House price scenario generation for the Danish mortgagor problem

Snorri Páll Sigurðsson

Kongens Lyngby 2007

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

Summary

A recent article written by Rasmussen and Clausen [13] publishes interesting results for mortgage loan portfolio optimization seen from the perspective of an individual mortgagor, for the Danish mortgage market. The purpose of this thesis is to develop a house price model to extend their results for multistage stochastic programming, by adding the option of selling the real estate as well as re-balancing the bond portfolio.

The purpose of this project is to get acquainted with the economic and econometric methods used for house price modeling, apply the methods to a simple benchmark relation and extend the results to a scenario tree structure. Secondly a more elaborate and economically real model is dissected and reproduced to give a relation able of forecasting house prices, with only a limited number of input variables available. The error of the reduced model is simulated and the resulting model applied to a scenario tree structure.

The final product should then be a scenario tree predicting the expected house price with known variance, using only interest rates and previous house prices as input.

Preface

This thesis was prepared at Informatics Mathematical Modelling, the Technical University of Denmark in partial fulfillment of the requirements for acquiring the degree, Master of Science in Engineering.

The project was carried out in the period from July 15th 2006 to April 15th 2007.

The subject of the thesis is house price prediction for scenario trees from applied macro house price models, limiting explanatory variables.

Lyngby, April 2007

Snorri Páll Sigurðsson

Acknowledgements

I thank my supervisors Kourosh Marjani Rasmussen and Jens Clausen for their help and guidance throughout this project. I would also like to thank Dan Knudsen at The Danish National Bank for taking the time to answer my questions about the MONA house price relation, and Klaus Kaae Andersen for taking the time to advice on the statistical aspects whenever I needed.

Contents

Chapter 1

Introduction

1.1 Background

Over the last half century or so there have been great strides in the advance of optimization and financial theory. Portfolio diversification strategy, combining the two fields, has been used for quite some time by investors in many parts of the financial sector with great success.

Purchasing real estate is one of the biggest financial decision an individual will make during his life. In Denmark there is an elaborate and diverse selection of mortgage loans allowing great flexibility when it comes to the financing of real estate investment.

In a recent article by Rasmussen and Clausen [13] the portfolio optimization technique is applied to the Danish mortgage loan system. The perspective is of a person which is faced with financing a real estate investment and has a diverse selection of mortgage loans available. They find that by creating a portfolio of bonds, instead of the current practise of only one bond, the investor can benefit by re-balance the portfolio at optimal points through to horizon.

1.2 Aim of Thesis

Initially the aim of this thesis was divided into two main parts, that is

1. To get acquainted with both the economics and econometrics of house price estimation and from a real model develop a simplified house price model and apply it to a scenario tree format.
2. To apply the house price trees along with a mortgage loan diversification optimization.

As the work on this thesis evolved part 1 took more time than expected and it was decided to drop part 2. Instead, more care would be taken in explaining and implementing the house price model as a prediction model and the theory behind such models.

It can therefore be said that the aim of this thesis is to deliver a house price scenario tree able to extend the Rasmussen and Clausen model by giving the investor a new option of selling the house, as well as the option of continuing by re-balancing the bond portfolio. This changes their problem since at horizon the objective was to minimize the cost of financing, while when adding the house price scenario tree the objective will be to maximize the profit from selling the house and paying the loans. The integration of house prices in Rasmussen and Clausen remains as further work.

1.3 Outline of Thesis

A flow diagram depicting the progression of the work done for the thesis is shown in Figure 1.1. Two main models were inspected, i.e. the simple Nykredit benchmark model and the MONA house price relation, taken from the Danish National Banks macro model called MONA. The up-down flow in the diagram represents the time line of the project work.

The structure of the thesis is as follows:

Chapter 1: Introduction. The background to the thesis is presented, as well as listing what is to be achieved by the work done and giving an overview of the material chapter by chapter.

Chapter 2: House Price Models. A discussion of house price development from the stand point of economics, showing a well known long term relationship for the development of house prices, the role of demand and supply in determining the price is also discussed. A short discussion of market expectation and real house price development in Denmark is also presented.

Chapter 3: House Price Dynamics I. The Nykredit Relation. The simple house price relationship, i.e. the Nykredit relation, is presented and formulated for a single time line. The definition of a scenario tree is presented. The one dimensional results are extended to a scenario tree structure and the results are investigated.

Chapter 4: Time Series and Econometric Theory. Before moving into more evolved and applied house price models a listing of the basic time series and econometrical definitions and methods are presented. The chapter gives a discussion on the relevant topics providing examples when necessary to demonstrate usability.

Chapter 5: House Price Dynamics II. The MONA Model. The more complicate House price model, adapted from the MONA model, is introduced. Numerous topics regarding the model are discussed such as data handling, theoretical derivation, parameter estimation and prediction capabilities. The chapter ends on a short discussion of the weaknesses of the model and the problem with out-of-sample data.

Chapter 6: Applying The MONA house price relation. Matters regarding aggregation of house price change, how to deal with missing explanatory data in the out-of-sample prediction and the estimation of the prediction error for out-of-sample forecasting.

Chapter 7: House Price Dynamics III Statistical Model. A new model is presented by modeling the data as is, i.e. dropping the long term economic intuition embedded in the MONA presentation. The MONA error-correction model, presented in Chapter 4, is used for the model.

Chapter 8: Validation and Results. The models are compared, first as single path models and later by inputting interest rate scenario trees. Results are analyzed and discussed.

Chapter 9: Conclusion. The conclusion of the modeling is summarized listing the pros and cons of the house price models, as well as a discussion of usability and further work is presented.

Appendix: Programming. The problem of implementing the scenario trees in a programming language is discussed, presenting solutions both in an object orientated language, i.e. C#, as well as a non objective orientated language, such as R and `Matlab`. Finally some examples of scripts showing how to use the numerous R function written for analysis of scenario trees and parameter estimation.

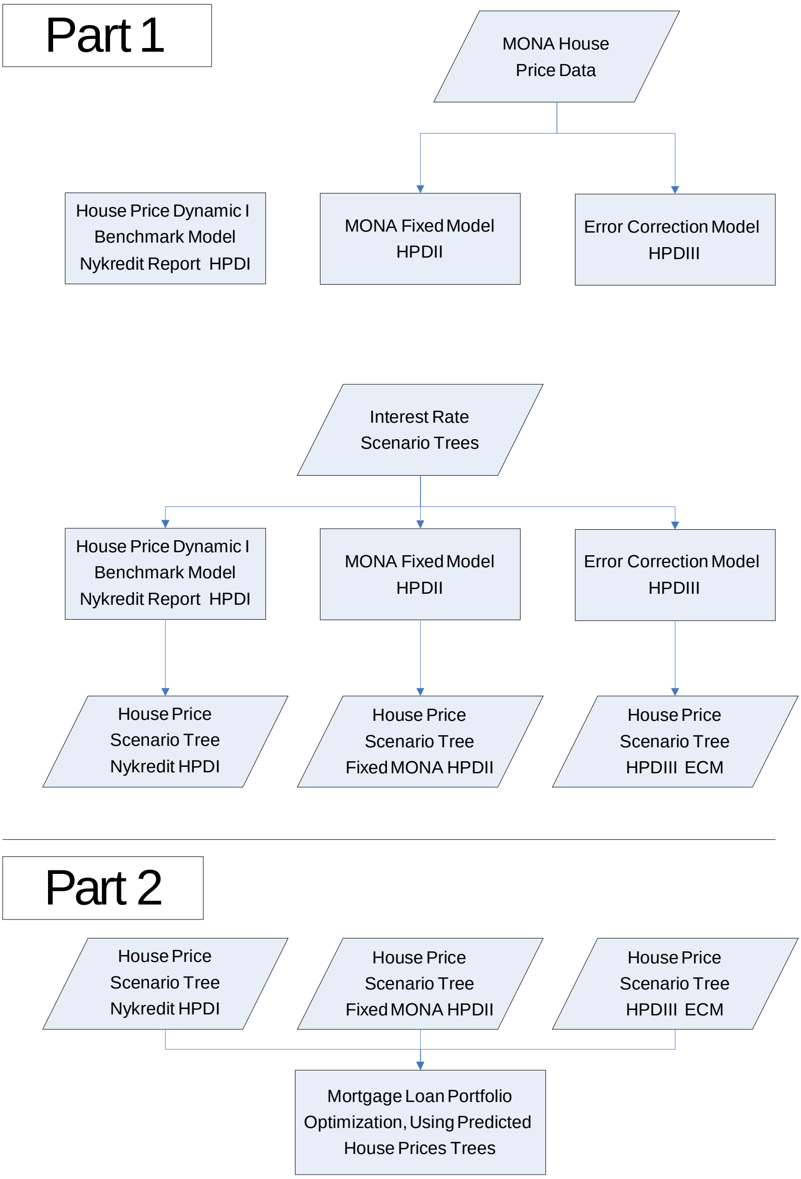
Part 1
MONA House Price Data
House Price Dynamic I Benchmark Model Nykredit Report HPDI
MONA Fixed Model HPDII
Error Correction Model HPDIII
Interest Rate Scenario Trees
House Price Dynamic I Benchmark Model Nykredit Report HPDI
MONA Fixed Model HPDII
Error Correction Model HPDIII
House Price Scenario Tree Nykredit HPDI
House Price Scenario Tree Fixed MONA HPDII
House Price Scenario Tree HPDIII ECM
Part 2
House Price Scenario Tree Nykredit HPDI
House Price Scenario Tree Fixed MONA HPDII
House Price Scenario Tree HPDIII ECM
Mortgage Loan Portfolio Optimization, Using Predicted House Prices Trees

Figure 1.1: An abstract view of the work performed for this thesis.

Chapter 2

House Price Models

2.1 Introduction

The main objective of this chapter is to give an introduction to the theoretical concepts used in economic house price models. As with most economic relationships the house price model is controlled by the supply and demand equilibrium. Both the demand and supply will be discussed in section 2.2 along with showing which variables are most relevant in each relationship. The equilibrium, created by demand and supply, is also discussed in section 2.2 where a visual example of the house price relation is given. The effects market expectations can have on the house price market are discussed briefly in section 2.3 along with a short description of real house price development in Denmark over the last 30 years.

2.2 House Price Economics Background

Most economic relationships depend on the equilibrium created between supply and demand to determine the price of a product and house price models are no exception. House price relations are usually formulated as ***stock-flow*** models, where the term ***stock*** refers to the amount of real estates on the market. By using supply and demand relationship for this stock the real estate price can be derived. The ***flow*** term refers to the flow or input of new assets added to the stock. The rest of this section focuses on how the theoretical supply and demand relation can be formulated for a stock-flow model.

2.2.1 Long Term Demand

A basic long term model for the housing demand can be seen e.g. in an article by Barot and Yang [1] and also in a report from the National Bank of Iceland [4] as

$$H^D = f\bigg(\frac{PH}{P},\ R,\ YD,\ WA,\ D \bigg) \tag{2.1}$$

where the terms on the right are the ***explanatory variables*** for the ***effect*** or ***response variable*** on the left. The response variable is housing demand (H^D). The explanatory variables are house price (PH), the long term interest rate (R), disposable income (YD), wealth other than real estate (WA), the debt of individual or household (D) and the ***consumption deflator*** (P). In economics inflation adjustment, or "deflation", is accomplished by dividing a time series by a price index such as the ***consumption deflator***. The deflator is then representative of consumer prices at each time. In the MONA report the consumption deflator is modeled especially. For further discussion see [12][1]. PH/P, or house price divided by the consumption deflator is therefore the real house price.

In housing models it is usually assumed that ***income elasticity*** is one in the long run. Income elasticity is defined as the ratio between the change in some demand, housing demand in this case, and the change in income. If the income elasticity is one, then the long run changes in income will result in proportional changes in demand. The idea behind this correlation has a strong intuitive nature since people will always need a place to live and what is more they must afford it, house price can therefore not increase more than proportional to wages in the long run. Empirical grounds for this assumption can be seen in MONA model from the Danish National Bank [12][2]. Making use of income elasticity constraint, the house price formula from Eq.(2.1) can be expressed as

$$\frac{H^D}{YD} = \phi\bigg(\frac{PH}{P},\ R,\ \frac{WA}{YD},\ \frac{D}{YD} \bigg) \tag{2.2}$$

where the income elasticity has been applied to both WA and D, since these two variables also have a long term elastic relationship with disposable income.

[1] On page 96 the components that make up the consumption deflator are described in detail.

[2] On page 43, in the MONA model [12], Chart II.3.1 it is shown that Real disposable income as a ratio of stock of houses has been approximately 1 the last 30 years in Denmark.

Isolating the real house price term (PH/P) from Eq.(2.2) gives

$$\frac{PH}{P} = \theta\left(\frac{H^D}{YD},\ R,\ \frac{WA}{YD},\ \frac{D}{YD}\right) \tag{2.3}$$

which is sometimes called the inverted demand function. Eq.(2.3) describes the development of house prices in the long run, derived from the demand relationship in Eq.(2.1).

2.2.2 Long Term Supply

The fundamental assumption made concerning the flow of new assets into the housing market is by use of a concept called ***Tobin's Q***, see Barot and Yang [1]. Tobin´s Q describes the ratio between the value of certain assets and the cost of replacing those assets, or construction cost in the case of the housing market

$$\mathrm{Q} = \frac{PH}{PB} = \frac{\textit{asset prices}}{\textit{construction cost}}\ . \tag{2.4}$$

In the long run the Q should have an equilibrium of around one. If Q>1 there is an incentive to build more houses, since market value of the assets is higher than the cost to build new assets per stock of houses. If Q<1 residential investment will decrease. According to Barot and Yang using Tobin's Q along with incorporating interest R, also known as the cost of finance, gives the relationship

$$\frac{IH}{H} = h(Q, R) \tag{2.5}$$

which is called the ***Augmented Tobin´s model*** of housing investment. In Eq.(2.5), IH and H represent housing investment and stock of house, respectively. IH and H are measured in monetary value, price adjusted to some fixed point. The assessment of IH and H differs between countries, the estimation for Denmark can be seen in Lunde [8].[3]

If Q in Eq.(2.5) increases, housing investment also increases. This can easily be seen from the definition of Tobin´s Q given before, i.e. an incentive for house builders is present since Q>1. If interest rates go up, on the supply side, housing investment will decrease since house builders need funding and interest rates influence their decision of construction.

The development of stock of houses, i.e. the supply of houses is given with the following error correction form

$$H^S = IH + (1 - \delta)H_{t-1} \tag{2.6}$$

[3] Box B on page 8.

where the supplied stock of houses (H^S) comprised of new houses , i.e. housing investment (IH) together with last periods stock of houses (H_{t-1}) after depreciation (δ). More precisely, the supply of houses is the stock of houses from last period adjusted for depreciation plus the housing investment.

2.2.3 Equilibrium

The fundamental equilibrium relationship in the housing market is created where $H^D = H = H^S$, i.e. when housing demand H^D, also known as the wanted stock of houses, is equal to the supply of houses H^S. There is however a considerable lag in the supply side since it takes some time to adjust from when there is a surge in demand until the flow is delivered. In the interval when the supply is working on increasing stock it is normal for house prices to go up, to maintain the equilibrium. This can be best explained with an example.

Example 2.1 (Example of Equilibrium)
Figures 2.1 and 2.2 show two possible situations on a housing market. Figure 2.1 shows an equilibrium situation where the y-axis describes the price of houses (PH) and the x-axis shows stock of houses (H). Equilibrium is at point A where the price is $PH = PH^* = PB$ and the stock of house on the market is $H = H^*$, i.e. where the demand and supply lines intersect. To account for the lag in supply there are three supply lines. The supply for the short term horizon is completely vertical to represent that no flow is delivered in the short term. For the medium term demand some of the flow initiated by the surge in demand has been delivered and finally the long term demand when all the requested houses have been delivered.

In Figure 2.2 there has been a shift in demand. Demand line D has shifted upward and the new demand is now described by the line D^*. In the short term the shift in demand causes an increase to the price PH^{**}, to maintain the equilibrium the prices rise since demand has increased while there is no supply to meet the new demand. In this new equilibrium point B there is a strong incentive to start building houses, i.e. Q $>$1.

Looking to the medium term supply curve the supply has managed to partially satisfy the demand, resulting in a decline in prices to PH^{***} along with a increase in stock of houses to H^{**}, i.e. the delivered supply initiated by the demand shift. At the new medium term, equilibrium point C, there is however still an incentive to build houses since asset prices are higher than construction cost, i.e. Q$>$1. Looking to the long term supply response, the supply has serviced all of the demand, and the prices have returned to the initial value,

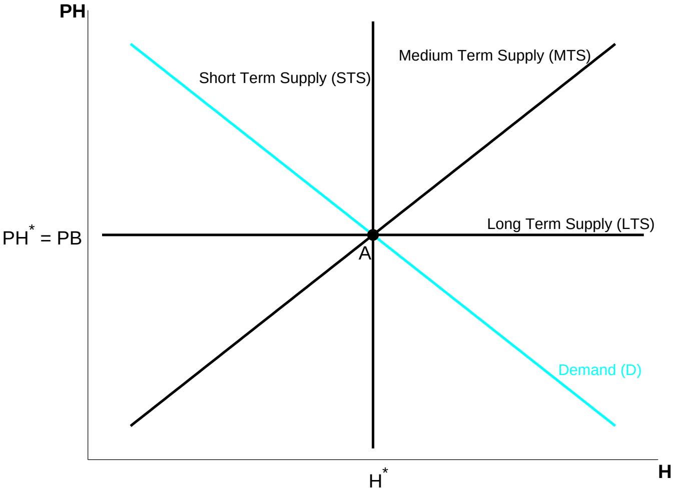

Figure 2.1: Shows a housing market in equilibrium at point A. The x-axis is the stock of houses H while the y-axis show the house price PH. Equilibrium is at the point $PH^* = H^*$.

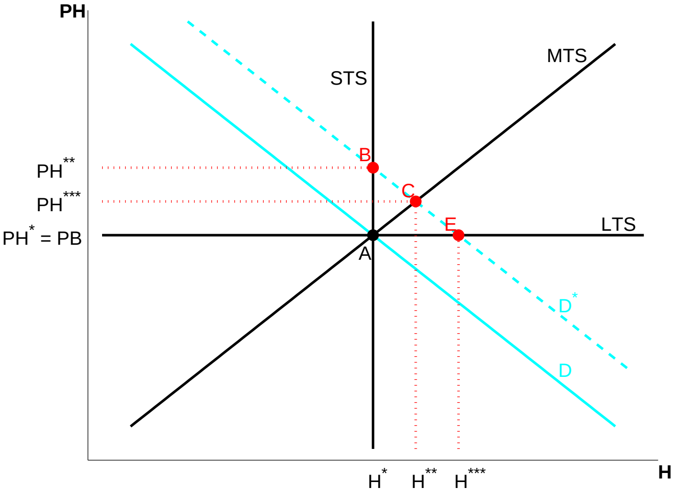

Figure 2.2: Shows the effects of increased demand on the equilibrium, figure adopted from [4].

$PH = PB$, resulting in Q=1 and new equilibrium point E with stock of houses at H^{***}.

□

The example above assumes that there is a sufficient supply of land for construction. According to the Icelandic National Bank [4] if land for construction is severely limited a permanent shift in the supply curve would take place and the long term equilibrium should take place at a higher price, e.g. C in Figure 2.2.

The short term supply is said to be ***completely inelastic***, i.e. vertical, since the immediate supply of houses compared to the existing amount of houses on the market is negligible. The long term supply is considered ***completely elastic***, i.e. horizontal, because Tobin´s Q controls the long term equilibrium, i.e. in the long term an equilibrium will be achieved at Q=1. Recall that elasticity measures the ratio of change between two elements.

Because of the steepness of the short term supply curve house prices are expected to oscillate greatly, especially if the demand curve is also steep. The dynamic nature of the system indicates that new changes in demand will usually have occurred before the supply flow from the previous change have arrived. This leads to an ever changing house price.

The power that interest rates have in this equilibrium is interesting. Interest rates have a dissuasive effect on both sides of the relation. For example high interest rates have a repelling effect on buyers on the demand side and also on contractors who need capital for their constructions on the supply side. Therefore it is obvious that the interest rate is an important factor in house price modeling.

The theoretical model above provides the macro economic long term relation for both the supply and demand side of the house price market. Applying the theory to data to get a viable applied house price model is however more complicated and requires the use of econometric methods, to capture the short term dynamics of the data. A well known problem with economic data is that it is often non-consistent with time and a limited amount of data is available, which causes further difficulty when modeling. The road from theory to application can often alter models drastically. However, the same main factors are always present in one form or another. The process of moving from theory to application in house price models is discussed further in section 5.4.

2.3 Market Expectation

Market expectations deserve special attention. The influence of market expectations on house prices is very hard to model. Usually market expectations should not present a problem in house price modeling since the market usually makes use of the information at hand, the factors mentioned before, which describe the market at each time. However, at times investors believe that the market has some untapped potential, or they expect it to rise even more and try to "ride" the rise to the end which is also known as ***herd behavior***. This can result in price changes which are inconsistent with the values of the other variables. This kind of behavior can in the long-run lead to the creation of a ***house price bubble***, which is a price increase not founded by the data believed to describe the development of house prices.

Recently in Denmark there has been a long run of rising real house prices, where before the market had behaved in cyclical periods, see Figure 2.3. The development of real house prices the last ten years or so has lead to an increase in discussion whether a house price bubble exists in the Danish housing market, or certain specific parts of it. Bubbles are quite hard to detect and the full extent of them is often not known until after they ***burst***. A burst is when the prices return to "normal" behavior from their over inflated state usually with a sharp decline. According to Lunde [8] the Danish housing market shows some signs of a housing bubble in some specified field of the housing market, such as urban flats and summer houses. This topic of herd behavior will be revisited when forecasting for out of sample house prices in subsection 5.6.1.

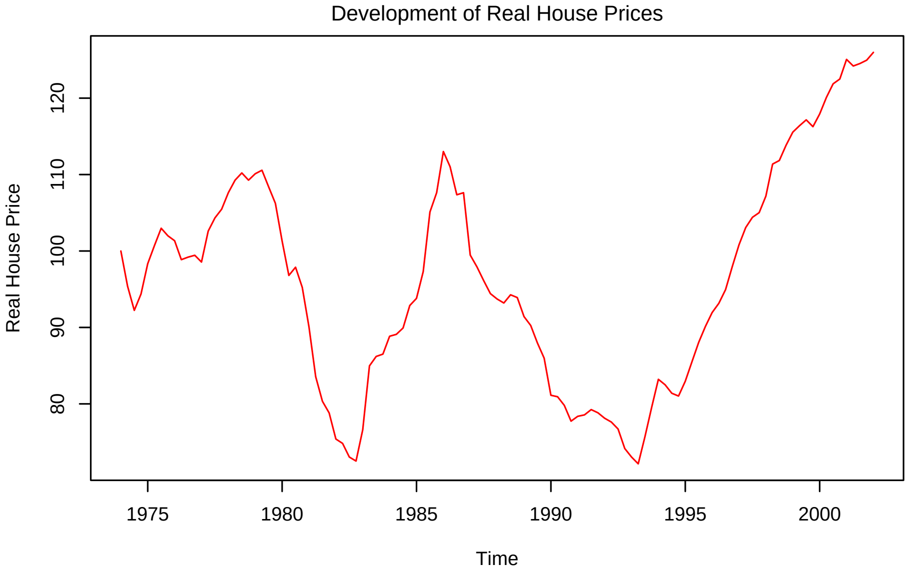

Figure 2.3: The development of real house prices the last 30 years in Denmark. Notice the break from the cycle around '97.

Chapter 3

House Price Dynamics I Modeling the Nykredit relationship

3.1 Introduction

In this chapter a simple benchmark relation for a house price dependant solely on interest rates will be formulated. Along with modeling the interest relation, the scenario tree structure which will be used through out this report is introduced. In section 3.2 a short account is given of the simple relation which will be modeled in this chapter. In section 3.3 the simple interest relation is applied to a one ***path***, i.e. a single time line scenario, to better realize the dynamics of the relation. Section 3.4 introduces the scenario tree concept along with a brief comment on the application of such a model. In section 3.5 the one path case is expanded to the scenario tree case. Finally in section 3.6 the model is compared to a simpler model, as well as giving examples of house price trees.

3.2 The Nykredit Relation

This first relationship between interest rates and house prices will be modeled and implemented to a trinomial scenario tree. The relationship used here is based on a very simple interest only relation, taken from a report published by Nykredit in May 2006 [10], which states:

$$\text{Nykredit result: } \begin{cases} 1\% \uparrow \text{ in short rates,} & 5\% \downarrow \text{ in house prices after one year;} \\ & 11\% \downarrow \text{ in house prices after two years;} \\ & \\ 1\% \downarrow \text{ in short rates,} & 5\% \uparrow \text{ in house prices after one year;} \\ & 11\% \uparrow \text{ in house prices after two years;} \end{cases}$$

This is a very simplified model where the only cause of changes in house prices is a change in interest rates, i.e. the only explanatory variable is change in interest rates. Although the relation is simple it will give a good idea of how to model more complex house price scenario trees and the programming done for this model will easily be extended to more complex models.

3.3 Modeling for one Path

Initially the Nykredit house price relation was considered as a single path relation, that is on a one dimensional time line. At each time on the time line there is a node holding observed and predicted information. Each node has a number, period, house price and interest rate. The modeling involves developing a relation for house prices based on interest rates and house prices from past periods, this sort of formulation is also known as a ***recursive*** relationship.

To calculate the effect of interest rate changes in the house price a few variables are needed. Firstly the change in interests rate between years is defined as ΔSR_t. More precisely, the interest rate change between any two points at time$=t$ and time$=t-1$ can be expressed as

$$\Delta SR_t = SR_t - SR_{t-1} \tag{3.1}$$

The Δ operator is called a ***difference*** operator and will be discussed further in section 4.2. Two other variables are also defined to express the change in house prices, i.e. the change after one year (OneYearEffect$_t$) and the change after two years (TwoYearEffect$_t$). These two house price changes are expressed as follows

$$\text{OneYearEffect}_t = -5HP_t(SR_t - SR_{t-1}) = -5HP_t \cdot \Delta SR_t \tag{3.2}$$

$$\text{TwoYearEffect}_t = -11HP_t(SR_t - SR_{t-1}) = -11HP_t \cdot \Delta SR_t \tag{3.3}$$

In Eq.(3.2) and (3.3) it is assumed that interest rates are expressed as decimal fractions. The minus is to account for the negative relationship between changes in interest rates vs. changes in house prices. If there is a change in interest rates between periods t and $t+1$ the effect of that change will not influence the house prices until at time $t+2$. The ***base*** house price, i.e. the price the change is applied to at each time, will be the house price from the previous period, e.g. at period t the base price is set to the result from period $t-1$. Eq.(3.2) and (3.3) along with knowledge of how much start up time the house price vs. interest rate lag needs, give the conditional formula for house prices, derived as

$$HP_t = \begin{cases} HP_0 & \text{if } t < 2 \\ HP_{t-1} - 5HP_{t-1} \cdot \Delta SR_{t-1} & \text{if } t = 2 \\ HP_{t-1} - 5HP_{t-1} \cdot \Delta SR_{t-1} - 11HP_{t-2} \cdot \Delta SR_{t-2} & \text{if } t > 2 \end{cases} \tag{3.4}$$

Eq.(3.4) assumes that time indexing (t) starts from 0. HP_0 is the startup house price, usually this would be set to 1 or 100. By using Eq.(3.2) and (3.3), Eq.(3.4) can be expressed as

$$HP_t = \begin{cases} HP_0 & \text{if } t < 2 \\ HP_{t-1} + \text{OneYearEffect}_{t-1} & \text{if } t = 2 \\ HP_{t-1} + \text{OneYearEffect}_{t-1} + \text{TwoYearEffect}_{t-2} & \text{if } t > 2 \end{cases} \tag{3.5}$$

The dynamic nature of Eq.(3.5) can best be viewed by showing the first special cases $t \in \{0, 1, 2\}$ along with the first general case $t = 3$ on a node graph.

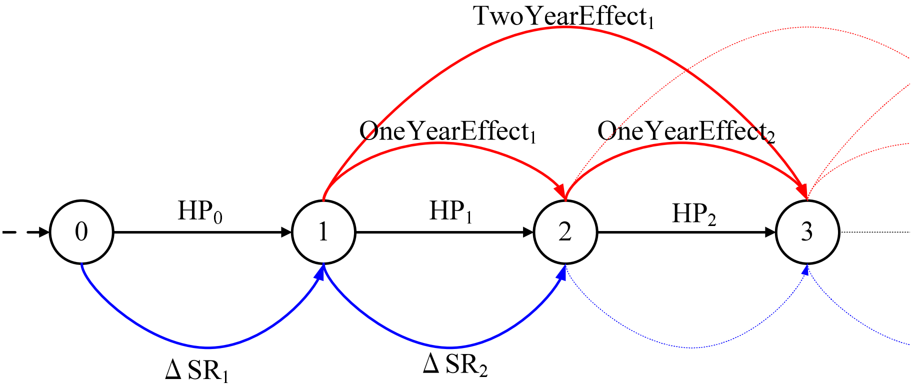

Figure 3.1: Visual representation of the first 4 periods in the conditional relationship, between interest rates and house prices, shown in Eq.(3.5).

Figure 3.1 shows the development of house price for the first 4 periods, including the special cases for $t < 3$. At time 0 the only input is the initial house price or HP_0. Between period 0 and 1 there is a change in interest rate, this change will effect the house price both at time 2 and 3, the change in interest rates will now be calculated at each period. At time 1 only HP_0 contributes to the new house price HP_1. The house price at time $t = 2$ has the first interest rate effect (OneYearEffect_1) which is added to HP_0.

$$HP_2 = HP_0 + \text{OneYearEffect}_1$$

At time 3 the first general case occurs, which means that the lag for HP vs. ΔSR is sufficient to give both the one and two year effects. At time 3 the base, or input, house price is the one from the previous year or HP_2. The OneYearEffect_2 from year two and the TwoYearEffect_1 from year one also affect the house price at HP_3

$$HP_3 = HP_2 + \text{OneYearEffect}_2 + \text{TwoYearEffect}_1$$

which is an example of the general case, i.e. when $t > 2$.

3.4 The Scenario Tree

Extending the model, in Eq.(3.5), to a tree structure is relatively easy. The relationship is still conditioned on the periods (t) as it was in Eq.(3.5). To account for the more complex recursive nature when dealing with the scenario tree format a new index is added along with formulating the tree structure in this section. The following notation for a scenario tree is borrowed from Rasmussen and Clausen [13].

A finite probability space $(\Omega, \mathcal{F}, P)$ is defined where the outcomes are a sequence of real-values (interest rates) over some discrete time period $t = 0, \cdots, T$. T is also sometimes called ***horizon***.

A ***scenario tree*** is generated by matching the probability outcomes $\omega \in \Omega$ to the corresponding nodes $n \in \mathcal{N}_t$ at time t in the tree.

Each node in the scenario tree $n \in \mathcal{N}_t$ for $1 \leq t \leq T$ has a unique ***parent node*** denoted by $a(n) \in \mathcal{N}_{t-1}$. Every node $n \in \mathcal{N}_t$ for $0 \leq t \leq T - 1$ also has a non-empty set of ***child nodes*** denoted by $\mathcal{C}(n) \subset \mathcal{N}_{t+1}$.

The nodes at horizon, $n \in \mathcal{N}_T$, are called ***leaf nodes***. The initial node $n \in \mathcal{N}_0$ is called the ***root node***. From each leaf node there is a unique recursive relationship to the root node, each such relationship is called a ***path***.

The recursive nature of the paths corresponds to the formula given in Eq.(3.5), the parent-child relationship must therefore be included into the Eq.(3.5) to preserve the scenario tree dynamics.

3.4.1 Example and Implementation

A full scenario tree can be of different types, these types are decided by the number of child nodes each parent node produces. For example if $n = 1$ is the root node then $|\mathcal{C}(1)| = 2$ is a binomial tree while $|\mathcal{C}(1)| = 3$ is a trinomial tree and so on.

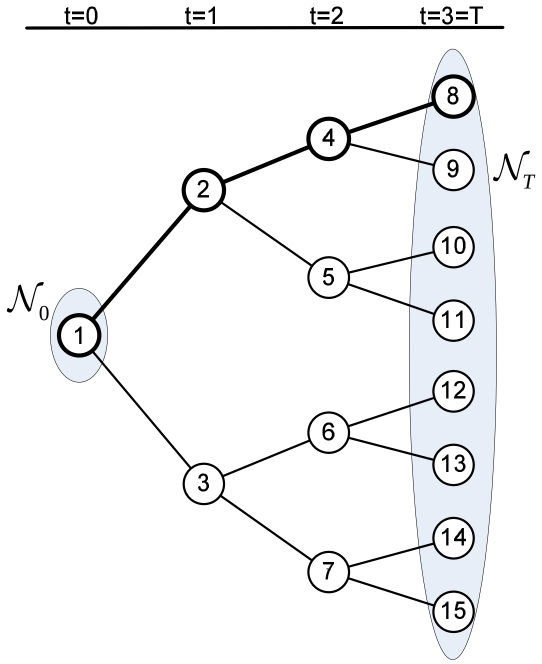

Figure 3.2: Example of a $|\mathcal{C}(1)| = 2$ tree or binomial tree. Here $N = 2^0 + 2^1 + 2^2 + 2^3 = 15$ and $T = 3$.

Using the tree type along with the period t can tell how many nodes are in an arbitrary set $\mathcal{N}_t$ by

$$|\mathcal{N}_t| = |\mathcal{C}(1)|^t \quad 0 \leq t \leq T$$

The total number of nodes in the tree N is therefore easily found by summing over all periods.

$$N = \sum_{t=0}^{T} |\mathcal{N}_t| \tag{3.6}$$

Figure 3.2 shows an example of scenario tree with $t \in \{0, 1, 2, 3\}$ and $n \in \{1, \cdots, 15\}$. It can be seen that the tree is binomial since each node, except for the leaf nodes, has two child nodes. The set of leaf nodes is shown as $\mathcal{N}_T$, the root node set, including only $n = 1$, is shown as $\mathcal{N}_0$.

When programming the scenario tree structure, two different methods were used. Originally an indexing method was applied in `Matlab` and R, which depends highly on the parent relationship as well as 3.6. The first version was later expanded by using an object oriented approach. The programming part of the scenario trees is given a thorough discussion in appendix A.

3.5 Applying to a Scenario Tree

The path concept from the tree structure corresponds very well with the single time line implementation given in Eq.(3.5).

Nodes in the scenario tree structure inherit house prices from the node in the previous period. This is the same as in the single path case, however since there are now multiple nodes at each time the recursive nature is preserved through the parent-child relationship as well as time. More precisely nodes inherit house prices from the parent node in the scenario tree.

The house price is now expressed as $HP_{n,t}$ where the n index indicates the node number and t, as before, indicates the period. Using the new indexing the tree can be expressed as $|\mathcal{N}_T|$, i.e. the number of leaf nodes, cases of a single path type. For example Figure 3.2 gives $2^3 = 8$ paths where the top path, in term of node indexes, is $1-2-4-8$ and the bottom path is $1-3-7-15$. The interest rate change between nodes is defined for the scenario tree as

$$\Delta SR_{n,t} = SR_{n,t} - SR_{a(n),t-1} \qquad 1 \leq t \leq T \tag{3.7}$$

Recall that $a(n)$ gives the parent of node n. Eq. (3.2) and (3.3) also become node dependant, shorten the names to One and Two

$$\begin{aligned} \text{One}_{n,t} &= -5HP_{n,t}(SR_{n,t} - SR_{a(n),t-1}) \\ &= -5HP_{n,t} \cdot \Delta SR_{n,t} \end{aligned} \tag{3.8}$$

$$\begin{aligned} \text{Two}_{n,t} &= -11HP_{n,t}(SR_{n,t} - SR_{a(n),t-1}) \\ &= -11HP_{n,t} \cdot \Delta SR_{n,t} \end{aligned} \tag{3.9}$$

Finally the model stated in Eq.(3.5), extended to the scenario tree becomes

$$HP_{n,t} = \begin{cases} HP_{n=1,t=0} & \text{if } t < 2 \\ HP_{a(n),t-1} + \text{One}_{a(n),t-1} & \text{if } t = 2 \\ HP_{a(n),t-1} + \text{One}_{a(n),t-1} + \text{Two}_{a(a(n)),t-2} & \text{if } t > 2 \end{cases} \tag{3.10}$$

Where n stands for the node number, $a(n)$ is the parent node of node n, $a(a(n))$ is the parent of $a(n)$ and the grandparent of n. Initially at $HP_{n=1,t=0}$ a initial house price is set, e.g. $HP = 100$. Because of the lag between interest rates and house prices, an interest rate tree of length T will result in a house price tree of length $T+1$.

3.6 Data

In this section a brief discussion will be given on implement Eq.(3.10) for optimal memory usage and comparability to the interest tree. Comparison of Eq.(3.10) to a simpler form of the relation is done and tests performed to see the difference between the two. The distribution of the node mass at time T is also inspected for both methods.

3.6.1 Lagged House Price Tree

When it comes to programming the relation in 3.10 it is a good idea to shift the house price tree, i.e. lag it by one time unit. Lagging the HP tree results in it being the same size as the interest rate tree, i.e having T periods instead of $T+1$. The one period lagged version of Eq.(3.10) for the house price tree is therefore achieved by moving the house price as follows:

$$\text{One}^*{}_{(n,t)} = -5HP_{(a(n),t-1)} \cdot \Delta SR_{(n,t)} \tag{3.11}$$

$$\text{Two}^*{}_{(n,t)} = -11HP_{(a(n),t-1)} \cdot \Delta SR_{(n,t)} \tag{3.12}$$

So the first node is cut of and the HP tree moved back one period. The resulting updated version of Eq.(3.10) is

$$HP_{n,t} = \begin{cases} HP_{n=1,t=0} & \text{if } t < 1 \\ HP_{a(n),t-1} + \text{One}^*{}_{N,t} & \text{if } t = 1 \\ HP_{a(n),t-1} + \text{One}^*{}_{N,t} + \text{Two}^*{}_{a(n),t-1} & \text{if } t > 1 \end{cases} \tag{3.13}$$

This is possible because of the HP lagged dependance on ΔSR and because the tree grows by q^t as time passes, where q is the tree type $|\mathcal{C}(1)| = q$. Because of the lag ΔSR results in q identical house price nodes when using Eq.(3.10), i.e. each price is replicated to q child nodes. This replication is not ideal as it makes the house price trees different from the interest rate trees in size as well as being a waste in memory, since there are only q^T unique nodes and $q^{T+1} - q^T$ are therefore wasted.

By shifting the tree back one period $q^{T+1} - q^T$ nodes are saved which is important when calculating for big trees. In Figure 3.3 an example of a full tree and a lagged tree is given for a $n = 3$ and $T = 3$ interest rate tree. Both trees are identical in shape and information, except for the redundant first node which has been cut out in the lagged tree. This method of lagging is the one that was applied. The HP trees will however be displayed with their right time horizon and be noted as $T + 1$ trees.

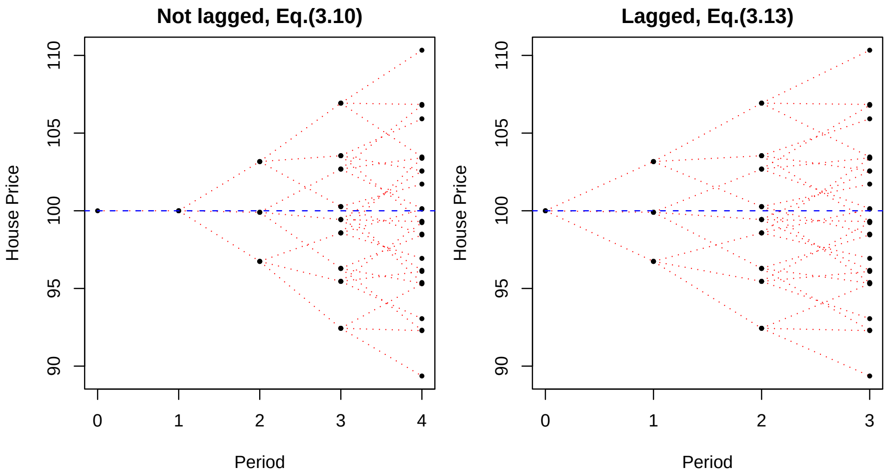

Figure 3.3: Here is an example of the original tree, trinomial and $T = 3$, using Eq.(3.10) to the left while the augmented version Eq.(3.13) is to the right. Both of these house price trees are so called non-recombining path trees.

