP INGARCH model, however, this issue is found even in the large sample sizes. In this model, only a few percentages of the standard errors are found to be positive values, even though this study does not really model the probability.

Table 12

Percentages of Complete Positive Standard Errors for Each Sample Size

Sample Size	Hurdle AR Model (S.E.)	ZIP AR Model (S.E.)	ZIP INGARCH Model (S.E.)
20	59%	100%	30%
50	90.2%	88.5%	22%
100	97.4%	98.9%	13.5%
200	100%	100%	21%
500	100%	100%	10.4%

Real Data Example

In this section, to illustrate the use of the developed hurdle autoregressive model, a real data example is analyzed. The data set is downloaded from (http://www.forecastingprinciples.com), where there are several data sets that have

been collected over many years from Pittsburgh, Pennsylvania and Rochester, New York. They represent numbers of different types of crimes include 6 million offense incident reports and the Pittsburgh computer-aided dispatch (CAD) calls. All the data sets have been processed into monthly time series data that were collected from January 1990 to December 2001. In the first grant in Pittsburgh, Pennsylvania, researchers collected all the crime offense reports and CAD calls from the police center from 1990 to 1998. After that, there was a second grant, and the data for the years 1999-2001 were added. Because Pittsburgh started using a new record management system at the time of the second grant, researchers had to reprocess all the data from the first grant to ensure that the data were treated identically and to make a smooth connection to the data collected for the second grant. The data were collected over five areas in Pittsburgh (named *census tracts*, *4,000 foot-grid cells*, *car beats*, an aggregation of car beats that is called *car beats plus*, and *precincts*) with several geographic areas under each of these parts. There are 24 types of crimes included in their data, including aggravated assault, arson, burglary, criminal mischief, weapons, family violence, gambling, larceny, liquor law violations, motor vehicle theft, murder/manslaughter, prostitution, public drunkenness, rape, robbery, simple assaults, trespassing, and vandalism. For the purpose of this study, the variable to include is selected based on the frequency of occurrence. The crime of arson is found to have a high frequency of zeros compared to the other crimes, and thus is the response variable in this study that was collected monthly over the 12 years. Moreover, since the focus of this study is not on spatial time series, only one of the 15 geographic areas in *car beats plus* is considered, which is the 13th police

car beats plus. Arson in Pittsburgh, Pennsylvania rarely happened in the period between 1990 and 2001, totaling 144 observations, with 80 of them recorded as zeros, which is around 55% of the data. The mean and variance of the data are 0.6667 and 0.8671, respectively. The time series plot is shown in figure (12). To evaluate the prediction of the hurdle autoregressive model, the previous ZIP AR model developed by Yang (2012) and the ZIP INGARCH model developed by Zhu (2012) are fit to the same dataset. The three models are compared in terms of their prediction capabilities. To calculate the predicted root mean squared error (PRMSE), the observations are divided into two parts, where the first part is used to estimate the parameters and the second part is used for prediction.

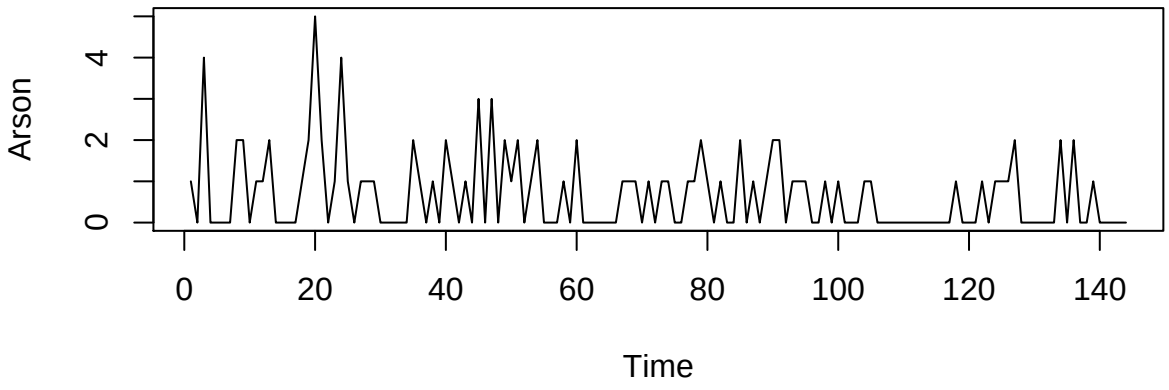

Figure 12. Time Series Plot for Monthly Arson Counts in Pittsburgh, Pennsylvania from 1990 to 2001

Table 13

Parameter Estimates and Standard Errors for Hurdle AR Model, ZIP AR Model, and ZIP INGARCH Models

Parameter	Hurdle AR Model		Parameter	ZIP AR Model		Parameter	ZIP INGARCH Model	
	Estimate	Standard Error		Estimate	Standard Error		Estimate	Standard Error
α	0.0642	0.3752	γ_0	-2.5555	2.975	π	0.4312	-
β	-0.7978	0.5874						
γ	-0.0638	0.3049	γ_1	1.5755	3.0593			
δ	-0.1649	0.2136	β_0	-0.2006	0.26251	δ	0.9385	1.5747
θ	0.0443	0.2765				θ	0.2525	0.8954
ψ	0.1181	0.1479	β_1	0.2496	0.35166	ψ	0.2975	0.3342

From Table (13), the standard errors for the parameters of the hurdle model appear to be small for all the parameters. In terms of the ZIP AR model, the standard errors are much larger in the logistic part than the ones in the count part. The ZIP INGARCH model larger standard error for the intercept than the other two parameters.

The first 110 observations in the crime data are used to fit the three models, and the remaining 34 are then used to find the PRMSE(h) for varying h. The PRMSE is calculated for the three models. It is calculated once using actual predicted observations (PRMSE1) and the second time using the predicted means (PRMSE2), as done in the simulation. In PRMSE1, the actual observations are compared to the predicted observations, whereas in PRMSE2, the observations are compared to the predicted means. Looking at Table (14), the PRMSE2 has lower

values than PRMSE1 for all the three models.The hurdle AR and the ZIP INGARCH models have similar PRMSE1 and PRMSE2 values, whereas the ZIP AR model has higher values. The prediction in general gets less accurate as further steps are performed. Plots of the predicted data sets are shown in figures (13) and (14). In this example, none of the models shows a good prediction of the number of arson situations happening in the next 34 months.

Table 14

Predicted Mean Squared Error for h Steps-Ahead Prediction for Arson Data

***h*-step**	Hurdle AR		ZIP AR		ZIP INGARCH	
	PRMSE1	PRMSE2	PRMSE1	PRMSE2	PRMSE1	PRMSE2
1	0.7276	0.7377	0.9074	0.7571	0.7276	0.7279
2	0.7385	0.7488	0.9211	0.7685	0.7385	0.7388
3	0.7500	0.7604	0.9354	0.7804	0.7500	0.7503
4	0.7620	0.7726	0.9503	0.7929	0.7620	0.7623
5	0.7745	0.7854	0.9660	0.8060	0.7745	0.7749

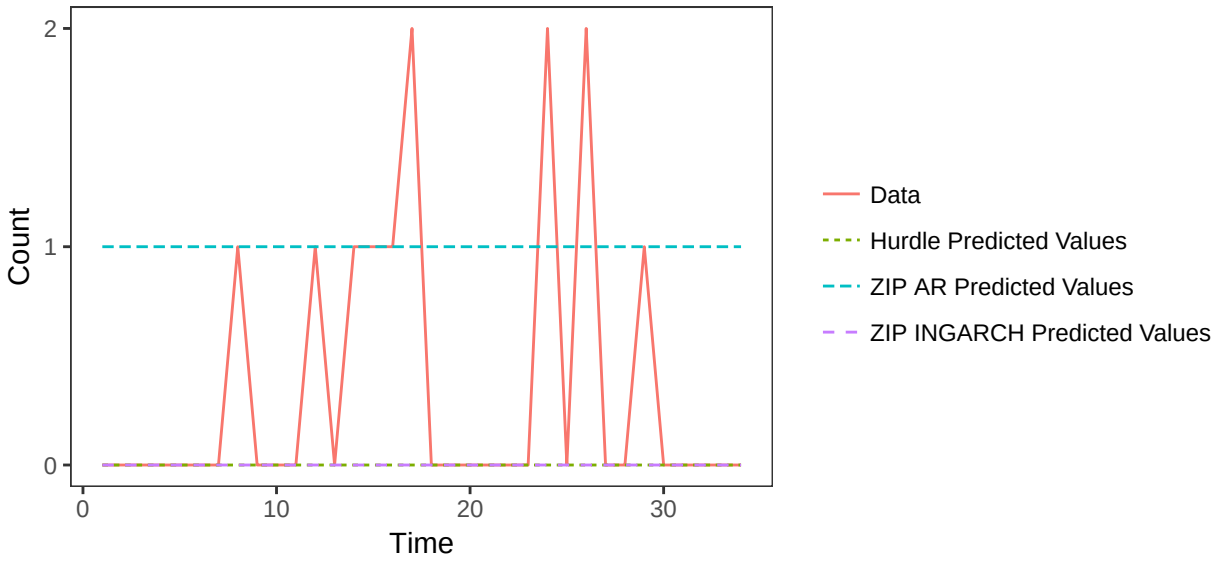

Figure 13. Predicted Time Series Arson Data

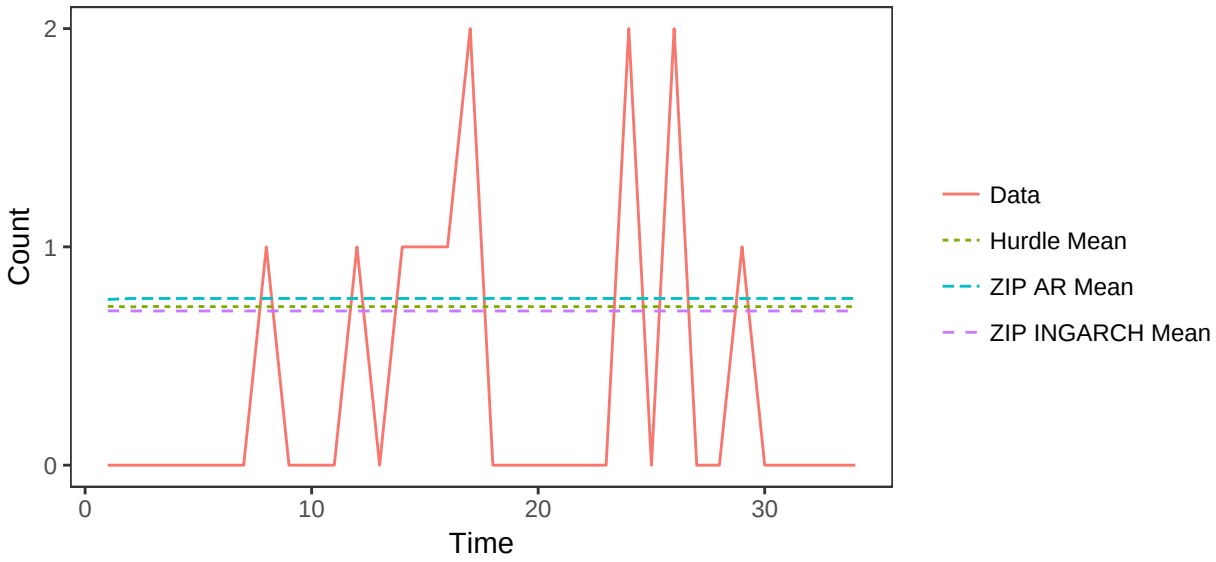

Figure 14. Predicted Time Series Arson Mean

CHAPTER V

CONCLUSIONS AND FUTURE RESEARCH

Conclusions

In this dissertation, a first-order hurdle autoregressive model is developed to fit hurdle time series data. A ZIP autoregressive model and a ZIP INGARCH model are fitted to hurdle time series data. The likelihood estimation method is used to estimate the regression coefficients. A simulation study of 1,000 replicates for the three models is conducted, and the results for the three models are compared in terms of prediction based on different sample sizes with different parameter settings.

Simulation results show that the three models perform better when the sample size is 100 or more. Thus, it is recommended to use considerably large sample sizes when working with count autoregressive time series models. For the developed hurdle AR model, the results in terms of standard errors are more sufficient compared to the other models, especially when the sample size is small (i.e. 20 and 50). The issue of standard errors, as mentioned in the previous chapter, is less noticeable in the hurdle model than in the ZIP AR and ZIP INGARCH models. The averaged standard error values for the hurdle model shown by the simulation get smaller as the sample size gets larger. The ZIP AR model results in very large standard error values for the logistic part, especially when the sample size is small, even though the exact calculations are provided, whereas this does not

appear in the hurdle model. In terms of the ZIP INAGRCH model, for the zero-inflation parameter π, there is no second derivative provided in the paper by Zhu (2012) in order to obtain the standard errors. The standard errors for the count parameters are small, although the percent of having simulations with all positive standard errors is very small for the 1,000 runs. As mentioned in chapter IV, it is recommended to calculate the standard errors by hand in order to obtain the exact values because the hurdle and ZIP INAGRCH models are recursive, and the calculation of the derivative by programming may not be as accurate.

Moreover, the hurdle model shows generally lower relative bias values for all the parameters as the sample size gets larger. that is applied to the MADE as well. The simulation shows that the estimation method gives small absolute deviation errors for larger sample sizes. Therefore, the estimates seem to converge to the true parameter values as the sample size increases.

The prediction of future observations is evaluated using what is called the predicted root mean squared error (PRMSE), where a smaller value indicates a better prediction. Generally, the PRMSE values for the hurdle AR and the ZIP AR are similar, and they are smaller than the values produced by the ZIP INGARCH model. Even though, the PRMSE values for the ZIP INGARCH model are not much higher than the other two models, the model is not represented as a generalized linear model, as the other two models are. The ZIP and hurdle models, in general, are analogues of GLMs. The means of the response in the GLMs are usually connected to the predictors using a specific link function, depending on the distribution. The two known link functions that are used in the ZIP distribution are

the logit function for the binary component and the log function for the count component. This idea of using the link functions in ordinary regression analysis should be generalized for time series analysis (Kedem and Fokianos, 2002). In the ZIP INGARCH model, however, the model is assumed to be linear, which is not consistent with GLMs in general. Therefore, the results would be more comparable if the link functions were used in defining the ZIP INGARCH model.

Even though the PRMSE values are not very different between the hurdle AR and ZIP AR models, plotting the predicted data based on the three models shows that the hurdle model is superior in terms of prediction of future observations compared to the other models. The ZIP AR and the ZIP INGARCH generally show straight lines when predicting future data. The potential reason behind that is that the ZIP INGARCH does not model the zero-inflation parameter (π) and thus less information is used to predict the future. The ZIP INGARCH model assumes that the π does not change over time, which may negatively affect the prediction because the same π value is used to predict all the future observations. The same point is applied to the ZIP AR model, where more parameters are needed in order to obtain a more accurate prediction.

The misspecification of the distribution is a possible important point that can have an impact on the results. In general, the ZIP data and the hurdle data are both excess-zero count data, but the assumption about the sources of the zeros is different. The hurdle distribution assumes that zeros come from one distribution and the positive counts have another distribution, whereas the ZIP distribution assumes that zeros can come from two distributions. In real life, however, it is

unusual to know which zeros come from which distribution which makes the ZIP distribution more complicated to study.

A real data example is analyzed in chapter IV to demonstrate how the developed model performs in real-life setting compared to other models. The data are divided to sets, one for estimation and the other for prediction in order to calculate the PRMSE values. PRMSE are lower for the hurdle AR and th ZIP INGARCH models than the ZIP AR model. Time series plots of the future counts of arson in Pittsburgh, Pennsylvania, however, indicate that the three models do not seem to predict the future data accurately.

In conclusion, the hurdle AR model has the best predictive capability compared to the ZIP AR and the ZIP INGARCH models. That can explain the importance of adding the prior observation to the model instead of just having an indicator that gives 0 or 1 values depending on whether the observation is a zero or a count, as the ZIP AR suggested. Another possible explanation is that adding the prior count parameter to the log-linear model part can contribute to better results, especially in terms of prediction. For the ZIP INGARCH model, the prediction is not as good as the hurdle AR model because Zhu (2012) only modeled the count part in ZIP not the logistic part.

Future Research and Limitations

In this dissertation, a hurdle autoregressive time series model is developed. Although autoregressive processes are useful and widely used in time series analysis, it is of future interest to generalize the developed model and incorporate the moving average or more complex processes to account for different correlation structures in

the data. Additionally, the hurdle autoregressive model in this study is assumed to be applied to univariate data. The idea of multivariate analysis, however, should be generalized and work well when there are multiple correlated outcome variables. Moreover, it is recommended to examine the proposed model for the spatial time series where the data are collected over different geographies and that is taken into account when analyzing the data.

Furthermore, the hurdle Poisson model is studied in this dissertation, where the count part of the hurdle is modeled using the truncated Poisson distribution. It is, however, recommended to study the hurdle model when the count portion is modeled using the negative binomial distribution as well. The developed hurdle autoregressive model has a current π_t that is assumed to be dependent on the prior π_{t-1} as well as the prior observation, and a current λ_t that is assumed to be dependent on the prior λ_{t-1} as well as the prior observation. It is of interest to extend this model to

$$\pi_t = \frac{exp(\alpha + \beta Y_{t-1})}{1 + exp(\alpha + \beta Y_{t-1})},$$

$$(\lambda_t | Y_t > 0) = exp(\delta + \theta\ ln(Y_{t-1} - \lambda_{t-1}) + \psi Y_{t-1}),$$

where the current π_t is assumed to be only dependent on the past observation as it is recommended to use a simpler model for the probability parameter, and the current λ_t is assumed to be dependent on the difference between the past λ_{t-1} and the past observation as well as the past observation.

In terms of predictors, in this study the model does not contain predictors, however the model should be able to handle predictors. It is recommended in the future to consider adding some covariates and perform hypothesis testing. Also, in this study the first order autoregressive model is examined and further extension of this model is to examine the autoregressive process of higher order. Finally, some criteria for evaluation of the performance are computed for the model. It would be useful to establish some tools to evaluate the structure of the correlation similar to the ACF and PACF for the traditional AR process.

REFERENCES

Agresti, A. (2007). *An introduction to categorical data analysis* (2nd ed.). Hoboken, NJ: Wiley-Interscience.

Al-Osh, M. A., and Alzaid, A. A. (1987). First-order integer-valued autoregressive (inar(1)) process. *Journal of Time Series Analysis*, *8*(3), 261-275.

Cameron, A. C., and Trivedi, P. K. (1998). *Regression analysis of count data* (Vol. no. 30.). Cambridge, UK;New York, NY, USA;: Cambridge University Press.

Chatfield, C. (2003). *The analysis of time series: an introduction*. CRC press.

Christou, V. (2013). *Statistical theory for mixed poisson time series models* (Doctoral dissertation). University of Cyprus.