3.6.2 Recombining Paths vs. Non-Recombining Paths

A ***recombining path*** scenario tree, also known as a ***lattice*** scenario tree, is where an up-down move in the scenario tree will result in the same value as a down-up move. This is best explained by a visual example see Figure 3.4 for a lattice tree, while Figure 3.3 shows an example of a ***non-recombining path*** tree, i.e. where a up-down move does not have to end in the same value as an up-down move. Recombining trees are often used in derivative pricing theory, as well as in dynamic programming and as decision trees. The main benefit that recombining trees have over non-recombining trees is that they are more recursively tractable and for the same horizon T have far fewer nodes than a non-combining tree.

In the next subsection, lattice as well as non-recombining, interest rate trees will be used as input to see what effect that has on the house price development.

3.6.3 ΔHP method

For contrast another method of modeling is compared to the relation in Eq.(3.10). The method used for comparison describes the percentage change in house price at each time irrelevant to the current house price at that time. The comparison method will be noted as ΔHP, while Eq.(3.10) will be noted as HP. To get

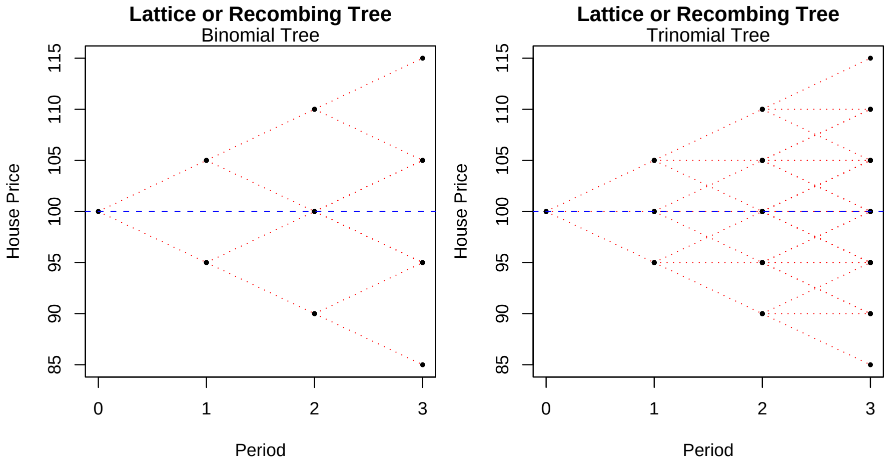

Figure 3.4: An example of a lattice or recombining tree, left panel is a binomial tree while the right show a trinomial tree. Notice that up-down and down-up result in the same house price.

the change from start to a certain period t where $0 \leq t \leq T+1$ the relation can be expressed as

$$\Delta 1_{n,t} = -5\Delta SR_{n,t} \qquad \Delta 2_{n,t} = -11\Delta SR_{n,t}$$

$$\Delta HP_{n,t} = \begin{cases} 0 & \text{if } t < 2 \\ \Delta HP_{a(n),t-1} + \Delta 1_{a(n),t-1} & \text{if } t = 2 \\ \Delta HP_{a(n),t-1} + \Delta 1_{a(n),t-1} + \Delta 2_{a(a(n)),t-2} & \text{if } t > 2 \end{cases} \tag{3.14}$$

Which can be viewed as change from some beginning index I by

$$HP_{n,t} = I \cdot (1 + \Delta HP_{n,t}) \tag{3.15}$$

The difference between these methods in essence is that the ΔHP method shows the change in house price from $t = 0$ to times $t = 1, ..., T+1$ in one step, i.e. without updating the base at each time. A short example for the two methods, given a vector of house price changes called $\Delta kp = [0.1, -0.1, 0.05]$ and an initial price of $kp_0 = 1$. Using the ΔHP and HP methods gives

$\underline{\Delta HP:}$	$\underline{HP:}$
$kp_1 = kp_0(1 + 0.1) = 1.1$	$kp_1 = kp_0(1 + 0.1) = 1.1$
$kp_2 = kp_0(1 + 0.1 - 0.1) = 1$	$kp_2 = kp_1(1 - 0.1) = 0.99$
$kp_3 = kp_0(1 + 0.1 - 0.1 + 0.05) = 1.05$	$kp_3 = kp_2(1 + 0.05) = 1.04$

This small example shows that the ΔHP method should give linear transformation of lattice interest rate trees resulting in lattice house price trees, since up-down result in the same value as down-up moves. The HP method is however more complex and has a compound nature. In the next subsection these two methods will be compared by using interest trees.

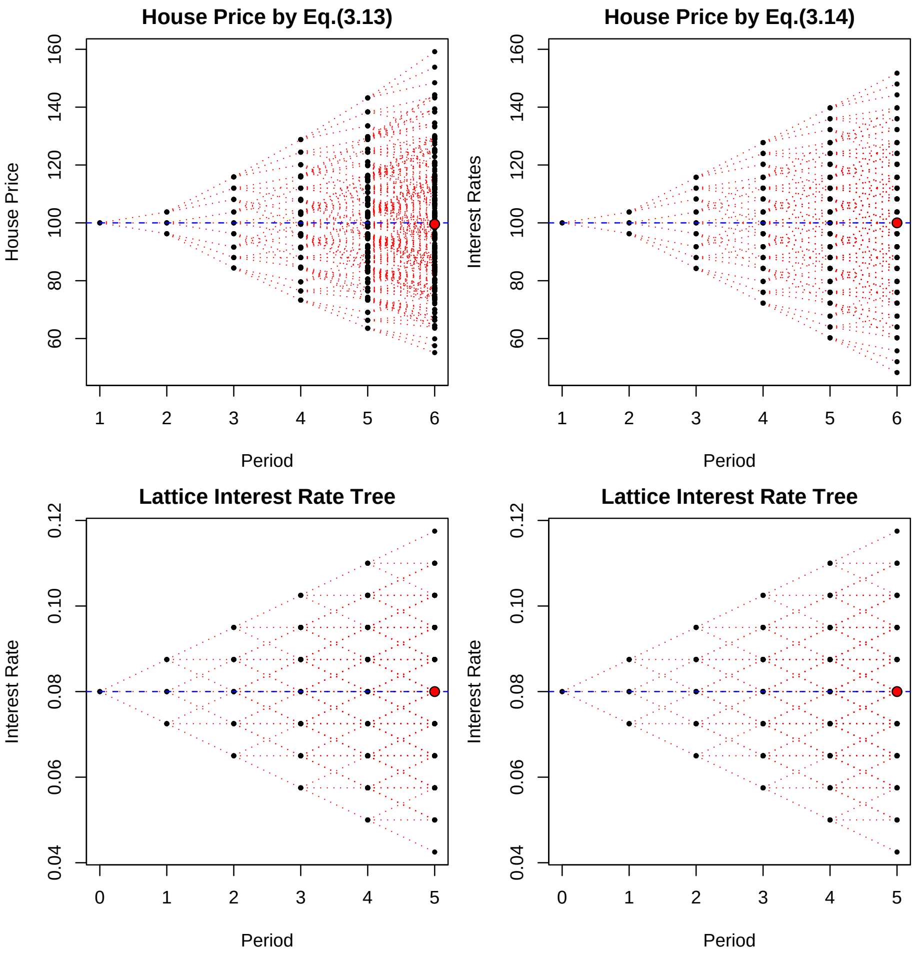

Figure 3.5: The upper half shows the house price trees. Upper left is the House Price tree where change is based on the house price at each time. Upper right is the ΔHP relation with $I = 100$. The lower graphs show the interest trees where each change is $a = 0.0075$, resulting in range of $0.1175 - 0.0425$ interest at time T.

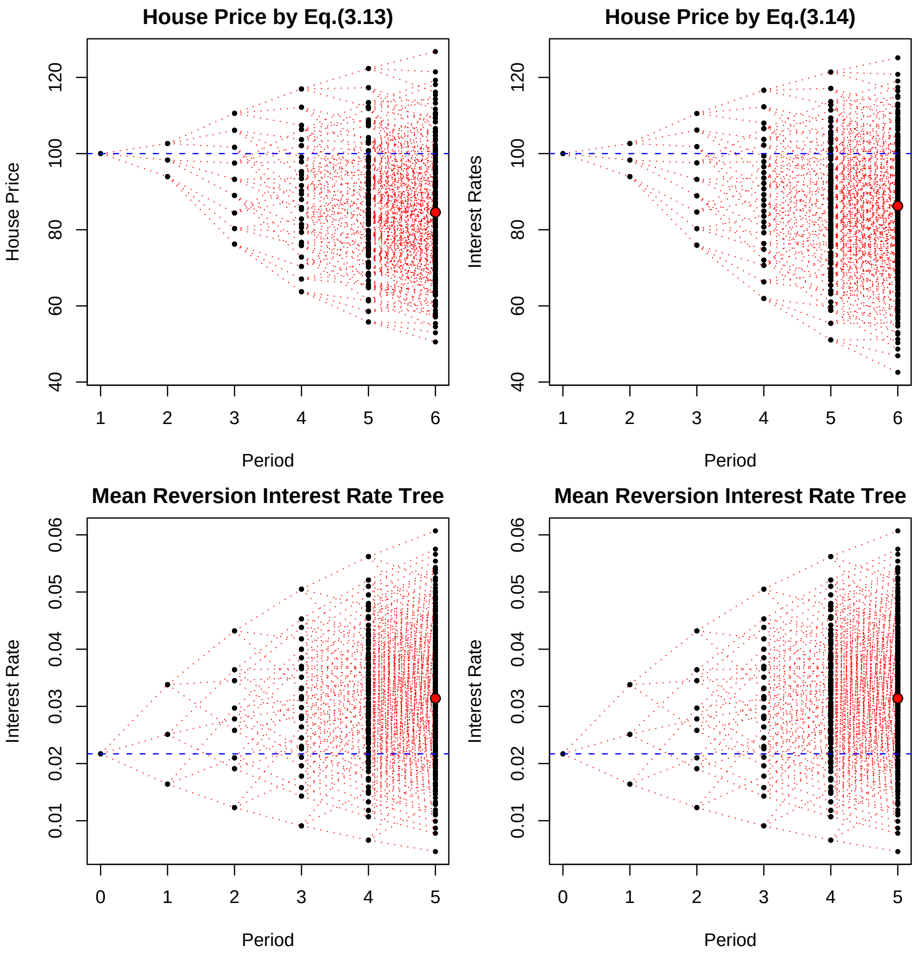

Figure 3.6: The upper half shows the house price trees. Upper left is the House Price tree where change is based on the house price at each time. Upper right is the ΔHP relation with $I = 100$. The lower graphs show interest rate trees .

3.6.4 Comparison

The two methods, HP and ΔHP, were tested together using identical trinomial interest rate trees. Both lattice trees as well as more diverse and real like interest trees were used as input. For the lattice tree interest rates can at each time rise by a, fall by a or stay the same. The range ($2a$) of each change for the case shown in Figure 3.5 is fixed to $2a = 0.015$. From Figure 3.5 it can be seen that using the HP method at each time introduces a certain nonlinearity to the relation, while using the ΔHP conserves the interest tree proportion to the HP tree, giving a lattice house price tree. The median, the red dot, which marks the center of density for the distribution of the nodes at time $T+1$ has slightly moved down for the HP case which is to be expected since compounding makes it harder to increase the house price once it has declined. The results maximum, minimum and median values can be seen in Tables 3.1 and 3.2 for the HP and ΔHP methods respectively, when using the lattice tree.

t	1	2	3	4	5	6
Max	100.00	103.75	115.89	128.80	143.19	159.18
Med	100.00	100.00	100.00	99.55	99.55	98.56
Min	100.00	96.25	84.39	73.29	63.57	55.14

Table 3.1: The maximum, median and minimum house price values for each period, using the HP method corresponding to Figure 3.5, upper left panel.

t	1	2	3	4	5	6
Max	100.00	103.75	115.75	127.75	139.75	151.75
Med	100.00	100.00	100.00	100.00	100.00	100.00
Min	100.00	96.25	84.25	72.25	60.25	48.25

Table 3.2: The maximum, median and minimum house price values for each period, using the ΔHP method corresponding to Figure 3.5, upper right panel.

In Figure 3.6 the input interest tree is a so called ***Mean reversion*** interest rate tree. Mean reversion is based on the mathematical premise that the initial price is not the mean but with time the process will eventually move back towards the mean or in this case some average interest rate.

The results for the house prices in Figure 3.6 show the same effects as the previous comparison, i.e. the HP method reduces (damps) the down turn and rises higher than the ΔHP tree. The median, for the HP tree, as before shows that the HP tree tends to bring the center of node density down, which is to be expected with the compounding effect. The median for ΔHP however represents the center of the interest rates tree. The corresponding maximum, minimum and median values, for each period, can be seen in Tables 3.3 and 3.4 for the ΔHP and HP methods respectively, when using the mean reversion interest rate tree. In Figure 3.7 a histogram for Figure 3.6, i.e. the house price when using mean reversion interest rates, is shown. It can be seen from the histogram how the transformation of the ΔHP is a linear transformation while the HP skews the the node distribution downward, giving an upward tail.

t	1	2	3	4	5	6
Max	100.00	102.65	110.53	116.64	121.41	125.16
Med	100.00	98.30	93.21	89.24	86.24	83.88
Min	100.00	93.95	75.94	61.95	51.07	42.55

Table 3.3: The maximum, median and minimum house price values for each period, using the ΔHP method corresponding to Figure 3.6, upper right panel.

t	1	2	3	4	5	6
Max	100.00	102.65	110.58	116.98	122.34	126.78
Med	100.00	98.30	93.23	89.38	84.58	82.18
Min	100.00	93.95	76.22	63.73	55.79	50.54

Table 3.4: The maximum, median and minimum house price values for each period, using the HP method corresponding to Figure 3.6, upper left panel.

ΔHP conserves the form of the interest rate tree, shown on the lower half, much better.

To summarize, three observation about the house price trees have been noticed from the above comparison. Firstly a sequence of downward changes in interest rates will give a higher house price with HP than ΔHP, i.e. the HP shows exponential growth while Δ conserves the interest change. A sequence of rises in interest rates will give a dampened decline in HP compared to the ΔHP one which again conserves the interest rate tree. Lastly the density mass of nodes will move downward at horizon $T+1$ for HP, while ΔHP will conserve the interest rate tree density. All of these differences between HP and ΔHP can be explained by the compounding effect when using HP. For short periods, e.g. $(T+1) < 4$, the ΔHP relation proves a good estimation to the HP compounding relation. However as $T+1$ increases the difference between the two also increase. The long term change of the leafs, given a lattice tree with $2a = 0.015$, is shown in Figure 3.8. As a, i.e. the change in interest rates, increases so does the difference between HP and ΔHP.

3.7 Summary

The conclusion of this analysis is that the Nykredit relation modeled in Eq.(3.10) is a rather crude relation for modeling the house price to interest rate relation. The relation is probably not meant to run over many years with compounding, without yearly correction to actual data. It is a good idea to plot the HP without compounding, i.e. ΔHP as expressed in Eq.(3.15) to benchmark Eq.(3.10)

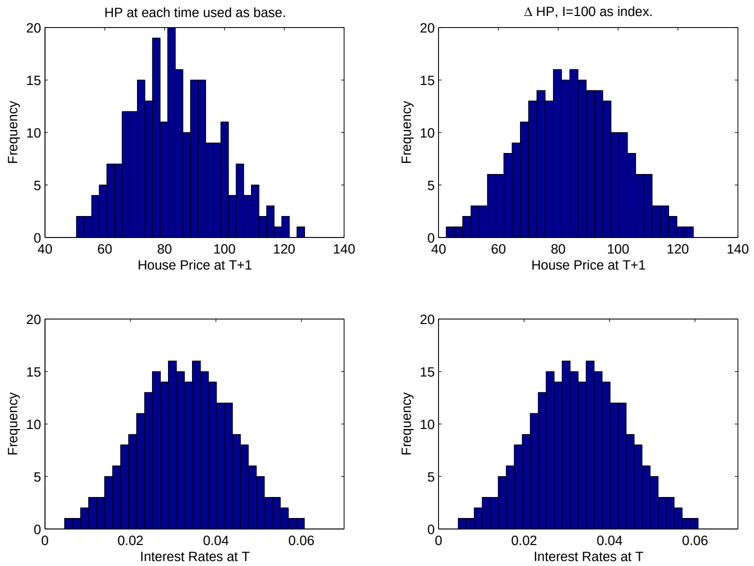

Figure 3.7: A histogram showing the distribution of the house price nodes at time (T+1) for the trees in Figure 3.6, in the upper half. The lower half shows the distribution of the interest rates in 3.6 at time T.

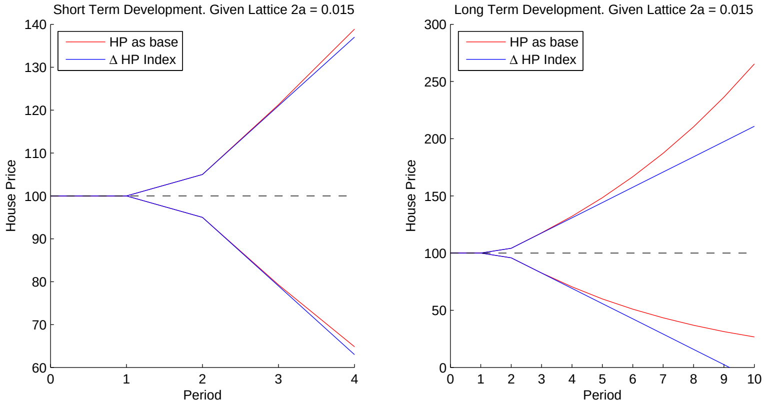

Figure 3.8: Shows the long term development of the leafs for the two ways of computing house prices, given a lattice tree with $2a = 0.015$.

to a linear transformation of the interest rate tree, when using the Nykredit relation. In subsequent chapters a more sophisticated relation for house price to interest rate relation will be inspected.

Chapter 4

Time Series and Econometric Theory

4.1 Introduction

Before moving into statistical analysis of the MONA house price model in the next section a few important concepts used frequently in time series and econometric analysis are listed and discussed. Most of the definitions and examples listed in this chapter are influenced or adapted from three time series books, i.e. Madsen [9], Tsay [15] and Hamilton [3].

In section 4.2 an account of basic econometric and time series concepts, needed to understand the models and terms used in empirical modeling of house prices is presented. Section 4.3 introduces two important time series models frequently encountered in econometric and financial analysis. Section 4.4 shows three well known methods for estimating parameters in time series models. Finally in section 4.6, methods of checking the quality of the estimated parameters are introduced.

4.2 Time Series Analysis

Economic time series data, as was mentioned in 2.2, often has some non ideal features making it hard to model, e.g. long term trends, periodic trends or even more general time varying behavior. Series exhibiting this sort of behavior are called ***non-stationary series***, forcing a series to be "stationary" is therefore important for analysis and modeling of the data. So called ***Weak Stationarity*** , which will be noted as ***stationarity*** from now on, is formally defined as;

Definition 4.1 (Weak Stationarity)
A series $\{r_t\}$ is said to be ***weakly stationary of order k*** if all first k moments are invariant to changes in time. A weakly stationary process of order 2 is simply called weakly stationary.

◇

If the mean and variance, the first two moments, are time invariant the series is stationary. Stationary series can be evaluated with classical time series methods and used to predict for future values.

Another definition used frequently is that of ***white noise***

Definition 4.2 (White Noise)
A series $\{\varepsilon_t\}$ is said to be ***completely random*** or ***white noise***, if ε_t is a sequence of mutual uncorrelated identically distributed stochastic variables with mean value 0 and constant variance σ_ε^2. This implies that

$$\mu_t = E[\varepsilon_t] \qquad \sigma_t^2 = V[\varepsilon_t] = \sigma_\varepsilon^2$$

$$\gamma_\epsilon(k) = Cov[\varepsilon_t, \epsilon_{t+k}] = 0 \quad \text{for} \quad k \neq 0$$

◇

To illustrate the stationarity along with white noise a small example is displayed, largely adapted from Madsen [9][1], showing a special case of a lag one autoregressive process (AR(1)) also known as ***random walk***.

Example 4.1 (AR(1) - Random Walk Series)
Let $\{\varepsilon_t\}$ be a normally distributed white noise sequence where $E[\varepsilon_t] = 0$ and

[1] see page 101

$V[\varepsilon_t] = \sigma^2$. Let $\{\varepsilon_t\}$ also be the input to dynamic relationship defined by a difference equation as

$$r_t = \phi r_{t-1} + \varepsilon_t \tag{4.1}$$

which then defines a new stochastic series $\{r_t\}$. By successively substituting $r_{t-1} = \phi r_{t-2} + \varepsilon_{t-1}$, $r_{t-2} = \phi r_{t-3} + \varepsilon_{t-2}$,... and so on, it is seen that Eq.(4.1) can be written as

$$r_t = \varepsilon_t + \phi\varepsilon_{t-1} + \phi^2\varepsilon_{t-2} + \cdots + \phi^i\varepsilon_{t-i} + \cdots \tag{4.2}$$

From Eq.(4.2) it can be seen that

$$\mu_r = E[r_t] = 0$$

and

$$\sigma_r^2 = V[r_t] = (1 + \phi^2 + \phi^4 + \cdots + \phi^{2i} + \cdots)\sigma^2 = \frac{\sigma^2}{(1-\phi^2)} \tag{4.3}$$

conditioned that $|\phi| < 1$. If $|\phi| \geq 1$ the variance is unbounded and the series is non-stationary, e.g. see Figure 4.1. A special case is when $\phi = 1$ where Eq.(4.1) is the so-called ***random walk series***, which is non-stationary.

The bounded variance in Eq.(4.3) is achieved by using the well known geometrical series

$$\frac{1}{1-x} = \sum_{n=0}^{\infty} x^n = 1 + x + x^2 + x^3 + \cdots \qquad \text{for} \qquad |x| < 1. \tag{4.4}$$

□

The coefficient ϕ acts as the memory of the process. For ϕ values close to 1 there is a long memory, small values of ϕ result in a short memory. The memory of a process is usually examined by the ***autocorrelation function*** (ACF), which gives a indication of how correlated, dependant, a series is to previous, lagged, values.

Example 4.2 (ACF and AR(1))

Consider the series shown in Figure 4.1, where four AR(1) series with $\phi \in \{0, 0.5, 0.9, 1\}$ have been simulated with white noise $a_t \sim N(0,1)$. The autocorrelation functions for each of the four different series is displayed in Figure 4.2. For $\phi = 0$ the series becomes $r_t = a_t$ i.e. only white noise. The ACF for $\phi = 0$, depicted in the upper left panel, shows that there is no dependency on previous values of r_t i.e. this process is without memory. The upper right and lower left panels show AR(1) with $\phi = 0.5$ and $\phi = 0.9$, respectively. The increasing height of the stems with increasing lags, i.e. previous observations, indicates that the two series are more dependant on previous values. The lower right panel shows the random walk with $\phi = 1$ which is non-stationary, notice how dependant the value at time t is to previous, lagged, values.

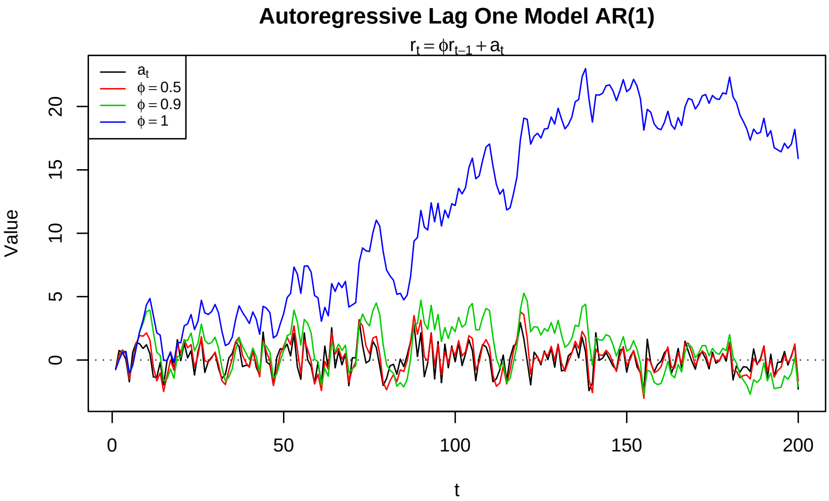

Figure 4.1: A simulation of a AR(1) process as described in Example 4.1 with different levels of the coefficient ϕ. The sequence $\{a_t\}$ is white noise where $a_t \sim \mathcal{N}(0, 1)$

□

The random walk model is listed in detail since it is considered as the model for many financial and economic series. The random walk series is also a perfect example of a special kind of non-stationarity called ***unit-root non-stationarity***. Given the unit-root non-stationary random walk series

$$r_t = r_{t-1} + a_t$$

it is seen that the current value r_t is based completely on the last value r_{t-1} plus the value of the equally likely plus/minus effect from the white noise (a_t). See Tsay [15][2] for a more detail description of unit-root non-stationarity.

An important operation used when analyzing unit-root non-stationary time series is called ***differencing***. The ***difference operator*** Δ is defined as

$$\Delta r_t = r_t - r_{t-1}$$

i.e. observing the change in level r_t instead of the level.

Example 4.3 (AR(1) Differencing)

Given the random walk process from Example 4.1 ($\phi = 1$) and taking the

[2] See chapter 2.7

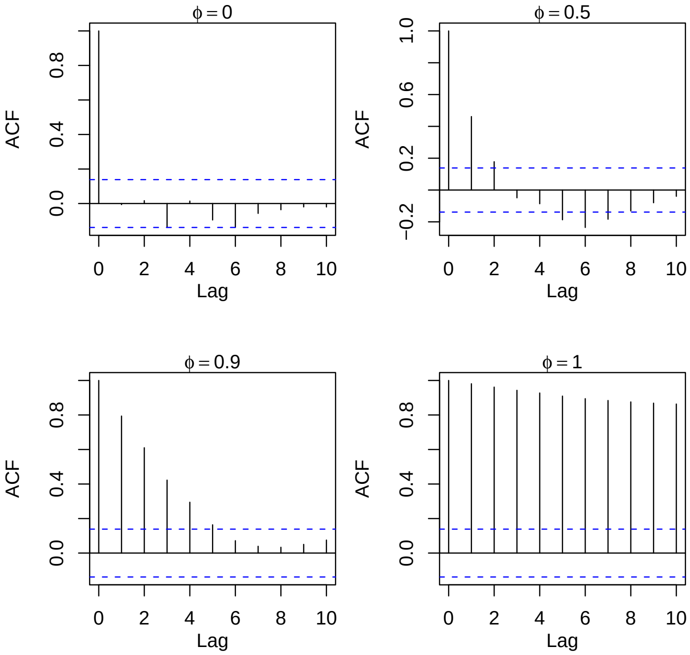

Figure 4.2: Autocorrelation functions for a simulated AR(1) process with different values of ϕ. Shows the different memory of a process.

difference of the left side of Eq.(4.1) it becomes

$$\begin{aligned} r_t - r_{t-1} &= r_{t-1} + a_t - r_{t-1} \\ r_t - r_{t-1} &= a_t \\ \Delta r_t &= a_t \end{aligned}$$

By differencing the unit-root non-stationary series $\{r_t\}$ it becomes a new series $p_t = \Delta r_t$ which is stationary. Removing the aggregation effect and giving the random effect at each time.

□

In the example above the series became stationary after one level of differencing, however this does not always apply.

Definition 4.3 (Integration $I(d)$)

A series which is non-stationary but becomes stationary after d levels of differencing is defined as being ***integrated*** of order d noted as $I(d)$.

◇

The AR(1) series in Example 4.3 is therefore said to be $I(1)$, or integrated of order one.

The terms above are all fundamental definitions in basic time series analysis and econometrics, needed to understand the rather complex nature of the applied house price model inspected in the following chapters. In the following subsection the error correction model (ECM) which is used to model many macro-economic relationships is presented.

4.3 Error-Correction Model (ECM)

For two stationary variables r_t and z_t, where z_t is the response of r_t, e.g. z_t is house prices and r_t is interest rates. Then the following can be assumed:

$$z_t = \delta + \theta z_{t-1} + \phi_0 r_t + \phi_1 r_{t-1} + \varepsilon_t \tag{4.5}$$

If ε_t is assumed white noise independent of $z_{t-1}, z_{t-2}, ...$ and $r_t, r_{t-1}, ...$ then Eq.(4.5) is sometimes known as an ***autoregressive distributed lag model*** (ADL). To estimate the parameters in the model, $(\delta, \theta, \phi_0, \phi_1)$, ordinary least squares (OLS) can be used, see section 4.4.1 for OLS. What is however of more interest is another form of ADL or the so called ***error-correction model*** (ECM). Following is the deduction of the ECM along with a discussion of the model properties, the deduction has been adopted largely from Verbeek [16][3].

By looking at Eq.(4.5) it is seen that z_t is described by lagged values z_{t-1} and by the change in r_t. Taking the partial derivative of z_t, z_{t+1} and z_{t+2} with regards to r_t gives:

$$\begin{aligned}\partial z_t/\partial r_t &= \phi_0\\ \partial z_{t+1}/\partial r_t &= \theta\ \partial z_t/\partial r_t + \phi_1 = \theta\phi_0 + \phi_1\\ \partial z_{t+2}/\partial r_t &= \theta\ \partial z_{t+1}/\partial r_t \quad = \theta(\theta\phi_0 + \phi_1)\end{aligned}$$

Continuing on like this and summing up over $t, t{+}1, t{+}2, ...$ a long run multiplier

[3] See e.g. chapter 9.1.

can be derived or:

$$\begin{aligned}\sum_{a=0}^{\infty}\frac{\partial z_{t+a}}{\partial r_t} &= \phi_0 + (\theta\phi_0 + \phi_1) + \theta(\theta\phi_0 + \phi_1) + \cdots \\ &= \phi_0 + (1 + \theta + \theta^2 + \cdots)(\theta\phi_0 + \phi_1) \\ &= \frac{\phi_0 + \phi_1}{1 - \theta} \quad \text{where} \quad |\theta| < 1 \end{aligned} \tag{4.6}$$

The long run multiplier described by Eq.(4.6) was gotten by using the geometrical series in Eq.(4.4). The relation in Eq.(4.6) therefore describes the long term change in z_t for a change in r_t.

There is another way of writing the ADL model described in Eq.(4.5), by subtracting z_t from both sides in Eq.(4.5) it becomes

$$\Delta z_t = \delta - (1-\theta)z_{t-1} + \phi_0\Delta r_t + (\phi_0 + \phi_1)r_{t-1} + \epsilon_t$$

or as the error-correction model (ECM)

$$\Delta z_t = \phi_0\Delta r_t - (1-\theta)[z_{t-1} - \alpha - \gamma r_{t-1}] + \varepsilon_t \tag{4.7}$$

where

$$\gamma = \frac{\phi_0 + \phi_1}{1-\theta} \qquad \text{and} \qquad \alpha = \frac{\delta}{1-\theta}$$

Eq.(4.7) has two main terms. The first term, i.e. the dynamic part is described by $\phi_0\Delta r_t$. The second term, known as the error correction term, includes the levels inside the brackets, i.e. the actual levels not the differenced values. The terms inside the bracket maintain the long run equilibrium for z_t. The ECM implies that z_t is decided by the change in r_t adjusted by the error correction term in the bracket, which speed of correction is controlled by $(1-\theta)$.

In subsection 5.6.3 the long run multiplier is applied to the house price model to derive what effect a small change in the variables, corresponding to r_t here, have on the response variable, z_t, in the long run.

4.4 Parameter Estimation

Given data and having prepared a model for the data, the model coefficients, or parameters, are estimated so the model describes, fits, the data as well as possible. There are different ways of performing parameter estimation. In this section two of the main methods, Ordinary Least Squares (OLS) and Maximum Likelihood Estimation (ML), are discussed in subsections 4.4.1 and 4.4.3

respectively. In subsection 4.4.2 a special case of OLS is described where linear constraints are implemented on the coefficients (ROLS). The derivation of the estimator for ROLS is largely borrowed from Judge *et. al.* [6].

Definition 4.4 (Linear Regression Model)
The ***linear regression model***, in matrix form, is expressed as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{4.8}$$

where

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \qquad \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix} \quad \text{and} \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

where $\boldsymbol{y}$ is a $(n \times 1)$ vector of ***observations*** also sometimes noted as the ***response variable***, $\boldsymbol{X}$ is $(n \times p)$ matrix of levels of the independent variables also noted as the ***design***- or ***explanatory*** matrix, where $p = k + 1$ i.e. the number of regressors k plus the intercept (β_0). The $(p \times 1)$ vector $\boldsymbol{\beta}$ holds the ***regression coefficients*** and $\boldsymbol{\varepsilon}$ is an $(n \times 1)$ vector of random errors, white noise.

◇

4.4.1 Ordinary Least Squares (OLS)

Isolating the error term from Eq.(4.8) it can be rewritten as

$$\boldsymbol{\varepsilon} = \boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}$$

A vector of least square estimators $\hat{\boldsymbol{\beta}}$ is sought so as it minimizes the following function $S(\boldsymbol{\beta})$

$$S(\boldsymbol{\beta}) \;= \sum_{i=1}^{n} \varepsilon_t^2 = \boldsymbol{\varepsilon}'\boldsymbol{\varepsilon} = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) \tag{4.9}$$

Where prime $(')$ indicates the transpose of a vector or matrix. By multiplying the matrices in the brackets, keeping in mind the fundamental matrix rule of

$(\boldsymbol{AB})' = \boldsymbol{B}'\boldsymbol{A}'$, Eq.(4.9) becomes

$$\begin{aligned} S(\boldsymbol{\beta}) &= \boldsymbol{y}'\boldsymbol{y} - \boldsymbol{y}'\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} \qquad (4.10) \\ &= \boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} \qquad (4.11) \end{aligned}$$

The step between Eq.(4.10) and Eq.(4.11) is explained by

$$\boldsymbol{y}'\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{y}'(\boldsymbol{X}')'(\boldsymbol{\beta}')' = \boldsymbol{y}'(\boldsymbol{\beta}'\boldsymbol{X}')' = (\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y})'$$

and the fact that the term $\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y}$ is a scalar as can be seen by

$$1 \times n \cdot p \times n \cdot n \times 1 = 1 \times n \cdot n \times 1 = 1$$

Taking the derivative of Eq.(4.11) with regards to $\boldsymbol{\beta}$ gives

$$\begin{aligned} \frac{\partial S}{\partial \boldsymbol{\beta}} &= \frac{\partial}{\partial \boldsymbol{\beta}}(\boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta}) \\ &= \frac{\partial}{\partial \boldsymbol{\beta}}(-2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta}) \\ &= -2\boldsymbol{X}'\boldsymbol{y} + 2\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} \end{aligned}$$

Setting the derivative $\partial S/\partial \boldsymbol{\beta}$ equal to zero, inserting $\boldsymbol{\beta} = \hat{\boldsymbol{\beta}}$ and solveing for $\hat{\boldsymbol{\beta}}$

$$\begin{aligned} \left.\frac{\partial S}{\partial \boldsymbol{\beta}}\right|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}} &= -2\boldsymbol{X}'\boldsymbol{y} + 2\boldsymbol{X}'\boldsymbol{X}\hat{\boldsymbol{\beta}} = 0 \\ \boldsymbol{X}'\boldsymbol{X}\hat{\boldsymbol{\beta}} &= \boldsymbol{X}'\boldsymbol{y} \\ (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{X}\hat{\boldsymbol{\beta}} &= (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y} \\ \hat{\boldsymbol{\beta}} &= (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y} \qquad (4.12) \end{aligned}$$

Eq.(4.12) is the ***ordinary least square estimator*** (OLS) of $\boldsymbol{\beta}$, i.e. $\hat{\boldsymbol{\beta}}$ holds the estimated coefficients to each of the factors in the relationship between $\boldsymbol{X}$ and $\boldsymbol{y}$, minimizing the second norm of the estimated standard error. An example of estimation of parameters by use of OLS in a economic relationship is shown in Example 4.4.

While the OLS method is easy to use and effective it is not as general as the Maximum Likelihood method mentioned in subsection 4.4.3. Furthermore OLS works only for problem that can be written on the regression model format.

Example 4.4 (Example of OLS)
Imagine a typical economic relationship of the following form

$$Q_t = AL_t^{\alpha}K_t^{\gamma}e^{\varepsilon_t}$$

where Q_t is output, L_t is labor, K_t is capital, A is some constant and ε_t is the error term, independent of K_t and L_t over the time period $t \in \{1, ..., n\}$. The parameters that are to be estimated are γ and α. Taking the logarithm (ln) of Q_t gives

$$\ln(Q_t) = \ln(A) + \alpha \ln(L_t) + \gamma \ln(K_t) + \varepsilon_t$$

It is easy to see that this relation can be transformed to the regression format as

$$y_t = \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} + \varepsilon_t$$

or in matrix form corresponding to Eq.(4.8) as

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$$

where

$$\boldsymbol{y} = \begin{bmatrix} \ln(Q_1) \\ \ln(Q_2) \\ \vdots \\ \ln(Q_n) \end{bmatrix}, \qquad \boldsymbol{X} = [\, \boldsymbol{I},\ \boldsymbol{x_{t1}},\ \boldsymbol{x_{t2}}] = \begin{bmatrix} 1 & \ln(L_1) & \ln(K_1) \\ 1 & \ln(L_2) & \ln(K_2) \\ \vdots & \vdots & \vdots \\ 1 & \ln(L_n) & \ln(K_n) \end{bmatrix}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \ln(A) \\ \alpha \\ \gamma \end{bmatrix} \qquad \text{and} \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}$$

and can be solved for $\hat{\boldsymbol{\beta}}$ by using Eq.(4.12).

□

4.4.2 Restricted Least Squares (ROLS)

In this subsection a special case of OLS is discussed. When a linear constraint, one or more, has been imposed on the coefficients in the $\boldsymbol{\beta}$ vector the ***Restricted Ordinary Least Squares*** (ROLS) method is used for estimating $\boldsymbol{\beta}$.

The objective function $S(\boldsymbol{\beta})$ given in Eq.(4.9) is the same except now it must be solved subject to the constraints presented as

$$\boldsymbol{R\beta} = \boldsymbol{r} \tag{4.13}$$

Where $\boldsymbol{R}$ is a $(q \times p)$ matrix, where p is the number of parameters, while q is the number of constraints, $\boldsymbol{r}$ is a $(q \times 1)$ vector of scalars. A coefficient vector $\hat{\boldsymbol{\beta}}^*$ is sought so as to minimizes $S(\boldsymbol{\beta})$, in Eq.(4.9), subject to the constraints imposed on $\boldsymbol{\beta}$ expressed in Eq.(4.13).

If the constraints in Eq.(4.13) are linear a Lagrange optimization process may be applied such that

$$\begin{aligned} L(\boldsymbol{\beta}, \boldsymbol{\lambda}) &= \boldsymbol{e}'\boldsymbol{e} - \boldsymbol{\lambda}'(\boldsymbol{r} - \boldsymbol{R}\boldsymbol{\beta}) \\ &= \boldsymbol{y}'\boldsymbol{y} - 2\boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{y} + \boldsymbol{\beta}'\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{\lambda}'(\boldsymbol{r} - \boldsymbol{R}\boldsymbol{\beta}) \end{aligned} \tag{4.14}$$

Where the Lagrangian multiplier $\boldsymbol{\lambda}$ is a $(q \times 1)$ vector. The derivative of Eq.(4.14) w.r.t. $\boldsymbol{\beta}$ and $\boldsymbol{\lambda}$ is taken, and set to 0, to find the optimal value of $\boldsymbol{\beta}$

$$L' = \begin{cases} \left.\dfrac{\partial L}{\partial \boldsymbol{\beta}}\right|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}^*, \boldsymbol{\lambda}=\boldsymbol{\lambda}^*} = -2\boldsymbol{X}'\boldsymbol{y} + 2\boldsymbol{X}'\boldsymbol{X}\hat{\boldsymbol{\beta}}^* + \boldsymbol{R}'\boldsymbol{\lambda}^* = 0 & (i) \\ \\ \left.\dfrac{\partial L}{\partial \boldsymbol{\lambda}}\right|_{\boldsymbol{\beta}=\hat{\boldsymbol{\beta}}^*, \boldsymbol{\lambda}=\boldsymbol{\lambda}^*} = -\boldsymbol{r} + \boldsymbol{R}\hat{\boldsymbol{\beta}}^* = 0 & (ii) \end{cases} \tag{4.15}$$

Using (i) and (ii) to solve for $\boldsymbol{\lambda}^*$ it can be seen that

$$\boldsymbol{\lambda}^* = -2(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}(\boldsymbol{r} - \boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y})$$

or if using the OLS result $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$

$$\boldsymbol{\lambda}^* = -2(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}(\boldsymbol{r} - \boldsymbol{R}\hat{\boldsymbol{\beta}}) \tag{4.16}$$

Combining Eq.(4.15) (i) and Eq.(4.16) and solving for $\hat{\boldsymbol{\beta}}^*$ gives

$$\hat{\boldsymbol{\beta}}^* = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y} + (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}'(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}(\boldsymbol{r} - \boldsymbol{R}\hat{\boldsymbol{\beta}}) \tag{4.17}$$

Or finally by using the OLS result again it becomes

$$\hat{\boldsymbol{\beta}}^* = \hat{\boldsymbol{\beta}} + (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}'(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}(\boldsymbol{r} - \boldsymbol{R}\hat{\boldsymbol{\beta}}) \tag{4.18}$$

which is the ***restricted ordinary least squares estimator*** giving the estimated values of $\boldsymbol{\beta}^*$.

Example 4.5 (Re-Parameterizations vs. ROLS)
Recall the regression model from Example 4.4, i.e.

$$\ln(Q_t) = \ln(A) + \alpha \ln(L_t) + \gamma \ln(K_t) + \varepsilon_t$$

Imagine now there exists a relationship between L and K, i.e. if both K and L increase with e.g. 10% then so will Q, (this is known in economics as a Cobb-Douglas function). This relation is equivalent to the constraint $\alpha + \gamma = 1$. Since this linear constraint is not very complex there is a re-parametrization alternative to the ROLS method.

Using re-parameterizations ξ: Using ξ instead of β. The constraint can be expressed as $\gamma = 1 - \alpha$ giving a new regression model as

$$\begin{aligned} \ln(Q_t) &= ln(A) + \alpha \ln(L_t) + (1-\alpha)\ln(K_t) + \varepsilon_t \\ \ln(Q_t) - \ln(K_t) &= ln(A) + \alpha(\ln(L_t) - \ln(K_t)) + \varepsilon_t \end{aligned}$$

which can be expressed as

$$y_t = \xi_0 + \xi_1 x_{t1} + \varepsilon_t$$

where

$$\boldsymbol{y} = \begin{bmatrix} \ln(Q_1) - \ln(K_1) \\ \ln(Q_2) - \ln(K_2) \\ \vdots \\ \ln(Q_n) - \ln(K_n) \end{bmatrix}, \qquad \boldsymbol{X} = [\, \boldsymbol{I},\ \boldsymbol{x_{t1}}] = \begin{bmatrix} 1 & \ln(L_1) - \ln(K_1) \\ 1 & \ln(L_2) - \ln(K_2) \\ \vdots & \vdots \\ 1 & \ln(L_n) - \ln(K_n) \end{bmatrix}$$

$$\boldsymbol{\xi} = \begin{bmatrix} \ln(A) \\ \alpha \end{bmatrix} \qquad \text{and} \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{bmatrix}.$$

Solve $\hat{\boldsymbol{\xi}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}$ where $E[\boldsymbol{\xi}] = \hat{\boldsymbol{\xi}}$.

Using ROLS β^*: Since there is only one constraint $\boldsymbol{R}$ is a $(1 \times p)$ vector, $p = 3$, and r only a scalar. The constraint equation Eq.(4.13), $\boldsymbol{R\beta} = r$, becomes

$$\begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \ln(A) \\ \alpha \\ \gamma \end{bmatrix} = 1$$

and can then be solved for $\hat{\boldsymbol{\beta}}$ by Eq.(4.18)

$$\hat{\boldsymbol{\beta}}^* = \hat{\boldsymbol{\beta}} + (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}'(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}(\boldsymbol{r} - \boldsymbol{R}\hat{\boldsymbol{\beta}})$$

□

In Example 4.5 it can be seen that the re-parametrization method is much easier to handle for one constraint. However, for a higher number of constraints (q), regression coefficients (p) or both, the re-paramiterization method quickly becomes difficult to implement while the ROLS method with the matrix representation is consistent in implementation.

4.4.3 Maximum Likelihood (ML)

Maximum likelihood (ML) estimation is a more general method of parameter estimation than that of OLS. The downside to using ML is that it can be complicated to derive the so called Likelihood function which is optimized for the estimated parameters. ML can be used to solve for coefficient in very complicated relations, using numerical optimization methods.

Maximum likelihood estimation was not used in this thesis but represent an interesting alternative to the OLS and ROLS methods and therefore warrants mentioning. For more information on ML estimation see Madsen [9][4], for an introduction, and Hamilton [3][5] for a more advanced treatment, including optimization methods.

4.5 Properties of the OLS and ROLS Estimators

Given an estimated $\hat{\boldsymbol{\beta}}$ coefficient, the ***fitted data*** ($\hat{\boldsymbol{y}}$) can be expressed as

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}} \tag{4.19}$$

The ***residual*** ($\boldsymbol{e}$), i.e. the difference between the fitted data and the observed data is denoted as

$$\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{y}} \tag{4.20}$$

it can be seen that if $E[\boldsymbol{\varepsilon}] = \boldsymbol{e}$ then $E[\boldsymbol{\beta}] = \hat{\boldsymbol{\beta}}$ so the condition that the residual behave like $\boldsymbol{\varepsilon}$, i.e. white noise, is crucial if $\hat{\boldsymbol{\beta}}$ is to be a correct estimation of $\boldsymbol{\beta}$. See subsection 5.5.1 for more on residual analysis.

The variance of the residual is often called the ***error*** or ***residual sum of squares*** (σ^2), it has $n-p$ number of degrees of freedom, where n represents the number of observations as before and p is the number of regression coefficients plus the intercept, as before. The σ^2 is estimated by

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)}{n-p} = \frac{\boldsymbol{e}'\boldsymbol{e}}{n-p} \tag{4.21}$$

The ***covariance matrix*** is a symmetric matrix representing the variance between different regression coefficients $\hat{\beta}_i$ and $\hat{\beta}_j$ at the (ij) and (ji) elements in

[4] section 2.2.2

[5] Chapter 5

the matrix. The diagonal, of the covariance represent the variance of estimated regressor β_{ii} where $1 \leq i \leq p$. The covariance matrix for OLS is expressed as

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}} = \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1} \tag{4.22}$$

The covariance matrix for the restricted case ROLS is

$$\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}^*} = \sigma^2\boldsymbol{M}^*(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{M}^{*\prime} \tag{4.23}$$

where

$$\boldsymbol{M}^* = \boldsymbol{I} - (\mathbf{X}'\boldsymbol{X})^{-1}\boldsymbol{R}'(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}\boldsymbol{R}$$

The proof for the OLS covariance matrix can be seen in Madsen [9][6]. The ROLS covariance matrix, which is more involved, can be found in Judge *et. al* [6][7].

4.6 Goodness of Fit

The ***Goodness of fit*** is a measurement of how well the fitted data using the estimated coefficients $\hat{\boldsymbol{\beta}}$ manage to represent the data. One measurement of goodness of fit is R^2 or ***R-squared*** calculated as follows

$$R^2 = \frac{\displaystyle\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\displaystyle\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y}$, also know as the sample mean, is calculated as $\bar{y} = (\sum_{i=1}^{n} y_i)/n$. The goodness of fit estimator R^2 gives a value in the interval $0 \leq R^2 \leq 1$, where 0 and 1 represent no and perfect correlation between the fitted data and the observed data, respectively.

The R^2 statistic is however biased to the number of regressors, i.e. the fit will become better as the number of regressors is increased. therefore another way of calculating the fit is R^2_{adj} ***adjusted R square*** which adjusts the statistic for the number of regressors used by taking p the number of regressor into account.

$$R^2_{adj} = 1 - \left(\frac{n-1}{n-p}\right)(1 - R^2)$$

The R^2 is not without fault and must by used with care and is not to be used as the only measure of goodness of fit or validation. For example R^2 will converge to one for a fit of an unit-root non-stationary processes, modeled directly, giving a good fit but useless parameters for forecasting.

[6] See page 35.

[7] See pages 238-239.

Chapter 5

House Price Dynamics II The MONA model

5.1 Introduction

In this chapter an actual house price model will be inspected, duplicated and used for prediction. The model under inspection is the house price relation from ***MONA-a quarterly model of the Danish economy*** [12], or the MONA model as it will be referred to here after. The MONA model was developed by Danmarks central bank, the Nationalbank, as a macro-economic model to forecast numerous economic relations and parameters. One of the many things the MONA model looks at is the development of house prices in Denmark. The idea behind macro models like MONA is to get a complete picture of how the economy works.

In section 5.2 a discussion of how the model is conceived is given, as well as listing a few of the well known elements and relationships that influence house prices. Section 5.3 describes the data used in the house price model, as well as giving an example of how the non-stationarity of the data can be handled. Section 5.4 deals with the modeling aspects of the relation from theory to application, the constraints in the model are also explained. In section 5.5 the results for the parameter estimation are presented, as well as the residual analysis for the fit is conducted. Section 5.6 focuses on how to use the model for prediction, as well as giving a short discussion of how general the MONA house price model results are and finally estimating the long term coefficients in the error-correction model format.

5.2 The MONA Model Background

On pages 41 to 52, in the MONA model [12], a relation for the Danish housing market is presented. The MONA house price relation is derived by using a theoretical model as a basis, while adding more elements where deemed necessary by the analysis of house price data.

Much like the model presented in section 2.2 the MONA house market model is split up into two parts. The first part is a house price relation which is the same as the demand side in section 2.2. The second part is a model of residential investment, equivalent to the supply side in section 2.2. As in the theoretical model the supply flow, in the MONA model, is controlled by the ratio between house prices and construction cost, also known as Tobin´s Q, or:

> *"On a fall in interest rates both house prices and housing construction go up, and the expanded supply of housing gradually forces house prices back towards equilibrium where they correspond to construction costs."*[1]

Much like the theoretical relation given in section 2.2 the main factors for house price development in the MONA model are considered to be interest rates, income and stock of houses.

Using data from the Danish economy from 1971 to 2001 it can be seen how interest rates, house prices, stock of houses and income have progressed. A graphical representation of the relationship between interest rates and house prices can be seen in Figure 5.1. The relationship between negative change in interest rates has been slowed down to show yearly change instead of quarterly change, i.e. the processes have been differenced 4 time periods to show correlation better graphically. The one period differenced correlation is $\rho = 0.6334$, where $-1 \leq \rho \leq 1$, one being completely positively correlated, minus one being completely negatively correlated and 0 showing no correlation. Another fundamental relationship between house prices, income and stock of houses is displayed in Figure 5.2. A ratio between income and stock of houses is calcu-

[1] *page 42, MONA-a quarterly model of the Danish economy* [12]

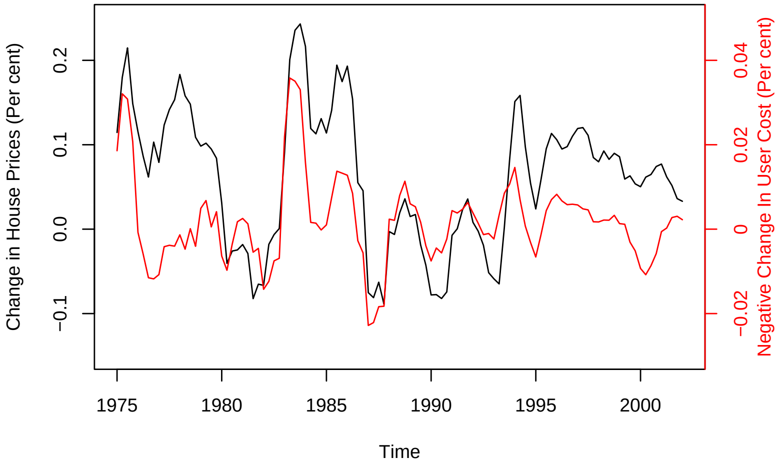

Figure 5.1: Shows the correlation between negative change in interest rates (red, right axis) and change in house prices (black). The data is differenced 4 periods to show the change better visually.

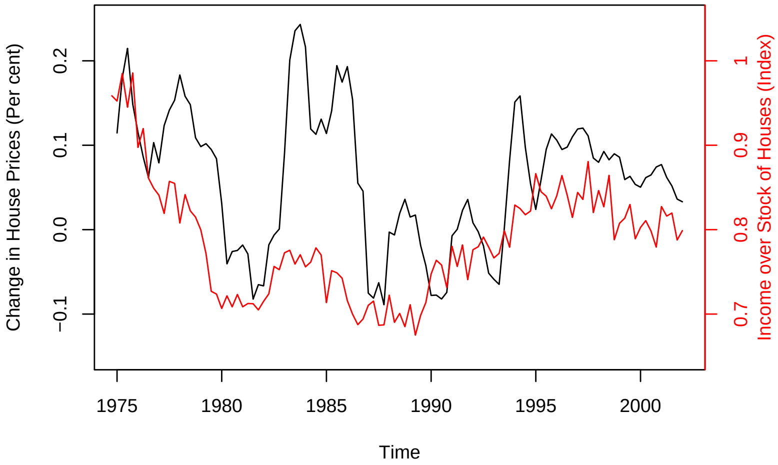

Figure 5.2: Shows the correlation between real disposable income over stock of house(red, right axis) against change in house prices (black).

lated and plotted against change in house prices. The correlation between these two time series is $\rho = 0.4095$.

Figure 5.1 shows that there is clearly a negative correlation between changes in interest rates and change in house prices. Figure 5.2 shows on the other hand that there is also a correlation between change in house prices and income as a ratio of stock of houses. What is more, Figure 5.2 shows that a high income ratio is usually followed by increases in house prices.

By inspecting the data as above, along with knowing in theory which are the main factors in house price modeling, the National Bank of Denmark has created an applied house price model whose derivation and assumptions are listed in the next sections.

5.3 The Data

This section is divided into two parts, firstly the data used is presented, giving a short description for each component. Secondly an example of how the series are analyzed from a time series point of view is shown.

5.3.1 Description of Data

Following is a listing of the components used in the house price model, for comparison the theoretical house price model, Eq.(2.3) from section 2.1 is repeated as

$$\frac{PH}{P} = \theta\left(\frac{H^D}{YD}, \; R, \; \frac{WA}{YD}, \; \frac{D}{YD}\right)$$

The data used in the MONA model is as follows

$\{kp_t\}$: This term describes the house price at time t, in Eq.(2.3) this is equivalent to PH.

$\{rente_t\}$: This is the interest rate term at time t, i.e. bond yield after tax.

$\{ssats_t\}$: The corresponding tax term for the bond yield term $rente_t$ at time t.

$\{pcp_t\}$: This is the level of the consumption deflator at time t recall the definition for consumption deflator in subsection 2.2.1, also the pcp term is the same as P is in Eq.(2.3).

$\{ipv_t\}$: This series represents the private investment at time t.

$\{ypd_t\}$: Private disposable income at time t.

$\{fwh_t\}$: This is the stock of houses at t which is noted as $H = H^D = H^S$ in Eq.(2.3).

$\{dkpe_t\}$: The expected increase in house price from t to $t-1$.

$\{dpcpe_t\}$: The expected increase in private deflator from t to $t-1$.

The added terms $rente_t + ssats_t$, i.e. interest rate plus tax rate, are noted as ***user cost*** and also referred to as $\{ibv_t\}$ in the MONA model. All these variables are observed changes except for the last two $(dkpe, dpcpe)$ which are internal variables to the MONA model, i.e. they are estimated with other relations at another place in the model[2].

The data is available quarter-yearly from 1971-2002, however not all data is available in this period and because of lagged data and differencing the so-called ***in-sample period***, also known as training period and off-line period, i.e. the period where the models parameters are estimated, is from 1974:$q1$ to 1997:$q4$. The ***out-of-sample period***, also known as the on-line period, used for validation and prediction, is from 1997:$q4$ to 2001:$q4$.

A quick inspection of the level plots along with the autocorrelation functions reveals that the processes shows signs of unit-root non-stationarity, i.e. a high correlation to lagged values. The next section shows an example of how to address the unit-root issue for the response series i.e. kp_t (house prices).

5.3.2 House Price Data

As can be seen from e.g. Eq.(2.1) a detailed house price model can include many elements. Although many series are also used in the MONA model only one will be shown here in detail i.e. the house price series $\{kp_t\}$ while similar methods were applied to the other series when modeling the MONA model.

The $\ln(kp_t)$ series is depicted in Figure 5.3 (a), along with the corresponding auto correlation function in (c). From the two graphs it can be seen how highly correlated the present values are to lagged values. The two panels show that the process has a long memory, which can indicate a unit-root behavior or ***trend stationarity***, which is when, using the AR(1) case for example, a constant has been added giving

$$r_t = \mu t + \theta r_{t-1} + a_t$$

where a_t is white noise and μt is a constant having a drift effect on the model. The drift effect can be estimated via OLS and removed to give the underlying

[2] see MONA [12] Page 196 and 197 for the estimation of *dkpe* and *dpcpe*.

process. The MONA report however uses the method of differencing, thereby removing the accumulation of values and modeling the change $\Delta \ln(kp_t)$ instead of the level kp_t.

In Figure 5.3 (b) the one period change in the $\ln(kp)$ series, i.e. $\Delta ln(kp)$, is displayed. Figure 5.3 (d) shows the autocorrelation function for the differenced series. It is obvious how much the memory of the process has been decreased by only one differencing.

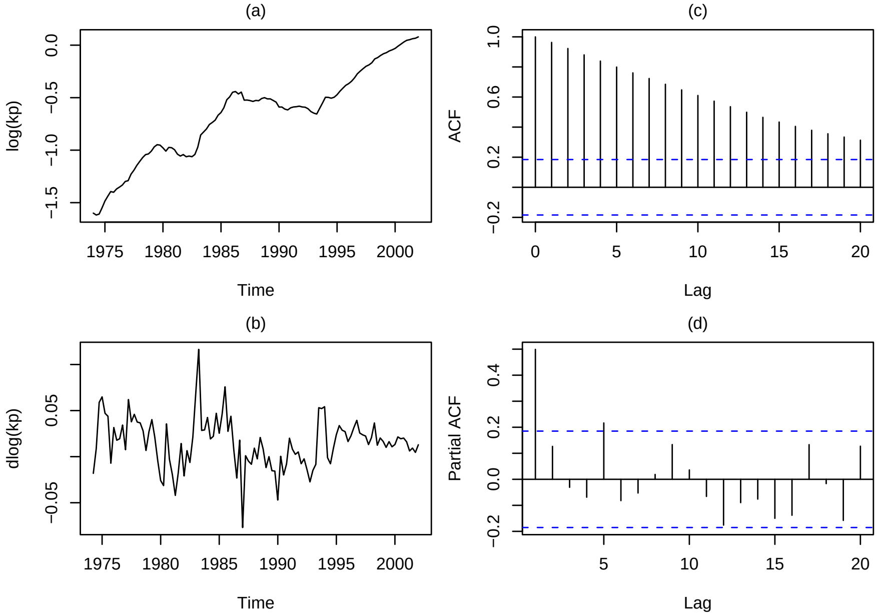

Figure 5.3: Log series of house prices (kp) from 1974:q1-2002:q1 : (a) log(kp), (b) time plot of the first differenced series log(kp) (c) sample auto correlation function for the log(kp) series, and (d) the sample partial auto correlation function for the differenced series.

A more accurate way of locating unit-roots, other than differencing once and viewing ACF plots, is by use of so-called ***Augmented Dicky Fuller***[3] tests (ADF) which test whether a series is dependant on previous values with $\phi = 1$, i.e. if it has a unit-root, for more details of ADF see Tsay [15][4].

Using the statistical software package R it can be seen that the test for unit-root in $\ln(kp)$ by the ADF method gives a Dickey-Fuller value = 1.6864 and ***p-value***

[3] See e.g. the function `adfTest()` in package {`fMultivar`} in R.

[4] see e.g. chapter 2.7.

$= 0.9768$, the p-value indicates that the hypothesis presented, in this case that there is a unit-root, can be rejected with approx 2.3% probability, i.e. it can not be rejected. If the series is differenced once the Dickey-Fuller value is -2.3078 with a p-value $= 0.02214$ indicating that the hypothesis of a unit-root can be rejected with aproximately 98% probability, therefore it can be said that $ln(kp_t)$ is $I(1)$, i.e. integrated of level one. Since there may be a unit-root in the levels $(\ln(kp_t))$ the first differenced levels $(\Delta \ln(kp))$ are modeled, the transformation back to $\ln(kp_t)$ is performed by

$$\ln(kp_t) = \Delta \ln(kp_t) + \ln(kp_{t-1}) \tag{5.1}$$

Further discussion will be given on the aggregation of the modeled differences in section 6.3.

5.4 The Model

This section focuses on numerous practical and theoretical items needed to understand and use the MONA house price relation. In subsection 5.4.1 the theoretical model is stated and derived to an initial regression format, along with some discussion of the constraints used in the model. The following subsection summarizes the model components, or explanatory variables, used to evaluate the models coefficients and presents the regression form of the model. Lastly the applied form of the constraint is presented in format suitable for solving with ROLS.

5.4.1 The Theoretical Model

Recall the house price relation presented in subsection 2.2.1 where the stock of houses can be expressed as

$$H^D = f\left(\frac{PH}{P},\ R,\ YD,\ WA,\ D\right) \tag{5.2}$$

Similar to this relation the theoretical relationship for long term house price development in the MONA[5] model is derived from the knowledge that the main factors are income, interest rates and stock of houses. A long term demand relation for the stock of houses in MONA is presented as

$$\ln(\text{stock of houses}) = \ln(\text{income}) - a \cdot \ln\left(\frac{\text{user cost}}{\text{consumer price}}\right) \tag{5.3}$$

[5] See the MONA model page 43.

It can be seen that the two relations have certain elements in common, although this form of Eq.(5.3) has fewer terms than Eq.(5.2) and seems more simple. The first mutual factor is wanted stock of houses (H^D) which is the same as the observed fwh or H. Other mutual elements are income (YD), user cost (R) and a price element (PH/P).

By rearranging the terms in Eq.(5.3) the relation becomes

$$\ln(\text{income}) - \ln(\text{stock of houses}) = a \cdot \ln\left(\frac{\text{user cost}}{\text{consumer price}}\right) \tag{5.4}$$

On the left side the stock of houses and income, using the MONA variables described in subsection 5.3.1, become

$$\begin{aligned} &= \ln(\text{income}) - \ln(\text{stock of houses}) \\ &= \ln((ypd - ipv)/pcp) - \ln(fwh) \end{aligned} \tag{5.5}$$

where income has been modeled as real income, i.e. *ydp* the private disposable gross income minus *ipv* the private investment will give the net income, and dividing by the consumption deflator *pcp* adjusts the value to the current period, giving real income.

It can be seen on the right side of Eq.(5.4) that the terms user cost and consumer price can be approximately expanded as follows, using the variables described in subsection 5.3.1

$$\begin{aligned} &= a \cdot \ln\left(\frac{\text{usercost}}{\text{consumerprice}}\right) \\ &\approx a_0 + a_1 \ln\left(\frac{kp}{pcp}\right) + a_2 \cdot (rente + ssats - infl) \\ &= a_0 + a_1 \ln\left(\frac{kp}{pcp}\right) + a_2 \cdot (rente + ssats) - a_2 \cdot infl \end{aligned} \tag{5.6}$$

In the first step an approximation is made so that the user cost divided by consumer price becomes real house price and real user cost, real user cost is user cost plus inflation ($infl$). In Eq.(5.6) the inflation term of the real user cost rate has been isolated. Next a relation is derived to simulate the inflation term, it is comprised of the elements that reflect the price increase

$$-a_2 \cdot infl \approx \left[a_3 \Delta \ln(pcp) + a_4 dpcpe + a_5 dkpe + a_6 \Delta \ln(kp)\right] \tag{5.7}$$

Inflation is therefore represented by four price changes. The change in consumption deflator from the last period ($\Delta \ln(pcp_{t-1})$), the expected change in consumption deflator from the last period ($dpcpe_{t-1}$), expected change in house prices from last period ($dkpe_{t-1}$) and the change in house prices from last period ($\Delta\ln(kp_t)$). This constraint is meant to ensure a real interest rate behavior, which is achieved by connecting the user cost coefficient a_2 to the weighing of the coefficients used in the estimation of the inflation. The constraint ensures that if there is a price increase of one percent it will result in a one percent fall in interest rates after tax, in the long run. The price coefficient constraint will be given more discussion in section 5.5.

Combining Eq.(5.5), Eq.(5.6) and Eq.(5.7) and isolating the house price term from the inflation constraint gives

$$\begin{aligned} a_6\Delta\ln(kp) = &-\Bigg(a_0 + a_1\ln\left(\frac{kp}{pcp}\right) + a_2(rente + ssats) \\ &- (a_3\Delta\ln(pcp) + a_4 dpcpe + a_5 dkpe)\Bigg) \\ &+ \ln((ydp - ipv)/pcp) - \ln(fwh) \end{aligned}$$

which when dividing through with a_6 becomes

$$\begin{aligned} \Delta\ln(kp) = &-\frac{a_0}{a_6} - \frac{a_1}{a_6}\ln\left(\frac{kp}{pcp}\right) - \frac{a_2}{a_6}(rente + ssats) + \frac{a_3}{a_6}\Delta\ln(pcp) \\ &+ \frac{a_4}{a_6}dpcpe + \frac{a_5}{a_6}dkpe + \frac{1}{a_6}\Bigg(\ln((ydp - ipv)/pcp) - \ln(fwh)\Bigg) \end{aligned} \tag{5.8}$$

This theoretical relation is then fitted to the available data by statistical analysis, i.e. using lagged values, including differenced values and levels where significant, resulting in a specific model which is described in the next subsection.

5.4.2 MONA Model Components

Recall the regression model in Definition 4.4 i.e.

$$\boldsymbol{y} = \boldsymbol{X\beta} + \boldsymbol{\varepsilon}$$

The response variable $\boldsymbol{y}$ and the column $\boldsymbol{x}_i$ of the explanatory matrix $\boldsymbol{X}$ where $(n \times p)$ and $1 \le i \le p$ are expressed as

$\boldsymbol{y}$: $\Delta\ln(kp)$

The modeled relation is changed from modeling house prices, or kp, to modeling the one period change in the log of house prices, or $\Delta\ln(kp)$ to see why this is done see 5.3.2.

$\boldsymbol{x}_1$: $\Delta\ln(pcp)$

Change in the consumption deflator.

$\boldsymbol{x}_2$ **:** $\Delta(rente + ssats)$
First differenced series of interest plus tax, i.e. user-cost change.

$\boldsymbol{x}_3$ **:** $\Delta(rente_{-1} + ssats_{-1})$
Lagged first differenced series of interest plus tax, i.e. lagged user-cost change.

$\boldsymbol{x}_4$ **:** $rente_{-1} + ssats_{-1} + 0.01$
Lagged user cost plus a logarithmic element (0.01). Interest rate plus tax element.

$\boldsymbol{x}_5$ **:** $dpcpe_{-1}$
Expected change in consumption, from last period, i.e lagged.

$\boldsymbol{x}_6$ **:** $dkpe_{-1}$
Expected change in house price, lagged.

$\boldsymbol{x}_7$ **:** $\ln(kp_{-1}/pcp_{-1})$
Real house price, i.e. house prices lagged adjusted with the lagged consumption deflator.

$\boldsymbol{x}_8$ **:** $\ln((ydp_{-1} - ipv_{-1})/pcp_{-1}) - \ln(fwh_{-1})$
Income elasticity constraint to stock of houses achieved by modeling together with only one regressor.

The new applied model can then be expressed as a regression model as follows

$$\begin{aligned}\Delta\ln(kp_t) = {} & \beta_0 + \beta_1\Delta\ln(pcp_t) + \beta_2\Delta(rente_t + ssats_t) + \beta_3\Delta(rente_{t-1} + ssats_{t-1}) \\ & + \beta_4(rente_{t-1} + ssats_{t-1} + 0.01) + \beta_5 dpcpe_{t-1} + \beta_6 dkpe_{t-1} \\ & + \beta_7\ln(kp_{t-1}/pcp_{t-1}) + \beta_8(\ln((ydp_{t-1} - ipv_{t-1})/pcp_{t-1}) - \ln(fwh_{t-1})) + \varepsilon_t\end{aligned} \tag{5.9}$$

The coefficients $\boldsymbol{\beta}$ have replaced the a coefficients and need to be estimated by the restricted least squares method since there is a constriction on their estimation.

The Constraints

In the MONA house price relation two constraints are applied. Firstly there is a constraint implemented by re-parameterization by modeling stock of houses and real income together, i.e. their ratio has only one regressor and will therefore always affect the price by the same weight.

The second constraint is not as easily implemented and requires the use of the restricted ordinary least squares method for the parameter estimation. Recall the inflation constraint modeled above to assure real interest rate behavior as

$$-a_2 \cdot infl \approx \Big[a_3 \Delta \ln(pcp) + a_4 dpcpe + a_5 dkpe + a_6 \Delta \ln(kp) \Big]$$

Now the theoretical a coefficients have been replaced by the β coefficient in the applied model. Where the corresponding β coefficient to the previous a coefficient can be found by comparing explanatory components $\boldsymbol{x}$ e.g. the previous a_2 coefficient to $(rente + ssats)$ is now β_4 the applied coefficient to $(rente_{t-1} + ssats_{t-1} + 0.01)$. The constraint represented with β coefficients is therefore

$$-\beta_4 = \frac{\beta_1}{4} + \beta_5 + \beta_6 - \frac{1}{4}$$

Where the scalar $(1/4)$ represents the house price increase quarter-yearly, now house price and consumption deflator changes always go hand in hand therefor β_1, the change in consumption deflator coefficient, is also divided by four to get a quarter-yearly change. The constraint can be used to calculate the expected inflation by dividing through with $-a_2$ and $-\beta_4$ in the theoretical and applied cases, respectively.

The constraint on $\boldsymbol{R\beta} = \boldsymbol{r}$ format for ROLS, is expressed as

$$\begin{array}{ccccccccccccc} \text{Const} & R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 & & & & \boldsymbol{r} \\ [0 & 0.25 & 0 & 0 & 1 & 1 & 1 & 0 & 0] & \cdot & \boldsymbol{\beta} & = & [0.25] \end{array}$$

The optimal coefficients can then be achieved by solving

$$\hat{\boldsymbol{\beta}}^* = \hat{\boldsymbol{\beta}} + (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}'(\boldsymbol{R}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{R}')^{-1}(\boldsymbol{r} - \boldsymbol{R}\hat{\boldsymbol{\beta}}) \tag{5.10}$$

where $\hat{\boldsymbol{\beta}}$ is the unconstrained coefficients estimated by OLS. This relation was derived in subsection 4.4.2.

Degrees of freedom, which are used for many statistical tests and estimators, must be handled with care when using ROLS. Degrees of freedom are usually described as $n - p$ where n is number of observations used in the modeling and $p = k + 1$ is the number of regressors including the constant. By deciding $\hat{\beta}_4^*$ implicitly from other coefficients it does not get a degree of freedom, this will have to be kept in mind when calculating test statistics and goodness of fit for the estimation.

5.5 The Results

Solving Eq.(5.10) with the constraint described above results in estimates of the coefficients displayed in Table 5.1. In the table there are three data columns, the first one shows the estimated value of the regression coefficient. The second column shows the estimated standard error for the coefficients, i.e. is the square root of the diagonal of the covariance matrix $\mathbf{\Sigma}_{\hat{\boldsymbol{\beta}}^*}$, expressed in Eq.(4.23). The third column shows the t-values calculated from the standard error and indicates whether the coefficient is significantly different from zero. For a 95% confidence interval $|t| > 1.96$.

		ROLS		
$\boldsymbol{X}$	$\hat{\boldsymbol{\beta}}^*$	Estimate	Std. Error	t value
I	$\hat{\beta}_0^*$	0.0663	0.0192	3.463
$\boldsymbol{x}_1$: $\Delta \ln(pcp)$	$\hat{\beta}_1^*$	0.3074	0.2122	1.449
$\boldsymbol{x}_2$: $\Delta(rente + ssats)$	$\hat{\beta}_2^*$	-3.7811	0.4358	-8.677
$\boldsymbol{x}_3$: $\Delta(rente_{-1} + ssats_{-1})$	$\hat{\beta}_3^*$	-0.7791	0.4468	-1.744
$\boldsymbol{x}_4$: $rente_{-1} + ssats_{-1} + 0.01$	$\hat{\beta}_4^*$	-0.7927	0.3187	-2.488
$\boldsymbol{x}_5$: $dpcpe_{-1}$	$\hat{\beta}_5^*$	0.7709	0.3575	2.156
$\boldsymbol{x}_6$: $dkpe_{-1}$	$\hat{\beta}_6^*$	0.1949	0.0671	2.905
$\boldsymbol{x}_7$: $\ln(kp_{-1}/pcp_{-1})$	$\hat{\beta}_7^*$	-0.1026	0.0268	-3.827
$\boldsymbol{x}_8$: $\ln((ydp_{-1} - ipv_{-1})/pcp_{-1}) - \ln(fwh_{-1})$	$\hat{\beta}_8^*$	0.0554	0.0282	1.963

Table 5.1: The coefficients in MONA house price relation estimated with restricted ordinary least squares (ROLS). The period for which this is estimated is 1974:q2 - 1997:q4 or 95 periods.

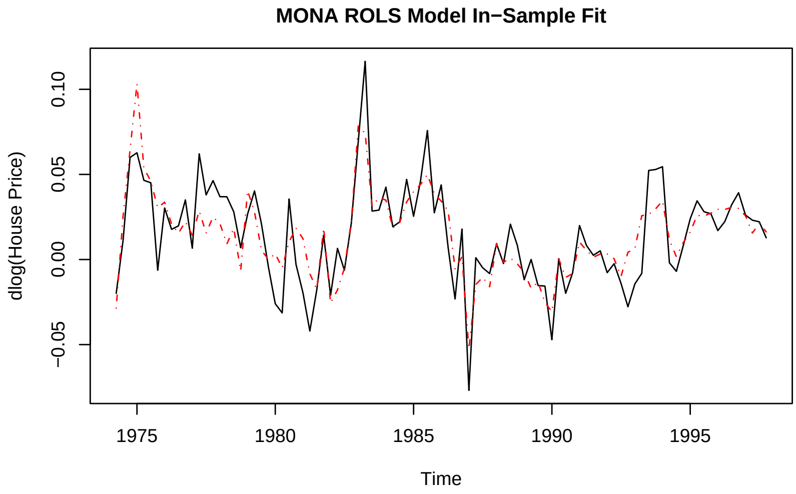

Figure 5.4: The black line is the actual $\boldsymbol{y} = \Delta \ln(kp)$ while the broken red line shows the fitted $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}^*$ using the estimates for $\hat{\boldsymbol{\beta}}^*$ calculated in Table 5.1.

The F-test statistic, which is a test of significance for all regression coefficients, indicates that the MONA model regression coefficients are very significant with $F(7, 87) = 27.9214$ and a very small p-value < 1e-13.

The *R-square*,*adjusted R-square* and error are shown in Table 5.2. The estimated model seems to fit the data quite well with a $R^2 = 0.692$. The *adjusted R-square* gives a lower value of $R^2_{adj} = 0.6672$, since it is adjusted to the number of regressors.

	ROLS
R^2	0.6920
R^2_{adj}	0.6672
$\hat{\sigma}$	0.0169

Table 5.2: The R^2, R^2_{adj} and $\hat{\sigma}^2$ for the ROLS fit shown in Table 5.1.

Example 5.1 (Calculations of change in house price)
Each line in the in-sample explanatory matrix $\boldsymbol{X}$ can be expressed as vector of all explanatory variables at a certain time t, where $1 \leq t \leq n$. More precisely

$$\boldsymbol{x}'_{t,1\ldots p} = \begin{bmatrix} 1 \\ \Delta \ln(pcp_t) \\ \Delta(rente_t + ssats_t) \\ \Delta(rente_{t-1} + ssats_{t-1}) \\ (rente_{t-1} + ssats_{t-1} + 0.01) \\ dpcpe_{t-1} \\ dkpe_{t-1} \\ \ln(kp_{t-1}/pcp_{t-1}) \\ \ln((ydp_{t-1} - ipv_{t-1})/pcp_{t-1}) - \ln(fwh_{t-1}) \end{bmatrix}$$

for a certain period or time the fitted change in house price can be calculated as follows

$$\hat{y}_t = \boldsymbol{x}_{(t,1\ldots p)}\hat{\boldsymbol{\beta}}^*$$

where $\hat{\boldsymbol{\beta}}^*$ is the estimated ROLS coefficients displayed in Table 5.1.

For a specific time e.g. if $t = 1987{:}q4$ fitted house price changes can be calculated as follows

$$\hat{y}_{1987:q4} = \boldsymbol{x}_{(1987:q4,1\ldots p)}\hat{\boldsymbol{\beta}}^*$$

where

$$\boldsymbol{x}'_{(1987:q4,1\ldots p)} = \begin{bmatrix} 1 \\ 0.00892 \\ 0.00148 \\ 0.00193 \\ 0.0848 \\ 0.0222 \\ 0.0622 \\ 0.125 \\ \text{-}0.492 \end{bmatrix}, \qquad \hat{\boldsymbol{\beta}}^* = \begin{bmatrix} 0.0663 \\ 0.3074 \\ \text{-}3.7811 \\ \text{-}0.7791 \\ \text{-}0.7927 \\ 0.7709 \\ 0.1949 \\ \text{-}0.1026 \\ 0.0554 \end{bmatrix}.$$

giving a fitted value of $\hat{y}_{1987:q4} = -0.01602$. The difference in fit and observed change, i.e. the residual, is then calculated as

$$\begin{aligned} e_{1987:q4} &= y_{1987:q4} - \hat{y}_{1987:q4} \\ &= -0.00826 - (-0.01602) \\ &= 0.00776 \end{aligned}$$

By exchanging the $\boldsymbol{X}$ matrix for the vector $\boldsymbol{x}$ a fit for the whole in-sample period can be achieved, which is depicted as the broken red line in Figure 5.4.

□

5.5.1 Residual Analysis

When analyzing the results from a regression model the residuals deserve attention since they need to be randomly distributed with mean 0 and constant variance σ^2_{res}. In the MONA report two well known econometric tests are used for analyzing the residuals. The first test is the so-called ***Durbin Watson***[6] test which tests for autocorrelation in the residuals, the second test is the ***Jarque-Bera***[7] test which is intended to check whether the residuals are normally distributed by using the third and fourth moments, skewness and kurtosis. A detailed account of these tests is outside the scope of this report but for more information see Kyhl & Nielsen [7] on the DW-test and Verbeek [16][8] for the JB-test. The ROLS model passes both of these tests. There is no significant autocorrelation in the residuals, $DW = 1.6924$ giving a p-value of 0.02, it can be asserted with 98% confidences that there does not exist autocorrelation among the residuals. The Jarque-Bera test gives a statistic of $JB = 0.8034$ and the null hypothesis, that the residuals are normally distributed, can not be rejected for all reasonable levels of confidence with a p-value = 0.6692.

Other ways of analyzing residuals, especially in engineering statistics and time series analysis, is by visual inspection of standardized residuals. Figure 5.5 shows four plots often inspected when analyzing residuals. In the upper left panel the residuals are plotted against the corresponding fitted value. The panel does not indicate anything suspicious such as funnel forming, which would indicate an increased variance with increased fitted values. The fact that the cluster is not taking on any obvious form indicates that the model is sufficient and no systematic effect (more regressors) are needed. The upper right plot show the so-called QQ-plot which is a normal probability plot of the standardized residuals, defined by

$$d_i = \frac{e_i}{\sqrt{\hat{\sigma}^2}}$$

[6] See R, package `lmtest`, function `dwtest()` .

[7] See R, package `tseries`, function `jarque.bera.test()` .