Cui, Y. (2009). *Integer-valued time series and renewal processes* (Doctoral dissertation). Clemson University.

Davis, R. A., Dunsmuir, W. T. M., and Streett, S. B. (2003). Observation-driven models for poisson counts. *Biometrika*, *90*(4), 777-790.

Davis, R. A., and Yau, C. Y. (2011). Comments on pairwise likelihood in time series models. *Statistica Sinica*, *21*(1), 255-277.

Delucchi, K. L., and Bostrom, A. (2004). Methods for analysis of skewed data distributions in psychiatric clinical studies: working with many zero values. *American Journal of Psychiatry*, *161*(7), 1159–1168.

Drost, F. C., Van den Akker, R., and Werker, B. J. (2008a). Note on integer-valued bilinear time series models. *Statistics and probability letters*, *78*(8), 992–996.

Drost, F. C., Van den Akker, R., and Werker, B. J. (2008b). Efficient estimation of auto-regression parameters and innovation distributions for semiparametric integer-valued ar (p) models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *71*(2), 467–485.

Ferland, R., Latour, A., and Oraichi, D. (2006). Integer-valued garch process. *Journal of Time Series Analysis*, *27*(6), 923–942.

Fokianos, K., Rahbek, A., and Tjøstheim, D. (2009). Poisson autoregression. *Journal of the American Statistical Association*, *104*(488), 1430–1439.

Jang, T. Y. (2005). Count data models for trip generation. *Journal of Transportation Engineering*, *131*(6), 444–450.

Jazi, M. A., Jones, G., and Lai, C.-D. (2012). First-order integer valued ar processes with zero inflated poisson innovations. *Journal of Time Series Analysis*, *33*(6), 954–963.

Jung, R. C., Liesenfeld, R., and Richard, J.-F. (2011). Dynamic factor models for multivariate count data: An application to stock-market trading activity. *Journal of Business and Economic Statistics*, *29*(1), 73–85.

Kedem, B., and Fokianos, K. (2002). *Regression models for time series analysis*. Hoboken, NJ: John Wiley Sons, Inc.

King, G. (1989). Event count models for international relations: Generalizations and applications. *International Studies Quarterly*, *33*(2), 123–147.

Lambert, D. (1992). Zero-inflated poisson regression, with an application to defects in manufacturing. *Technometrics*, *34*(1), 1–14.

Liboschik, T., Fokianos, K., and Fried, R. (2015). tscount: An r package for analysis of count time series following generalized linear models.

Liu, H. (2012). *Some models for time series of counts* (Doctoral dissertation). Columbia University.

Maiti, R., Biswas, A., Guha, A., and Ong, S. H. (2014). Modelling and coherent forecasting of zero-inflated count time series. *Statistical Modelling*, *14*(5), 375–398.

Martin, T. G., Wintle, B. A., Rhodes, J. R., Kuhnert, P. M., Field, S. A., Low-Choy, S. J., ... Possingham, H. P. (2005). Zero tolerance ecology: improving ecological inference by modelling the source of zero observations. *Ecology letters*, *8*(11), 1235–1246.

McCullagh, P., and Nelder, J. A. (1989). *Generalized linear models* (2nd ed.). New York;London;: Chapman and Hall.

McKenzie, E. (1985). Some simple models for discrete variate time series. *JAWRA Journal of the American Water Resources Association*, *21*(4), 645–650.

McKenzie, E. (1988). Some arma models for dependent sequences of poisson counts. *Advances in Applied Probability*, *20*(4), 822-835.

McKenzie, E. (2003). Ch. 16. discrete variate time series. *Handbook of statistics*, *21*, 573–606.

Miller, J. M. (2007). *Comparing poisson, hurdle, and zip model fit under varying degrees of skew and zero-inflation* (Doctoral dissertation). University of Florida.

Min, Y., and Agresti, A. (2005). Random effect models for repeated measures of zero-inflated count data. *Statistical Modelling*, *5*(1), 1–19.

Mullahy, J. (1986). Specification and testing of some modified count data models. *Journal of econometrics*, *33*(3), 341–365.

Neelon, B., Ghosh, P., and Loebs, P. F. (2013). A spatial poisson hurdle model for exploring geographic variation in emergency department visits. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *176*(2), 389–413.

Nelson, K. P., and Leroux, B. G. (2006). Statistical models for autocorrelated count data. *Statistics in Medicine*, *25*(8), 1413–1430.

Thomas, S. J. (2010). *Model-based clustering for multivariate time series of counts* (Doctoral dissertation). Rice University.

Thombs, L. A. (1986). *Bootstrap prediction intervals for autoregressive processes* (Doctoral dissertation). S. M. U.

Tjøstheim, D. (2012a). Rejoinder on: Some recent theory for autoregressive count time series. *Test*, *21*(3), 469-476.

Tjøstheim, D. (2012b). Some recent theory for autoregressive count time series. *Test*, *21*(3), 413–438.

Tooze, J. A., Grunwald, G. K., and Jones, R. H. (2002). Analysis of repeated measures data with clumping at zero. *Statistical methods in medical research*, *11*(4), 341–355.

Van den Broek, J. (1995). A score test for zero inflation in a poisson distribution. *Biometrics*, *51*(2), 738-743.

Ver Hoef, J. M., and Jansen, J. K. (2007). Space time zero inflated count models of harbor seals. *Environmetrics*, *18*(7), 697–712.

Wang, P. (2001). Markov zero-inflated poisson regression models for a time series of counts with excess zeros. *Journal of Applied Statistics*, *28*(5), 623–632.

Wang, Y. (2002). *Modeling time series of count data* (Doctoral dissertation). Colorado State University,.

Weiler, H. (1964). A significance test for simultaneous quanta1 and quantitative responses. *Technometrics*, *6*(3), 273–285.

Weiß, C. H. (2008). Thinning operations for modeling time series of counts a survey. *AStA Advances in Statistical Analysis*, *92*(3), 319–341.

Welsh, A. H., Cunningham, R. B., Donnelly, C., and Lindenmayer, D. B. (1996). Modelling the abundance of rare species: statistical models for counts with extra zeros. *Ecological Modelling*, *88*(1), 297–308.

Xiao-Hua, Z., and Tu, W. (1999). Comparison of several independent population means when their samples contain log-normal and possibly zero observations. *Biometrics*, *55*(2), 645–651.

Yang, M. (2012). *Statistical models for count time series with excess zeros* (Doctoral dissertation). University of Iowa.

Yang, M., Zamba, G. K., and Cavanaugh, J. E. (2013). Markov regression models for count time series with excess zeros: A partial likelihood approach. *Statistical Methodology*, *14*, 26–38.

Yau, K. K., Lee, A. H., and Carrivick, P. J. (2004). Modeling zero-inflated count series with application to occupational health. *Computer Methods and Programs in Biomedicine*, *74*(1), 47–52.

Zeger, S. L., and Qaqish, B. (1988). Markov regression models for time series: A quasi-likelihood approach. *Biometrics*, *44*(4), 1019-1031.

Zhu, F. (2012). Zero-inflated poisson and negative binomial integer-valued garch models. *Journal of Statistical Planning and Inference*, *142*(4), 826–839.

Zorn, C. J. (1996). Evaluating zero-inflated and hurdle poisson specifications. *Midwest Political Science Association*, *18*(20), 1–16.

APPENDIX A

R CODE FOR SIMULATIONS

```
####################
####  Packages ###
####################
install.packages('splines')
install.packages('ZIM')
install.packages('stats')
install.packages('MASS')

library(MASS)
library(splines)
library(ZIM)
library(stats)

##############################

####       HURDLE AR      ####

##############################

##############################
####Parameter settings####
##############################

Theta_all=matrix(c(.47,.42,.52,
-.8,-.85,-.75,
-.08,-.13,-.03,
-.2,-.25,-.15,
```

```
.06,.01,.11,
.13,.08,.18
),nrow=3)
Pschem = 1 # 1,2,3
Theta = Theta_all[Pschem,]
#######################################
###Constraints on the parameters###
#######################################
uilinear=matrix(0,nrow=8,ncol=6)
 #matrix for the linear constraints
uilinear[1,2]=1
uilinear[2,2]=-1
uilinear[3,5]=1
uilinear[4,5]=-1
uilinear[5,6]=1
uilinear[6,6]=-1
uilinear[7,5]=1
uilinear[7,6]=1
uilinear[8,5]=-1
uilinear[8,6]=-1
cilinear=rep(-.999,8)
#constant vector for the linear constraints
uilinear %*% Theta - cilinear

#######################################
#####Generate first observation#####
#######################################
```

```
n=125 # 25,70,125,250,600
m=25
lambda1=.1
pi1=.05
rtpois<-function(num,lambda)
qpois(runif(num,ppois(0,lambda),1),lambda)
certain_zero = rbinom(1,1,pi1)
data1= ifelse(certain_zero,0,rtpois(1,lambda1))
data1

#################################
#####Generate the data########
#################################
hurdle.ts=function(Theta,n)
{

data=rep(NA,n)
lambda=rep(NA,n)
pi=rep(NA,n)
data[1]=data1
pi[1]=pi1
lambda[1]=lambda1

for(i in 2:n)
{
certain_zero = rbinom(1,1,pi[i-1])
data[i]= ifelse(certain_zero,0,rtpois(1,lambda[i-1]))
```

```
G=log(pi[i-1]/(1-pi[i-1]))
B=exp(Theta[1]+Theta[2]*G+Theta[3]*data[i-1])
pi[i]=B/(1+B)
lambda[i]=exp(Theta[4]+
Theta[5]*log(lambda[i-1])+
Theta[6]*data[i-1])
}
# return(cbind(data,lambda,pi))
output=list(data=data,lambda=lambda,pi=pi)
output
}
output=hurdle.ts(Theta,n)
data=output$data
lambda=output$lambda
pi=output$pi
data
lambda
pi
range(pi)
range(lambda)
mean(pi)

###Divide the data into ###
### estimation and prediction###

data_est=data[1:100]
pi_est=pi[1:100]
```

```
lambda_est=lambda[1:100]
data_pred=data[101:125]
pi_pred=pi[101:125]
lambda_pred=lambda[101:125]

######## HAR(1) MODEL ########
#####################################
#####################################
#### Log-Likelihood Function###
#####################################
A=rep(NA,length(data_est))

for(i in 1:length(data_est))
{
A[i]= if (data_est[i]==0) 1 else 0
}
A
loglikfng<- function
(Theta,data_est,pi1,lambda1,A){
loglik=rep(0,length(data_est))
pi_est=rep(NA,length(data_est))
lambda_est=rep(NA,length(data_est))
pi_est[1]=pi1
lambda_est[1]= lambda1

for (i in 2:length(data_est))
{
```

```
G=log(pi_est[i-1]/(1-pi_est[i-1]))
B=exp(Theta[1]+
Theta[2]*G+Theta[3]*data_est[i-1])
pi_est[i]=B/(1+B)
lambda_est[i]=exp(Theta[4]+
Theta[5]*log(lambda_est[i-1])+
Theta[6]*data_est[i-1])
loglik[i]= A[i]*log(pi_est[i])+
(1-A[i])*log(1-pi_est[i])+
(1-A[i])*(data_est[i]*log(lambda_est[i])-
log(exp(lambda_est[i])-1))
}
sum(loglik)
}

#loglikfng(Theta,data_est,pi1,lambda1,A)

###############
#####MLE######
###############
results1=constrOptim(theta=c(.3,.2,.4,.33,-.2,.25),
f=loglikfng,grad=NULL,data=data_est,
pi1=pi1, A=A, lambda1=lambda1,
ui=uilinear,ci=cilinear,method="Nelder-Mead",
outer.iterations=100000,outer.eps=1e-9,
control=list(fnscale=-1,reltol= 1e-8))
#########################################
```

```
# Calculation of \hat{\lambda}\hat{pi}#
##########################################
Theta_hat=results1$par
lambda_hat=rep(NA,length(data_est))
pi_hat=rep(NA,length(data_est))
lambda_hat[1]=lambda1
pi_hat[1]=pi1
for (i in 2:length(data_est))
{
G=log(pi_hat[i-1]/(1-pi_hat[i-1]))
B=exp(Theta_hat[1]+Theta_hat[2]*G+
Theta_hat[3]*data_est[i-1])
pi_hat[i]=B/(1+B)
lambda_hat[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_hat[i-1])+
Theta_hat[6]*data_est[i-1])
}
#lambda_hat
#pi_hat



######################
###Standard Errors###
######################
HES<-tryCatch( optimHess(par=results1$par ,
fn=loglikfng ,
data=data_est,pi1=pi1, A=A, lambda1=lambda1),
```

```
error=function(e) e )
if(!inherits(HES, "error")){
SE1=sqrt(diag(-solve(HES)))
Theta_hat=results1$par}
#################################
###Predicted observations####
#################################3
datap<-function(data_pred2)
{
data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
pi_0=pi_hat[length(data_est)]
data_0=data_est[length(data_est)]
lambda_0=lambda_hat[length(data_est)]
G=log(pi_0/(1-pi_0))
B=exp(Theta_hat[1]+
Theta_hat[2]*G+Theta_hat[3]*data_0)
pi_pred2[1]=B/(1+B)
lambda_pred2[1]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_0)+
Theta_hat[6]*data_0)
data_pred2[1]=
if (pi_pred2[1]< mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]
```

```
for (i in 2:length(data_pred))
{
G=log(pi_pred2[i-1]/(1-pi_pred2[i-1]))
B=exp(Theta_hat[1]+
Theta_hat[2]*G+Theta_hat[3]*data_pred2[i-1])
pi_pred2[i]=B/(1+B)
lambda_pred2[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_pred2[i-1])+
Theta_hat[6]*data_pred2[i-1])
data_pred2[i]=
if (pi_pred2[i]< mean(pi)) lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
}
datap(data_pred2)

##############################
###PRMSE using the data###
##############################
PRMSE<-function(data,datapr){
data=c(data_est,data_pred)
datapr=c(data_est,datap(data_pred2))
prmse=rep(NA,5)

for(h in 1:5)
{
```

```
for(i in length(data_est)+h:n)
{
prmse[h]=sqrt(1/(m-h+1)*sum((data-datapr)^2))
}
}
prmse
}

#PRMSE(data,datapr)

############################
######Predicted mean#####
############################
hurdle_mean<-function(data_pred2)
{
mu_hatt=rep(NA,length(data_pred))
data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
pi_0=pi_hat[length(data_est)]
data_0=data_est[length(data_est)]
lambda_0=lambda_hat[length(data_est)]
G=log(pi_0/(1-pi_0))
B=exp(Theta_hat[1]+Theta_hat[2]*G+Theta_hat[3]*data_0)
pi_pred2[1]=B/(1+B)
lambda_pred2[1]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_0)+Theta_hat[6]*data_0)
```

```
data_pred2[1]=
if (pi_pred2[1]< mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]

for (i in 2:length(data_pred))
{
G=log(pi_pred2[i-1]/(1-pi_pred2[i-1]))
B=exp(Theta_hat[1]+Theta_hat[2]*G+
Theta_hat[3]*data_pred2[i-1])
pi_pred2[i]=B/(1+B)
lambda_pred2[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_pred2[i-1])+
Theta_hat[6]*data_pred2[i-1])
data_pred2[i]= if (pi_pred2[i]<
mean(pi)) lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2

for(i in 1:length(data_pred))
{
mu_hatt[i]=(1-pi_pred2[i])*
lambda_pred2[i]/(1-exp(-(lambda_pred2[i])))
}
mu_hatt
}
```

```
#hurdle_mean(data_pred2)
############################
###PRMSE using the mean###
############################
PRMSE_hat<-function(data,mu_hat){
data=c(data_est,data_pred)
mu_hat=c(data_est,hurdle_mean(data_pred2))
prmsem=rep(NA,5)

for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmsem[h]=sqrt(1/(m-h+1)*sum((data-mu_hat)^2))
}
}
prmsem
}

#PRMSE_hat(data,mu_hat)
############

############################

####    ZIP AR MODEL  ####

############################
```

```
############################
ar1 <- bshift(data_est > 0)
results2<-zim(data_est~ar1|ar1)
Theta_hat2=results2$par
se=results2$se
Theta2=c(1.2, 0.6, 0.4,-0.8)
##########################################
# Calculation of \hat{\lambda}\hat{pi}#
##########################################
Theta_hat2=results2$par
lambda_hat2=rep(NA,length(data_est))
pi_hat2=rep(NA,length(data_est))
lambda_hat2[1]=lambda1
pi_hat2[1]=pi1
I1=rep(NA,length(data_est))
for(i in 1:length(data_est))
{
I1[i]= if (data_est[i]==0) 0 else 1
}
I1
for (i in 2:length(data_est))
{
lambda_hat2[i]=exp(Theta_hat2[1]+
Theta_hat2[2]*(I1[i-1]))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I1[i-1]))
pi_hat2[i]=g/(1+g)
```

```
}
lambda_hat2
pi_hat2
##########################
################################
###Predicted observations####
################################
datap2<-function(data_pred2)
{
I=rep(NA,length(data_pred))
I_0=I1[length(data_est)]

data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
lambda_pred2[1]=exp(Theta_hat2[1]+
Theta_hat2[2]*(I_0))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I_0))
pi_pred2[1]=g/(1+g)
data_pred2[1]=
if (pi_pred2[1]< mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
I[1]=if (data_pred2[1]==0) 0 else 1
for (i in 2:length(data_pred))
{
lambda_pred2[i]=
exp(Theta_hat2[1]+Theta_hat2[2]*(I[i-1]))
```

```
g=exp(Theta_hat2[3]+
Theta_hat2[4]*(I[i-1]))
pi_pred2[i]=g/(1+g)
data_pred2[i]=
if (pi_pred2[i]< mean(pi)) lambda_pred2[i] else 0
I[i]= if (data_pred2[i]==0) 0 else 1
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
}
datap2(data_pred2)

##############################
###PRMSE using the data###
##############################
PRMSE2<-function(data,datapr2){
data=c(data_est,data_pred)
datapr=c(data_est,datap2(data_pred2))
prmse=rep(NA,5)

for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmse[h]=sqrt(1/(m-h+1)*sum((data-datapr)^2))
}
}
```

```
prmse
}

PRMSE2(data,datapr)

############################
######Predicted mean#####
############################
ZIPAR_mean<-function(data_pred2)
{


mu_hatt=rep(NA,length(data_pred))
I=rep(NA,length(data_pred))
I_0=I1[length(data_est)]

data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
lambda_pred2[1]=exp(Theta_hat2[1]+
Theta_hat2[2]*(I_0))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I_0))
pi_pred2[1]=g/(1+g)
data_pred2[1]= if (pi_pred2[1]<
mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
I[1]=if (data_pred2[1]==0) 0 else 1
```

```
for (i in 2:length(data_pred))
{
lambda_pred2[i]=
exp(Theta_hat2[1]+Theta_hat2[2]*(I[i-1]))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I[i-1]))
pi_pred2[i]=g/(1+g)
data_pred2[i]=
if (pi_pred2[i]< mean(pi)) lambda_pred2[i] else 0
I[i]= if (data_pred2[i]==0) 0 else 1
data_pred2=round(data_pred2, digits = 0)
}
data_pred2

for(i in 1:length(data_pred))
{
mu_hatt[i]=(1-pi_pred2[i])*lambda_pred2[i]
}
mu_hatt
}
ZIPAR_mean(data_pred2)
##############################
###PRMSE using the mean###
##############################
PRMSE_hat2<-function(data,mu_hat){
data=c(data_est,data_pred)
mu_hat=c(data_est,ZIPAR_mean(data_pred2))
prmsem=rep(NA,5)
```

```
for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmsem[h]=sqrt(1/(m-h+1)*sum((data-mu_hat)^2))
}
}
prmsem
}

PRMSE_hat2(data,mu_hat)
############

##############################

#### ZIP INGARCH MODEL ####

##############################


################################################
###   Get the initial parameters for
###    the model to start EM
################################################
lambd= matrix(0,1,length(data_est))
tau=matrix(0,1,length(data_est))
```

```
true_alpha=c(1,0.3,0.4)
true_pii=0.1
pii=1/2
alpha=true_alpha+0.2*runif(1)-0.1
##################################################
############## Expectation#######################
##################################################
expectation <- function(data_est,alpha,pii)
{

lambd[1]=alpha[1]
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*lambd[i-1]+
alpha[3]*data_est[i-1]
tau[i]= if (data_est[i]==0) pii/(pii+
(1-pii)*exp(-lambd[i])) else
tau[i]=0
}
return(tau)
}