[8] See e.g. page 174

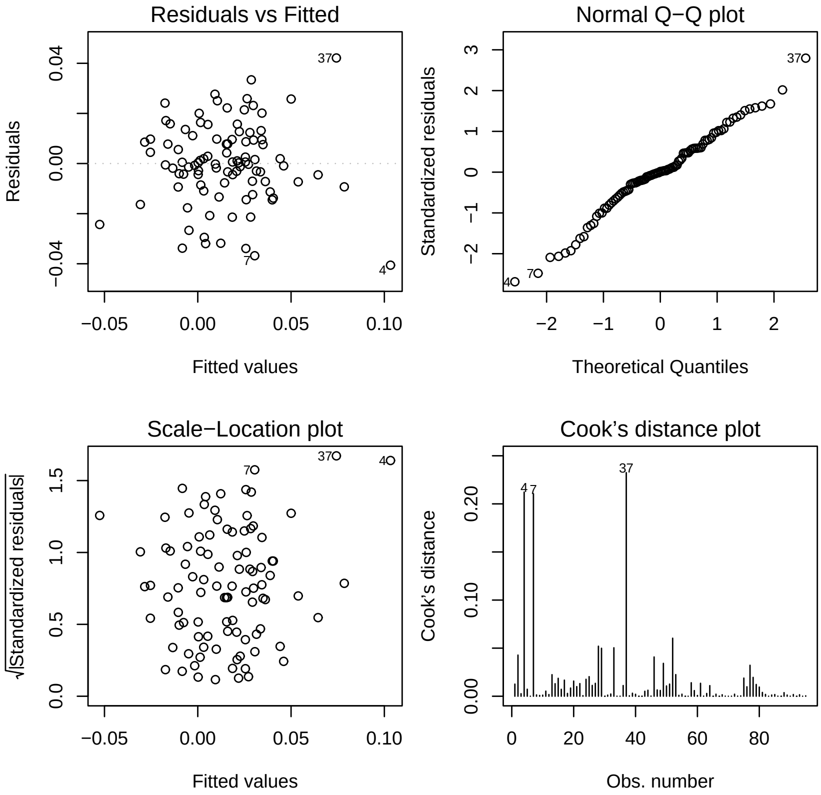

Figure 5.5: Visual residuals analysis from the $\boldsymbol{e} = \mathbf{y} - \hat{\mathbf{y}}$.

Using the standardized residuals also reveals whether the there are any outliers present, i.e. since all d_i should be inside the interval $-3 \leq d_i \leq 3$, or else they may be having an outlier effect on the regression. The residuals on the QQ-plot should fall to a straight line from -3 to 3 if they are normally distributed, this seems to be the case which has also been indicated by the JB-test. The bottom left plot shows the square root of the absolute value of the standardized residuals, which makes it easier to see if there is any trend in the residual cluster, same as for the for d_i no suspicious clustering can be seen in the bottom left graph. The bottom right plot shows the Cook distance for the residuals, Cooks distance measures the effect a single observation can have on the regression, i.e. it finds the outliers. According to Montgomery and Runger [11][9] the Cook distance with a value of $D_i > 1$ indicates that a single outlier is influential in

[9] See section 12-5.1 Residual Analysis.

the regression. As the bottom right graph shows all $D_i < 0.25$, the suggestion of certain outliers affecting the regression is dismissed.

5.6 Prediction

The subject of using the regression models to forecast for new variables is one of the main reasons the MONA house price model has been listed and dissected in such detail. Since there is data available from 1972:$q2$ to 2001:$q3$ the out-of-sample period, 1998:$q1$ to 2001:$q3$, will be used to show how a prediction is made when new observations for the explanatory variables are available. The following is largely adopted from Montgomery and Runger [11][10] and Madsen [9][11].

When predicting l-steps ahead, where $1 \leq l \leq k$ and k is the prediction horizon, given the estimated coefficients the predicted response value can be expressed as

$$\hat{y}_{t+l} = E[y_{t+l}|\boldsymbol{X}_{t+l} = \boldsymbol{x}_{t+l}] = \boldsymbol{x}_{t+l}\hat{\boldsymbol{\beta}} \tag{5.11}$$

where $\boldsymbol{x}_{t+l}$ represent a vector of new observed values for the explanatory variables. Eq.(5.11) gives the so-called ***point estimates*** for the future response corresponding to $\boldsymbol{x_{t+l}}$. The ***prediction error*** $e_{t+l} = y_{t+l} - \hat{y}_{t+l}$ has the variance

$$V_{OLS}[e_{t+l}] = V[y_{t+l} - \hat{y}_{t+l}] = \sigma^2(1 + \boldsymbol{x}'_{t+l}(\boldsymbol{X'X})^{-1}\boldsymbol{x}_{t+l})$$

for the OLS method, this can be seen from

$$\begin{aligned} V[y_{t+l} - \hat{y}_{t+l}] &= V[\boldsymbol{x}'_{t+l}\boldsymbol{\beta} + \varepsilon_{t+l} - \boldsymbol{x}'_{t+l}\hat{\boldsymbol{\beta}}] \\ &= V[\boldsymbol{x}'_{t+l}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}) + \varepsilon_{t+l}] \\ &= \boldsymbol{x}'_{t+l}V[\hat{\boldsymbol{\beta}}]\boldsymbol{x}_{t+l} + \sigma^2 + 2Cov[\boldsymbol{x}'_{t+l}(\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}), \varepsilon_{t+l}] \\ &= \sigma^2 + \boldsymbol{x}'_{t+l}V[\hat{\boldsymbol{\beta}}]\boldsymbol{x}_{t+l} \end{aligned}$$

where $V[\hat{\boldsymbol{\beta}}]$ is the covariance matrix $\boldsymbol{\Sigma}_{\hat{\boldsymbol{\beta}}} = \sigma^2(\boldsymbol{X'X})^{-1}$. This result can be extended to the ROLS method by inserting the ROLS covariance matrix which gives

$$V_{ROLS}[e_{t+l}] = \sigma^2(1 + \boldsymbol{x}'_{t+l}\boldsymbol{M}^*(\boldsymbol{X'X})^{-1}\boldsymbol{M}^{*'}\boldsymbol{x}_{t+l}) \tag{5.12}$$

A $100(1-\alpha)\%$ ***confidence interval for future values*** of $\hat{y}_{t+l}$ is given by

$$\hat{y}_{t+l} \pm t_{(\alpha/2,n-p)}\sqrt{V[\varepsilon_{t+l}]} \tag{5.13}$$

[10] Section 12-4, Prediction of new observations.

[11] Section 2.3

which for ROLS becomes

$$\hat{y}_{t+l} \pm t_{(\alpha/2,n-p+q)}\hat{\sigma}\sqrt{(1+\boldsymbol{x}'_{t+l}\boldsymbol{M}^*(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{M}^{*\prime}\boldsymbol{x}_{t+l})} \tag{5.14}$$

when using the estimate $\hat{\sigma}$ for the residual variance of error. The term $t_{\alpha/2,n-p+q}$ is from the t-distribution with $(n-p+q)$ degrees of freedom, where q is the number of constraints since q regressors are linear combinations of other regressors and therefore q of the p regressors return their degrees of freedom.

Using the out-of-sample period 1998:$q1$ to 2001:$q3$ the point estimate, along with a 95% prediction interval is calculated and plotted in Figure 5.6.

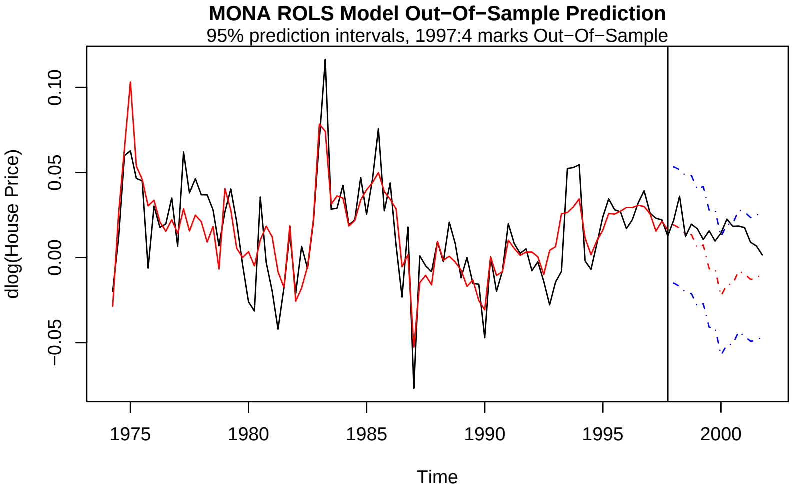

Figure 5.6: The in-sample estimation is represented with a red whole line $\hat{\boldsymbol{y}}$, the black line is the actual observed change $\boldsymbol{y}$, the red broken line is the point estimate for new observations $\hat{\boldsymbol{y}}_{new}$ along with a 95% prediction interval shown by the broken blue lines. The vertical line marks where the in-sample ends and the new observations (out-of-sample) begins.

Figure 5.6 shows that the out-of-sample prediction seems to be performing poorly, a measure often used for analyzing predictions is the ***Mean square error*** defined as

$$MSE(\hat{y}_{t+k}) = \frac{1}{k}\sum_{l=1}^{k}(y_{t+l}-\hat{y}_{t+l})^2. \tag{5.15}$$

Calculating the MSE for the prediction in the out-of-sample an estimate of the error $\hat{\sigma}$ can be found. The error in the out-of-sample period gives an error estimate of 0.0213, which is higher than the in-sample error of $\hat{\sigma} = 0.0169$. The out-of-sample performance is considerably worse than for the in-sample, such a big shift in accuracy indicates that the out-of-sample data is different from the in-sample data. This will be discussed further in the next subsection.

5.6.1 MONA Out-Of-Sample failure

The out-of-sample performance is not expected to be as good as the in-sample, since that is where the coefficients are estimated, however a large shift in error suggest that the out-of-sample data is significantly different from the in-sample period. This seems to be the case for the out-of-sample data, a large shift in error and visual analysis of the out-of-sample data shows that the variance of the house price change has decreased dramatically and the mean has increased, see Figure 5.7. All observed house price changes after 1994:$q4$ are increments and the variance has changed considerably from the in-sample variance, see cyan colored broken lines in Figure 5.7. The explanatory variables suggest that the price of houses should drop while it does not, this continues for some time creating a gap between the predicted price and observed house price, which is typical of a housing bubble such as was mentioned in section 2.3. The fact that the data seems to be non-homogenous between the in- and out-of-sample periods makes validation, of the parameter estimation, in the out-of-sample period useless. In the theoretical economic models the bubble-free condition is assumed.

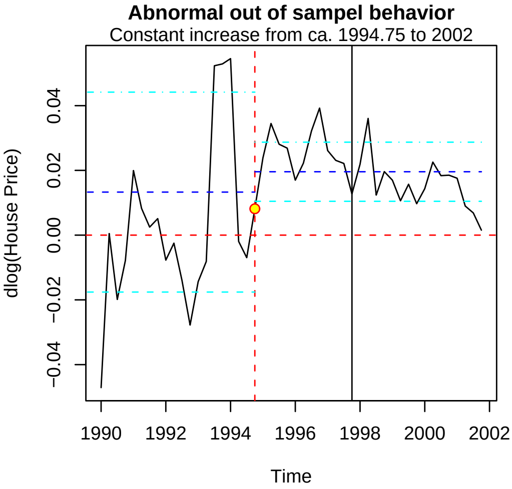

Figure 5.7: Shows the strange behavior of the house price data after 1994:$q4$ the process seems to slow down considerably resulting in less variance and higher mean. Only increments after 1994:$q4$.

Ways of dealing with this discrepancy could e.g. be to include the abnormal data period in the parameter estimation. The parameters will then be able to deal better with presence of such behavior. However, the goodness of fit will drop and all the data is then used for estimation which makes validation hard. Another way to deal with the bubble behavior is to move the time window, i.e. include more of recent years and less of the previous years, however that would also result in out-of-sample validation problems since the out-of-sample data would then most likely not resemble the in-sample data.

Yet another method would be to use another parameter estimation method, i.e. so-called recursive least squares (RLS) where the parameter estimation is consistently being updated with a rolling time window, or a forgetting factor which reduces the impact of old data has on the parameter estimation giving ever changing, but relatively accurate estimations, see Madsen [9][12].

5.6.2 MONA model and certain markets

Something to keep in mind when looking at the results of the MONA model is that the house price data is an average of diverse house markets. For example the urban flats markets in Copenhagen may behave differently than the rural or summerhouse market. The difference in these two markets can e.g. be traced back to the theoretical model described in chapter 2 where house price is considered to achieve a higher equilibrium price where construction land is limited. There are however many other things other than location that influence the price such as building age, building style, size, number of bathrooms and so on. If a prediction is sought for a certain part of the market, that section of the market has to be modeled specifically, with corresponding data acquired from sales prices in that region.

The MONA is thought of as a general macro model to indicate the long term direction of the Danish house price market as a whole, not to give dynamic short term predictions for specific parts of the Danish market.

5.6.3 The ECM with the ROLS model

As was mentioned before in section 4.3, the ROLS coefficients are used in an error-correction model format to give an idea of the long term effects in the housing market. These long term trends are shown in the MONA report [12][13].

[12] e.g. page 278.

[13] See top of page 45.

Recall the ECM format given in section 4.3 as

$$\Delta \boldsymbol{z}_t = \phi_0 \Delta \boldsymbol{r}_t - (1-\theta)[\boldsymbol{z}_{t-1} - \alpha - \gamma \boldsymbol{r}_{t-1}] + \epsilon_t$$

where $\boldsymbol{z}_t$ is some response variable and $\boldsymbol{r}_{t-1}$ is a explanatory component. The ECM relation is divided into a dynamic part, i.e. the $\phi_0 \Delta \boldsymbol{r}_t$ part, and the error correction part, i.e. $(1-\theta)[\boldsymbol{z}_{t-1} - \alpha - \gamma \boldsymbol{r}_{t-1}]$.

To use the error correction form for the MONA house price relation the components of the explanatory matrix $\boldsymbol{X}$ needed to be sorted into dynamic parts and the error correction or long term effects. The short term changes are indicated by modeling the change (differenced components) while the long term effects are taking into account the level at each time (nominal series).

The i-th component of the explanatory matrix $\boldsymbol{X}$ and estimated coefficient vector $\boldsymbol{\beta}^*$ are noted as $\boldsymbol{x}_i$ and β_i^* respectively. The estimated change in house price is calculated as $\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}^*$. The ECM format of $\hat{\boldsymbol{y}}$ is therefore

$$\hat{\boldsymbol{y}} = \left[\hat{\beta}_1^* \boldsymbol{x}_1 + \hat{\beta}_2^* \boldsymbol{x}_2 + \hat{\beta}_3^* \boldsymbol{x}_3 + \hat{\beta}_5^* \boldsymbol{x}_5 + \hat{\beta}_6^* \boldsymbol{x}_6\right] - \hat{\beta}_7^* \left(\boldsymbol{x}_7 - \frac{\hat{\beta}_4^*}{\hat{\beta}_7^*}\boldsymbol{x}_8 - \frac{\hat{\beta}_8^*}{\hat{\beta}_7^*}\boldsymbol{x}_4 - \frac{\hat{\beta}_0^*}{\hat{\beta}_7^*}\right) \quad (5.16)$$

In Eq.(5.16) the terms inside the [ ] bracket represent the dynamic part of the model i.e. price and interest changes. The second part, or the () bracket, has the terms which cause a deviation from $\hat{\boldsymbol{y}}$ in a long run equilibrium, i.e. the levels and the part which corresponds to the long run multiplier γ, derived in section 4.3. Recall that $\hat{\beta}_7^*$ is the coefficient for real house price, while $\hat{\beta}_4^*$ corresponds to user cost and $\hat{\beta}_8^*$ is for real income over stock of houses. Inserting the estimated coefficients from Table 5.1 gives the following long run multipliers for the levels of $\boldsymbol{x}_4$ and $\boldsymbol{x}_8$:

$$-\frac{\hat{\beta}_4^*}{\hat{\beta}_7^*} = -\frac{-0.7927}{-0.1026} = -7.726, \qquad -\frac{\hat{\beta}_8^*}{\hat{\beta}_7^*} = -\frac{0.0554}{-0.1026} = 0.540.$$

If either of the elements corresponding to β_4^* or β_8^* were to increase by some small dx element the house price change will in the long run change by the dx times the ratios above, given that all other things stay fixed.

The nature of the error-correction format is to include levels and differenced values, even though the level is non-stationary as long as the response variable is stationary.

Chapter 6

Applying The MONA house price relation

6.1 Introduction

The purpose of this chapter is to get an applied version of the MONA house price relation. To get a robust prediction model from the MONA house price relation some relaxations must be made, this chapter discusses the concessions made and what results they have in regards to precision in prediction.

In section 6.2 a regression model based only on the interest terms in the MONA model is formulated, which will be used to benchmark other models. Section 6.3 discusses the aggregation of house price changes, using updating with or without observed house prices, to get house price levels. Section 6.4 addresses the fact that when predicting, only interest rates are available, other explanatory variables must therefore be fixed in some sensible way. In section 6.5 the aggregate error is simulated and compared for three different models.

6.2 Interest Rate Regression

Using only the interest rate terms from the MONA house price model a smaller, simpler, benchmark model is developed. The main reason for performing this simpler regression is to get a model where all the information is available, i.e. the model will only be dependent on interest rates, which are available through the interest rate tree. Later when the MONA model as whole will be used, it can be seen that all the missing data has to be fixed to some level which increases the error of the house price estimate. The fact that missing observations of the explanatory variables do not have to be fixed also allow for simpler error calculations that can be calculated via analytical methods compared to the simulated error for the fixed model.

The simplified regression model based on the MONA house price relation is expressed as follows

$$\widehat{\Delta ln(kp_t)}^{I} = \hat{\beta}_0^I + \hat{\beta}_1^I \Delta rente_t + \hat{\beta}_2^I \Delta rente_{t-1} + \hat{\beta}_3^I rente_{t-1} \tag{6.1}$$

Where $\Delta rente_t$ and $rente_t$ are the change in interest rates and actual interest rate respectively. Notice that the tax rate $ssats$ has also been removed from the interest relation. From this reduced model two results can be expected. Firstly a lower value for both goodness of fit estimators R^2 and R^R_{adj}, in comparison to the MONA model. Secondly the residuals are more likely to show signs of autocorrelation since it is known from the MONA house price relation that this smaller model is missing many proven systematic effects, e.g. income over stock of houses ($\boldsymbol{x}_8$) and the consumption deflator ($\boldsymbol{x}_1$) to name only two.

Using the in-sample period, 1974:q2-1997:q4, that was used in the MONA house price relation, an ordinary least squares (OLS) regression is performed to estimate the coefficients $\boldsymbol{\beta}^{I'} = [\beta_0^I,\ \beta_1^I,\ \beta_2^I,\ \beta_3^I]$ by solving

$$\hat{\boldsymbol{\beta}}^I = (\boldsymbol{X}^{I'}\boldsymbol{X}^I)^{-1}\boldsymbol{X}^{I'}\boldsymbol{y}$$

where the explanatory matrix $\boldsymbol{X}^I$ is only composed of interest terms as follows

$$\boldsymbol{X}^I = \begin{bmatrix} 1 & \Delta rente_2 & \Delta rente_1 & rente_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \Delta rente_n & \Delta rente_{n-1} & rente_{n-1} \end{bmatrix}$$

After having performed the regression the t-statistic shows that $\hat{\beta}_3^I$ is not significantly different from zero, with p-value $= 0.9965$. When the regression is repeated, leaving $\hat{\beta}_3^I$ out, it gives the estimated coefficients $\hat{\boldsymbol{\beta}}^I$ shown in Table 6.1

	Estimate	Std. Error	t value	Pr(>|t|)
$\hat{\beta}_0^I$	0.0125	0.0023	5.37	0.0000
$\hat{\beta}_1^I$	-3.6539	0.5885	-6.21	0.0000
$\hat{\beta}_2^I$	-1.6934	0.5767	-2.94	0.0042

Table 6.1: The estimated coefficients in the reduced MONA house price relation, using only interest rates, estimated with ordinary leat squares (OLS). Estimated for the sample period 1974:2 - 1997:4 or 95 periods. The first column is the estimate, second is the standard error of the estimate, third is the t-statistic and fourth is the p-value.

All the coefficients estimate in Table 6.1 are highly significant, i.e. all p-values are less than one percent which indicates that all coefficients are significant with a confidence of $> 99\%$. The F-statistic also indicates that the model is significant with $F(2, 92) = 32.72$ which gives a p-value $= 1.854e - 11$.

	ROLS	OLS_{Int}
R^2	0.6920	0.4156
R^2_{adj}	0.6672	0.4029
$\sqrt{\hat{\sigma}^2}$	0.0169	0.0226

Table 6.2: Comparison of the the goodness of fit, R^2 and R^2_{adj}, for the MONA house price relation (ROLS) and the reduced interest rate only regression (OLS_{Int}).

The goodness of fit statistics can be seen in Table 6.2, the results from the MONA house price fit is also displayed for comparison. As expected there is a considerable fall in R^2 since many known explanatory variables are skipped in the reduced model. When comparing two regression models with different number of coefficients the R^2_{adj} is a better way of comparing the two fits than R^2. The difference in R^2_{adj} is not as much as for R^2 but is still considerable or approximately 0.165.

The Jarque-Bera and Durbin Watson tests indicate whether or not the residuals pass the claim of being normally distributed and without any significant autocorrelation. The Jarque-Bera statistic is $JB = 2.42$, i.e. the hypothesis that the residuals are normally distributed can not be rejected since p-value$= 0.2978$. The Durbin Watson test is used to detect any autocorrelation in the residuals, i.e. is the residual e_t dependant on previous residuals $e_{t-1,\ldots,0}$. The Durbin Watson gives $DW = 2.0274$ and a p-value $= 0.6098$ which means that the hypothesis of no-autocorrelation in the residuals can not be dismissed. When comparing Figure 6.1 to the residual plot in Figure 5.5, which is for the full model, it can be seen that the variance of the residuals seems to be bigger in the reduced model. The left panels in Figure 6.1 also show less dispersion in the

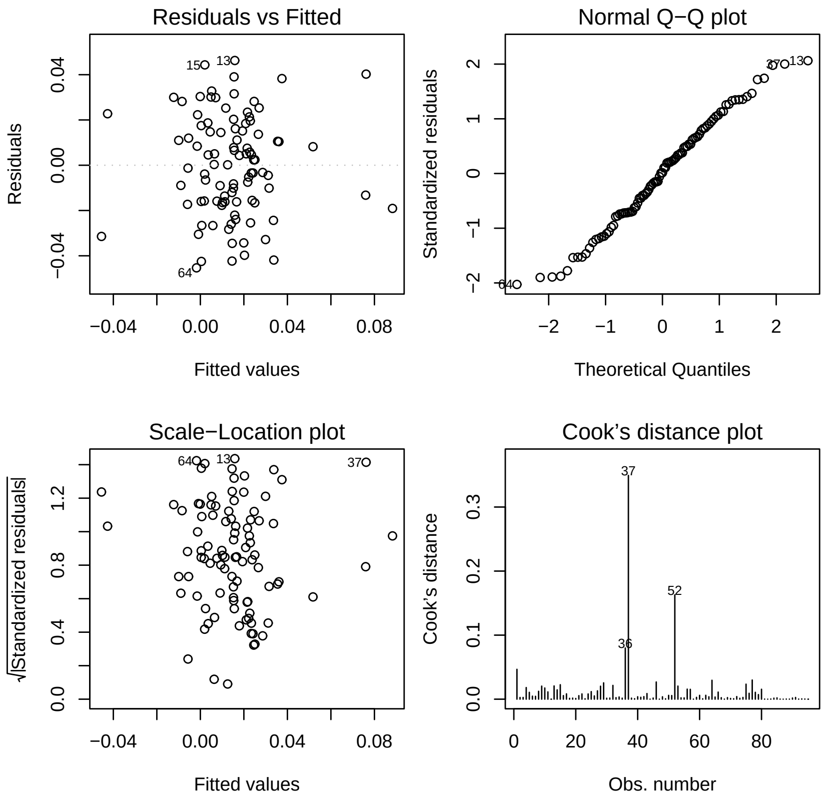

Figure 6.1: Visual residuals analysis from the $e = \mathbf{y} - \hat{\mathbf{y}}_{\mathbf{I}}$.

cluster than in 5.5, which might indicate autocorrelation. The normality curve is not visually different from the full model. The Cook plot shows that there are bigger outliers in the reduced model, but still nothing to be worried about according to the $D_i > 1$ limit.

As expected there appearers to be some autocorrelation in the residuals, for this reduced regression, however judging by the QQ-plot and the JB it is safe to say that the residuals can be viewed as approximately normal distributed.

Since no fixing of any explanatory variables is performed the point estimate and prediction interval for new observations can be achieved by using Eq.(5.14), although because of the autocorrelation the prediction will most likely not be good. The results for such a point estimate along with prediction intervals is shown in Figure 6.2.

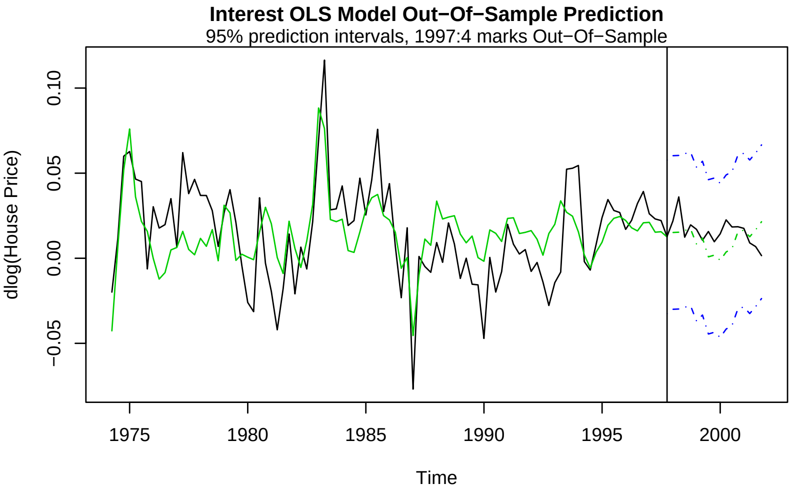

Figure 6.2: The sample period is the period 1974:$q2$-1997:$q4$ and is shown by the green whole line. The black whole line represents the actual change at each time. During 1997:$q4$-2001:$q4$, the out of sample period, the broken green line is the point estimation, while the red lines represent a 95% prediction interval for future observations.

The reduced regression model is not as accurate as the MONA house price relation. It does not represent the data as well as the MONA model and all economic intuition used in the MONA is dropped. Despite these shortcomings the reduced model will be used to benchmark the fixed MONA model throughout this chapter.

6.3 Aggregated House Prices

The estimated change, according to the MONA house price model, at some time t can be expressed as

$$\hat{y}_t = \Delta \widehat{\ln(kp_t)} = \hat{\beta}_0^* + \sum_{i=1}^{k} \hat{\beta}_i^* x_{ti} \qquad t = 1, 2, ..., n$$

where $\hat{y}_t$ is the estimated change in house prices, from $t-1$ to t, by using the regression coefficients $\hat{\beta}_i^*$ times the corresponding explanatory variable x_{ti}. The house price scenario tree, which is to be produced, is meant to hold the nominal value of house prices not the change in house prices between periods. The MONA results must therefore be accumulated over the prediction period.

The transformation from house price change, to aggregated house price change will be discussed in this section.

As was mentioned in subsection 5.3.2 the differenced series must be accumulated to give the actual house price. According to MONA [12][1] the observed house price can be calculated from house price change by

$$\ln(kp_t) = \Delta \ln(kp_t) + \ln(kp_{t-1}) \tag{6.2}$$

i.e. by adding the house price change to last periods house price.

There are two ways of performing this transformation. The first method involves updating the estimate of aggregated house prices with actual observed house prices (kp_{t-1}), this greatly reduces the error and gives a very stable prediction, i.e. basically a one step prediction with updating at each step. The second way, which will be of interest in this thesis, is comparable to a k step prediction without updating, i.e. the prediction is updated not with observed values but last periods predicted values $(\widetilde{kp}_{t-1})$. Both methods will be given some discussion, beginning with the one step updating.

6.3.1 Updating with observed house prices, k=1

Recall that the difference between the actual change and the estimated change is the residual, i.e.

$$\begin{aligned} e_t &= y_t - \hat{y}_t \\ &= \Delta \ln(kp_t) - \Delta \widehat{\ln(kp_t)} \\ &= (\ln(kp_t) - \ln(kp_{t-1})) - \Delta \widehat{\ln(kp_t)} \end{aligned}$$

When rearranging the terms in the last relation and $\ln(kp_t)$ is isolated on the left side it becomes

$$\ln(kp_t) = \Delta\widehat{\ln(kp_t)} + \ln(kp_{t-1}) + e_t \tag{6.3}$$

Which is the relation for one step updating for the house price level using the modeled house price change. Since the residuals should follow $e_t \sim N(0, \hat{\sigma}^2)$ it is easy to see that the aggregation should give an expected value, point estimate, of

$$\ln(\widetilde{kp}_t) = \Delta \widehat{\ln(kp_t)} + \ln(kp_{t-1}) \tag{6.4}$$

Where $\ln(\widetilde{kp}_t)$ represents the point estimate of $\ln(kp_t)$ for one period and updating with last periods observed house prices. The accumulation has no effect

[1] See page 196.

on the variance of $\ln(\widetilde{kp_t})$, i.e. the only contribution to the error is from the current estimation of $\Delta \widehat{\ln(kp_t)}$. Prediction intervals for the one step aggregate house price can be calculated in the same way as was done in section 5.6 using Eq.(5.14). Figure 6.3 shows how the one step method has very little effect when transforming to the aggregate house price both for the MONA house price relation and the relatively inaccurate interest rates only model.

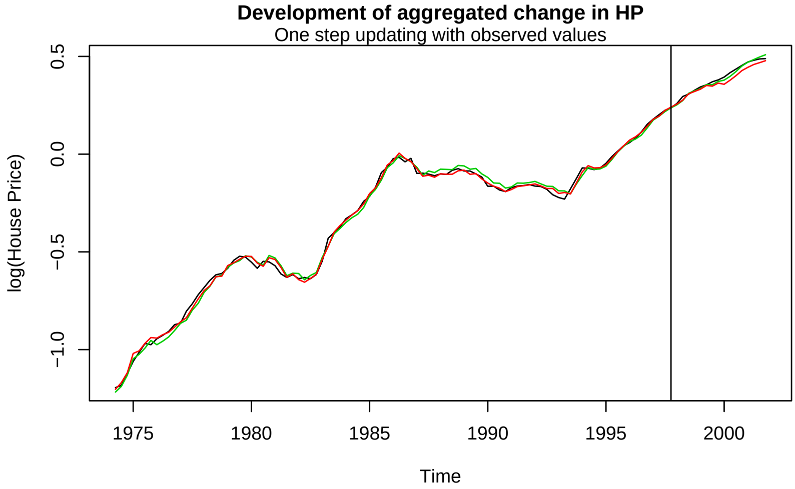

Figure 6.3: The graph shows how the cumulative house price develops when updating with actual observed house price values at each time. The black line is the actual house price, red line is the MONA ROLS model and the green line is the interest rate only regression from section 6.2.

6.3.2 Updating with estimated house prices, k>1

If the observed house price is not available at each period, or only occasionally, the change in house price must be compounded and last periods estimated house price level used for updating.

Given some initial house price, $A = ln(kp_0)$, and using the updating formula given in Eq.(6.2), the following can be shown:

$$
\begin{aligned}
\ln(\widetilde{kp}_0) &= A \\
\ln(\widetilde{kp}_1) &= A \\
\ln(\widetilde{kp}_2) &= A + \Delta\widehat{\ln(kp_2)} \\
\ln(\widetilde{kp}_3) &= \ln(\widetilde{kp}_2) + \Delta\widehat{\ln(kp_3)} = A + \Delta\widehat{\ln(kp_2)} + \Delta\widehat{\ln(kp_3)} \\
\ln(\widetilde{kp}_4) &= \ln(\widetilde{kp}_3) + \Delta\widehat{\ln(kp_4)} = A + \Delta\widehat{\ln(kp_2)} + \Delta\widehat{\ln(kp_3)} + \Delta\widehat{\ln(kp_4)} \\
\vdots \; &= \quad \vdots \\
\ln(\widetilde{kp}_t) &= A + \sum_{i=2}^{t} \Delta\widehat{\ln(kp_i)} \qquad \text{where} \qquad t \geq 2 \qquad (6.5)
\end{aligned}
$$

Eq.(6.5) shows the relation for house price development when using last periods estimated house price as base for the change for $t > 2$. Notice that 2 periods are needed before the house price can be evaluated. The reason for this start up time is that for the evaluation of $\Delta\widehat{\ln(kp_t)}$, the lagged change in user cost ($\boldsymbol{x}_3$) is needed. More precisely

$$
\begin{aligned}
\hat{\beta}_3^{\,*} x_{t3} &= \hat{\beta}_3^{\,*} \cdot \Delta(rente_{t-1} + ssats_{t-1}) \\
&= \hat{\beta}_3^{\,*} \cdot (rente_{t-1} - rente_{t-2} + ssats_{t-1} - ssats_{t-2})
\end{aligned}
$$

The relation above shows the calculation of the third term, lagged user cost, the one which requires the most start up time and therefore decides the start up for the evaluation of both $\Delta\widehat{\ln(kp_t)}$ and thereby $\widetilde{kp}_t$. The conditional form for aggregate house prices, updating with predictions, is therefore

$$
\ln(\widetilde{kp}_t) = \begin{cases} A & \text{if } t < 2 \\ A + \sum\limits_{i=2}^{t} \Delta\widehat{\ln(kp_i)} & \text{if } t \geq 2 \end{cases} \tag{6.6}
$$

Eq.(6.6), is very important since it describes how to calculate the one path case for house prices, given an initial index price of A and using the MONA house price relation. In Figure 6.4, upper panel, the development of aggregate house prices using the compounding method in Eq.(6.6) can be seen for both Interest rate only regression, green line, and the MONA ROLS house price relation, red line. Comparing the upper panel from Figure 6.4 to the development in Figure 6.3 it can be seen how the aggregation of error has a much bigger effect, especially for the interest only regression which has a considerably higher error, $\hat{\sigma}$, see Table 6.2.

The main problem with using the relation shown in Eq.(6.6) is the estimation of the error. The relation shown in Eq.(6.6) is actually the point estimate, i.e.

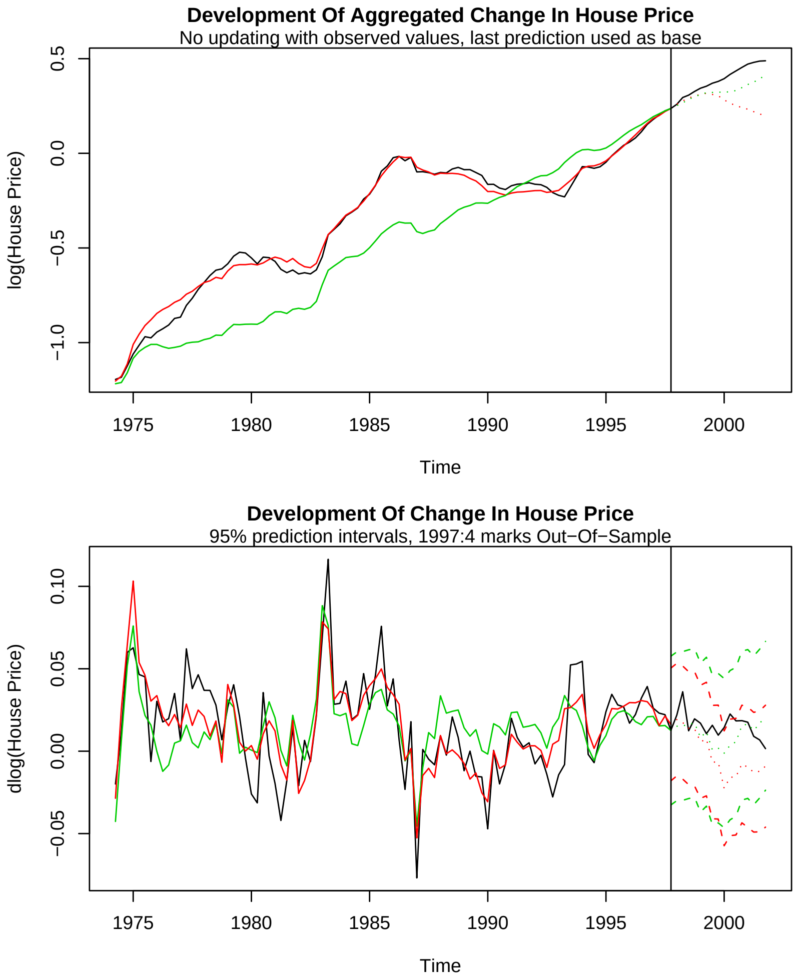

Figure 6.4: The lower panel shows the development of the modeled variable $\Delta ln(kp_t)$. The upper panel shows the aggregated change without updating. The red line is the MONA ROLS, green line is the interest only model described in section 6.2 and black is the observed change. The black vertical line represent the boundary between the in-sample and out-of-sample periods. The point estimates, for the out of sample period, are shown as broken lines.

the expected value of the estimation, since $E[e_t] = 0$. If the residual element for each estimation is included, Eq.(6.6) has the following form

$$\ln(kp_t) = A + \sum_{i=2}^{t}[\Delta \widehat{\ln(kp_i)} + e_i] \qquad \text{where} \qquad t \geq 2$$

The point estimate represents the expected value of the forecast and is simple to calculate as was shown above, however the variance of the prediction is non-trivial. The effect of aggregating the MONA house price change estimates will lead to an ever growing variance of the prediction in accumulated house price estimates. Simulation was used to evaluate the aggregate variance for the compound method. A detailed discussion of how the simulation is performed is given in section 6.5.

6.3.3 Analogy to interest compounding

Before continuing with the discussion of applying the MONA house price relation to a tree structure, a short digression to give an intuitive analogy is presented. The method described in subsection 6.3.2 can be compared to an interest rate compounding relation i.e.

$$V = A \cdot (1 + r)^n \tag{6.7}$$

where A is the initial amount, r is the interest rate and V the total value after n years. By taking the exponential of Eq.(6.5) it becomes

$$\widetilde{kp}_t = A \cdot \prod_{i=2}^{t} e^{\Delta \widehat{\ln(kp_i)}} \qquad \text{where} \qquad t \geq 2$$

The term $e^{\Delta \widehat{\ln(kp_t)}}$ expresses all changes based from one, since $e^0 = 1$, by changing this such that all changes are base from zero

$$r_t = e^{\Delta \widehat{\ln(kp_t)}} - 1$$

where r_t is the percentage change, or rate, from time $t-1$ to t. It can therefore be seen that the exponential form of Eq.(6.5) is the same as Eq.(6.7) with different rates for each period.

$$\widetilde{kp}_t = A \cdot \prod_{i=2}^{t}(1 + r_t) \qquad \text{where} \qquad t \geq 2 \tag{6.8}$$

6.3.4 Numerical Example

To demonstrate the aggregate house price development, using the two methods mentioned above, i.e. updating with observed values and updating with previous predictions, a small numerical example has been prepared. All the data used in the example is fictional.

An initial house price of $A = 100$ is given at time $t = 0$. Interest rate time series I_t start at $t = 0$ and ends at $t = 6$, so the differenced interest series starts at $t = 1$, i.e.

$$\begin{bmatrix} \boldsymbol{I} \\ \boldsymbol{\Delta I} \\ \boldsymbol{\Delta I_{-1}} \end{bmatrix} = \begin{bmatrix} I_0 & I_1 & I_2 & I_3 & I_4 & I_5 & I_6 & \\ & \Delta I_1 & \Delta I_2 & \Delta I_3 & \Delta I_4 & \Delta I_5 & \Delta I_6 & \\ & & \Delta I_1 & \Delta I_2 & \Delta I_3 & \Delta I_4 & \Delta I_5 & \Delta I_6 \end{bmatrix}$$

As was mentioned before the MONA house price relation needs the lagged change of interest rates, which is available at time $t = 2$, to calculate the estimated change in house prices.