#########################################


loglikfng3<- function(alpha,data_est){
loglik=rep(NA,length(data_est))
loglik[1]=0
```

```
tau<-expectation(data_est,alpha,pii)
lambd=rep(NA,length(data_est))
lambd[1]=alpha[1]
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*
lambd[i-1]+alpha[3]*data_est[i-1]
loglik[i]= tau[i]*log(pii)+
(1-tau[i])*(log(1-pii)+
data_est[i]*log(lambd[i])-lambd[i])
}
return(sum(loglik))
}

##########################################
### Score functions for other parameters ##
##########################################
gr<-function(alpha,data_est)

{
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*lambd[i-1]+
alpha[3]*data_est[i-1]
tau[i]= if (data_est[i]==0) pii/(pii+
(1-pii)*exp(-lambd[i])) else
tau[i]=0
```

```
}
lambd=rep(NA,length(data_est))
lambd[1]=alpha[1]
first=rep(NA,length(data_est))
first[1]=0
second=rep(NA,length(data_est))
second[1]=0
third=rep(NA,length(data_est))
third[1]=0
s1=rep(NA,length(data_est))
s2=rep(NA,length(data_est))
s3=rep(NA,length(data_est))
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*lambd[i-1]+
alpha[3]*data_est[i-1]
first[i]=(1+alpha[2]*first[i-1])
second[i]=(lambd[i-1]+alpha[2]*second[i-1])
third[i]= (data_est[i-1]+alpha[2]*third[i-1])
s1[i]=(1-tau[i])*((data_est[i]/lambd[i])-1)*first[i]
s2[i]=(1-tau[i])*((data_est[i]/lambd[i])-1)*second[i]
s3[i]=(1-tau[i])*((data_est[i]/lambd[i])-1)*third[i]
}
ss1=sum(s1[-1])
ss2=sum(s2[-1])
ss3=sum(s3[-1])
c(ss1,ss2,ss3)
```

```
}

##################################################
######constraints on the parameters#######
##################################################
uilinear3=matrix(0,nrow=3,ncol=3) #matrix for
#the linear constraints
uilinear3[1,1]=1
uilinear3[2,2]=1
uilinear3[3,3]=1
cilinear3=rep(0,3)
#constant vector for the linear constraints
#################################
#####################################
########Maximization##############
#####################################
maximization <- function(data_est,tau,gr){
tau<-expectation(data_est,alpha,pii)
pii_hat= sum(tau)/(length(data_est)-1)

###other parameters###
res=constrOptim(theta=alpha,f=loglikfng3,
grad=gr,data=data_est,ui=uilinear3,
ci=cilinear3,method="BFGS",
outer.iterations=100000,
outer.eps=1e-9,hessian=TRUE,
control=list(fnscale=-1,reltol= 1e-8,maxit=1000))
```

```
result<-list
(matrix(c(res$par,pii_hat),4,1),res$hessian )
result
}

#############
EM <- function(data_est,tau,gr,pii,alpha,maxit=100,tol=.5)
{

flag <- 0


# Iterate between expectation and maximization parts

for(i in 1:maxit){
alpha=true_alpha+0.2*runif(1)-0.1
pii=1/2
newest <-
maximization
(data_est, expectation(data_est,alpha,pii),gr)[1]
new<- unlist(newest)
new_h<-
maximization
(data_est, expectation(data_est,alpha,pii),gr)[2]
alpha1_new <-
new[1]; alpha2_new <- new[2]; alpha3_new <- new[3];
```

```
pii_new <- new[4]
new_step <-
c(alpha1_new,alpha2_new,alpha3_new,pii_new)
initial=c(alpha,pii)
# Stop iteration if the
#difference between the current and
#new estimates is less than a tolerance level
if( all(abs(initial  - new_step) < tol))
{ flag <- 1; break}
###################



}
if(!flag) warning("Didn't converge\n")

list(alpha1_new,alpha2_new,alpha3_new, pii_new ,new_h)


}
results3<-EM(data_est,tau,gr,alpha,maxit=100,tol=.5)

#################
##########################################
# Calculation of \hat{\lambda}#
##########################################
Theta_hat3=unlist(results3[1:4])
lambda_hat3=rep(NA,length(data_est))
lambda_hat3[1]=alpha[1]
```

```
for (i in 2:length(data_est))
{
lambda_hat3[i]=Theta_hat3[1]+
Theta_hat3[2]*lambda_hat3[i-1]+
Theta_hat3[3]*data_est[i-1]
}
lambda_hat3

############################
##### Standard Errors#####
############################
HES3=matrix(unlist(results3[5]),3,3)
SE3<-sqrt(diag(-solve(HES3)))

################################
###Predicted observations####
################################
datap3<-function(data_pred2)
{
data_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
data_0=data_est[length(data_est)]
lambda_0=lambda_hat3[length(data_est)]
lambda_pred2[1]=Theta_hat3[1]+
Theta_hat3[2]*lambda_0+Theta_hat3[3]*data_0
data_pred2[1]=
if (unlist(results3[1:4]))[4]<
```

```
 mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]

for (i in 2:length(data_pred))
{
lambda_pred2[i]=Theta_hat3[1]+
Theta_hat3[2]*lambda_pred2[i-1]+
Theta_hat3[3]*data_pred2[i-1]
data_pred2[i]=
if (unlist(results3[1:4])[4]<
 mean(pi)) lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
}
datap3(data_pred2)

##############################
###PRMSE using the data###
##############################
PRMSE3<-function(data,datapr){
data=c(data_est,data_pred)
datapr=c(data_est,datap3(data_pred2))
prmse=rep(NA,5)

for(h in 1:5)
```

```
{
for(i in length(data_est)+h:n)
{
prmse[h]=sqrt(1/(m-h+1)*sum((data-datapr)^2))
}
}
prmse
}

PRMSE3(data,datapr)

##############################
######Predicted mean#####
##############################
ZIPINGARCH_mean<-function(data_pred2)
{
mu_hatt=rep(NA,length(data_pred))
data_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
data_0=data_est[length(data_est)]
lambda_0=lambda_hat3[length(data_est)]
lambda_pred2[1]=Theta_hat3[1]+
Theta_hat3[2]*lambda_0+Theta_hat3[3]*data_0
data_pred2[1]=
if (unlist(results3[1:4])[4]<
mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
```

```
data_pred2[1]

for (i in 2:length(data_pred))
{
lambda_pred2[i]=Theta_hat3[1]+
Theta_hat3[2]*lambda_pred2[i-1]+
Theta_hat3[3]*data_pred2[i-1]
data_pred2[i]=
if (unlist
(results3[1:4])[4]< mean(pi)) lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
for(i in 1:length(data_pred))
{
mu_hatt[i]=(1-unlist(results3[1:4])[4])*lambda_pred2[i]
}
mu_hatt
}
ZIPINGARCH_mean(data_pred2)
##############################
###PRMSE using the mean###
##############################
PRMSE_hat3<-function(data,mu_hat){
data=c(data_est,data_pred)
mu_hat=c(data_est,ZIPINGARCH_mean(data_pred2))
prmsem=rep(NA,5)
```

```
for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmsem[h]=sqrt(1/(m-h+1)*sum((data-mu_hat)^2))
}
}
prmsem
}

PRMSE_hat3(data,mu_hat)
############
#################
###Simulation###
#################
Pschem = 1 # parameter setting, 1 to 3
n=125 #n=25,70,125,250,600
rep=1000 #number of replications
out1 = matrix(NA,rep,6)
out2 = matrix(NA,rep,6)
out3=matrix(NA,rep,5)
out4=matrix(NA,rep,5)
out5=matrix(NA,rep,4)
out6=matrix(NA,rep,4)
out7=matrix(NA,rep,5)
out8=matrix(NA,rep,5)
```

```
out9=matrix(NA,rep,4)
out10=matrix(NA,rep,3)
out11=matrix(NA,rep,5)
out12=matrix(NA,rep,5)
out13=matrix(NA,rep,length(data_pred))
out14=matrix(NA,rep,length(data_pred))
out15=matrix(NA,rep,length(data_pred))
out16=matrix(NA,rep,length(data_pred))
out17=matrix(NA,rep,length(data_pred))
out18=matrix(NA,rep,length(data_pred))
out19=matrix(NA,rep,length(data_pred))


Theta = Theta_all[Pschem,]
strt<-Sys.time()
for (r in 1:rep){

#compute some info to be used in optimization
lambda1
pi1
rtpois<-function(num,lambda)
qpois(runif(num,ppois(0,lambda),1),lambda)
certain_zero = rbinom(1,1,pi1)
data1= ifelse(certain_zero,0,rtpois(1,lambda1))
output=hurdle.ts(Theta,n)
data=output$data
lambda=output$lambda
```

```
pi=output$pi
####Divide the data to estimation and prediction####
data_est=data[1:100]
pi_est=pi[1:100]
lambda_est=lambda[1:100]
data_pred=data[101:125]
pi_pred=pi[101:125]
lambda_pred=lambda[101:125]
###Fit HAR(1)model###
A=rep(NA,length(data_est))

for(i in 1:length(data_est))
{
A[i]= if (data_est[i]==0) 1 else 0
}
A
loglikfng(Theta,data_est,pi1,lambda1,A)
results1=constrOptim(theta=c(.3,.2,.4,.33,-.2,.25),
f=loglikfng,grad=NULL,data=data_est,
pi1=pi1, A=A, lambda1=lambda1,
ui=uilinear,ci=cilinear,method="Nelder-Mead",
outer.iterations=100000,outer.eps=1e-9,
control=list(fnscale=-1,reltol= 1e-8))
### \hat{\lambda}\hat{pi}###
Theta_hat=results1$par
lambda_hat=rep(NA,length(data_est))
pi_hat=rep(NA,length(data_est))
```

```
lambda_hat[1]=lambda1
pi_hat[1]=pi1
for (i in 2:length(data_est))
{
G=log(pi_hat[i-1]/(1-pi_hat[i-1]))
B=exp(Theta_hat[1]+Theta_hat[2]*G+
Theta_hat[3]*data_est[i-1])
pi_hat[i]=B/(1+B)
lambda_hat[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_hat[i-1])+
Theta_hat[6]*data_est[i-1])
}

###Standard Errors###
#######################
HES<-tryCatch( optimHess(par=results1$par ,
fn=loglikfng ,
data=data_est,pi1=pi1, A=A, lambda1=lambda1),
error=function(e) e )
if(!inherits(HES, "error")){
out1[r,]=sqrt(diag(-solve(HES)))
out2[r,]=Theta_hat}
#########################
out13[r,]<-data_pred
###Predicted observations####
out14[r,]<-datap(data_pred2)
###PRMSE using the data###
```

```
out3[r,]<-PRMSE(data,datapr)
######Predicted mean#####
out15[r,]<-hurdle_mean(data_pred2)
###PRMSE using the mean###
out4[r,]<-PRMSE_hat(data,mu_hat)
######Fit ZIP AR MODEL#######
ar1 <- bshift(data_est > 0)
results2<-tryCatch
(zim(data_est~ar1|ar1),error=function(e) e )
if(!inherits(results2, "error")){
Theta_hat2=results2$par
out5[r,]=results2$par
out6[r,]=results2$se
### Calculation of \hat{\lambda}\hat{pi}###
Theta_hat2=results2$par
lambda_hat2=rep(NA,length(data_est))
pi_hat2=rep(NA,length(data_est))
lambda_hat2[1]=lambda1
pi_hat2[1]=pi1
I1=rep(NA,length(data_est))
for(i in 1:length(data_est))
{
I1[i]= if (data_est[i]==0) 0 else 1
}
I1
for (i in 2:length(data_est))
{
```

```
lambda_hat2[i]=exp(Theta_hat2[1]+Theta_hat2[2]*(I1[i-1]))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I1[i-1]))
pi_hat2[i]=g/(1+g)
}

###Predicted observations####
out16[r,]<-datap2(data_pred2)
###PRMSE using the data###
out7[r,]<-PRMSE2(data,datapr2)
######Predicted mean#####
out17[r,]<-ZIPAR_mean(data_pred2)
###PRMSE using the mean###
out8[r,]<-PRMSE_hat2(data,mu_hat)
}
#######Fit ZIP INGARCH MODEL######
###get the initial parameters for the model to start EM###
lambd= matrix(0,1,length(data_est))
tau=matrix(0,1,length(data_est))
true_alpha=c(1,0.3,0.4)
true_pii=0.1
pii=1/2
alpha=true_alpha+0.2*runif(1)-0.1
############## Expectation###################
expectation(data_est,alpha,pii)
loglikfng3(alpha,data_est)
gr(alpha,data_est)
########Maximization#############
```

```
maximization(data_est,tau,gr)
##EM##

results3<-tryCatch
(EM(data_est,tau,gr,alpha,maxit=100,tol=.5),
error=function(e) e )
if(!inherits(results3, "error")){
Theta_hat3=unlist(results3[1:4])
out9[r,]= Theta_hat3
# Calculation of \hat{\lambda}#
Theta_hat3=unlist(results3[1:4])
lambda_hat3=rep(NA,length(data_est))
lambda_hat3[1]=alpha[1]
for (i in 2:length(data_est))
{
lambda_hat3[i]=Theta_hat3[1]+Theta_hat3[2]*
lambda_hat3[i-1]+Theta_hat3[3]*data_est[i-1]
}
##### Standard Errors#####

HES3=matrix(unlist(results3[5]),3,3)
out10[r,]<-sqrt(diag(-solve(HES3)))}

###Predicted observations####
out18[r,]<-datap3(data_pred2)
###PRMSE using the data###
#PRMSE3(data,datapr)
```

```
out11[r,]<-PRMSE3(data,datapr)
######Predicted mean#####
out19[r,]<-ZIPINGARCH_mean(data_pred2)
###PRMSE using the mean###
#PRMSE_hat3(data,mu_hat)
out12[r,]<-PRMSE_hat3(data,mu_hat)
}

print(Sys.time()-strt)
print(c("n =", n))
print(c("Pschem=",Pschem))
print(c("rep=",rep))

print("SE1")
out1=out1[complete.cases(out1),]
nrow(out1)
print("SE1")
colMeans(out1)
print("Theta Hat 1")
out2=out2[complete.cases(out2),]
nrow(out2)
print("Theta Hat 1")
colMeans(out2)
MADEH1<- sweep(out2,2,Theta)
MADEH1abs<-abs(MADEH1)
MADEH1sum<- colSums(MADEH1abs)/rep
print("MADE")
```