The house price changes have been calculated using the MONA house price model, with all explanatory variables available. The estimated change can be seen as $e^{\boldsymbol{\Delta} \widehat{\ln(\boldsymbol{kp})}}$ based from one or as $\boldsymbol{r}$ based from zero

$$e^{\boldsymbol{\Delta} \widehat{\ln(\boldsymbol{kp})}} = \begin{bmatrix} 1 & 1 & 1.03 & 0.99 & 1.01 & 0.97 & 0.98 \end{bmatrix}$$

$$\boldsymbol{r} = \begin{bmatrix} 0 & 0 & 0.03 & \text{-0.01} & 0.01 & \text{-0.03} & \text{-0.02} \end{bmatrix}$$

Using the exponential form of the compounding equation given in Eq.(6.8), i.e. using previous predictions as basis for future estimates (compounding method), gives an aggregate house price as follows

$$\begin{aligned}
\widetilde{kp}_0 &= A = 100 \\
\widetilde{kp}_1 &= A = 100 \\
\widetilde{kp}_2 &= A \cdot (1 + 0.03) = 103 \\
\widetilde{kp}_3 &= A \cdot (1 + 0.03)(1 - 0.01) = 101.97 \\
\widetilde{kp}_4 &= A \cdot (1 + 0.03)(1 - 0.01)(1 + 0.01) = 102.99 \\
\widetilde{kp}_5 &= A \cdot (1 + 0.03)(1 - 0.01)(1 + 0.01)(1 - 0.03) = 99.9 \\
\widetilde{kp}_6 &= A \cdot (1 + 0.03)(1 - 0.01)(1 + 0.01)(1 - 0.03)(1 - 0.02) = 97.90
\end{aligned}$$

Now imagine that the observed house prices from last period are available for $t = 0, ..., 5$ such as

$$\boldsymbol{kp} = \begin{bmatrix} 100 & 98 & 99 & 101 & 99.5 & 102 \end{bmatrix}$$

Using the one period updating given in Eq.(6.4), taking the exponential and inserting r_t gives

$$\widetilde{kp}_t = e^{\Delta \widehat{\ln(kp_t)}} \cdot kp_{t-1} = kp_{t-1}(1 + r_t) \tag{6.9}$$

which when used with the data above gives the following, i.e. estimated house prices with one period updating.

$$\begin{aligned}
\widetilde{kp}_0 &= 100\\
\widetilde{kp}_1 &= kp_0 \cdot (1+0) = 100 \cdot 1 = 100\\
\widetilde{kp}_2 &= kp_1 \cdot (1+0.03) = 98 \cdot 1.03 = 100.94\\
\widetilde{kp}_3 &= kp_2 \cdot (1-0.01) = 99 \cdot 0.99 = 98.01\\
\widetilde{kp}_4 &= kp_3 \cdot (1+0.01) = 101 \cdot 1.01 = 102.01\\
\widetilde{kp}_5 &= kp_4 \cdot (1-0.03) = 99.5 \cdot 0.97 = 96.52\\
\widetilde{kp}_6 &= kp_5 \cdot (1-0.02) = 102 \cdot 0.98 = 99.96
\end{aligned}$$

It is apparent when looking at the results from this small example how dependant on the previous house price value the estimates are when using the compounding method. The one step updating gives house prices that are independent of the last estimated house price, since the observed value is used for updating. A visual demonstration of this independence is given in Figure 6.5 and Figure 6.6 for Eq.(6.4) and Eq.(6.2) respectively.

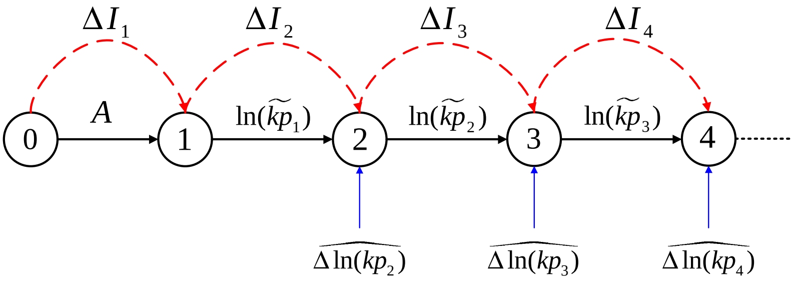

Figure 6.5: A visual representation of the first four house prices when using the compounding method without updating.

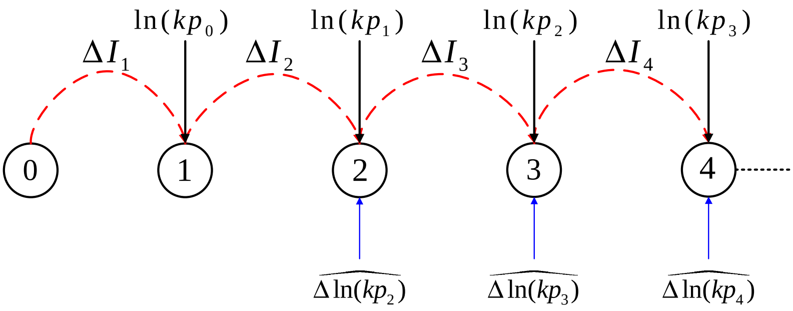

Figure 6.6: A visual representation of the first four house prices when using the one period updating.

To summarize the discussion on aggregation, when aggregating estimated change, it is better to have observed values for updating than previous estimates. Updating, with observed values, is equivalent to resetting the prediction error and thereby resetting the aggregate prediction variance. Using observed values therefore results in a much more accurate prediction, where the change is equivalent to that of the estimated change.

Compounding the change without updating will result in difficulties when estimating the variance of the predicted, aggregated, variable. Further discussion on the estimation of the aggregated variance is given in 6.5.

6.4 Unavailable Explanatory Variables

To apply the MONA house price relation as a prediction model there are some practical aspects that need considering. The most important of these aspects is the lack of information. When predicting with the MONA house price relation, the only new explanatory variables available, during the prediction, are the ones including interest rate. This section deals with ways of compensating for missing information and discusses what effects the lack of new observations have on the prediction.

Recall that the MONA house price relation regression was performed with the design, or explanatory, matrix $\boldsymbol{X}$ which is of size $(n \times p)$, where p is the number of explanatory variables and n the number of observations. Each line $t \in \{1, ..., n\}$ in $\boldsymbol{X}$ can be expressed as

$$X_t = \begin{bmatrix} 1 & x_{t1} & x_{t2} & x_{t3} & x_{t4} & x_{t5} & x_{t6} & x_{t7} & x_{t8} \end{bmatrix}$$

When predicting for future observations of house price change, using the MONA relation, all eight variables must be available. However, as was mentioned before only the interest rates are available in the house price scenario tree prediction. Out of the eight explanatory series in $\boldsymbol{X}$ three include interest rates $(rente_t)$:

$$\begin{aligned} x_{t2} &= \Delta(rente_t + ssats_t) \\ x_{t3} &= \Delta(rente_{t-1} + ssats_{t-1}) \\ x_{t4} &= rente_{t-1} + ssats_{t-1} + 0.01 \end{aligned}$$

The other five explanatory variables, $[\ x_{t1}\ x_{t5}\ x_{t6}\ x_{t7}\ x_{t8}\ ]$, along with the tax terms $(ssats)$ in $[\ x_{t2}\ x_{t3}\ x_{t4}\ ]$ are unavailable when predicting in a house price scenario tree relation. Ways of compensating for the lack of new observations, when forecasting, must therefore be devised.

Dealing with Unavailable Variables

In section 5.6 the MONA house price relation was used to predict for new observations where all the explanatory variables are present for the forecast. When predicting for some response $\hat{y}_t^+$, where $+$ indicates out-of-sample period, a corresponding vector of new explanatory variables can be expressed as

$$X_t^+ = \begin{bmatrix} 1 & x_{t1}^+ & x_{t2}^+ & x_{t3}^+ & x_{t4}^+ & x_{t5}^+ & x_{t6}^+ & x_{t7}^+ & x_{t8}^+ \end{bmatrix}$$

for the Full MONA model, i.e. when all variables are available. In the house price tree generation, where the MONA model is used as basis but only interest rates are available, the vector of new explanatory variables is expressed as

$$A_t^+ = \begin{bmatrix} 0 & 0 & \Delta rente_t^+ & \Delta rente_{t-1}^+ & rente_{t-1}^+ & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (6.10)$$

Subtracting the available A_t^+ from the full X_t^+ gives the missing variables, transposed to

$$F_t^{'} = (X_t^+ - A_t^+)^{'} = \begin{bmatrix} 1 \\ x_{t1}^+ \\ x_{t2}^+ \\ x_{t3}^+ \\ x_{t4}^+ \\ x_{t5}^+ \\ x_{t6}^+ \\ x_{t7}^+ \\ x_{t8}^+ \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \Delta rente_t^+ \\ \Delta rente_{t-1}^+ \\ rente_{t-1}^+ \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ x_{t1}^+ \\ \Delta ssats_t^+ \\ \Delta ssats_{t-1}^+ \\ ssats_{t-1}^+ \\ x_{t5}^+ \\ x_{t6}^+ \\ x_{t7}^+ \\ x_{t8}^+ \end{bmatrix} \qquad (6.11)$$

The vector F_t includes all the variables not available when forecasting. There are numerous ways of dealing with missing or unavailable observations in forecasting. The most simple and straight forward method is to fix the data to a certain period. This method involves fixing all the missing variables to the observed values at time T when predicting for $T+k$ periods ahead, i.e fix all the variables to their value at the prediction origin. This method is a bit cumbersome to apply, since all variables must be aligned at the prediction origin. Fixing missing variables to their values at prediction origin will likely give a good approximation, to the case where new data is available for all explanatory variables, but only for short prediction horizons k.

Example of Fixing at Prediction Origin

Given an in-sample explanatory matrix $\boldsymbol{X}$ and a coefficient vector $\hat{\boldsymbol{\beta}}$, an out-of-sample prediction is sought for five periods ahead, $k = 5$. All out-of-sample data, except for the interest rates, is not available and will be fixed to the last in-sample observations at time $t = n$. The explanatory variables that are fixed at time $t = n$ are therefore

$$F_n = \begin{bmatrix} 1 & x_{n1} & \Delta ssats_n & \Delta ssats_{n-1} & ssats_{n-1} & x_{n5} & x_{n6} & x_{n7} & x_{n8} \end{bmatrix}$$

The available out-of-sample available variables are described, as before, by

$$A_t^+ = \begin{bmatrix} 0 & 0 & \Delta rente_t^+ & \Delta rente_{t-1}^+ & rente_{t-1}^+ & 0 & 0 & 0 & 0 \end{bmatrix}$$

Adding these two vector, i.e. the available variables A_t^+ and the fixed variables F_n, gives the full out-of-sample covariate matrix $X_t^{F'}$ as

$$X_t^{F'} = (A_t^+ + F_n)' = \begin{bmatrix} 1 \\ x_{n1} \\ \Delta rente_t^+ + \Delta ssats_n \\ \Delta rente_{t-1}^+ + \Delta ssats_{n-1} \\ rente_{t-1}^+ + ssats_{n-1} \\ x_{n5} \\ x_{n6} \\ x_{n7} \\ x_{n8} \end{bmatrix} \tag{6.12}$$

where $t = n + 1, ..., n + k$. Using the fixed out-of-sample explanatory matrix to forecast will give predicted change in house price according to

$$\hat{\boldsymbol{y}}^+ = \boldsymbol{X}^F \hat{\boldsymbol{\beta}}^*$$

Effects Of Fixing

By fixing explanatory variables in predictions a certain concession to the full model is made. The fixed model, for short prediction horizons, should prove a good approximation to the full model, however for long prediction horizons the fixed model should be used with much care since it is likely to diverge from the full model and thereby the observed response. Figure 6.7 show the point estimate for out-of-sample predictions using the MONA house price model fixing explanatory variables at forecast origin, 1997:$q4$, for the blue line and using all available data for the red line. The out-of-sample period proves very bad for the MONA model since this is the period which considered to have very "heard like" behavior. The fixed model seems to be much more conservative, which is as expected since many of the variables are fixed and are therefore always giving the same effect, the interest rates control the movement. Fixing will increase the error estimates for the predictions. Fixing variables also makes it hard to

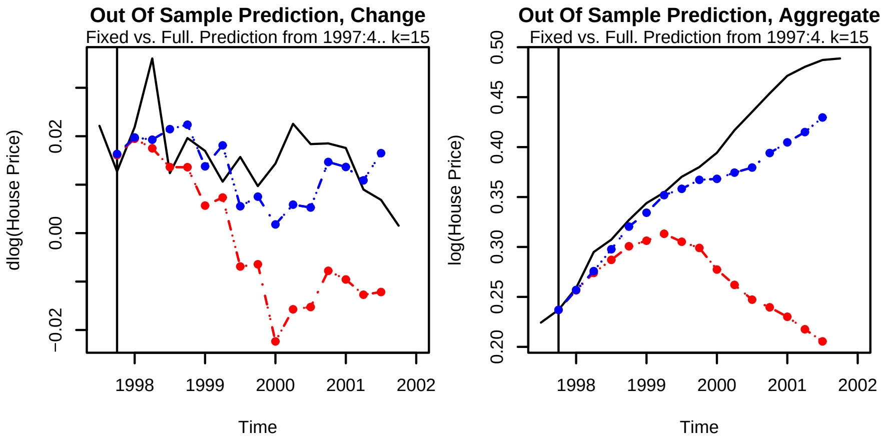

Figure 6.7: Prediction using fixed variables is shown as the blue line. The full model with all data available as red, black is the observed change. Left panel shows the development of predicted values for the change in house prices. The right panel show the aggregate development of house prices.

evaluate the prediction intervals with traditional analytical methods, such as those used in section 5.6. In section 6.5 a thorough discussion about the error is given.

The fixing method can be used to show the individual effect interest rates have in the house prices model, since when the other variables are fixed they act only as a constant. This can be better realized by splitting Eq.(6.12) again up into the fixed and time dependant vectors

$$\hat{y}_t^+ = A_t^+ \hat{\boldsymbol{\beta}}^* + F_n \hat{\boldsymbol{\beta}}^* \qquad (6.13)$$

Notice that the only time dependant effect is the interest rates in X_t^{+A} while F_n only contributes constant value throughout the prediction, i.e. for $t = n + 1, ..., n + k$.

6.4.1 Modeling Explanatory Variables

An alternative to fixing the variables is to model the explanatory variables and use the predicted value, of those models, as the unavailable explanatory variables. The degree of sophistication for modeling of the explanatory variables can also vary greatly, care must however be taken since not all of the processes are stationary. Having to model the explanatory variables also increases the complexity of the prediction model and thereby reduces the usability of the applied scenario house price tree.

The main dissuasive factor, for modeling all the explanatory variables, remains however that proper economic models for these variables tend to have a chain reaction effect, i.e. economic models of the explanatory variables need other variables that also need estimation, requiring new models for those variables and so on. It is therefore essential to make a sensible compromise between model precision and usability. Simple models for the explanatory relationships can be derived, however it is arguable whether they are beneficial or only increase complexity and even the uncertainty. The explanatory, in-sample data is depicted in Figure 6.8. As can be seen there is no simple general way of modeling all these relationships. For example a very simple model could be devised to capture the the expected change in consumption deflator x_{t5} (*dpcpe*) as a time dependant drift model, i.e.

$$\hat{x}_{t5} = \hat{\theta}_0 + \hat{\theta}_1 t$$

however to stop the drift from going below zero more elaborate modeling would be required.

The decision of modeling explanatory variables was abandoned since it would be to time consuming and would have to be done with great care to avoid bad input. Involved modeling would also increase the complexity and decrease usability of the final scenario tree forecasting product. The method of fixing variables at prediction origin was therefore used.

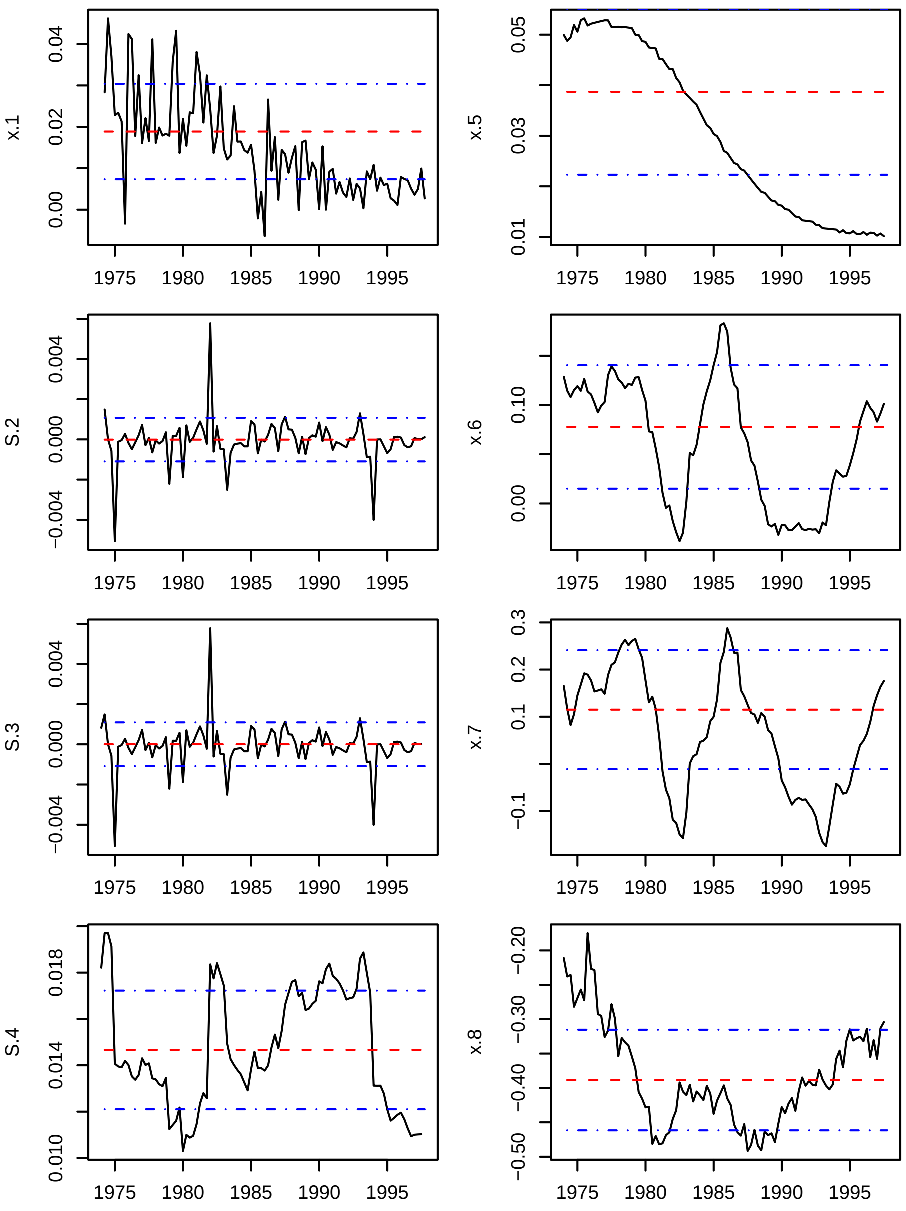

Figure 6.8: The eight series that are fixed. The black line shows the development of the series during the sample period. The broken blue line is the mean of the series, broken red lines are $\mu \pm \sigma$ or mean plus minus one standard deviation. To see what actual economic series x represents, in the MONA model, see subsection5.4.2.

6.5 Estimating the Error

In this chapter topics regarding the extension of the MONA house price model to an aggregated house price tree structure have been discussed. To achieve the scenario tree structure some concession have had to be made to the original MONA house price relation. These concessions have raised question as to how the error should be estimated. This section discusses the components contributing to the error and use simulation methods to quantify the prediction intervals which will give the scenario tree predictions more credibility.

It should be obvious that the actions described in both section 6.4, i.e fixing unavailable variables, as well as aggregating the estimated change, discussed in subsection 6.3.2, will cause an increase in error for the estimation of predicted values. To help quantify and benchmark the house price predictions three models have been devised.

- Model 1: The ideal model. Model for aggregate house price change, using the MONA house price model.
  - All observations available.
- Model 2: The applied model. Model for aggregate house price change, using the MONA house price model.
  - Only interest rates available, other factor fixed at prediction origin, see Eq.(6.12).
- Interest Only model: The interest only regression performed in section 6.2, i.e. the interest rates modelde with new coefficients.
  - Only interest rates explanatory variables needed and are available.

Both the predicted estimated change and the predicted aggregate house price will be investigated for all three models. The most interesting results should be from Model 2 when aggregating the house price, i.e. since in that model both the fixing and the aggregation is applied, also since Model 2 with aggregate house prices is the format that can be applied to the scenario tree.

An expected distribution of predicted change and the predicted aggregate house price for the three models is shown in Figure 6.9. For the predicted estimated change, in house prices, a fixed variance is expected, since no direct recursive or feedback relationship is present in the estimation of the change. The expected outcome for the predicted change in house prices is depicted in Figure 6.9 (a).

In Figure 6.9 (b) the expected development for aggregate house price is shown, where the variance is expected to increase, mainly because of the feedback effect of previous predicted values without updating, see subsection 6.3.2. This aggregation will be different for the three models since different assumptions are made in each model, e.g. the fixing of explanatory variables in Model 2 should act to increase the variance even more.

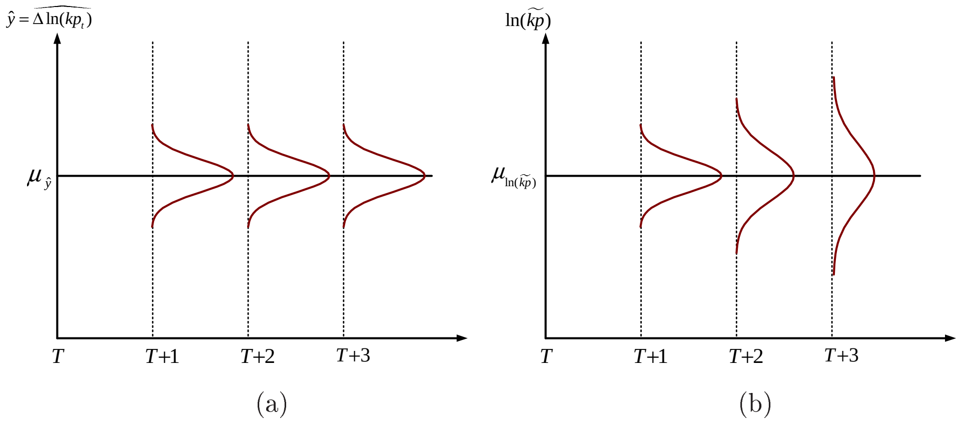

Figure 6.9: Expected error behavior for aggregated house price changes without updating (b). Panel (a) shows the error given by the estimated change at each time.

6.5.1 Bootstrapping

Linear regression models are often used to predict future values. The product of such a prediction is a point estimate and often a prediction interval, such as was discussed in section 5.6. The method described in section 5.6 is an analytical method that uses the variance of the regression to give prediction intervals. When deviation are made to the traditional regression framework, such as fixing variables as is done in Model 2, the analytical methods described in 5.6 no longer apply. Calculations for deriving a formula for the prediction interval can be made, however the more changes that are made from the original framework, the harder and more error prone will its estimation be.

The ideal tool for estimating prediction intervals, when considerable adjustments to the original model have been made, is to use so called bootstrapping methods. The idea behind bootstrapping is to sample from the original data sets to create replicated data sets. From the replicated data sets the variability of the variables of interest can then be estimated without having to deduct long error prone analytical formulas for the variance. For more information about bootstrapping methods in linear regression models see Davidson and Hinkley [2][2].

As was mentioned before the variance analysis will be split into two main scenarios. Firstly the variance for the predicted house price change, for all three models will be estimated. Secondly the changes will be aggregated by sampling the in-sample data, i.e. bootstrapping.

Prediction Interval Estimation

The variance in regression models comes from two terms, i.e. the regression coefficients and the residual

$$\sigma_T^2 = \sigma_R^2 + \sigma_E^2 \tag{6.14}$$

Where σ_T^2, σ_R^2 and σ_E^2 are the total, regression and error or residual variances, respectively. The estimate of σ_E^2 is calculated as $\hat{\sigma}_E^2$ see Eq.(4.21) for the calculation in the MONA restricted ordinary least squares (ROLS) case.

Since the ROLS estimator $\hat{\boldsymbol{\beta}}^*$ is a linear combination of the observations, it can be seen that $\hat{\boldsymbol{\beta}}^*$ is normally distributed with mean $\boldsymbol{\beta}^*$ and covariance matrix $\Sigma_{\boldsymbol{\beta}^*}$, which for ROLS is given as

$$\boldsymbol{\Sigma}_{\boldsymbol{\beta}^*} = \sigma^2 \boldsymbol{M}^* (\boldsymbol{X}'\boldsymbol{X})^{-1} \boldsymbol{M}^{*\prime}$$

where

$$\boldsymbol{M}^* = \boldsymbol{I} - (\mathbf{X}'\mathbf{X})^{-1}\mathbf{R}'(\mathbf{R}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{R}')^{-1}\boldsymbol{R}$$

The diagonal of $\boldsymbol{\Sigma}_{\boldsymbol{\beta}^*}$ gives the variance of the regressors, σ_R^2. The square root of the diagonal of $\boldsymbol{\Sigma}_{\boldsymbol{\beta}^*}$ gives the standard error of the regressors, expressed as $se(\boldsymbol{\beta}^*)$. Recall that the ROLS coefficients, $\boldsymbol{\beta^*}$, were estimated as $\hat{\boldsymbol{\beta}}^{\boldsymbol{*}}$ and displayed in Table 5.5, giving the point estimate and standard error displayed as Estimate and Std.Error respectively. The results are repeated in Table 6.3 .

	Estimate	Std.Error
Int	0.0663	0.0192
$\hat{\beta_1}^*$	0.3074	0.2122
$\hat{\beta_2}^*$	−3.7811	0.4358
$\hat{\beta_3}^*$	−0.7791	0.4468
$\hat{\beta_4}^*$	−0.7927	0.3187
$\hat{\beta_5}^*$	0.7709	0.3575
$\hat{\beta_6}^*$	0.1949	0.0671
$\hat{\beta_7}^*$	−0.1026	0.0268
$\hat{\beta_8}^*$	0.0554	0.0282

Table 6.3: The coefficient part of Table 5.1 repeated.

[2] See e.g. chapter 6.

The estimated variance, $\hat{\sigma}_R^2$ and $\hat{\sigma}_E^2$ therefore represent the variance in the data or the coefficients and the residual error for the model, respectively. When bootstrapping these estimated variance are used to create empirical distribution that replicate the behavior of the in-sample data and the model. The empirical distributions can then be sampled to simulate results of the regression model, what is more special conditions can be applied and their effects observed by simulation, e.g. how the fixing of some of the explanatory variables effects the development of the prediction intervals when forecasting.

6.5.2 Simulating Change In House Prices

The first simulation was done without aggregating the estimated house price change. The main objective of this simulation is to achieve prediction intervals for Model 2, i.e. estimated predicted house price change when fixing unavailable explanatory variables. Simulations were also performed for the predicted house price change for Model 1 and the Interest rate only regression. The Model 1 and Interest only simulation can validate the simulation method by comparing the results to the ones already calculated by analytical methods in sections 5.6 and 6.2. The method used to perform the estimates is presented in Algorithm 1.

Algorithm 1 estimates the prediction for the three models by bootstrapping. For the predictions where all explanatory variables are available α and γ, i.e. Model 1 and Int Only respectively, no variance of the data needs to be introduced, the residual variance is however added. Model 2 is estimated by

$$\delta_{r,n+l} = A_{n+l}^{+}\hat{\boldsymbol{\beta}}^{*} + F_n\hat{\boldsymbol{\beta}}_r^s + e_r^\delta$$

Here certain data is available A_t^+ and does therefore not need to added variance. The fixed component $F_n\hat{\boldsymbol{\beta}}_r^s$ is however altered according to empirical distribution, created by the observed dispersion of the in-sample data. More precisely by sampling $\hat{\boldsymbol{\beta}}_r^s \sim N(\hat{\boldsymbol{\beta}}^*, se(\hat{\boldsymbol{\beta}}^*)^2)$. The model is then expected to behave like model 1 and the same residual error term can be applied. The simulation starts at the prediction origin n where F_n is fixed, k describes the prediction horizon. Each prediction at time $t = n+l$ is simulated R times. The results for the three predictions, (α, δ, γ), are then summarized by taking the mean and standard

Algorithm 1 Re-sampling and bootstrapping of prediction for change in house prices.

```
$X_t^+$ describes all explanatory variables at time $t$.
$A_t^+$ describes available variables at time $t$, see Eq.(6.10).
$F_n$ describes the explanatory variables fixed at time $n$, see Eq.(6.11).
$\alpha_{r,t}$ predicted full model response, Model 1, at time $t$ and simulation $r$.
$\delta_{r,t}$ predicted fixed response, Model 2, at time $t$ and simulation $r$.
$\gamma_{r,t}$ predicted Interest rate only response, Model 3, at $t$ and simulation $r$.
$n$ Prediction origin.
$k$ Prediction horizon.
$R$ Number of simulations done.
for $l = 1$ to $k$ do
  for $r = 1$ to $R$ do

    Sample the MONA residual error as $e_r^\alpha \sim N(0, \hat{\sigma}_E^2)$
    Sample the MONA residual error as $e_r^\delta \sim N(0, \hat{\sigma}_E^2)$
    Sample the Interest rate only residual error as $e_r^I \sim N(0, \hat{\sigma}_{EI}^2)$
    Sample the coefficients as $\hat{\boldsymbol{\beta}}_r^s \sim N(\hat{\boldsymbol{\beta}}^*, \hat{\sigma}_R^2)$

    $\alpha_{r,n+l} = X_{n+l}^+ \hat{\boldsymbol{\beta}}^* + e_r^\alpha$
    $\delta_{r,n+l} = A_{n+l}^+ \hat{\boldsymbol{\beta}}^* + F_n \hat{\boldsymbol{\beta}}_r^s + e_r^\delta$
    $\gamma_{r,n+l} = A_{n+l}^+ \hat{\boldsymbol{\beta}}^I + e_r^I$

  end for
end for
```

deviation for each predicted period l, e.g. these calculations for δ are

$$
\begin{aligned}
E[\boldsymbol{\delta}_{n+l}] = \bar{\boldsymbol{\delta}}_{n+l} &= \frac{1}{R}\sum_{i=1}^{R} \delta_{i,n+l} \\
Var(\boldsymbol{\delta}_{n+l}) &= \frac{1}{R}\sum_{i=0}^{R} (\delta_{i,n+l} - \bar{\delta}_{n+l})^2 \\
se(\boldsymbol{\delta}_{n+l}) &= \sqrt{Var(\delta_{n+l})}
\end{aligned}
$$

Results

Simulations were performed using Algorithm 1, where the component of F_t are fixed at $n = 1997{:}q4$, i.e. the last in-sample period and then $F_{1997:q4}$. The prediction horizon was set to $k = 10$ giving the prediction horizon date at $n + k$

$=$ 2000:q2. Each prediction was performed $R = 10.000$ times. The results for the three models is displayed in Table 6.4.

The most interesting result from Table 6.4 is the comparison of variances for the three methods. From Table 6.4 it can be seen, as was speculated in Figure 6.9 (a), that the variance of the prediction of the estimated change in house price is a constant. The estimated variance for Model 1 is $E[se(\alpha)] = 0.0169$, for Model 2 using fixing $E[se(\delta)] = 0.0332$ and for Interest Only model $E[se(\gamma)] = 0.0226$. Which for Model 1 and Interest Only are the same as the $\hat{\sigma}_{MONA}$ and $\hat{\sigma}_{INT}$ that were estimated earlier, see Table 6.2.

The results are displayed in Figure 6.10 were the larger prediction variance, for the same confidence interval, can be clearly seen for Model 2. The analytically calculated point estimate and confidence intervals are also shown in Figure 6.10 and it can be seen that the simulated intervals and point estimates of Model 1 and Int Only fit them perfectly.

	Mean			Standard Deviation		
i	$E[\alpha_i]$	$E[\delta_i]$	$E[\gamma_i]$	$se(\alpha_i)$	$se(\delta_i)$	$se(\gamma_i)$
1998 Q1	0.0194	0.0193	0.0153	0.0167	0.0333	0.0226
1998 Q2	0.0175	0.0204	0.0151	0.0169	0.0333	0.0227
1998 Q3	0.0141	0.0212	0.0164	0.0169	0.0330	0.0226
1998 Q4	0.0134	0.0219	0.0168	0.0168	0.0329	0.0224
1999 Q1	0.0058	0.0138	0.0083	0.0170	0.0329	0.0227
1999 Q2	0.0073	0.0185	0.0121	0.0170	0.0331	0.0227
1999 Q3	−0.0066	0.0063	0.0011	0.0170	0.0329	0.0227
1999 Q4	−0.0067	0.0078	0.0018	0.0168	0.0336	0.0225
2000 Q1	−0.0227	0.0013	−0.0008	0.0168	0.0333	0.0226
2000 Q2	−0.0159	0.0056	0.0037	0.0168	0.0342	0.0226

Table 6.4: Results for the simulation according to Algorithm 1, α ,δ and γ describe Model 1, Model 2 and Int Only respectively. Prediction horizon $k = 10$.

6.5.3 Simulating The Aggregate Change In House Prices

The main conclusion taken from the previous simulation is that the variance from a prediction of house price changes with fixing according to Model 2 will result in normally distributed value with standard deviation $se(\delta) = 0.0332$ and that the variance is fixed for all prediction horizons (k).

Using the results from the previous simulation the effects the aggregation of predicted values has on the variance can now be inspected.

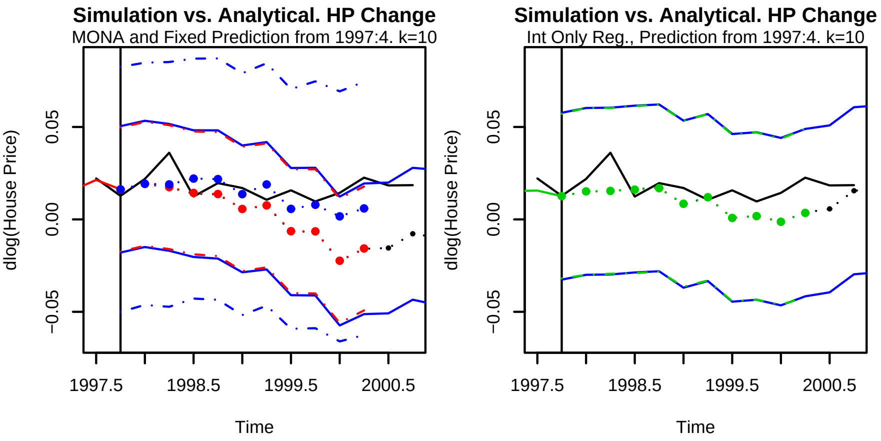

Figure 6.10: Left panel shows the simulated fixed Model 2 (blue dots) and MONA Model 1 (red dots) with 95% confidence of the prediction interval, the broken blue line is for Model 2 while the broken red line is for Model 1. Right panel show the simulated Interest Only (green dots) model with 95% confidence of the prediction interval. Black whole line is the observed change.

As was discussed in section 6.3, forecasting house prices without updating, i.e. using previous forecast as bases will lead to an increase in prediction variance. Here the increasing prediction variance will be estimated by way of bootstrapping. By using the aggregation formulas for house prices derived in Eq.(6.5) the house price will be given at each time from the estimated house price change. An empirical distribution will then be generated from the house price at that time and a sample from that distribution used as basis for next periods house price, see Algorithm 2 for more detail.

Algorithm 2 Estimating variance in aggregate house price predictions.

```
$y_{rt}$ Aggregate house price at time $t$ simulation $r$.
$u_{rt}$ is any one of three models from Algorithm 1 at time $t$ and rep $r$.
$\hat{\mu}_{y_t}$ mean value of house price at time $t$ over $R$.
$\hat{\sigma}_{y_t}$ standard error of house price at time $t$ over $R$.
$A$ initial ln(house price) at time $n$, i.e. $\ln(kp_n)$
$n$ Prediction origin.
$k$ Prediction horizon.
$R$ Number of simulations done.
for $l = 0$ to $k$ do

  for $r = 1$ to $R$ do
    if $l = 0$ then
      $y_{rn} = A$
    else
      Sample last house price $y_{r,n+l-1}$ as $p_r^* \sim N(\hat{\mu}_{y_l}, \hat{\sigma}_{y_l}^2)$
      $y_{r,n+l} = u_{r,n+l} + p_r^*$
    end if
  end for

  $\hat{\mu}_{y_{n+l}} = E[y_{\cdot,n+l}]$
  $\hat{\sigma}_{y_{n+l}} = se(y_{\cdot,n+l})$

end for
```

Results

Algorithm 2 was used to investigate the development of house price prediction intervals. The output from Algorithm 1 was used as input to the simulation performed listed below. The simulation replication was set to $R = 10.000$ for Algorithm 2 and the initial house price $A = \ln(kp_n)$ or $A = ln(kp_{1997:4})$.

Programming was performed with the statistical package R, the source code can be seen in Appendix C.2. The results for the three models is displayed in Table 6.5.

		Mean			Standard Deviation		
	t	Model 1	Model 2	Int Only	Model 1	Model 2	Int Only
n+0	1997 Q4	0.2370	0.2370	0.2370	0.0000	0.0000	0.0000
n+1	1998 Q1	0.2562	0.2566	0.2523	0.0167	0.0331	0.0228
n+2	1998 Q2	0.2736	0.2769	0.2669	0.0236	0.0471	0.0322
n+3	1998 Q3	0.2872	0.2975	0.2829	0.0291	0.0566	0.0394
n+4	1998 Q4	0.3009	0.3206	0.2998	0.0337	0.0658	0.0455
n+5	1999 Q1	0.3067	0.3351	0.3084	0.0378	0.0737	0.0507
n+6	1999 Q2	0.3139	0.3547	0.3209	0.0415	0.0811	0.0559
n+7	1999 Q3	0.3070	0.3594	0.3224	0.0446	0.0877	0.0604
n+8	1999 Q4	0.3004	0.3650	0.3238	0.0473	0.0940	0.0639
n+9	2000 Q1	0.2771	0.3664	0.3220	0.0498	0.0996	0.0674
n+k	2000 Q2	0.2604	0.3710	0.3247	0.0528	0.1053	0.0716

Table 6.5: Results for the simulation according to Algorithm 2 using Model 1, Model 2 and Int Only. Prediction horizon $k = 10$. First observation 1997 Q4 is not a forecast, initial value of house prices.

The data in Table 6.5 show the mean and standard deviation for the predicted aggregate log(house price), i.e. $\ln(\widetilde{kp_t})$. Comparing the estimated change of house price $\Delta\widehat{\ln(kp_t)}$, i.e. dlog(house price), in Table 6.4 to those in Table 6.5 it can be seen that the variance increases with prediction horizon k, as was expected see e.g. Figure 6.9.

The right panel of Figure 6.11 shows how the predictions progress from forecasting $k = 1$ period ahead up to $k = 10$ periods ahead. Although the point estimate varies greatly the variance of the predictions are only dependant on the prediction horizon or k. The dependance on k is as expected since it is an aggregation of the fixed variance of the estimated change in house prices, shown in section 6.4.

The right panel of Figure 6.11 shows the, $k = 1$ and $k = 10$, prediction distributions for all three models, centered around zero at $k = 1$ and $k = 10$. Each prediction horizon in the out-of-sample data from $k = 1, ..., 10$ has distribution as is shown in Figure 6.12, for Model 2, centered around zero, i.e. the point estimate at any time. Finally the predictions and the prediction intervals are given for $k = 10$ fixing $n = 1997{:}q4$ in Figure 6.13 with a 95% confidence intervals for the prediction.

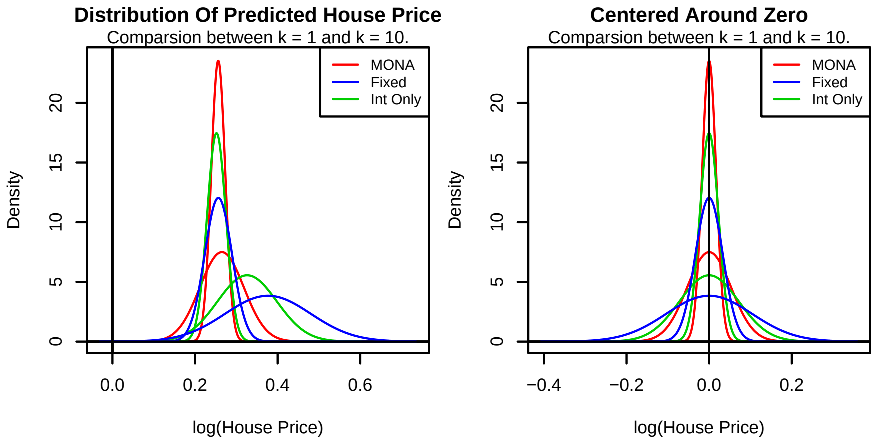

Figure 6.11: The left panel shows distribution of the forecasted house price for all three models, for one period ahead $k = 1$ and secondly for ten periods ahead $k = 10$. The right panel show the same distributions as the left only centered around zero.