MADEH1sum

```
rel.bH1<- sweep(out2,2,Theta)
rel.bH1d<-sweep(rel.bH1,2,Theta,'/')
rel.bH1sum<- colSums(rel.bH1d)/rep
print("Relative.bias")
rel.bH1sum

print("PRMSE")
out3=out3[complete.cases(out3),]
colMeans(out3)
print("PRMSE Hat")
out4=out4[complete.cases(out4),]
colMeans(out4)
print("Theta Hat 2")
out5=out5[complete.cases(out5),]
nrow(out5)
print("Theta Hat 2")
colMeans(out5)

print("SE2")
out6=out6[complete.cases(out6),]
nrow(out6)
print("SE2")
colMeans(out6)
print("PRMSE 2")
out7=out7[complete.cases(out7),]
```

```
colMeans(out7)
print("PRMSE Hat 2")
out8=out8[complete.cases(out8),]
colMeans(out8)
print("Theta Hat 3")
out9=out9[complete.cases(out9),]
nrow(out9)
print("Theta Hat 3")
colMeans(out9)
print("SE 3")
out10=out10[complete.cases(out10),]
nrow(out10)
print("SE 3")
colMeans(out10)
print("PRMSE 3")
out11=out11[complete.cases(out11),]
colMeans(out11)
print("PRMSE Hat 3")
out12=out12[complete.cases(out12),]
colMeans(out12)
print("Data_Pred")
nrow(out13)
out13[100,]
out13[150,]
out13[200,]
out13[400,]
out13[450,]
```

```
out13[550,]
out13[800,]
print("Data_P")
out14=out14[complete.cases(out14),]
out14[100,]
out14[150,]
out14[200,]
out14[400,]
out14[450,]
out14[550,]
out14[800,]
print("Hurdle_Mean")
out15=out15[complete.cases(out15),]
out15[100,]
out15[150,]
out15[200,]
out15[400,]
out15[450,]
out15[550,]
out15[800,]
print("Data_P  2")
out16=out16[complete.cases(out16),]
out16[100,]
out16[150,]
out16[200,]
out16[400,]
out16[450,]
```

```
out16[550,]
out16[800,]
print("ZIPAR_Mean")
out17=out17[complete.cases(out17),]
out17[100,]
out17[150,]
out17[200,]
out17[400,]
out17[450,]
out17[550,]
out17[800,]
print("Data_P  3")
out18=out18[complete.cases(out18),]
out18[100,]
out18[150,]
out18[200,]
out18[400,]
out18[450,]
out18[550,]
out18[800,]
print("ZIPINGARCH_Mean")
out19=out19[complete.cases(out19),]
out19[100,]
out19[150,]
out19[200,]
out19[400,]
out19[450,]
```

```
out19[550,]
out19[800,]
```