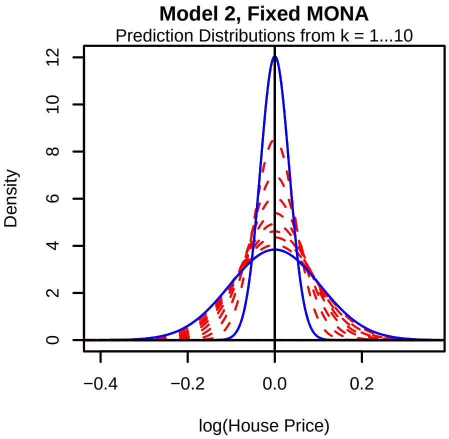

Figure 6.12: Shows the prediction distributions for $k = 1, ..., 10$ for any prediction in the out-of-sample data.

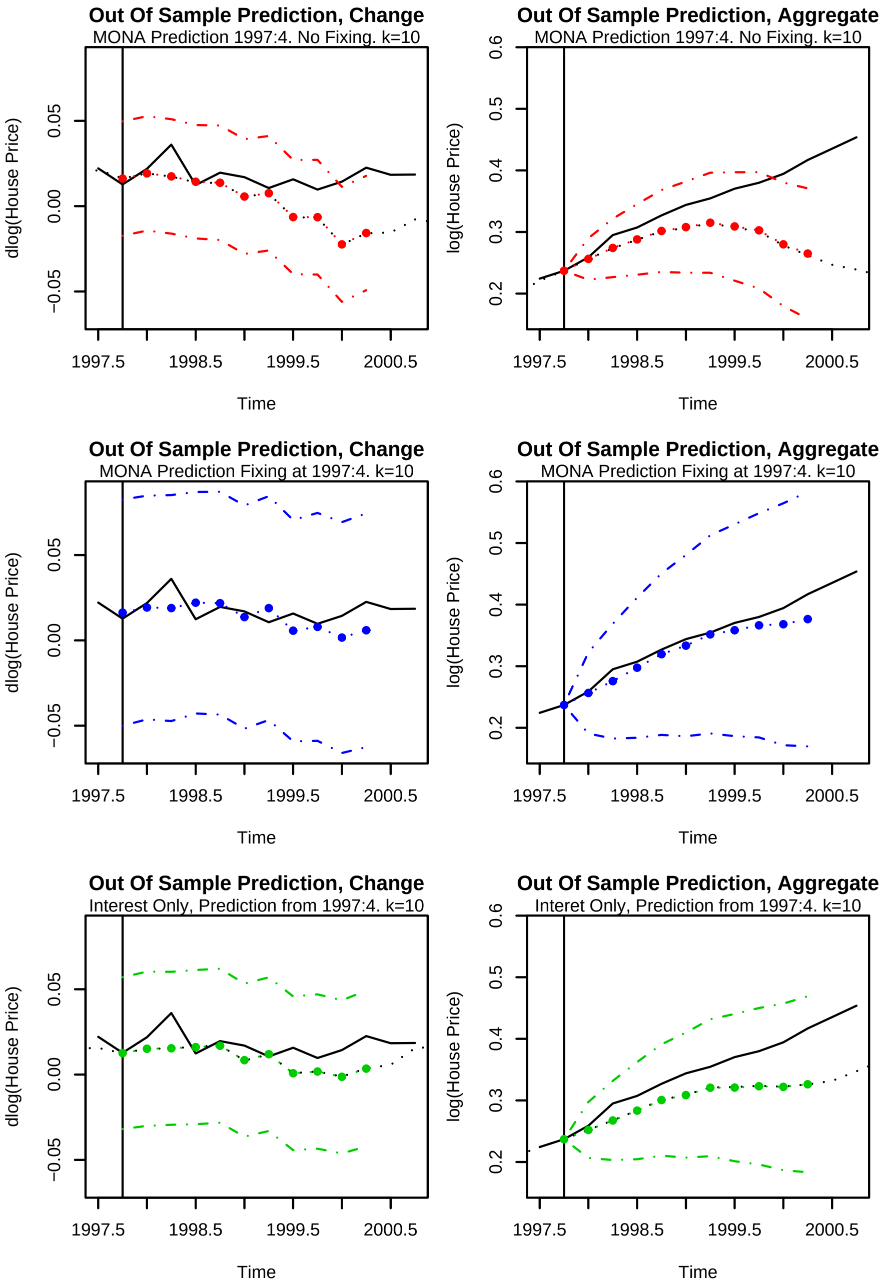

Figure 6.13: The left panels show the estimated change in house prices with $\pm 1.98\sigma$ which corresponds to about 95% confidence prediction intervals. The right panels show the estimated aggregate house price development also with 95% confidence prediction intervals.

6.5.4 Summary of Results

The result from this error estimation is that in the case of predicting for the change in house price a fixed variance can be expected, irrelevant of the prediction horizon k. The prediction can therefore be expected to have an normal distribution around it´s point estimate with a variance listed in Table 6.6. When aggregating the estimate change, i.e. calculating the actual house price without updating the prediction is also normally distributed around the point estimate, since the point estimate is essentially the accumulation of the change in the house price point estimate. The variance however increases with an increase in prediction horizon k. For any out-of-sample prediction of aggregate house prices, the prediction variance can be expected to be a function of k as listed in Table 6.4. The results for the Fixed MONA model are summarized in Figure 6.14, for $k = 1, ..., 20$.

	se
MONA no fixing of explanatory variables, Model 1	0.0169
MONA with fixing certain variables to prediction origin n, Model 2	0.0332
Interest rate only Regression	0.0226

Table 6.6: The expected variance for the prediction of change in house prices, $\hat{y}_t = \widehat{\Delta \ln kp_t}$
.

When comparing the three models the fixed model will give the highest uncertainty of the three models when forecasting. The interest rate model is second and the MONA model with all explanatory variables is likely to give the most secure prediction.

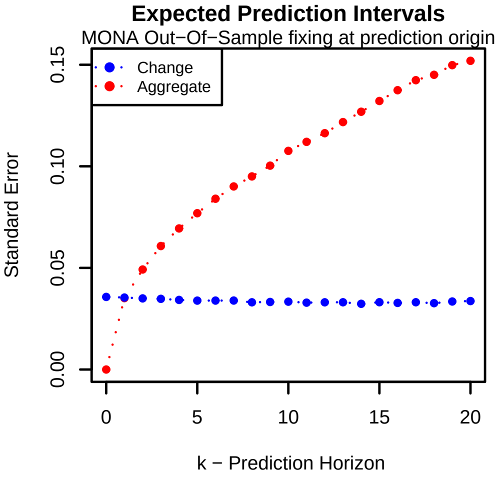

Figure 6.14: The expected variance for the Fixed MONA model as a function of prediction horizon.

Chapter 7

House Price Dynamics III Statistical Model

7.1 Introduction

In this chapter a new reduced statistical model is devised, using the error-correction model format and the data from MONA. This new model will be noted as ***HPDIII***, the new model focuses more on modeling the house price to interest rate relationship than attempting to develop a model which completely encapsulates the economic long term relationship.

In section 7.2 the outline of the ***Box-Jenkins*** statistical modeling process is presented, the section also gives a brief discussion of which steps in the Box-Jenkins framework have been investigate previously in this thesis. Section 7.3 introduces the data and uses correlation plots to decide the level of differencing and beginning level of lags to include in the model. Section 7.4 discusses how the model is reduced from the initial guess, in section 7.3, to a usable model including only the relevant terms, the parameters of the final model are also estimated, the fit plotted and goodness of fit investigated. In section 7.5 the residuals are investigated as in previous chapters to assert the model quality. Finally in section 7.6 a short summary is presented on what benefits the $HPDIII$ poses over pervious models.

7.2 Statistical Modeling

A method of modeling based on the ***Box-Jenkins*** modeling approach is applied to systematically identify, estimate and validate a statistical model for house price development. A flow diagram, illustrating the Box-Jenkins modeling procedure, is shown in Figure 7.1. The Box-Jenkins method is described by the following main ideas:

1. Identification of the data which involves asking question such as, what are the main factors, does the data need to be transformed, is the stationarity assumption a reasonable one.

2. Chose a suitable model type, to fit the data.

3. Estimate Parameters in the selected model.

4. Validate model, residual analysis and out of sample fitting.

If validation of the model fails something has gone wrong and the model must be reevaluated.

Throughout this thesis some of these rules have been applied already without mentioning the Box-Jenkins framework directly. For example the identification of the factors in the MONA house price relation, as well as theoretical model describing house price development were discussed in sections 5.3.1 and 5.4, respectively. These actions are equivalent to the first step in Box-Jenkins. Estimation of parameters and residual validation has also been performed for previous models.

The goal of this chapter is to develop a model based solely on previous levels, and differenced levels, of house prices and interest rates. In doing so the theoretical framework mentioned in section 5.4 is largely dropped. The statistical model of choice for this chapter is chosen as the error-correction model, inspired by the use in MONA. The ECM allows for the inclusion of the levels as well as the stationary differences, which ensures the long term trend is captured as well as short term dynamics.

The *HPDIII* model is meant to improve on the shortcomings of the reduced MONA models, i.e. the interest only regression model and the MONA fixed model, from chapter 6. All the house price models will be compared in the next chapter, first for single branch and later for a scenario trees.

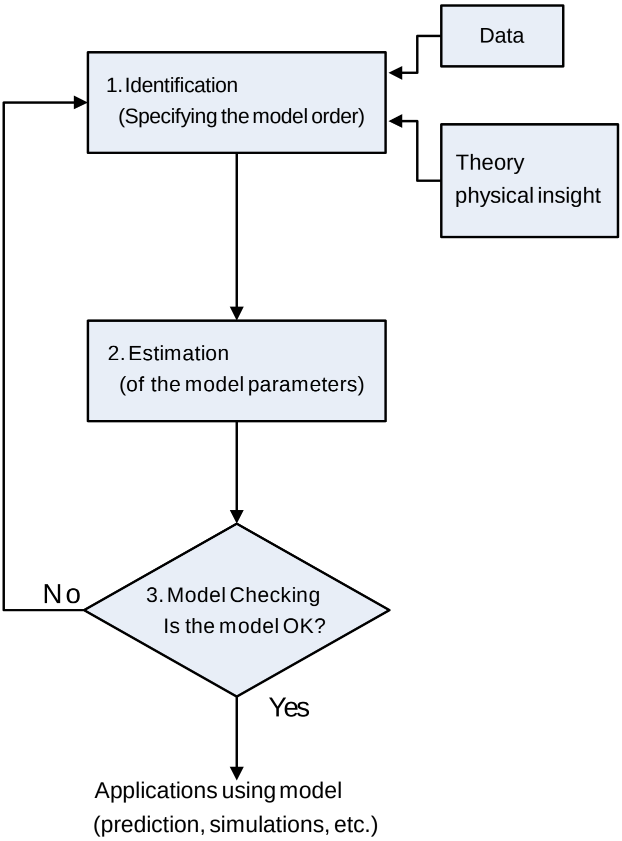

Figure 7.1: Box-Jenkins framework for statistical model building. Adopted from Madsen [9], page 148.

7.3 Data and Identification

In chapter 5 it was shown that there exists a negative relationship between house prices and interest rates. This section investigates the relationship between interest rates and house price further, with the intention of constructing an error correction model for the change in house prices.

In Figure 7.2 the level and first difference of the series House Price: $\ln(kp_t)$ and interest rates: $rente_t$ are shown. Both series are $I(1)$, i.e stationary after one level of differencing. The correlation between the levels and differenced values

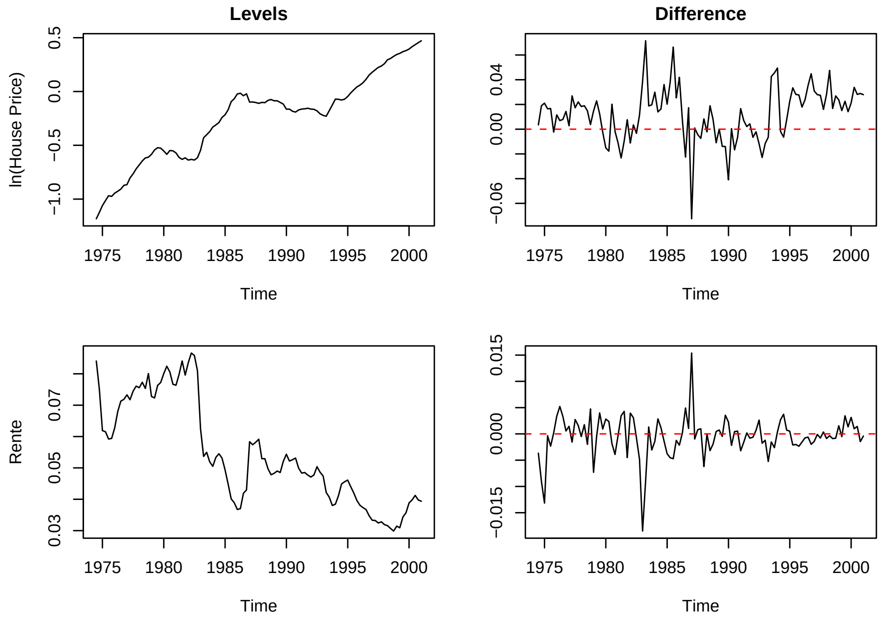

Figure 7.2: Upper left panel shows the $\ln(kp)$ i.e. log house prices. Lower left panel shows the interest rates $rente_t$, the right panels show the change in the levels on the left or the differenced series. The data spans 1974:q3-2001:q1.

is shown in Table 7.1, there it can be seen that the respond variable $\Delta\ln(kp_t)$ shows some correlation to all of the three series.

	$\Delta\ln(kp_t)$	$-\Delta rente_t$	$\ln(kp_t)$	$-rente_t$
$\Delta\ln(kp_t)$	1.000			
$-\Delta rente_t$	0.500	1.000		
$\ln(kp_t)$	0.251	−0.115	1.000	
$-rente_t$	0.356	0.050	0.835	1.000

Table 7.1: Correlation matrix for the four series used.

Investigating the correlation further, the autocorrelation and cross-correlation functions are shown in Figure 7.3. The graph diagonal in Figure 7.3 represents the autocorrelation of the four series, while the off-diagonal represents the correlation between the row and column series, called cross correlation. It can be seen from from the top line in Figure 7.3 that some significant correlation between $\Delta\ln(kp_t)$ and all three other series is present. There also seems to be some autocorrelation as can be seen in the top left panel.

From Figure 7.3 an initial guess to the level of the model can be made as including 3 lags from $\Delta \ln(kp_t)$, 2 lags from $\Delta(rente_t)$, 1 lag of $\ln(kp_t)$ and 1 lag of $rente_t$.

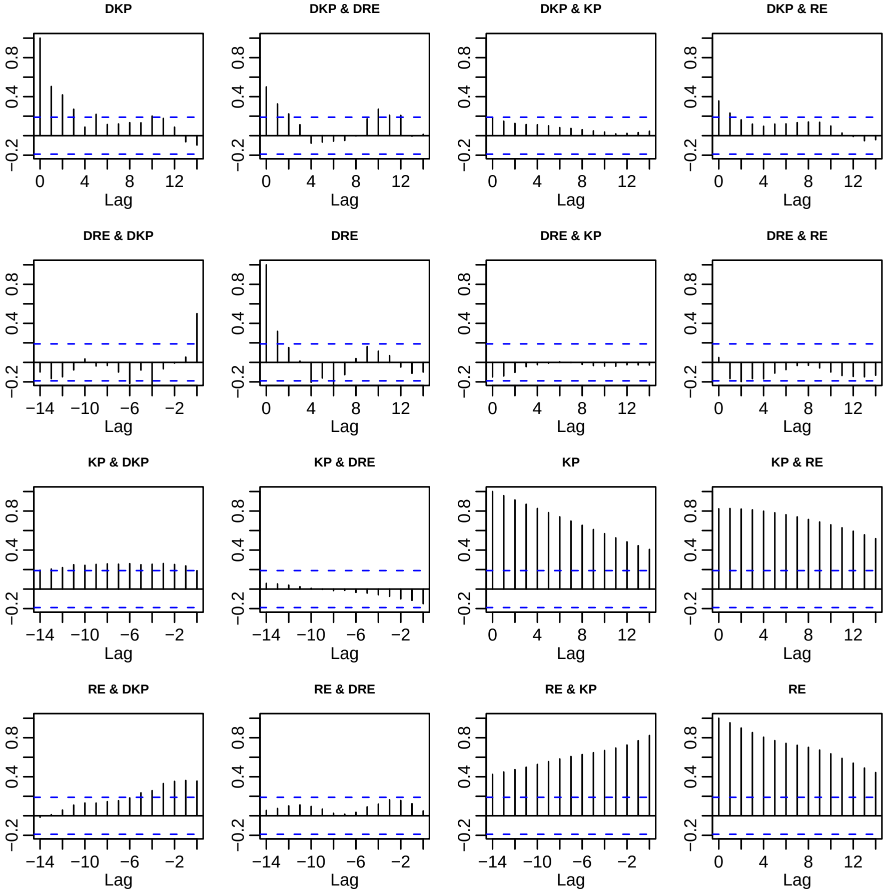

Figure 7.3: Cross Correlations between the lags of the four series, diagonal is the auto correlation functions. KP : $\{\ln(kp_t)\}$, DKP : $\{\Delta \ln(kp_t)\}$, RE : $\{rente_t\}$ and DRE : $\{\Delta rente_t\}$.

7.4 The Model

Using the information from Figure 7.3 the initial model can be expressed as

$$\begin{aligned}\Delta\ln(kp_t) &= \theta_0 + \theta_1\Delta\ln(kp_{t-1}) + \theta_2\Delta\ln(kp_{t-2}) + \theta_3\Delta\ln(kp_{t-3}) + \theta_4\Delta(rente_t)\\ &\quad + \theta_5\Delta(rente_{t-1}) + \theta_6\Delta(rente_{t-2}) + \theta_7\ln(kp_{t-1}) + \theta_8 rente_{t-1} + \varepsilon_t\end{aligned}$$

Where the parameter of interest is $\boldsymbol{\theta}$ estimated by OLS to give $E[\boldsymbol{\theta}] = \hat{\boldsymbol{\theta}}$. Some of the parameters in the initial model may be unnecessary, by estimating the parameters and removing those which are not significant, reevaluating the model, and removing the parameters again, a model including only relevant terms can be derived, the process is described in Example 7.1.

Example 7.1 (Estimation of initial model)

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.031454   0.014811   2.124  0.03660 *
Off$DKP.1    0.260881   0.104628   2.493  0.01459 *
Off$DKP.2    0.242424   0.105489   2.298  0.02401 *
Off$DKP.3   -0.009673   0.084247  -0.115  0.90886
Off$DRE     -4.115852   0.502050  -8.198 2.26e-12 ***
Off$DRE.1   -0.332784   0.661691  -0.503  0.61631
Off$DRE.2    0.879325   0.629211   1.398  0.16590
Off$KP.1    -0.029084   0.010857  -2.679  0.00887 **
Off$RE.1    -0.532613   0.255111  -2.088  0.03981 *
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

Residual standard error: 0.01872 on 85 degrees of freedom
Multiple R-Squared: 0.6294,     Adjusted R-squared: 0.5945
F-statistic: 18.05 on 8 and 85 DF,  p-value: 1.813e-15
```

The R output above is for the estimation of the coefficients in the initial model, the stars show the level of significance calculated from the $p - value$. The parameter that seems to be contributing the least to the model is `Off$DKP.3` or $\theta_3\ln(kp_{t-3})$. The next step would be to remove `Off$DKP.3`, re-estimate the parameters, and removing the "worst" parameter if there are still non-significant parameters, until all the parameters left are significant.

□

Using the process of eliminating non-significant parameters as described in Example 7.1, the following final model was derived

$$\begin{aligned}\Delta \ln(kp_t) = \theta_0 + \theta_1 \Delta \ln(kp_{t-1}) + \theta_4 \Delta(rente_t) \\ + \theta_7 \ln(kp_{t-1}) + \theta_8 rente_{t-1} + \varepsilon_t \end{aligned} \tag{7.1}$$

Estimation for the parameters in the final version of the $HPDIII$ model, Eq.(7.1), are displayed in Table 7.2. The comparison of goodness of fit statistics is displayed in Table 7.3. From the goodness of fit it can be seen that the $HPDIII$ model fits the data much better than the naive interest rate only regression, see section 6.2, and not far from the intricate MONA model, see chapter 5. The three models are compared graphically in Figure 7.4, where it can be seen that $HPDIII$ clearly manages to adapt better to the data than the interest only regression model. The $HPDIII$ also seems to adapt better to the out of sample data anomaly, which can be explained by the autoregressive nature of the $HPDIII$ model.

		Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	$\hat{\theta}_0$	0.0384	0.0140	2.75	0.0073
$\Delta \ln(kp_{t-1})$	$\hat{\theta}_1$	−0.0343	0.0106	−3.25	0.0017
$\Delta(rente_t)$	$\hat{\theta}_4$	−4.0416	0.4799	−8.42	0.0000
$\ln(kp_{t-1})$	$\hat{\theta}_7$	0.3421	0.0753	4.54	0.0000
$rente_{t-1}$	$\hat{\theta}_8$	−0.6326	0.2434	−2.60	0.0109

Table 7.2: The estimated coefficients for the $HPDIII$ model based on ECM for change in house price, estimated with ordinary leat squares (OLS). For the in-sample period 1974:$q2$ - 1997:$q4$ or 95 periods. First column is the estimate, second is the standard error of the estimate, thirdly is the t-statistic and fourthly is the p-value.

	ROLS	OLS_{Int}	$HPDIII$
R^2	0.6920	0.4156	0.6028
R^2_{adj}	0.6672	0.4029	0.5849
$\hat{\sigma}$	0.0169	0.0226	0.0189

Table 7.3: Comparison of the the goodness of fit, R^2 and R^2_{adj}, for the MONA house price relation (ROLS) and the reduced interest rate only regression (OLS_{Int}) as well as the $HPDIII$ model estimated above.

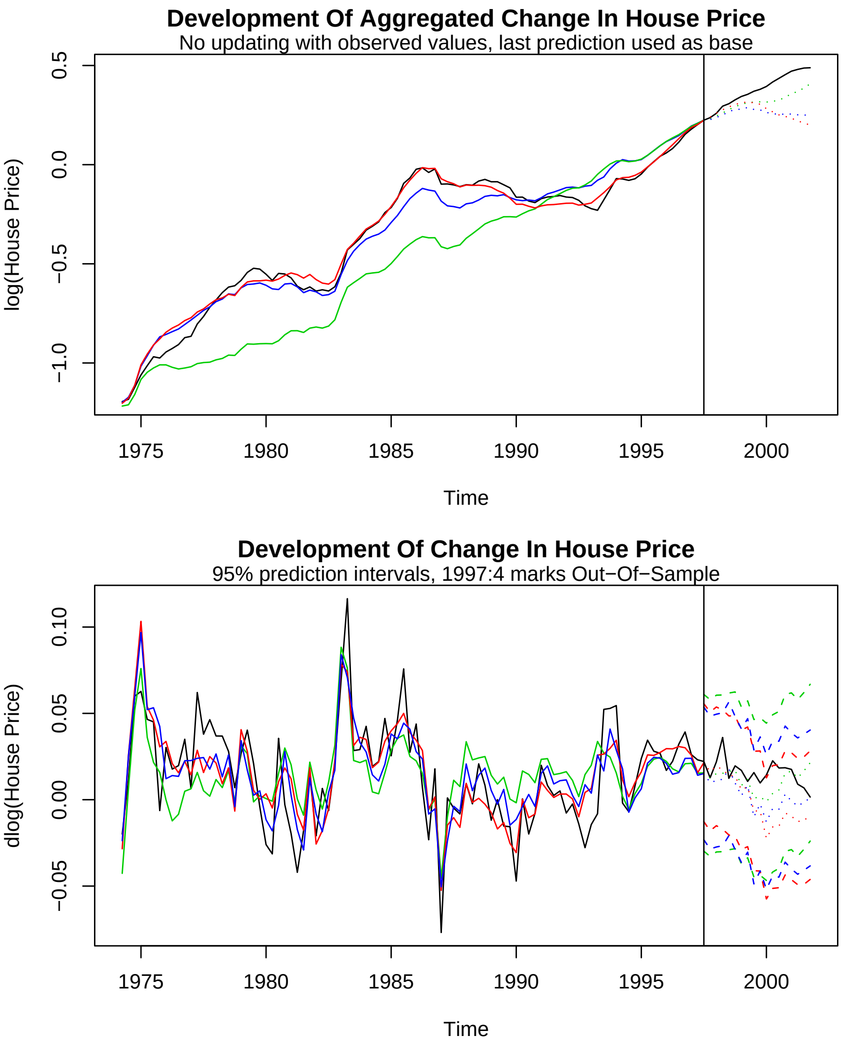

Figure 7.4: The bottom graph shows the development of the modeled variable $\Delta ln(kp_t)$. The upper graph shows the aggregated change without updating. The red line is the MONA ROLS, green line is the interest only model described in 6.2, blue is the *HPDIII* model and black is the observed change. The black vertical line represent the boundary between the in-sample and out-of-sample periods.

7.5 Residual Analysis

Same as in sections 5.5.1 and 6.2 the residuals are investigated to assert the model dependability. The residual graph can be seen in Figure 7.5. From the residual plot there appears to be no apparent auto correlation from examining the left panels. The cook plot shows that no outliers are causing trouble and the QQ-plot, indicates normality.

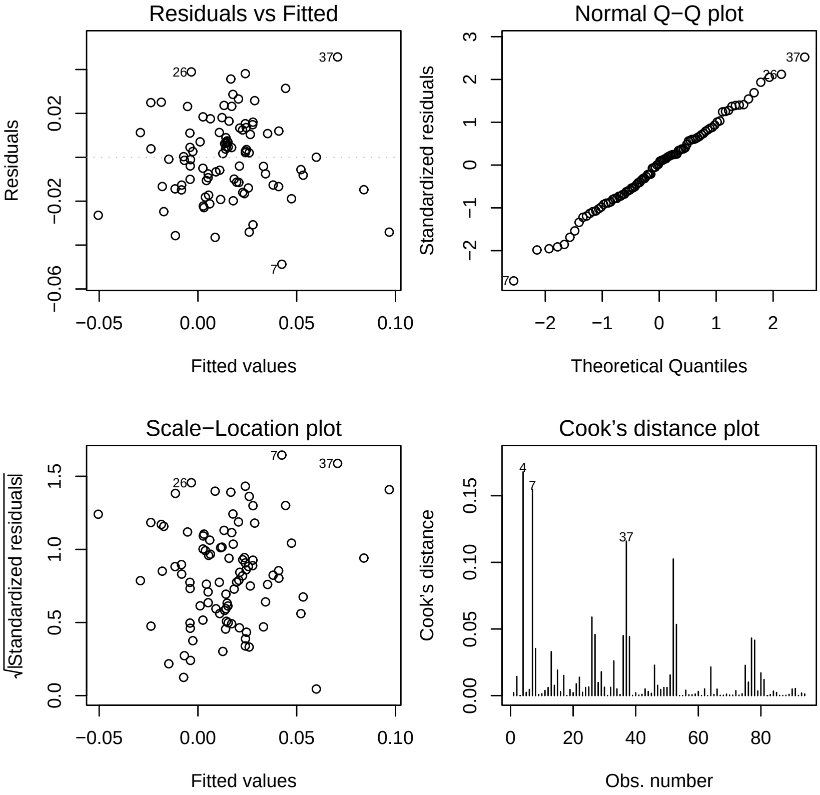

Figure 7.5: Visual residuals analysis from the $\boldsymbol{e} = \mathbf{y} - \hat{\mathbf{y}}_{\mathbf{ECM}}$.

The two test performed in previous chapters i.e. DW-test and JB-test, see subsection 5.5.1, are also conducted to investigate the behavior of the residuals. The Durbin Watson gives $DW = 1.8017$ and a p-value $= 0.1603$ which means that the hypothesis of no-autocorrelation in the residuals cannot be dismissed. The fact that there may be autocorrelation in the residuals can be explained by the fact that important systematic effect such as income and stock of houses are omitted. The Jarque Bera test gives a value $JB = 0.1075$ with a p-value $=$

0.9477 which indicates the hypothesis that the residuals are normally distributed can not be dismissed for any reasonable level of confidence. The JB along with the QQ-plot indicates that the residuals can be considered normal.

The residual for the $HPDIII$ and interest only regression is shown in Figure 7.6, upper panels. There appears to be quite a bit of autocorrelation in the naive Interest rate only regression model, see lower left panel. The $HPDIII$ residual shows signs of small significant autocorrelation on lags 2 and 5. The autocorrelation can be remedied by modeling the residual, that sort of modeling is called moving average (MA). However, since there is very little autocorrelation, in $HPDIII$ and adding a MA term increases complexity considerably the small autocorrelation is disregarded. In the case of the interest only regression model, MA terms would have to be added to given a sensible prediction. For more information about MA see Madsen [9].

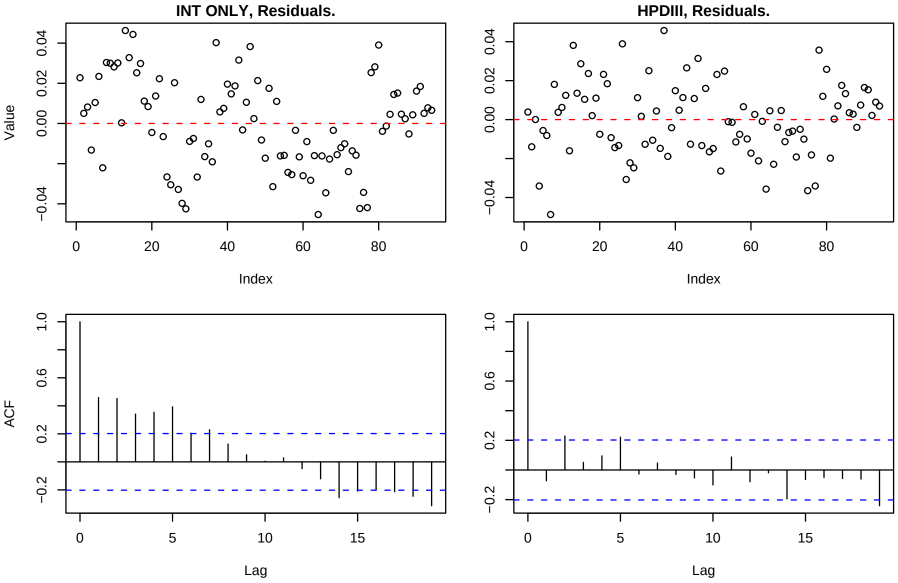

Figure 7.6: Residual time plot and corresponding autocorrelation plots for the interest only regression, section 6.2, and $HPDIII$.

7.6 Summary

The $HPDIII$ model estimated in this chapter is a more easily manageable model than the alternative i.e. MONA fixed model. The $HPDIII$ model has a lower estimated error, $HPDIII = 0.0189$, than the Fixed MONA, Fixed = 0.033. The $HPDIII$ model also has some practical advantages to the Fixed MONA model, such as it is not dependant on as many variables. The downfalls of the $HPDIII$ are that is seems to show some signs of autocorrelation and it needs calibration to the prediction origin, same as the Fixed MONA.

The next chapter compares all the models and applies the best ones to a scenario tree structure.

Chapter 8

Validation and Results

8.1 Introduction

In previous chapters numerous house price models, most based on the MONA house price relation have been devised. So far model checking has mainly been performed by residual analysis. Another important aspect of model checking is called validation, i.e. checking the prediction performance of the models. The main purpose of this chapter is to remove the benchmark models by comparing the models through validation and then apply the models which pass the validation to a house price scenario tree.

In section 8.2 the models are compared with different prediction horizons for a single path or time line, the prediction capabilities of the different models are discussed and the pros and cons of the models listed. Section 8.3 extends the one path results by implementing the models which capture house price behavior from interest rates. Using interest rate scenario trees, house price scenario trees are produced. The house price trees are validated using observed interest rates and house prices. Finally in section 8.4 the results of the chapter are summarized for both cases.

8.2 One Path Validation

In this section the four models, inspected in previous chapters, i.e. the full MONA, the Fixed MONA, interest only regression and HPDIII (ECM) are compared for one path, or time line, validation. The purpose of the validation is to see how the prediction changes with increased prediction horizon and to compare the model together. The models that pass the validation will then be implemented to a scenario tree structure in the next section.

Validation involves seeing how the model performs, given new explanatory variables, i.e. how well the model predicts for new explanatory observation. This sort of validation was performed in section 5.6 where, because of discrepancy between the in-sample and out-of-sample data, the model was shown to deliver poor results.

Since the out-of-sample data is not suited for validation, see subsection 5.6.1, the in-sample period is used. In-sample validation has some disadvantages and numerical results should be taken with reserve. The main downfalls of using the in-sample period is that it is the same period as used for parameter estimation, which will give a very good fit for validation, in fact a too good or misleading fit.

Although the in-sample numerical results of the validation should not be taken at face value, the validation can still give indications to the quality of the models. More precisely the validation can be used to compare the models to each other, the in-sample validation will also show which models are truly capturing the house price by changing the initial point of the validations.

8.2.1 The Validation

The validation is performed as follows, all the models have all explanatory information available. Instead of using the whole period from 1974:q2-1997:q4, the data is incremented in small periods and a new prediction is performed, this way it can be seen whether the model captures the house price or diverges, which would be a cause of model inadequacy. The error between the observed house price and the predicted value is measured by the mean square error (MSE), see Eq.(5.15). Two ways of measuring the error are used, first the MSE is calculated as function of different prediction horizon k, i.e. how much error can be expected when predicting $1, ..., k$ periods ahead. Secondly the sum of the mean square error or the total error of the k prediction is calculated.

Two different prediction horizons are considered for the in-sample validation, first a $k = 5$ period ahead prediction. The prediction origin is also incremented by one period through the in-sample period, also known as a rolling time window. The second prediction is a long term or $k = 20$ prediction, also incremented by one period through the in-sample period.

The results for the five steps ahead in-sample prediction, or validation, can be seen in Figure 8.1. Notice how the rolling window progresses through the in-sample data, predicting $k = 5$ periods ahead, then incrementing the prediction origin and performing a new prediction. From 8.2 it can be seen that the green line or interest only regression seem not be capturing the dynamics of the house price, but only the upward trend of the model. The red and cyan, full MONA and $HPDIII$ respectively, seem to capture the drift and the dynamics relatively well throughout the in-sample prediction. The blue line or the Fixed MONA model also seems to capture the house price well, for such a short horizon. The Fixed MONA however shows that it does not cope well with dynamic changes, which can be expected since 5 of 8 explanatory variables are fixed. The results

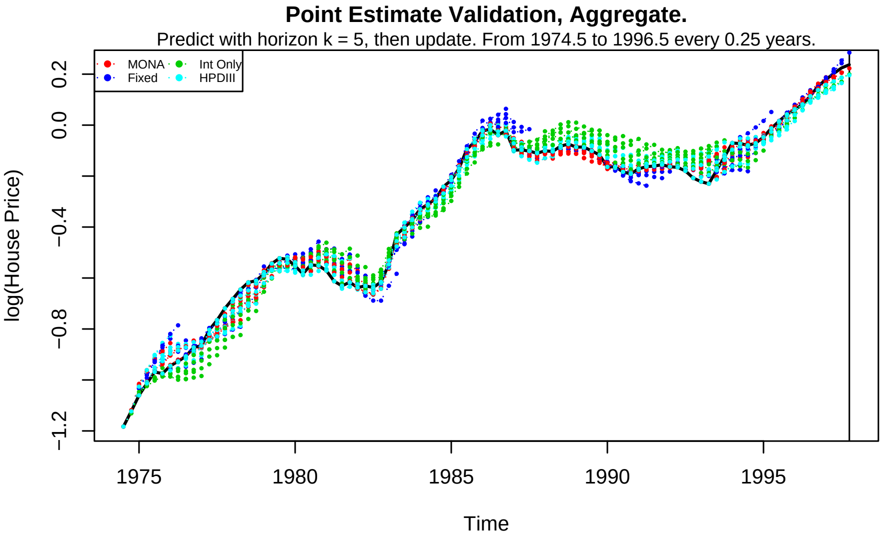

Figure 8.1: The in-sample forecast or validation for a horizon $k = 5$. The red line is the Full MONA model, the blue line is the Fixed MONA model, the green line is the interest Only regression and the cyan line is the HPDIII or ECM model.

from Figure 8.1 are summarized graphically in Figure 8.2. The left panel shows a scatter plot where each dot represent the aggregate squared error for a prediction initiated at time t, the lines show the mean error that can be expected for a $k = 5$ prediction. The green line, interest only regression, gives the highest error

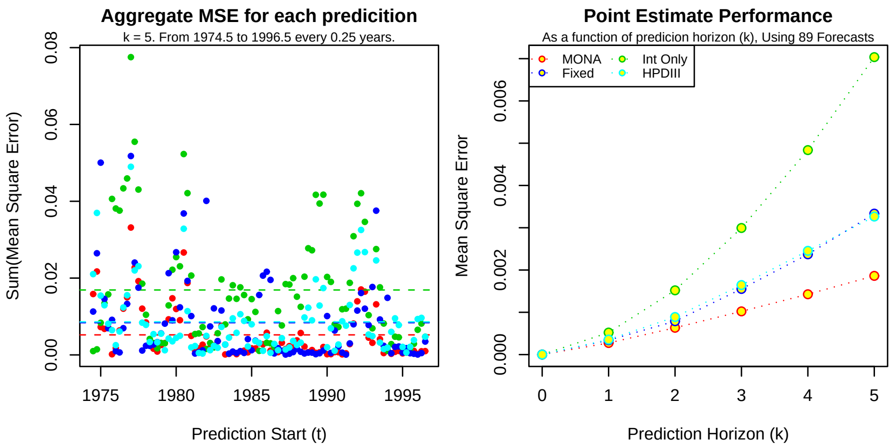

Figure 8.2: Left panel shows the aggregate sum of squares for each forecast, the lines show the mean of those forecasts. The right panel show the estimated mean square error for prediction horizon l=1,...,k.

followed by HPDIII and Fixed MONA shown as cyan and blue, respectively. The red line, which represent the Full MONA model, has the lowest error.

The right panel of Figure 8.2 shows the mean square error, from the predictions in Figure 8.2, expressed as a function of prediction horizon. The error increases with prediction horizon, for all the methods, as can be expected. The interest only method however seems to be giving the highest error for the $k = 5$ horizons, the $HPDIII$ and Fixed MONA giving very similar results and the full MONA capturing the house price the best.

Performing the prediction again using a horizon of $k = 20$ as a long term prediction, i.e. $20 * 0.25 = 5$ years ahead. The results for $k = 20$ are shown in Figure 8.3, comparing the $k = 20$ and $k = 5$, in Figure 8.1 it is obvious that for longer predictions some of the methods seem to be diverging quite a bit from the observed value, which can be expected for methods where no updating is used.

The results from the $k = 20$ in-sample prediction are summarized in Figure 8.4. The left panel shows that the interest only regression method give the worst aggregate error for the $k = 20$ prediction. The right panel however shows that the Fixed MONA model has exceeded the interest only regression model after $k = 17$.

The Interest rate only method obviously only captures the drift, as can be seen

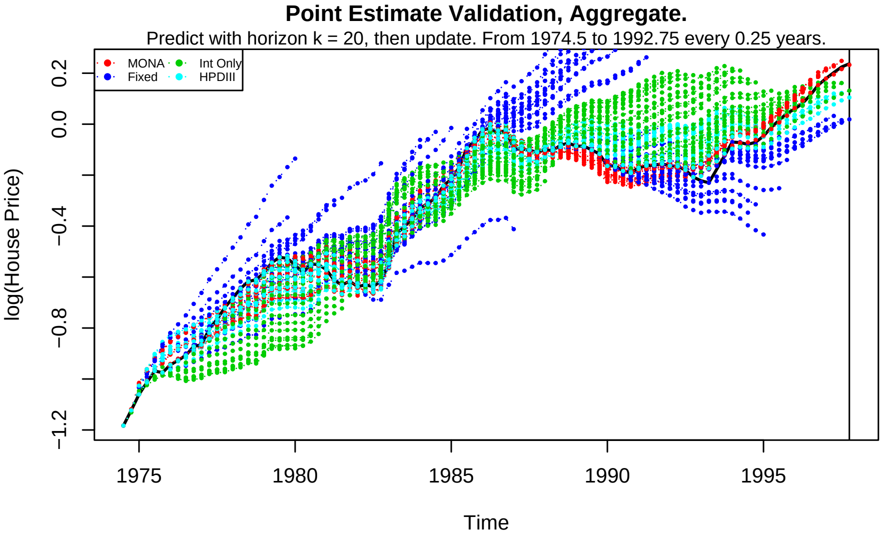

Figure 8.3: The in-sample forecast or validation for a horizon $k = 20$. The red line is the Full MONA model, the blue line is the Fixed MONA model, the green line is the interest Only regression and the cyan line is the HPDIII or ECM model.

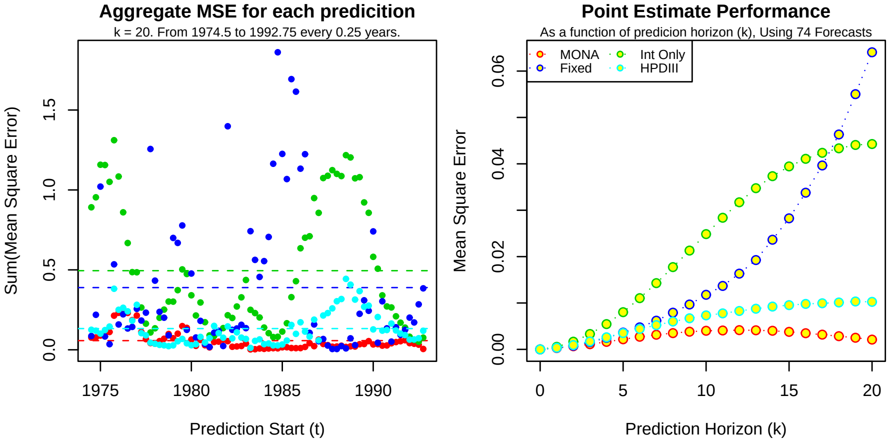

Figure 8.4: Left panel shows the aggregate sum of squares for each forecast, the lines show the mean of those forecasts. The right panel show the estimated mean square error for prediction horizon l=1,...,k.

by the constant upward trend. The reason for the poor performance of the interest only regression can be explained by the fact that the levels, both house price and interest rates, are not included leaving only the constant to capture the trend, the constant seems however not versatile enough to capture the dynamics of the trend and over fits the house price.

The full MONA model gives the best performance and the smallest error. However, as has been mentioned before not all data is available for the MONA model. The closest match is the Fixed MONA model which seems to perform well for short prediction horizons $k = 5$ but diverges away with increased prediction horizon. The fixing of the explanatory variables, is equivalent of adding a fixed amount to the constant, i.e. fixing the course of the process. The alternating explanatory variables, interest rates, then oscillate around the course set by the fixed variables or new constant. This explains why for long periods, the Fixed MONA house price model may diverge from the observed house price. The model does not have the capability to respond to large dynamic changes. However, by estimating the prediction interval as was done in section 6.5.2, the Fixed MONA model can be applied.

The $HPDIII$ or error-correction method, also seems to capture both the trend and the short term effects relatively well. It does not only represent the trend, as the Interest rate only regression method does for example.

8.2.2 Nykredit Relation

The Nykredit relation from chapter 3 was also compared to the house price data from the MONA model. Two extreme scenarios were considered for the Nykredit relation, first a one period forecast with updating, i.e. $k = 1$, and secondly a prediction for the whole period without updating or $k = 120$. The results for these two validations can be seen in Figure 8.5.

From Figure 8.5 it can be seen that for the $k = 1$ the Nykredit relation performs well with a very high level R-square of around $R^2 \approx 0.99$. However, in this model the unit-root non-stationarity has been overlooked, which deems the model useless for predictions without updating. The long term prediction shows that when the model does not get observed values for updating it performs very poorly, see red line in Figure 8.5.

The conclusion from the validation of the Nykredit relation is that non-stationarity of house prices is not considered, resulting in a useless prediction model except for very short horizons, e.g. $k = 1, 2$. This conclusion for the Nykredit model is the same as discussed in section 3.7.

Despite the downfalls of the Nykredit model it was useful for developing and

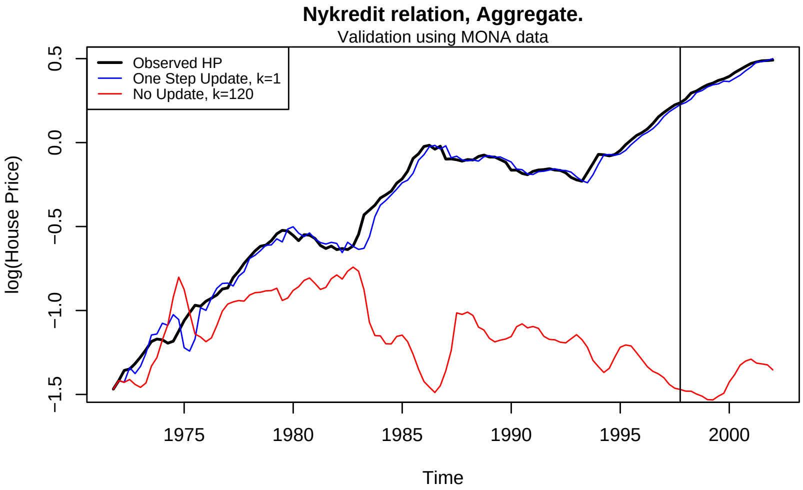

Figure 8.5: Using the MONA house price data to validate the Nykredit relation. Blue line shows the one step prediction with updating, the red line shows the Nykredit relation using previous prediction as bases for new predictions.

understanding the more complex relations, it was especially useful as a starting point for the programming conducted, which later was extended to the more elaborate models quite easily.

8.2.3 Cross Validation

An alternative to the in-sample validation could be to use cross validation. Cross validation in this case could be achieved by dividing the in-sample period into two smaller periods, then estimate the parameters on one part of the data and validate on the other. Cross validation for this data set however, like the in-sample validation, has some drawbacks. The main of which is that the number of observations are rather low for estimation and validation, if the cross validation method would be applied.

The idea behind cross validation is to validate the model structure irrelevant of placement in data, i.e. validating the terms in the model and not focusing so much on specific estimation of the parameters. Obviously for this to work the data has to be quite homogenous, which is not the case for the house prices.

8.3 Scenario Tree Validation

In this section house price scenario trees are developed from the one path versions of the Fixed MONA and $HPDIII$ models. Since it was shown that the Nykredit and interest only relations do not capture the house price for one path, except for very short predictions they are not applied to the tree structure.

The section is structured as follows. First a short description of how to extend the two models to the scenario tree structure. Secondly a short discussion of the input interest rate scenario trees and a discrepancy in time steps. Thirdly the interest rate trees are applied to give house price trees, and the results inspected and discussed.

8.3.1 House Price Formulas For Scenario Trees

Given a scenario tree of interest rates, and applying each path from that tree as single path in the house price models, a house price scenario tree can be derived.

As was discussed in section 6.3.2 the response of interest is the house price level, not the change, the models results must be accumulated.

$$\ln(\widetilde{kp}_{t,n}) = A + \sum_{i=1}^{t} \Delta \widehat{\ln(kp_{t,n})} \tag{8.1}$$

where $\ln(\widetilde{kp}_{t,n})$ is the aggregate estimated house price at time t and node n. A is the initial house price index at prediction start, set to some intuitive value e.g. $A = \ln(100)$. The term $\Delta \widehat{\ln(kp_{t,n})}$ represents the estimated change in house price, which is represented by the two models

Fixed MONA model Fixed 5 of 8 at time $t = T$

$$\begin{aligned}
\Delta \widehat{\ln(kp_{t,n})} = {} & \hat{\beta}_0^* + \hat{\beta}_1^* \Delta \ln(pcp_T) + \hat{\beta}_2^* \Delta(rente_{t,n} + ssats_T) \\
& + \hat{\beta}_3^* \Delta(rente_{t-1,a(n)} + ssats_{T-1}) \\
& + \hat{\beta}_4^* (rente_{t-1,a(n)} + ssats_{T-1} + 0.01) + \hat{\beta}_5^* dpcpe_{T-1} \\
& + \hat{\beta}_6^* dkpe_{T-1} + \hat{\beta}_7^* \ln(kp_{T-1}/pcp_{T-1}) \\
& + \hat{\beta}_8^* (\ln((ydp_{T-1} - ipv_{T-1})/pcp_{T-1}) - \ln(fwh_{T-1}))
\end{aligned}$$

HPDIII (ECM)

$$\begin{aligned}\Delta \widehat{\ln(kp_{t,n}}) = \hat{\theta}_0 + \hat{\theta}_1 \Delta \ln(kp_{t-1,a(n)}) + \hat{\theta}_4(\Delta rente_{t,n}) \\ + \hat{\theta}_7 \ln(kp_{t-1,a(n)}) + \hat{\theta}_8 rente_{t-1,a(n)}\end{aligned}$$

Notice that for the models above there are only two variables i.e. house price (kp) and interest rate ($rente$), which are node dependant, all other variables are fixed for all nodes $\mathcal{N}_t$ to their value at time $t = T$.

The assumption is made that all data is available before $t = 0$, i.e. before the prediction start, and can be used as correct input for the first node. There after the estimates are used, so there is no updating with observed values.

8.3.2 Interest Rate Scenario Trees

The input variables to the house price trees are interest rate trees generated with a variation of the Vasicek interest rate model, generation of interest rate scenario trees is out side the scope of this thesis, for more detail see Jensen and Poulsen [5].

The input data used for validation are scenario trees of interest rates, where the bonds have a maturity of 0-10, 15, 20, 25 and 30 years. The interest scenario trees are in yearly increments, while the house price models use quarter yearly steps, so to use the estimated models an interpolation is applied to the paths of the interest rates, to get quarterly rates usable in the models. An explanatory diagram of the interpolation is shown in Figure 8.6. To the left of $t = 0$ the observed MONA data is available for model initialization, after $t = 0$ the interest rates are provided yearly and must be estimated quarter yearly with interpolation, giving the small nodes on each path. The horizon on the input interest rate trees is 5 years which is equivalent to $5 \cdot 4 = 20$ in the quarterly model, i.e. the interest rate scenario trees are correspond to a $k = 20$ prediction tree for the house price tree.

Having many bonds with different rates is different to the MONA model where only one rate is used. The structure of the interest term *rente*, used in the estimation of the MONA model, compared to the rates used for input here is not exactly known. The *rente* term will be plotted together with the bond scenario trees to see a comparison between the rate modeled as the "true" rate in house prices, and the input generated rates.

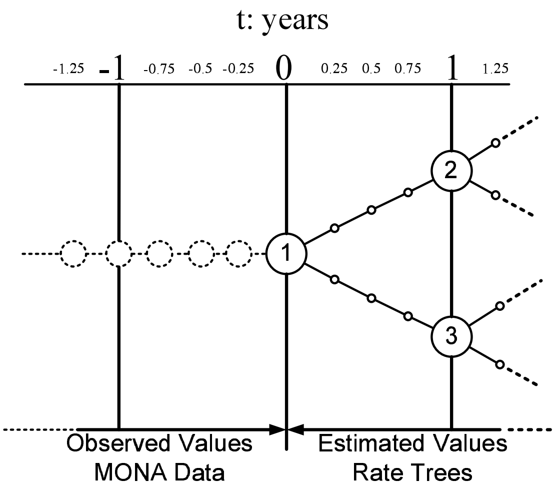

Figure 8.6: Example of a linear interpolation from a yearly data to get quarter yearly data, for a binomial.

8.3.3 Results

The validation performed here is a way of seeing if the house price scenario trees capture the house price, given the house price models and a scenario tree of estimated interest rates.

Three models were initially applied to the scenario tree structure for validation, i.e. Fixed MONA, *HPDIII* and interest Only regression. However, both the one path and preliminary scenario tree show the interest only model to perform poorly. The results for the Int only regression are omitted here, but shown in Appendix B.1.

The next three pages show the scenario trees for the 1995 – 2005 interest rates and the corresponding estimated house prices.

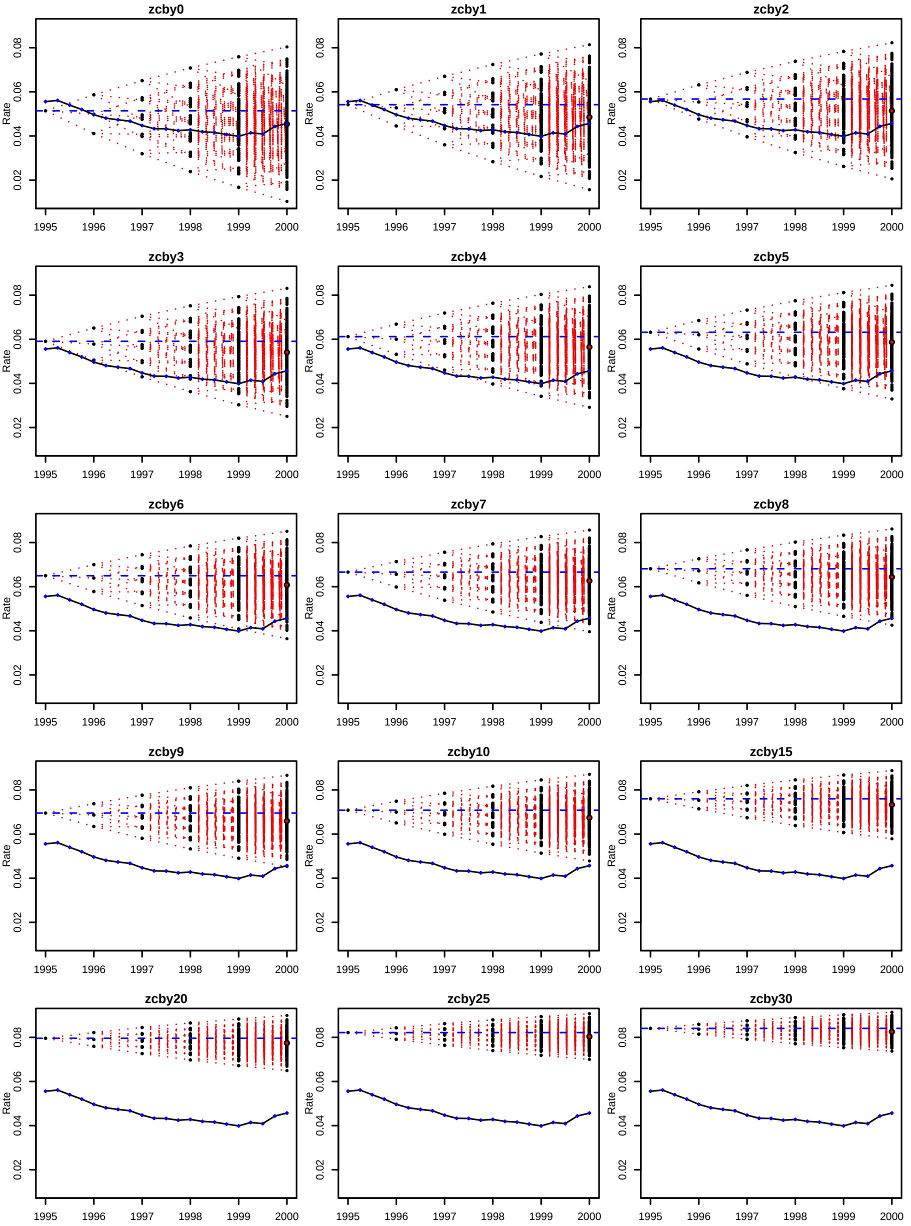

Figure 8.7: Interest rate scenario trees estimated from 1995 – 2000, zcbyXX where XX corresponds to the time to maturity on the bonds, 0-10,15,20,25 and 30 years. The blue line is the development of the MONA interest term *rente*.

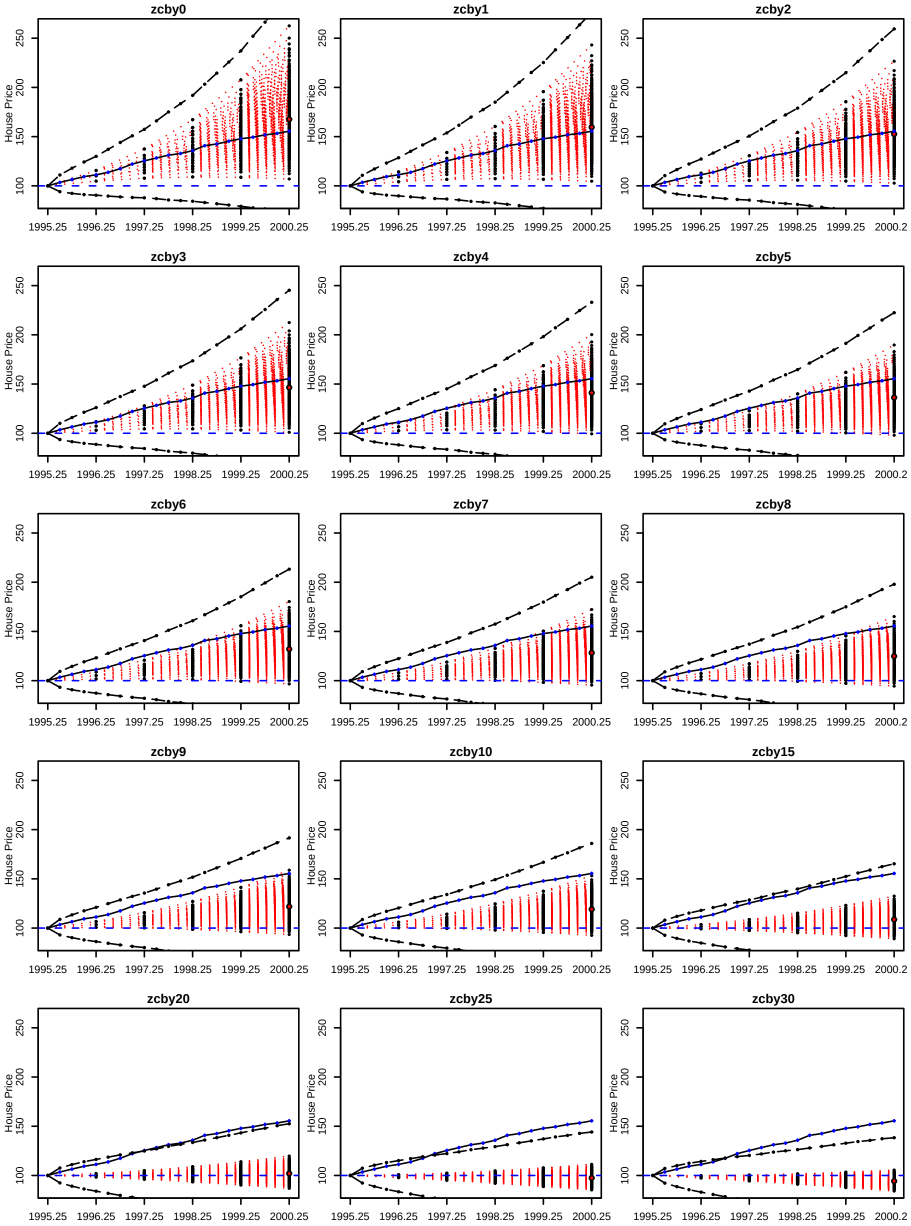

Figure 8.8: Estimate house price using the FIXED MONA method, each panel corresponds to the interest scenario tree in with same header from Figure 8.7. The blue line describes the observed house price. The broken black lines are the prediction error bars for the extreme paths, with 95% confidence interval.

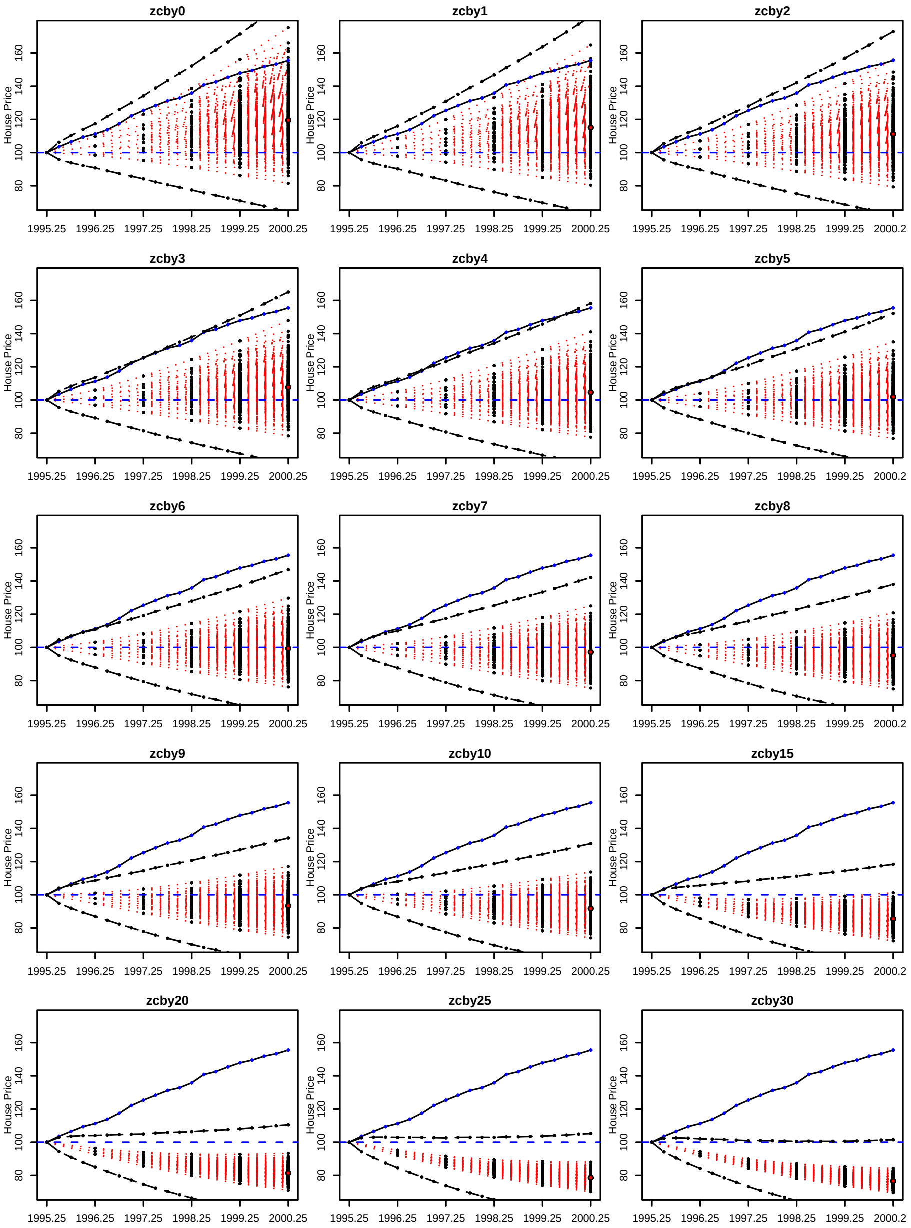

Figure 8.9: Estimate house price using the HPDIII (ECM) method, each panel corresponds to the interest scenario tree in with same header from Figure 8.7. The blue line describes the observed house price. The broken black lines are the prediction error bars for the extreme paths, with 95% confidence interval.

Figure 8.7 shows the estimated scenario trees with the observed MONA interest rate for comparison. From the figure it can be seen that with increasing bond maturity the mean level of interest rates increases while the variance or volatility decreases. For $zcby30$, i.e. the 30 year bond, the rate has a relatively low volatility and a high mean of ca. 8% which is considerably higher than the MONA rate. The MONA rate seems to be decreasing in this period $1995-2000$ and the rate trees do not seem to represent the rate particularly, the MONA rate might be a downward path in the $zcby0-5$ bond scenario trees, i.e. the short term bonds. For the other scenario trees the MONA rate seems represent a substantiality lower rate than shown by the trees.

Figure 8.8 shows the response from the Fixed MONA model given the corresponding scenario trees in Figure 8.7 as input, the broken black lines show the error bars as calculated in section 6.5, with $k = 1, ..., 20$. The Fixed MONA model captures the house price well for the short term bonds, where the MONA rate was also captured. However, the volatility of the house price at horizon is quite high, the most extreme being a rise from 100 to 250 in five years, with a range from ca. 300-80, with 95% prediction horizon. This high volatility can however be expected from the Fixed MONA model for long predictions, as was discussed in the one path validation in section 8.2. What is more, if the predication origin were to be shifted slightly it might have a considerable effect since the variables would be fixed to new levels. Obviously the long term bond trees are not expected to yield good house price results, since the corresponding interest rate trees do not capture the MONA rate which the models uses to describe the interest rate to house price relation.

Figure 8.9 gives the results from the $HPDIII$ model given the interest rate scenario trees in Figure 8.7. The $HPDIII$ does not seem to be capturing the house price as well as the Fixed MONA. The house price at horizon however has a much smaller volatility. In the cases where the $HPDIII$ model captures the house price is on the extreme paths, more precisely the maximum house price path. The house price response is not so strange since the MONA rate is non-increasing throughout, and usually close to the lowest interest rate path, which in turn should give the max house price path in the house price model.

The period $1995-2000$, which is inspected in Figure 8.7, is not well suited for validation because of the constantly increasing house price. Recall from subsection 5.6.1 that during this period the data shows abnormal behavior and the response breaks away from the information of the explanatory variables. Even though the data is not ideal there are two main results that can be deduced from this validation

1. The house price model respond directly to the volatility of the interest

rate trees, i.e. if there is a large variance of rates at horizon there is also a large variance of house prices at horizon.

2. A second interesting observation is how the house price trees respond to the level of interest rate, if the rate is on average high such as for 30 year bond ($zcby30$) the house prices will yield a downward house price, which is in accordance with the economical theory of high interest will show a decline in house prices. This crucial relationship between the interest level and and the trend of house prices is captured by both the Fixed MONA model as well as the $HPDIII$ model, the interest only regression however does not capture this behavior, see Figure B.1.

Another experiment is conducted by approximating the interest rate trees to another time. That is, instead of being from 1995-2000, the scenario trees are noted as 1989-1994, with corresponding MONA interest rate and observed house price.

As can be seen in Figure 8.10 during the 1989-1994 period there seem to be more variation in the MONA rate, than the downward 1995-2000 rate, what is more the fixing of the MONA model does not give an extreme addition from the fixed variables resulting in the fixed model capturing the house price very well, see Figure 8.11. The $HPDIII$ form is the same as before since it is only dependant on the input interest rate tree. However, where the interest rate trees capture the MONA rate, the house price trees seems to capture the house price.

8.3.4 Prediction Errors

The errors or estimated prediction intervals were estimated according to Algorithm 2, in subsection 6.5.3, for $k = 20$ and the results for all three methods are listed in Appendix B.2, Table B.1.

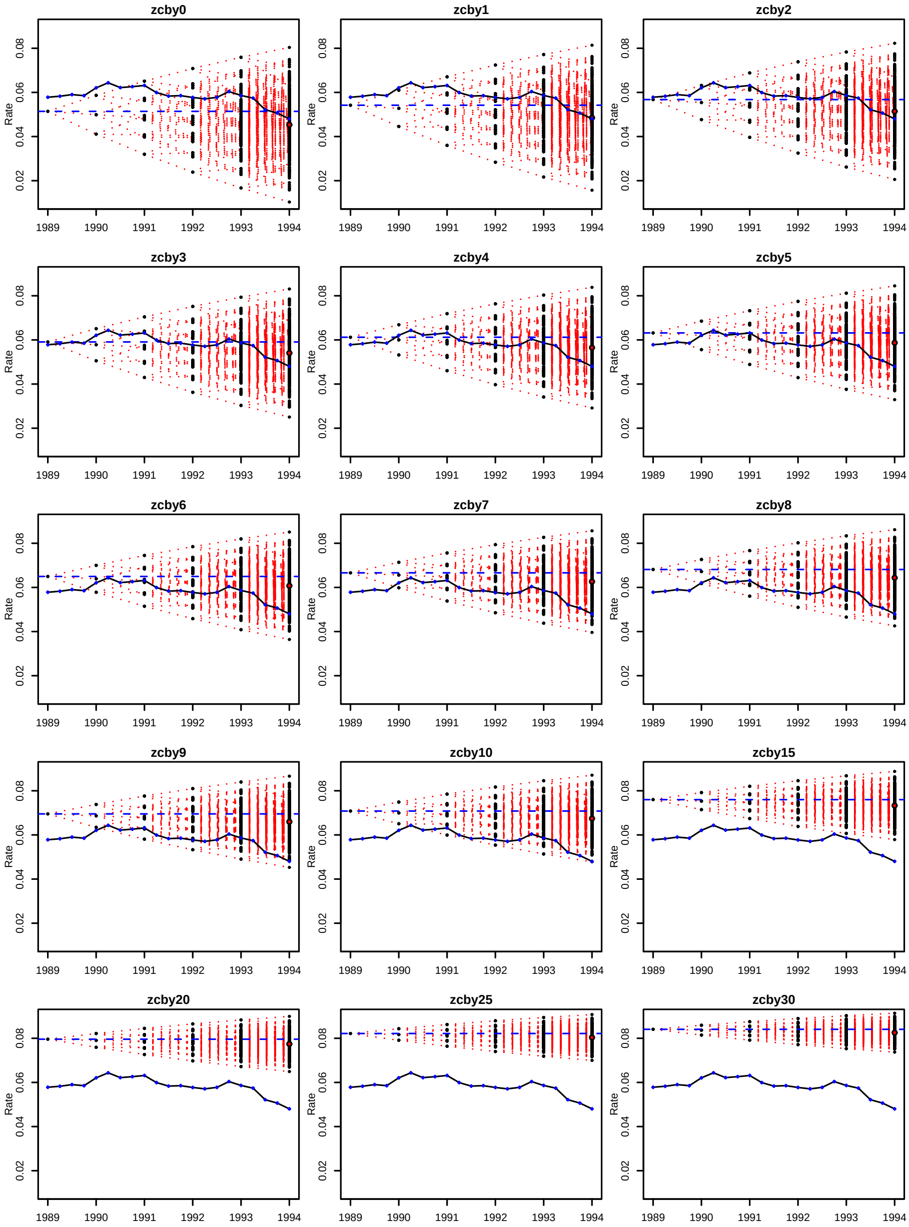

Figure 8.10: Interest rate scenario trees estimated from 1989 – 1994, zcbyXX where XX corresponds to the time to maturity on the bonds, 0-10,15,20,25 and 30 years. The blue line is the development of the MONA interest term *rente*.

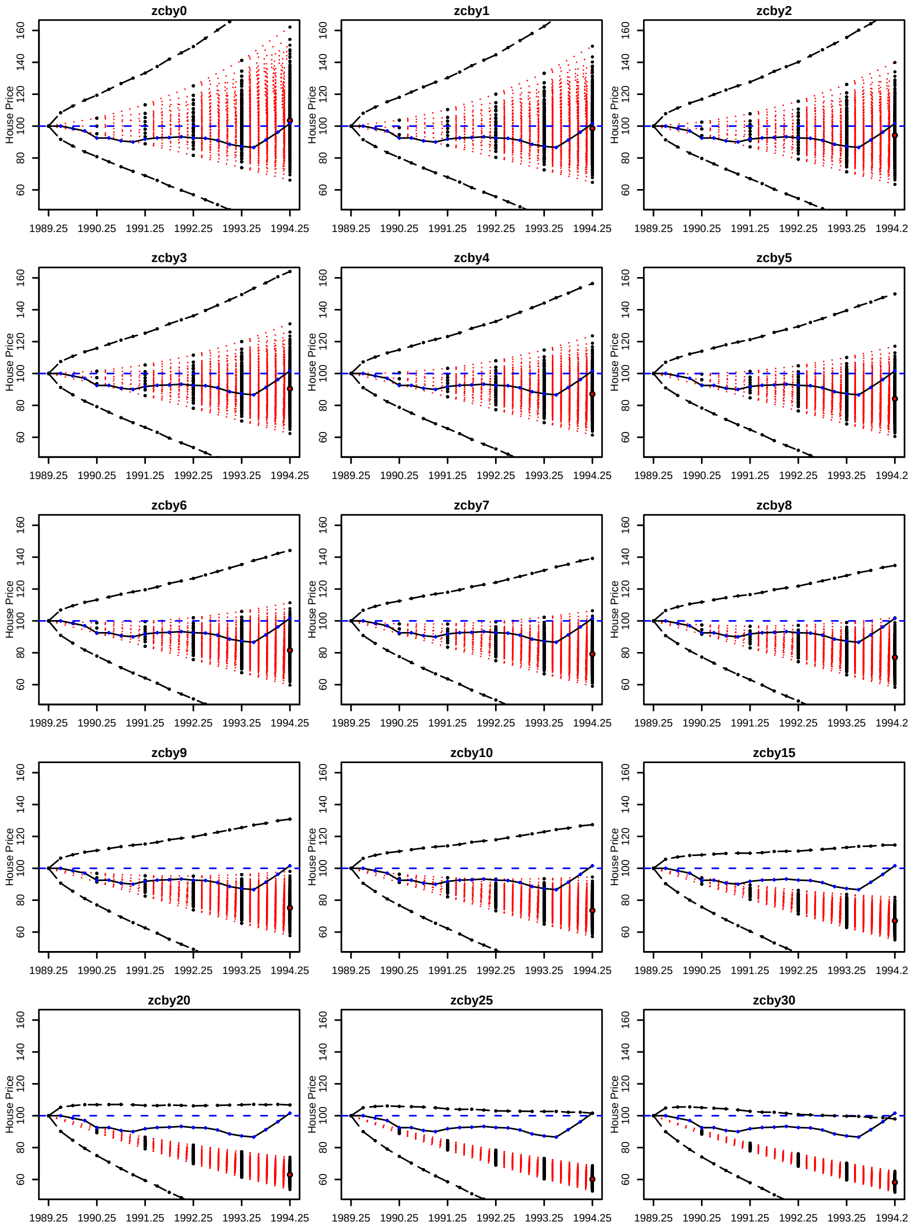

Figure 8.11: Estimate house price using the FIXED MONA method, each panel corresponds to the interest scenario tree in with same header from Figure 8.10. The blue line describes the observed house price. The broken black lines are the prediction error bars for the extreme paths, with 95% confidence interval.

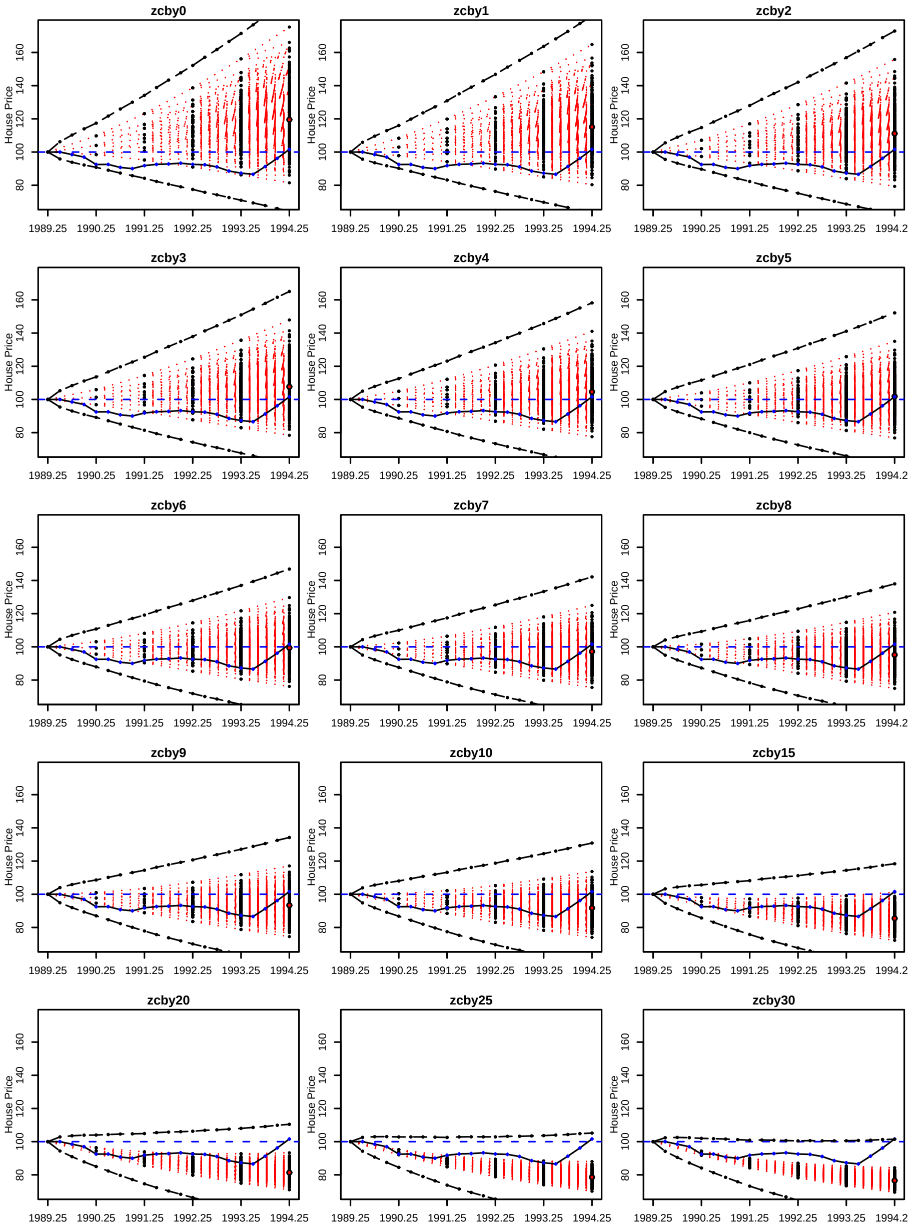

Figure 8.12: Estimate house price using the HPDIII (ECM) method, each panel corresponds to the interest scenario tree in with same header from Figure 8.10. The blue line describes the observed house price. The broken black lines are the prediction error bars for the extreme paths, with 95% confidence interval.

8.4 Summary

This chapter has listed the validation of the models first as single path models, or normal time series models, and later as scenario trees. Response relationships between the interest rates and house prices are developed for the one path and then applied to a scenario tree of interest rate paths. This section summaries the main results for the two validations.

Single path The validation for the single path reveals the Nykredit and Interest Rate only regression models as not suitable for predicting house prices. The non-stationary nature of the Nykredit relation results in unreliable results. The Interest Only regression is missing terms and only captures the upward trend of house prices. The Fixed MONA model appears to approximate the ideal Full MONA model for short to medium term predictions, see Figure 8.4. However for longer predictions $k > 15$ the precision decreases rapidly since the model is not well equipped to respond to dynamic change over a long period with many explanatory variables fixed. The $HPDIII$ model seems to be performing well according to the single path validation. Only the $HPDIII$ and Fixed MONA model are applied to the scenario tree structure.

Scenario tree In short if the input interest rate scenario trees capture the MONA rate, which can be modeled from data, the house price models capture the house price. However, this is dependant on the data not being significantly different from the in-sample period, where the models parameters are estimated. A sudden change in house prices not explained by the model factors, such as a bubble, will likely cause a discrepancy between the rates and house prices.

Both the Fixed MONA and $HPDIII$ models captured the house price well in the absence of bubble behavior, fixing the MONA and predicting for $k = 20$ can give very volatile house prices at horizon if the fixed explanatory variables were indicating a strong change at the time of fixing, prediction horizon.

Chapter 9

Conclusion

"All models are wrong, some are usefull."[1]

In this thesis the problem of modeling house prices to a degree was considered. House price is a non-stationary process, dependant on many economic variables. The three main factors affecting house price are interest rates, income and the amount of houses available.

Throughout this thesis the process of house price modeling is described from basic economic theory to applied house price scenario model, with estimated prediction interval.

Initially a basic theoretical economic model was devised. The complexity of the model was increased by replicating and analyzing the house price relation from a complex macro model (MONA). The theory and intuition from the MONA model was then applied to derive a MONA-like model which is more suited to the data available in the mortgagor problem, namely only interest rates. Two single path models are devised from the intuition acquired from the MONA model. The Fixed MONA model and the $HPDIII$ model.

1. The Fixed MONA model, used all the information in the Full MONA house price relation, while fixing many of the explanatory variables used and using only the interest rate variables as input. This fixing increased the error of the MONA prediction, the fixing also maked it hard to estimate the error with analytical methods. Bootstrapping was used to estimate the prediction error when using the Fixed model.

[1] George Box, one of the most influential statisticians of the 20th century.

2. The $HPDIII$ model was based on the same time series model as was used in the MONA house price relation, i.e. the error correction model, mixing together both levels and differences to capture both the short term dynamics and the long term trend. The $HPDIII$ model was modeled from data and did not use the MONA relation directly, unlike the fixed model.

Although these were the only models that were finally applied to a scenario tree structure, other benchmark models were also created. The benchmark models, The Nykredit relation and the Interest only regression, both served a certain purpose but in the end did not capture the house price well enough such that they could be used for prediction.

Validation was especially hard since the data was both scarce as well as very non-consistent. This lead to an in-sample validation which showed that the Fixed MONA and $HPDIII$ model were the ones that captured the house price best. However both methods have down sides. The Fixed MONA is non-respondent to dynamics changes, for long prediction horizons, and is therefore not very affective for long prediction horizons $k > 10$. This feature was incorporated into the evaluation of the prediction intervals for the Fixed MONA. The $HPDIII$ showed small signs of autocorrelation which did not seem to reduce the prediction performance significantly, e.g. as in the case of the interest only regression.

Both models showed the ability of capturing the two main elements in house price movements. Firstly both models captured the trend, which is related to the interest level at each time. Secondly and more importantly both models show signs of capturing the dynamics, with estimated prediction error. However modeling the short term dynamics with great precision is impossible.

Initially all models were treated as one path models or univariate time series. However, to be able to use the results in the Mortgagor problem a house price scenario tree must be devised from the single path model.

The house price trees were tested against interest rates with different maturities. There it could be seen that the two house price models capture the house price development, i.e. if the interest rate tree captures the interest rate. More precisely the output is only as good as the input, where the quality of the interest rate trees is fundamental in the quality of the house price trees.

The house price and interest elements in the MONA model are both very abstract. More specified models, e.g. for specified sector of the real estate market and certain bonds, can however be achieved quite easily using the same ideas applied in this thesis. The models developed in this thesis are considered as "correct models", i.e. they include the right terms, giving new parameter estimations for different data.

The thesis fulfills the aim that was set out with in the beginning, i.e. to develop a house price scenario tree(s), with known prediction intervals, that can be applied to the Danish Mortgagor problem [13].

9.1 Further Work

There are numerous aspects that can be investigated further, continuing from the results given in this thesis. The most interesting of these is to apply the house price trees to the Mortgagor problem and see what affect the possibility of adding house price will have on the results.

Another interesting issue is to investigate the composition of the interest term (*rente*) used by the National Bank in the estimation of the MONA model. There is obviously no, one, true interest rate and the MONA rate is some sort of weighted average of the rates of the bonds available. Given historical data of rates, an approximation to the *rente* term can be made from available rates. Giving the weights each bond has in the composition of the *rente* term. The weights could then be used to combine estimted house price trees to give a interest rate *rente* tree, resulting in a more correct scenario tree for house prices.

Appendix A

Programming

A.1 Introduction

The main topic of this chapter is the implementation of the scenario tree and a description of the reusable programs written for modeling and analysis.

In section A.2 the scenario tree from section 3.4 is revisited, describing the problem less formally as well as the different methods of implementation for such a tree. Sections A.3 and A.5 describe the two different ways the scenario tree was implemented. Firstly the ***indexing method***, implemented initially in `Matlab`, later moved to R, and described in section A.3. A short introduction to object oriented programming (OOP) is given in A.4. The second implementation of the scenario tree uses OOP for the more robust method, called the ***object oriented*** approach, implemented in *C#* and described in section A.5.

The analysis, parameter estimation and simulations performed in this thesis was performed in the statistical package R. Section A.3 discusses the programs written for modeling and analysis. Many of the functions written in R are highly reusable and therefore deserve some discussion. Section A.3 also provides example scripts, illustrating how to use the numerous functions written especially for this thesis.

A.2 Scenario Tree Revisited

From the start of the project the objective was to implement the tree structure in an object oriented language, i.e. $C\#$, see A.4 for further details. However, since having more experience with `Matlab` a more brute force method was attempted initially. The initial method is based on applying a tractable indexing scheme to the scenario tree. The purpose of the first implementation called the ***indexing method*** was initially intended to give insight into the tree structure and meant as a draft for the creation of the $C\#$ program.

There are two main elements to a scenario tree, i.e the shape (q) and number of periods (T). For example a binomial tree or trinomial tree would be $q = 2$ and $q = 3$, respectively. Recall from section 3.4 that the set of nodes in the tree at any time $0 \leq t \leq T$ can be described by the set $\mathcal{N}_t$. Corresponding to the formal definition of the tree the shape can be found from $q = \mathcal{C}(1)$. The two fundamental equations for implementing the indexing method can then be defined as the number of nodes at each time

$$|\mathcal{N}_t| = q^t \tag{A.1}$$

and the total number of nodes in the tree

$$N = \sum_{i=0}^{T} q^i \tag{A.2}$$

which e.g. for a $q = 2$, binomial tree, and $T = 8$ gives

$$\{q^t\} = \{1, 2, 4, 8, 16, 32, 64, 128, 256\} \qquad N = 511$$

These two equations, i.e. Eq.(A.1) and Eq.(A.2), allow for the formulation of the indexing method described in the next section.

Although the indexing method was only intended to give an intuition towards the scenario tree, it became very useful for validating the $C\#$ results, analyzing output from plotting the trees. Eventually both methods worked for generating scenario trees.

A.3 The Index Method

`Matlab` and S, the language used in R, are non object oriented programming languages which, when used correctly, can be very effective. The key to effective function programming is to write small, robust and specialized functions.

The functions can then be applied inside more complex functions to accomplish more involved tasks. This programming procedure also makes the code quite transparent and intuitive.

The final versions of the indexing method were very valuable in validating the results from the *C#* program, since by then they had captured most of the *C#* programs functionality.

The index method was initially implemented in `Matlab`, however since all statistical analysis, predicting and simulation was performed in R the indexing programs were moved to R for consistency, since the syntax of R and `Matlab` is very similar the transformation was easy.

In this section a short discussion will be given on the functionality of the most important indexing functions. The code for the following functions is available in C.1 in the Appendix.

$\texttt{seq} = \texttt{GeoSequence}(\texttt{q},\texttt{T})$: The first function that was created, calculates and returns a sequence $\{q^i\}$ where i=0,...,T. This sequence shows at time i how many nodes are at that time. If `q`=3 and `T`=5 for example, it would give:

$$3^5 = [1,\ 3,\ 9,\ 27,\ 81,\ 243]$$

So this is Eq.(A.1) and is used in all of the following indexing functions.

$\texttt{Sum} = \texttt{GeoSum}(\texttt{n},\texttt{T})$: This is Eq.(A.2) and sums up the results of the sequence given by `GeoSequence`, i.e. gives the total number of nodes in a tree. For example if `n`=3 and `T`=5 the function returns

$$\sum_{i=0}^{5} 3^i = 1 + 3 + 9 + 27 + 81 + 243 = 364$$

$\texttt{t} = \texttt{WhatPeriod}(\texttt{q},\texttt{T},\texttt{i})$: This function uses `GeoSequence` and `GeoSum` to find in which period, i.e. $0 \leq \texttt{t} \leq \texttt{T}$, node `i` is positioned. For example given `n`=3, `T`=5 and `i`=6, the program delivers an output of t=2.

$\texttt{p} = \texttt{Parent}(\texttt{n},\texttt{T},\texttt{i})$: This function is probably the most important program of the indexing functions. The function finds the parent index number `p` of a certain node `i` given the tree type `q` and length `T`. The algorithm uses `GeoSequence`, `GeoSum` and `WhatPeriod`. An example of output from this function is