APPENDIX B

R CODE FOR THE APPLICATION

```
Theta=c(-.33,.2,.25,.3,-.2,-.4)
data=as.vector(mydata$Arson)
lambda=rep(NA,length(data))
pi=rep(NA,length(data))
loglik=rep(NA,length(data))
lambda1=.1
pi1=.05
pi[1]=pi1
lambda[1]=lambda1
##################################
###Obtaining pi and lambda###
##################################
for (i in 2:length(data))
{
G=log(pi[i-1]/(1-pi[i-1]))
B=exp(Theta[1]+Theta[2]*G+Theta[3]*data[i-1])
pi[i]=B/(1+B)
lambda[i]=exp(Theta[4]+
Theta[5]*log(lambda[i-1])+Theta[6]*data[i-1])
}
#################################################################
###Dividing the Data to Estimation and Prediction###
#################################################################
data_est=data[1:110]
pi_est=pi[1:110]
lambda_est=lambda[1:110]
data_pred=data[111:144]
```

```
pi_pred=pi[111:144]
lambda_pred=lambda[111:144]
#####################################
###Defining Indicator Variable###
#####################################
A=rep(NA,length(data_est))

for(i in 1:length(data_est))
{
A[i]= if (data_est[i]==0) 1 else 0
}
A
#################################
#### Log-Likelihood Function###
#################################

loglikfng<- function(Theta,data_est,pi1,lambda1,A){
loglik=rep(0,length(data_est))
pi_est=rep(NA,length(data_est))
lambda_est=rep(NA,length(data_est))
pi_est[1]=pi1
lambda_est[1]= lambda1

for (i in 2:length(data_est))
{
G=log(pi_est[i-1]/(1-pi_est[i-1]))
B=exp(Theta[1]+Theta[2]*G+Theta[3]*data_est[i-1])
```

```
pi_est[i]=B/(1+B)
lambda_est[i]=exp(Theta[4]+Theta[5]*
log(lambda_est[i-1])+Theta[6]*data_est[i-1])
loglik[i]= A[i]*log(pi_est[i])+(1-A[i])*log(1-pi_est[i])+
(1-A[i])*(data_est[i]*log(lambda_est[i])-
log(exp(lambda_est[i])-1))
}
sum(loglik)
}
################################################
## Constrains to obtain the MLE        ##
##  -1<Theta[2]<1, -1<Theta[5]<1,      ##
##-1<Theta[6]<1,-1<Theta[5]+Theta[6]<1##
################################################
uilinear=matrix(0,nrow=8,ncol=6)
#matrix for the linear constraints
uilinear[1,2]=1
uilinear[2,2]=-1
uilinear[3,5]=1
uilinear[4,5]=-1
uilinear[5,6]=1
uilinear[6,6]=-1
uilinear[7,5]=1
uilinear[7,6]=1
uilinear[8,5]=-1
uilinear[8,6]=-1
cilinear=rep(-.999,8)
```

```
#constant vector for the linear constraints
################################
#####################################
#####MLE of Hurdle Model######
#################################
results1=constrOptim
(theta=Theta, f=loglikfng, grad=NULL, data=data_est,
pi1=pi1, A=A, lambda1=lambda1,
ui=uilinear, ci=cilinear, method="Nelder-Mead",
outer.iterations=100000, outer.eps=1e-9,
control=list(fnscale=-1, reltol= 1e-8))
##############################################
# Calculation of \hat{\lambda}\hat{pi}#
############################################
Theta_hat=results1$par
lambda_hat=rep(NA, length(data_est))
pi_hat=rep(NA, length(data_est))
lambda_hat[1]=lambda1
pi_hat[1]=pi1
for (i in 2:length(data_est))
{
G=log(pi_hat[i-1]/(1-pi_hat[i-1]))
B=exp(Theta_hat[1]+Theta_hat[2]*G+
Theta_hat[3]*data_est[i-1])
pi_hat[i]=B/(1+B)
lambda_hat[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_hat[i-1])+
```

```
Theta_hat[6]*data_est[i-1])
}
########################
###Standard Errors###
########################
HES= optimHess(par=results1$par ,
fn=loglikfng ,data=data_est ,pi1=pi1 ,
A=A, lambda1=lambda1 )
SE1<-sqrt(diag(-solve(HES)))
##################################
###Predicted observations####
##################################3
datap<-function(data_pred2)
{
data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
pi_0=pi_hat[length(data_est)]
data_0=data_est[length(data_est)]
lambda_0=lambda_hat[length(data_est)]
G=log(pi_0/(1-pi_0))
B=exp(Theta_hat[1]+Theta_hat[2]*G+Theta_hat[3]*data_0)
pi_pred2[1]=B/(1+B)
lambda_pred2[1]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_0)+
Theta_hat[6]*data_0)
data_pred2[1]= if (pi_pred2[1]< mean(pi))
```

```
lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]

for (i in 2:length(data_pred))
{
G=log(pi_pred2[i-1]/(1-pi_pred2[i-1]))
B=exp(Theta_hat[1]+Theta_hat[2]*G+
Theta_hat[3]*data_pred2[i-1])
pi_pred2[i]=B/(1+B)
lambda_pred2[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_pred2[i-1])+
Theta_hat[6]*data_pred2[i-1])
data_pred2[i]= if (pi_pred2[i]< mean(pi))
 lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
}
############################
###PRMSE using the data###
############################
PRMSE<-function(data,datapr){
data=c(data_est,data_pred)
datapr=c(data_est,datap(data_pred2))
prmse=rep(NA,5)
```

```
for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmse[h]=sqrt(1/(34-h+1)*sum((data-datapr)^2))
}
}
prmse
}
##############################
######Predicted mean#####
##############################
hurdle_mean<-function(data_pred2)
{
mu_hatt=rep(NA,length(data_pred))
data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
pi_0=pi_hat[length(data_est)]
data_0=data_est[length(data_est)]
lambda_0=lambda_hat[length(data_est)]
G=log(pi_0/(1-pi_0))
B=exp(Theta_hat[1]+Theta_hat[2]*G+Theta_hat[3]*data_0)
pi_pred2[1]=B/(1+B)
lambda_pred2[1]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_0)+Theta_hat[6]*data_0)
data_pred2[1]= if (pi_pred2[1]< mean(pi))
```

```
 lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]

for (i in 2:length(data_pred))
{
G=log(pi_pred2[i-1]/(1-pi_pred2[i-1]))
B=exp(Theta_hat[1]+Theta_hat[2]*G+
Theta_hat[3]*data_pred2[i-1])
pi_pred2[i]=B/(1+B)
lambda_pred2[i]=exp(Theta_hat[4]+
Theta_hat[5]*log(lambda_pred2[i-1])+
Theta_hat[6]*data_pred2[i-1])
data_pred2[i]= if (pi_pred2[i]<
mean(pi)) lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2

for(i in 1:length(data_pred))
{
mu_hatt[i]=(1-pi_pred2[i])*
lambda_pred2[i]/(1-exp(-(lambda_pred2[i])))
}
mu_hatt
}
##############################
```

```
###PRMSE using the mean###
############################
PRMSE_hat<-function(data,mu_hat){
data=c(data_est,data_pred)
mu_hat=c(data_est,hurdle_mean(data_pred2))
prmsem=rep(NA,5)

for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmsem[h]=sqrt(1/(34-h+1)*sum((data-mu_hat)^2))
}
}
prmsem
}
############################
######ZIP AR MODEL#######
############################
library (ZIM)
library(splines)
ar1 <- bshift(data_est > 0)
results2=zim(data_est~ar1|ar1)
Theta2=c(1.2, 0.6, 0.4,-0.8)
#########################################
# Calculation of \hat{\lambda}\hat{pi}#
#########################################
```

```
Theta_hat2=results2$par
lambda_hat2=rep(NA,length(data_est))
pi_hat2=rep(NA,length(data_est))
lambda_hat2[1]=lambda1
pi_hat2[1]=pi1
I1=rep(NA,length(data_est))
for(i in 1:length(data_est))
{
I1[i]= if (data_est[i]==0) 0 else 1
}
I1
for (i in 2:length(data_est))
{
lambda_hat2[i]=exp(Theta_hat2[1]+Theta_hat2[2]*(I1[i-1]))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I1[i-1]))
pi_hat2[i]=g/(1+g)
}
################################
###Predicted observations####
################################
datap2<-function(data_pred2)
{
I=rep(NA,length(data_pred))
I_0=I1[length(data_est)]

data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
```

```
lambda_pred2=rep(NA,length(data_pred))
lambda_pred2[1]=exp(Theta_hat2[1]+Theta_hat2[2]*(I_0))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I_0))
pi_pred2[1]=g/(1+g)
data_pred2[1]= if (pi_pred2[1]<
mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
I[1]=if (data_pred2[1]==0) 0 else 1
for (i in 2:length(data_pred))
{
lambda_pred2[i]=exp(Theta_hat2[1]+Theta_hat2[2]*(I[i-1]))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I[i-1]))
pi_pred2[i]=g/(1+g)
data_pred2[i]= if (pi_pred2[i]<
mean(pi)) lambda_pred2[i] else 0
I[i]= if (data_pred2[i]==0) 0 else 1
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
}
##############################
###PRMSE using the data###
##############################
PRMSE2<-function(data,datapr2){
data=c(data_est,data_pred)
datapr=c(data_est,datap2(data_pred2))
prmse=rep(NA,5)
```

```
for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmse[h]=sqrt(1/(34-h+1)*sum((data-datapr)^2))
}
}
prmse
}
############################
######Predicted mean#####
############################
ZIPAR_mean<-function(data_pred2)
{


mu_hatt=rep(NA,length(data_pred))
I=rep(NA,length(data_pred))
I_0=I1[length(data_est)]

data_pred2=rep(NA,length(data_pred))
pi_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
lambda_pred2[1]=exp(Theta_hat2[1]+Theta_hat2[2]*(I_0))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I_0))
pi_pred2[1]=g/(1+g)
```

```
data_pred2[1]= if (pi_pred2[1]< mean(pi))
lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
I[1]=if (data_pred2[1]==0) 0 else 1
for (i in 2:length(data_pred))
{
lambda_pred2[i]=exp(Theta_hat2[1]+Theta_hat2[2]*(I[i-1]))
g=exp(Theta_hat2[3]+Theta_hat2[4]*(I[i-1]))
pi_pred2[i]=g/(1+g)
data_pred2[i]= if (pi_pred2[i]< mean(pi))
lambda_pred2[i] else 0
I[i]= if (data_pred2[i]==0) 0 else 1
data_pred2=round(data_pred2, digits = 0)
}
data_pred2

for(i in 1:length(data_pred))
{
mu_hatt[i]=(1-pi_pred2[i])*lambda_pred2[i]
}
mu_hatt
}
##############################
###PRMSE using the mean###
##############################
PRMSE_hat2<-function(data,mu_hat){
data=c(data_est,data_pred)
```

```
mu_hat=c(data_est,ZIPAR_mean(data_pred2))
prmsem=rep(NA,5)

for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmsem[h]=sqrt(1/(34-h+1)*sum((data-mu_hat)^2))
}
}
prmsem
}
#################################
#######ZIP INGARCH MODEL######
#################################
#####################################################
###get the initial parameters for the model to start EM###
#####################################################
lambd= matrix(0,1,length(data_est))
tau=matrix(0,1,length(data_est))
true_alpha=c(1,0.3,0.4)
true_pii=0.1
pii=1/2
alpha=true_alpha+0.2*runif(1)-0.1
#################################################
############## Expectation###################
#################################################
```

```
expectation <- function(data_est,alpha,pii)
{

lambd[1]=alpha[1]
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*
lambd[i-1]+alpha[3]*data_est[i-1]
tau[i]= if (data_est[i]==0) pii/(pii+
(1-pii)*exp(-lambd[i])) else
tau[i]=0
}
return(tau)
}
###################################
#### Loglikelihood Function####
###################################
loglikfng3<- function(alpha,data_est){
loglik=rep(NA,length(data_est))
loglik[1]=0
tau<-expectation(data_est,alpha,pii)
lambd=rep(NA,length(data_est))
lambd[1]=alpha[1]
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*
lambd[i-1]+alpha[3]*data_est[i-1]
```

```
loglik[i]= tau[i]*log(pii)+
(1-tau[i])*(log(1-pii)+data_est[i]*
log(lambd[i])-lambd[i])
}
return(sum(loglik))
}

#################################################################
#########Score functions for other parameters############
#################################################################
gr<-function(alpha,data_est)

{
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+
alpha[2]*lambd[i-1]+
alpha[3]*data_est[i-1]
tau[i]= if (data_est[i]==0) pii/
(pii+(1-pii)*exp(-lambd[i])) else
tau[i]=0
}
lambd=rep(NA,length(data_est))
lambd[1]=alpha[1]
first=rep(NA,length(data_est))
first[1]=0
second=rep(NA,length(data_est))
```

```
second[1]=0
third=rep(NA,length(data_est))
third[1]=0
s1=rep(NA,length(data_est))
s2=rep(NA,length(data_est))
s3=rep(NA,length(data_est))
for (i in 2:length(data_est))
{
lambd[i]=alpha[1]+alpha[2]*
lambd[i-1]+alpha[3]*data_est[i-1]
first[i]=(1+alpha[2]*first[i-1])
second[i]=(lambd[i-1]+alpha[2]*second[i-1])
third[i]= (data_est[i-1]+alpha[2]*third[i-1])
s1[i]=(1-tau[i])*((data_est[i]/lambd[i])-1)*first[i]
s2[i]=(1-tau[i])*((data_est[i]/lambd[i])-1)*second[i]
s3[i]=(1-tau[i])*((data_est[i]/lambd[i])-1)*third[i]
}
ss1=sum(s1[-1])
ss2=sum(s2[-1])
ss3=sum(s3[-1])
c(ss1,ss2,ss3)
}

##################################################
######constraints on the parameters#######
##################################################
uilinear3=matrix(0,nrow=3,ncol=3)
```

```
#matrix for the linear constraints
uilinear3[1,1]=1
uilinear3[2,2]=1
uilinear3[3,3]=1
cilinear3=rep(0,3)
#constant vector for the linear constraints
##################################
#####################################
########Maximization#############
#####################################
maximization <- function(data_est,tau,gr){
tau<-expectation(data_est,alpha,pii)
pii_hat= sum(tau)/(length(data_est)-1)

###other parameters###
res=constrOptim
(theta=alpha,f=loglikfng3,grad=gr,data=data_est,
ui=uilinear3,ci=cilinear3,method="BFGS",
outer.iterations=100000,outer.eps=1e-9,hessian=TRUE,
control=list(fnscale=-1,reltol= 1e-8,maxit=1000))

result<-list(matrix(c(res$par,pii_hat),4,1),res$hessian )
result
}

############
EM <- function
```

```
(data_est,tau,gr,pii,alpha,maxit=100,tol=.5)
{

flag <- 0


# Iterate between expectation and
 maximization parts

for(i in 1:maxit){
alpha=true_alpha+0.2*runif(1)-0.1
pii=1/2
newest <-maximization
(data_est, expectation(data_est,alpha,pii),gr)[1]
new<- unlist(newest)
new_h<-maximization
(data_est, expectation
(data_est,alpha,pii),gr)[2]
alpha1_new <- new[1]; alpha2_new
 <- new[2]; alpha3_new <- new[3];
pii_new <- new[4]
new_step <- c(alpha1_new,alpha2_new,alpha3_new,pii_new)
initial=c(alpha,pii)
# Stop iteration if the difference
 between the current and
  new estimates is less than
   a tolerance level
```

```
if( all(abs(initial  - new_step) < tol)  )
{ flag <- 1; break}
####################



}
if(!flag) warning("Didn't converge\n")

list(alpha1_new,alpha2_new,alpha3_new, pii_new  ,new_h)


}
results3<-EM(data_est,tau,gr,alpha,maxit=100,tol=.5)
################
#############################################
# Calculation of \hat{\lambda}#
#############################################
Theta_hat3=unlist(results3[1:4])
lambda_hat3=rep(NA,length(data_est))
lambda_hat3[1]=alpha[1]
for (i in 2:length(data_est))
{
lambda_hat3[i]=Theta_hat3[1]+
Theta_hat3[2]*lambda_hat3[i-1]+
Theta_hat3[3]*data_est[i-1]
}
############################
##### Standard Errors#####
```

```
##############################
HES3=matrix(unlist(results3[5]),3,3)
SE3<-sqrt(diag(-solve(HES3)))
SE3
##################################
###Predicted observations####
##################################3
datap3<-function(data_pred2)
{
data_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
data_0=data_est[length(data_est)]
lambda_0=lambda_hat3[length(data_est)]
lambda_pred2[1]=Theta_hat3[1]+
Theta_hat3[2]*lambda_0+Theta_hat3[3]*data_0
data_pred2[1]= if (unlist(results3[1:4])
[4]< mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]

for (i in 2:length(data_pred))
{
lambda_pred2[i]=Theta_hat3[1]+
Theta_hat3[2]*
lambda_pred2[i-1]+
Theta_hat3[3]*data_pred2[i-1]
data_pred2[i]= if
```

```
(unlist(results3[1:4])
[4]< mean(pi)) lambda_pred2[i]
 else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
}
##############################
###PRMSE using the data###
##############################
PRMSE3<-function(data,datapr){
data=c(data_est,data_pred)
datapr=c(data_est,datap3(data_pred2))
prmse=rep(NA,5)

for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmse[h]=sqrt(1/(34-h+1)*sum((data-datapr)^2))
}
}
prmse
}
##############################
######Predicted mean#####
##############################
```

```
ZIPINGARCH_mean<-function(data_pred2)
{
mu_hatt=rep(NA,length(data_pred))
data_pred2=rep(NA,length(data_pred))
lambda_pred2=rep(NA,length(data_pred))
data_0=data_est[length(data_est)]
lambda_0=lambda_hat3[length(data_est)]
lambda_pred2[1]=Theta_hat3[1]+
Theta_hat3[2]*lambda_0+
Theta_hat3[3]*data_0
data_pred2[1]= if (unlist(results3[1:4])
[4]< mean(pi)) lambda_pred2[1] else 0
round(data_pred2[1], digits = 0)
data_pred2[1]

for (i in 2:length(data_pred))
{
lambda_pred2[i]=Theta_hat3[1]+
Theta_hat3[2]* lambda_pred2[i-1]+
Theta_hat3[3]*data_pred2[i-1]
data_pred2[i]= if (unlist(results3[1:4])
[4]< mean(pi)) lambda_pred2[i] else 0
data_pred2=round(data_pred2, digits = 0)
}
data_pred2
for(i in 1:length(data_pred))
{
```

```
mu_hatt[i]=(1-unlist(results3[1:4])[4])*
lambda_pred2[i]
}
mu_hatt
}
##############################
### PRMSE using the mean ###
##############################
PRMSE_hat3<-function(data,mu_hat){
data=c(data_est,data_pred)
mu_hat=c(data_est,ZIPINGARCH_mean(data_pred2))
prmsem=rep(NA,5)

for(h in 1:5)
{
for(i in length(data_est)+h:n)
{
prmsem[h]=sqrt(1/(34-h+1)*sum((data-mu_hat)^2))
}
}
prmsem
}
############
```