```
Parent(n=3,T=10,i=3400)=1133
Parent(n=3,T=10,i=3401)=1134
Parent(n=3,T=10,i=3402)=1134
Parent(n=3,T=10,i=3403)=1134
Parent(n=3,T=10,i=3404)=1135
```

num = `NumBranches(q, T)` : This function takes the usual tree type `q` and tree length `T` as input. It returns a structured array in `Matlab` and list in R with two variables. The first one describes the number of leafs and the second the index number of the top leaf. An example of output for the function, call `NumBranches(n=3,T=10)`, is

```
NBranch: 59049
FBranch: 29524
```

i.e. there are 59049 leafs on this tree and `i`=29524 is the node index of the top leaf.

`mat` = `BranchParents(q, T, i)` : This function uses `Parent` and `NumBranches` and returns index numbers for whole branches. An example of output given the following function call `BranchParents(n=3,T=8,i=1)`, i.e. the `i` = 1 is the first leaf at `T`, gives

```
mat =
  Columns 1 through 5
          1            2            5            14            41
  Columns 6 through 9
        122          365         1094          3281
```

i.e. the output vector holds all the node indices of index=1, or the top leafs branch.

These are the main sub-functions used in making a scenario tree with the indexing method. Initially intended to be a exercise, for the more evolved *C#* programs, the indexing method evolved into a full fledged scenario tree generation method able of validating the results from the *C#* program. In the end, all house price models had working implementations both in R as well as *C#*. In R the house price dynamics are called `HPDI`, Nykredit model, `HPDINT`, Interest only and `HPDFIX`, Fixed MONA. An example of using the `TreeFunctions.R` bundle of functions is given below. The `TreeFunctions.R` code can be viewed in Appendix C.1.

Example of using the Tree Function library

```
###################################################################################
#                                                                                 #
#   Example of using the functions in the TreeFunctions.R file in the Appendix.   #
#                                                                                 #
###################################################################################

# Place the file TreeFunctions.R in directory or accsses via path and source:
source('TreeFunctions.R')
# Now all the functions in the TreeFunctions.R file are available for use.


# Initiating

q = 3                                          # Tree of type q = 3, i.e. trinomial.
T = 5                                                # Time T = 5, i.e. 0 <= t <= 5.
Indexes = Indexer(q,T)              # Matrix holding the indexes of a scenario tree.


# Generate Lattice Tree of test rates

Start.Rate = 0.04;                                                  # Begining Rate.
Range = 0.014;                                       # Range of change at each time.
LattTreeV = GenerateRates(q, T, Start.Rate, Range)     # Lattice Tree Vector Format.
LattTreeM = TreeForm(Indexes,LattTreeV)                # Lattice Tree Matrix Format.


# House Price Tree Generation. Using the lattice tree above.

NykreditTree = HPDI(q,T,LattTreeV)          # HPDI, the Nykredit House Prices model.

MONAFixed = HPDFIX(q,T,LattTreeV)                              # HPDFIX, MONA fixed.
# Fixed model uses 1997:75 values as default, other values can be used for fixing
# by adding HPDFIX(...., FIX = new.vector).

InterestOnlyReg = HPDINT(q,T,LattTreeV)      # HPDINT, The Interest Only regression.


# Simple Plot of house price.

INT.H = InterestOnlyReg$H        # The house price from InterestOnlyReg list object.
INT.H.MAT = TreeForm(Indexes,INT.H)                   # Get vector to matrix format.
PlotTree(INT.H.MAT)                                                # Plot INT.H.MAT.

Data.INT.H.MAT = MMM(INT.H.MAT)     # Matrix showing min,max and median at each time
                                    # 0...T in the tree.
```

The example above handles the scenario tree in two formats, i.e. the vector form $(1 \times N)$ and the matrix form $(q^T \times (T+1))$. All calculations use the vector form which allows for much bigger calculations than the heavy matrix form. The matrix form is derived from the vector format through the function `TreeFormat`. The matrix form is mainly used for plotting the trees, it is not recommended to manipulate big trees in matrix form or plot very big trees.

A.4 Object Oriented Programming

Another more sophisticated approach of programming the house price scenario trees is by use of so-called ***Object oriented programming*** (***OOP***). Objected orientation is an approach to build programs that mimic how actual objects are assembled in the real world. OOP procedure is often used along with ***The Unified Modeling Language*** (***UML***) which is a collection of successfully proven practises when it comes to programming large and complicated systems. The idea behind using OOP and UML is to create more reusable, reliable and understandable programs. More precisely object oriented programming portions big problems into more easily understandable parts. OOP´s standardized way of reducing problems through the use of UML makes it also possible for different people to maintain or extend already existing code with relative ease.

Here only a brief discussion will be given to a few OOP terms relative to the programming done in the thesis. For further discussion see Bennett, McRobb and Farmer [14][1]. These relative concepts here are ***Class***, ***Object***, ***Inheritance*** and ***Abstraction***.

Class: is the abstract definition of a "thing", including the "things" characteristics and what the "thing" can do. An example of this will be given in the object definition.

Object: is a particular instance of a class. An example of a class object relation is e.g. if a dog is a class then Lassie is an object of that class, i.e. the Lassie *is* a dog.

Inheritance: Often it is convenient to specify classes in more detail, which can be done by creating sub-classes. The sub classes then ***inherit*** the characteristics and attributes of the super class. An example of inheritance is that Lassie is a Collie. Collie can therefore be a sub class of dog. Since all Collies have the attributes of dogs, Lassie is therefore a object of the class Collie which inherits from the class Dog.

Abstraction: When programming complex relationship ***Abstraction*** is a good quality to have. Abstraction can be achieved by working at the appropriate level of inheritance, e.g. Lassie is a Animal - Mammal - Dog - Collie, each class becomes more specific when moving down in the hierarchy, i.e. adding more specific attributes and functions.

The next section uses the concepts expressed above when explaining the object oriented version of the house price scenario tree.

[1] See e.g. chapter 4 called What Is Object-Orientation

A.5 C# programming

As was mentioned in the previous sections, initial formulations for the scenario trees were drafted using Matlab. From the start the goal was however to build a program in an objective oriented language. Using the Matlab ideas of how a scenario tree structure works, along with the OOP framework a house price tree was programmed in the OOP language C#. There were two versions of the house price tree in C#, the class diagram for the first one can be seen in Figure A.1. The first version did not use concepts such as inheritance and abstraction there were only two classes, i.e. ***Tree*** and ***Node***. The first version begins by initializing a ***Tree*** object, e.g. HouseTree, next it calls a function to import the data from a XML file. For each new input supplied by the XML file an object is instantiated from the ***Node*** class, until all the data has been read from the XML file. Functions were then used on the HouseTree object, now holding all the XML data, to calculate corresponding house prices. Comparing to the Matlab version, which uses an elaborate indexing scheme to calculate the house prices the C# is a much more elegant solution with a much lower level of involvement required before it can be used by someone other than the author. The first implementation had room for improvement, since the level of abstraction was to high and there was a possibility of delegating the responsibility of the two classes further. Version one was also quite involved, though not as much as the Matlab version, i.e. if some one other than the author would want to edit or extend the program, that same person would have to acquire a full understanding of the whole system first.

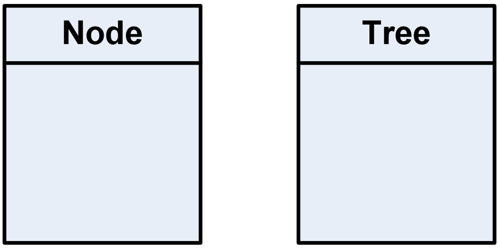

Figure A.1: An abstract class diagram of the initial version of the scenario tree program, performed in C#.

The second model was developed mainly by re-thinking the responsibilities of each class baring the OOP concepts in mind. As with the simple example given with the dog class above, a refined class for node and tree are derived where they only contain the most abstract terms common to scenario trees and nodes.

An example of this is that all nodes in a tree have a number while not all nodes should have an interest rate attribute. In the second version a new node and tree type are formulated as ***IR Tree*** and ***IR Node*** or interest rate tree, since interest rates are not common to node and tree but needed for calculating house prices. IR Tree and IR Node inherit the basic attributes of a Tree and Node respectively, same as for the Collie class does from the Dog class in the example above. A house price tree and node are formulated in the same way inheriting from the interest tree and node. The second version class diagram and the final version is displayed in Figure A.2, the arrows in the diagram represent an inheritance relationship. The benefits of the second model should be obvious,

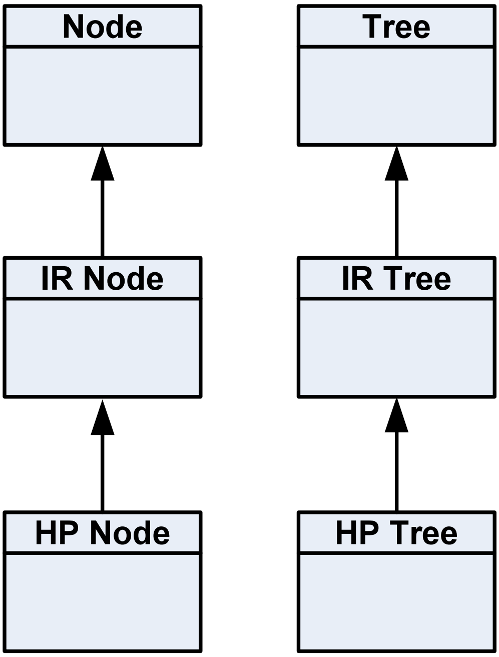

Figure A.2: An abstract class diagram of the second, and final, version of the scenario tree program, performed in C#.

e.g. if an individual would want to add a new tree say a pension tree, the pension node and tree could inherit from anywhere in the class hierarchy allowing the developer to achieve a certain level of abstraction. The developer would not have to know everything about the programm, only how the super class works. The full class diagram is given in Appendix D, for C# code see also Appendix D.

A.6 R Functions and Scripts

R is a language and environment for statistical computing and graphics. It is part of the GNU Project and therefore free[2]. R strengths lie mainly in the statistical and time series analysis, where it supersedes Matlab. R is also a fully fledged programming language and offers a flexible syntax for programming specialized functions. The main power of R comes from the open source nature which leads to very powerful discussion forums for problem solving. R is today considered the de-facto language when dealing with statistics.

The R package was used for replicating the MONA house price relation results, as well as for all tests, predictions and error estimation. Following is a script demonstrating the use of the numerous functions written for R. The code for the functions used can be seen in the Appendix section C.2.

[2] For more information see the R home page at ***http://wwww.r-project.org/***

Example of using Modeling Functions

```
###################################################################################
#                                                                                 #
#   Example of using the functions in the Functions.R file in the Appendix.       #
#                                                                                 #
###################################################################################

source('Functions1.R')

zz = read.csv("New.csv",sep = ";")             # Importing data from file New.csv.
attach(zz)
zz = ts(zz,frequency=4,start=c(1971,1))        # Make time series object.
zz = zz[,-1]

# Setting up data.
data =list('KP'=ts(KP,frequency=4,start=c(1971,1)),
           'RENTE'=ts(RENTE,frequency=4,start=c(1971,1)),
           'PCP'=ts(PCP,frequency=4,start=c(1971,1)),
           'IPV'=ts(IPV,frequency=4,start=c(1971,1)),
           'FWH'=ts(FWH,frequency=4,start=c(1971,1)),
           'SSATS'=ts(SSATS,frequency=4,start=c(1971,1)),
           'DPCPE'=ts(DPCPE,frequency=4,start=c(1971,1)),
           'DKPE' =ts(DKPE,frequency=4,start=c(1971,1)),
           'YDP' =ts(YDP,frequency=4,start=c(1971,1)),
           'RENTE.SSATS' = ts(RENTE+SSATS+0.01,frequency=4,start=c(1971,1)))

time = list( 'Sta' = 1974.25,
             'End' = 1997.75,
             'Clo' = 2001.75)


# Ordinary Least Squares And ROLS, formulate data.
i.m = Int.Only(data,time)              # Interest Only model estimated.
pi.m = Pred.OLS(i.m,alpha=0.05)        # Interest Only model predicted.
r.m = MONA.Model(data,time)            # MONA model estimated.
pr.m = Pred.ROLS(r.m,alpha=0.05)       # MONA model predicted.
ecm = ECM.Model(data,time)             # ECM model estimated.
pecm = Pred.OLS(ecm,alpha=0.05)        # ECM model predicted.

# Aggregation, moving from differences to levels.
Fit.all = i.m$All$Y
Fit.off = cbind(i.m$Hat$Off,r.m$Hat$Off,i.m$Off$Y,ecm$Hat$Off)
Fit.on  = cbind(i.m$Hat$On,r.m$Hat$On,i.m$On$Y,ecm$Hat$On)

Nom.all = Nominal.Dev(data$KP,Fit.all)            # All data.
Nom.off = Nominal.Dev(data$KP,Fit.off)            # In Sample, Offline.
Nom.on = Nominal.Dev(data$KP,Fit.on,time$End)     # Out Of Sample, Online.
```

Appendix B

Tables and Graphs for Results

B.1 Scenario Trees For Interest Only

The interest only regression model, did not capture the house price development, it only seemed to capture the upward trend as can be seen in section 8.2.

The interest only regression on scenario tree format is expressed as

Interest Only Regression

$$\Delta \widehat{\ln(kp_{t,n})} = \hat{\beta}_0^I + \hat{\beta}_1^I \Delta rente_{t,n} + \hat{\beta}_2^I \Delta rente_{t-1,a(n)} + \hat{\beta}_3^I rente_{t-1,a(n)}$$

an example of the development of interest only regression house price scenario trees for the interest rate scenario trees in Figure 8.7, can be seen in Figure B.1. The scenario trees show how the model does not respond to different levels in interest rates resulting in a upward trend, from the intercept.

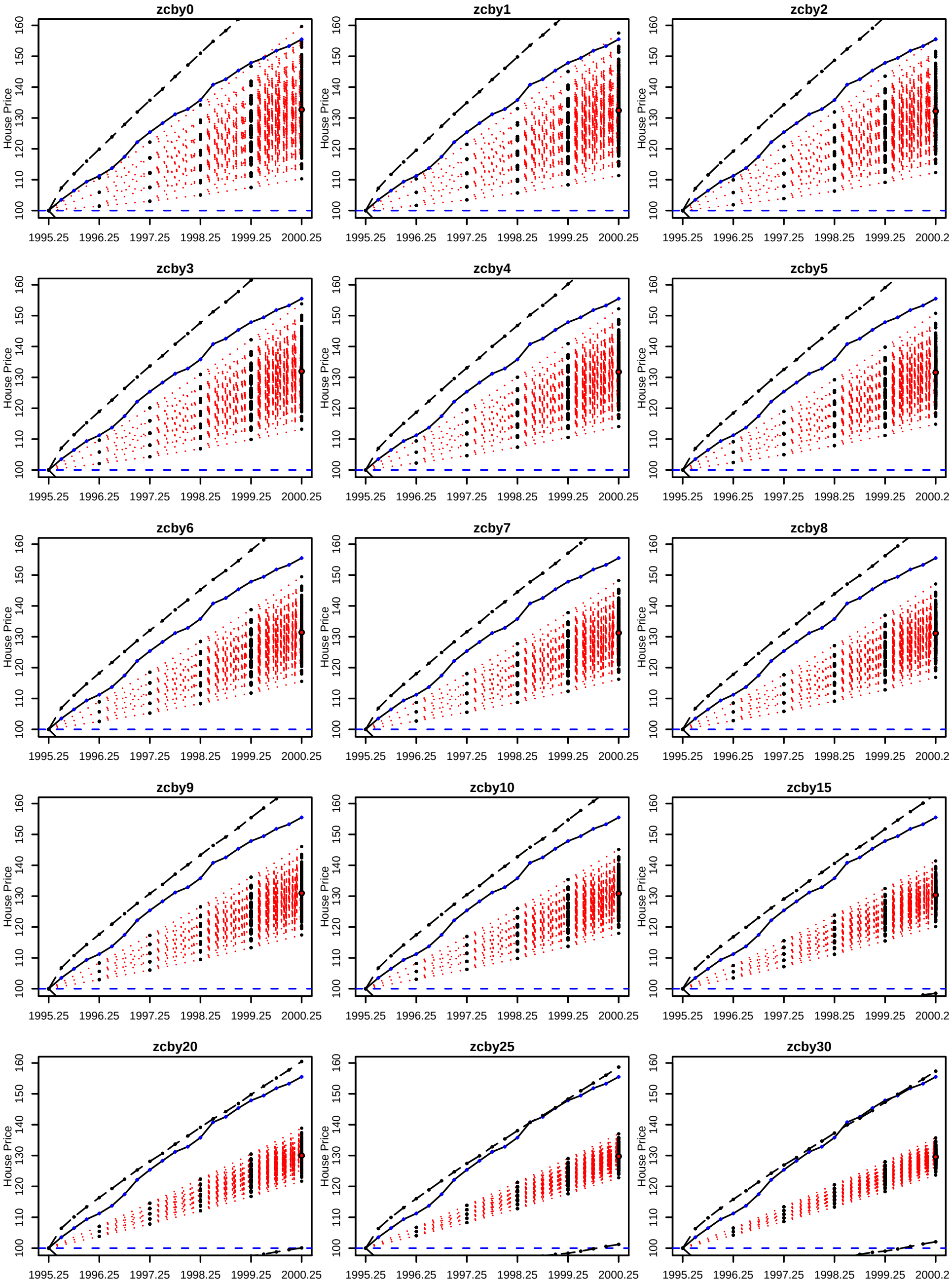

Figure B.1: Interest Only regression model corresponding to the interest rate scenario trees in Figure 8.7.

B.2 Error Bars

The error bars in the House Price figures are simulated according to Algorthm 2 in subsection 6.5.3. For a five year horizon using quarterly data corresponds to $k = 20$ periods. The numerical values for the four methods can be seen in Table B.1.

t: years	k	Full MONA	Fix MONA	Int Only	HPDIII
0	0	0.0000	0.0000	0.0000	0.0000
0.25	1	0.0172	0.0351	0.0231	0.0187
0.5	2	0.0240	0.0492	0.0324	0.0266
0.75	3	0.0295	0.0608	0.0395	0.0328
1	4	0.0342	0.0694	0.0454	0.0376
1.25	5	0.0380	0.0769	0.0505	0.0420
1.5	6	0.0416	0.0840	0.0560	0.0464
1.75	7	0.0452	0.0901	0.0612	0.0500
2	8	0.0477	0.0950	0.0655	0.0536
2.25	9	0.0508	0.1003	0.0685	0.0576
2.5	10	0.0536	0.1076	0.0734	0.0604
2.75	11	0.0562	0.1120	0.0767	0.0628
3	12	0.0592	0.1163	0.0805	0.0653
3.25	13	0.0614	0.1218	0.0839	0.0680
3.5	14	0.0634	0.1269	0.0854	0.0694
3.75	15	0.0661	0.1322	0.0882	0.0716
4	16	0.0684	0.1375	0.0921	0.0738
4.25	17	0.0706	0.1424	0.0950	0.0757
4.5	18	0.0727	0.1450	0.0973	0.0775
4.75	19	0.0751	0.1498	0.0997	0.0804
5	20	0.0763	0.1519	0.1026	0.0823

Table B.1: The estimated standard deviations, for aggregate house prices, estimating up to $k = 20$. The data is calculated according to Algorithm 2, in subsection 6.5.3.

Appendix C

R Code

C.1 Tree Functions.R

```
##############################################################################
#
#   Functions for plotting and analysis of scenario trees.
#   In the following order:
#
#   GeoSequence, GeoSum, GenerateProb, Parent, Mod, WhatPeriod,
#   NumBranches, BranchParents, Indexer, TreeForm and GenerateRates.
#
##############################################################################

GeoSequence = function(type,years)
{
# Generates a sequence of numbers i.e. [type^0,...,type^years], i.e. the number
# of nodes at any time i in the scenario tree.

    u = numeric(years+1)
    for(i in 0:years){
        u[i+1]=type^i;                                  # q^t where 0 <= t <= T
    }

    return(u)                                                      #Return seq.
}



GeoSum = function(type,years)
{
# Sums up the geometrical sequence [type^0,...,type^years]. i.e. sum up the seq
# from GeoSequence giving the total number of nodes in the tree.
```

```
    return(sum(GeoSequence(type,years)));    # Use GeoSequence and sum elements.
}



GenerateProb = function(type,years)
{
# Returns a array with probabilities that fit a tree of type type and of length
# such that a any time t the probabities sum to one and for any node the 1/q^t
# for t.

    twos = GeoSequence(type,years);             # Get the sequence of the tree.
    Sum = GeoSum(type,years);                   # Get total number of nodes.
    Prob = rep(0,Sum);
    b = 1;                                                          # counter.
    a = 0;                                                          # counter.

    for(i in 1:length(Prob)){                               # Loop over all nodes.

        if(a == twos[b]){                       # If a has been looped through all
            b = b+1;                            # nodes in periods. Move period up
            a = 0;                              # and set a to zero.
        }
        Prob[i] = 1/(twos[b]);                # Add a probability to current node.
        a = a + 1;                            # Increment a.
    }

    return(Prob)                                          # Return the array Prob.
}



Parent = function(type,years,index)
{
# Return the index of parent to node index. Needs tree type, number of years as
# input.

    Seq = GeoSequence(type,years);                   # Get the sequence of the tree.
    Sum = GeoSum(type,years);                          # Get total number of nodes.
    Vec = 1:Sum;                                               # A indexing vector.

    TotalIndex = index;                                              # Node number.
    mat = WhatPeriod(type,years,index);                   # Returns in which period.

    if(mat == 0){                           # Periods are 0,1,.. so if first Period
        parent = c();                       # the node has no parent.
    }else{
        IndexToPrevYear = sum(Seq[0:(mat-1)]); # Number of nodes to the year before.
        IndexToYear = sum(Seq[0:mat]);                    # Number of nodes to year.
        IndexOnYear = TotalIndex - IndexToYear;               # Nodes index on year.

        if(Mod(IndexOnYear,type)==0){     # If modulus of type and IndexOnYear is 0.
            num = IndexOnYear/type;           # Parent number in the period before.
        }else{
            num = floor(IndexOnYear/type)+1;  # parent number in the period before.
        }

        parent = IndexToPrevYear + num;                # Find total index of parent.
    }

    return(parent)                                 # Return index number of parent.
}
```

```
Mod = function(x,m)
{
# Calculates the modulus for x and m.
    t1<-floor(x/m)

    return(x-t1*m)
}



WhatPeriod = function(type,years,index)
{
# Returns the period number of which node number index is in. Also takes type
# of tree and number of years as input.

    Seq = GeoSequence(type,years);             # Get the sequence of the tree.
    Sum = GeoSum(type,years);                     # Get total number of nodes.
    counter = 0;                                                    # Counter.

    for(i in 0:length(Seq)){                    # Loop over number of periods.
    counter = counter + Seq[i+1];        # Add number of nodes for period i+1.
        if(counter >= index){            # If counter is bigger then node num.
            mat = i;                           # Return that period and break.
            break
        }
    }

    return(mat)                                               # Return period.
}



NumBranches = function(type,years)
{
# NumBranches  returns the index of the top leaf and the number of leafs in a
# list object. Input is type of tree (q) and years (T).

    num = list();                                           # Empty list.
    tmp = GeoSequence(type,years);                          # Tree sequence.
    n = length(tmp);
    num$N = tmp[n];
    num$F = sum(tmp[1:(n-1)]);   # Sum up the number of nodes pervious to T.

    return(num)                                        # Return list object.
}



BranchParents = function(type,years,index)
{
# NumBranches  returns the indexes of the branch from leaf of number index. The
# input variables are type of tree (q), years (T) and the leaf number, index.

    num = NumBranches(type,years);  # Find number of leafs and first leaf index.
    NumberBranches = num$N;
    FirstBranch = num$F;
    mat = rep(0,years+1);                                  # Empty index vector.
    index = index + FirstBranch; # Setting correct node index to the leaf index.

    for(i in (years+1):1){                          # Loop backwards over years.
        mat[i] = index;            # Set the index into the branch index vector.
        parent = Parent(type,i-1,index);                 # Find parent of index.
        index = parent;                                   # Set parent as index.
    }
    return(mat);    # Return the vector of indexes from leaf index to root note.
}



Indexer = function(type,years)
{
# NumBranches returns the indexes of the branch from leaf of number index. The
# input variables are type of tree (q), years (T) and the leaf number, index.

    num = NumBranches(type,years);
    indexer = matrix(0,nrow=(num$N),ncol=years+1);
    index = (1:(num$N));
    for(i in 1:length(index)){
        indexer[i,]=BranchParents(type,years,index[i]);
    }
    return(indexer)
}



TreeForm = function(Ind,Tree)
{
# Use the output of Indexer to return a indexed matrix form of the Tree vector.
# The input is Ind a matrix of indexes and Tree a scenario tree on the vector
# format.

    n=nrow(Ind)
    Out = Ind;
    for(i in 1:n){
    Out[i,]=Tree[Ind[i,]];
    }

    return(Out)                                         # Return Matrix Out.
}



GenerateRates = function(type,years,first,rang)
{
# This function is used to generate lattice interest rate trees for testing.
# The input variables are
#   type : the type of tree, q.
#  years : the numer of periods, T.
#  first : from what interst value is the tree to start.
#   rang : the range of a up to down change for one node from t-1 to t.

    Sum = GeoSum(type,years);                           # Number of nodes.
    Rates = rep(0,Sum);                         # Create Rates as 0 vector.
    Rates[1] = first;                           # Set first value in Rates.

    # Generate a vector from range/2 to -range/2 in type many parts.
    inc = seq(rang/2,-rang/2,length=type);

    # Repeate a sequence of vector 1:type in matrix tmp.
    tmp=rep(seq(1,type),(Sum-1)/2)
    Incs = c(0,inc[tmp]);                              # Index inc by tmp.

    for(i in 2:length(Rates)){                          # Loop over tree.
        parent = Parent(type,years,i);                 # Find parent node.

        # Calculate current rate by use of parent rate and change.
        Rates[i] = Rates[parent]+Incs[i];
    }
    return(Rates)            # Return lattice scenario tree of interest rates.
}


##############################################################################
#
#   Functions for plotting and analysis of scenario trees.
#   In the following order:
#
#   PlotTree, MMM and Pretty.
#
##############################################################################



PlotTree = function(Tree,lag=1,ylab="",xlab="Period",cex=0.5,
                    lty=3,main="",ylim=c(0,0),point=TRUE,year=0)
{
# Plots a matrix of the form from TreeForm.
# Input :  Tree - a matrix from TreeForm.
#          lag  - the number of lag on the x-axis.
#          point - switch whether the median point is plotted.
#          ylab,xlab,cex,lty,main,ylim same as in plot().
# All inputs have a default value so only the Tree matrix is needed to plot.

    n=nrow(Tree);
    m=ncol(Tree);
    if(ylim[1]==0 & ylim[2] ==0){                   # If ylim not specified.
    ylim = range(Tree);                             # Set to range of matrix.
    }
    xlim = c(lag,m-1+lag)                           # lag x-axis by lag.

    plot(Tree,xlim=xlim,ylim=ylim,type="n",ylab=ylab,
         xlab=xlab,main=main,cex=cex,xaxt="n")      # Set up empty grahpic device.
    axis(1,c(0:10),c(0:10)+year)

    Pret = Pretty(Tree)            # Removes repetition in Tree for better graphs.
    n = nrow(Pret)

    for(i in 1:n){                    # For each line in Pret plot line and point.
    lines(Pret[i,1:2]+lag,Pret[i,3:4],col=2,lty=lty,cex=cex*0.7)
    points(Pret[i,1:2]+lag,Pret[i,3:4],col=1,pch=19,cex=cex*0.7)
    }
    abline(h=Tree[1],col=4,lty=2); # Ad a horizontal line marking the first value.
    if(point){                                  # If point=T plot median of leafs.
    points(m-1+lag,MMM(Tree)[2,m],pch=21,col=1,bg="red",cex=cex*2)
    }

    # Returns nothing.
}



MMM = function(Mat)
{
# Simple function used for calculating the Min,Max and Median at each time in the
# tree. The input is a tree matrix.

    Min = apply(Mat,2,min)      # apply(Mat,2,operation) mean the opertion is used
    Max = apply(Mat,2,max)      # on the 2 dimension (column) of the Mat object.
    Med = apply(Mat,2,median)

    return(rbind(Max,Med,Min))                          # Return matrix (3 x T+1)
}
```

```
Pretty = function(Ind)
{
# A function used to simplify the a Tree matrix for plotting, input is a Tree matrix.
# Output is a matrix with four columns [line1.start line1.end line2.start line2.end].
# Used to remove repetition in the Tree matrix, making plotting faster and easier.
# Possible by inspecting the Ind tree matrix and reducing the Ind matrix to a matrix sec
# where each unique line segment only appears once.

    n = nrow(Ind)
    m = ncol(Ind)
    tmp = Ind[,1:2];                             # Set tmp as fist two columns of Ind.
    tmp1 = Ind[1,1:2]                         # Set tmp1 as the first line segment of
                                              # Ind i.e. 0 to 1

    sec = matrix(0,nrow=1,ncol=4)           # The reduced matrix created and set to 0.
    sec[1,1] = 0; sec[1,2] = 1; sec[1,3:4]=tmp1;

    for(i in 1:(m-1)){                          # Loop over all columns except last.
        for(j in 1:n){                                        # Loop over all lines.
        tmp2 = Ind[j,i:(i+1)]            # tmp2 the line segment Ind(j,i) to Ind(j,i+1).

            if(!all(tmp1==tmp2)){           # If tmp1 and tmp2 are not identical then.
            sec=rbind(" "=sec,c(i-1,i,tmp2))                         # Ad tmp2 to sec.
            }
        tmp1 = tmp2;                                          # Update tmp1 as tmp2.
        }

    }

    return(sec)                                             # Return the sec matrix.
}

##############################################################################
#
#   Scenario Tree House Price Dynamic Funcions
#   In the following order:
#
#   HPDFIX, HPDI, HDINT and HPDEcm
#
##############################################################################


HPDFIX = function(n,T,Rates,bbb=0,FIX=c(1, 0.002713868,0.000111711,8.054e-06,
                 0.01102516,0.01013059, 0.1011561, 0.1757178, -0.3041972),Ti=4)
{
# This function is very similar to the fuction used in C#.
# Calculating the Fixed MONA Relationship for House prices.
#     n : Type of tree n aka q.
#     T : Number of periods in the tree, T.
# Rates : Tree of interest rates.
#   FIX : The fixed explanatory matrix F.
#   bbb : Initial value of laged interest rates.
#
#   Mat : List including H the house price tree, DH changes in house price and
#         DSR Delta Rates.

    SR = Rates;                            # Interest Rates are SR.
    NodePlusOnePeriod = GeoSum(n,T);       # Number of nodes. T+1.
    I = 100;
    H = numeric(NodePlusOnePeriod)
    DH= numeric(NodePlusOnePeriod)
    D = numeric(NodePlusOnePeriod)
    Dtemp.Old = numeric(NodePlusOnePeriod)
    H[1] = 0;
    DH[1] = 0;

    c = c(0.06632852,0.30744099,-3.78106433,
          -0.77908085,-0.79271964,0.77091843,
           0.19494096,-0.10257190,0.05538029)
    Ti = Ti +1 ;

    Dtemp = numeric(Ti)
    SRtemp = numeric(5)
    DHtemp = numeric(5)
    Htemp  = numeric(5)

    Comp = list();

    HH = numeric(Ti);
    DHH = numeric(Ti);
    tt = numeric(Ti);

    for(i in 2:NodePlusOnePeriod){      #  Loop over 2:n^(T+1) nodes.

        t  = WhatPeriod(n,T,i);         # Returns the period t of node i.
        P  = Parent(n,T,i);             # P is the index of the parent node.
        GP = Parent(n,T,P);             # GP index of the Parent(Parent).
                                        # As long as t < (T+1).
        D[i] = SR[i] - SR[P];           # Difference in Current Rate and
                                        # Parent rate.

        DD = c(0,rep(D[i]/4,Ti-1))
        SS = cumsum(DD)+SR[P]
        tt = seq(t-1,t,length.out=Ti);
        DD[1] = D[P]/4
        HH[1] = H[P]
        DHH[1] = DH[P]

        for(j in 2:length(Dtemp)){
            if(tt == 0.25){
                int =  c(0,0,DD[j],bbb,SS[j-1],0,0,0,0);
                DHH[j] =  c%*%(FIX+int);
            }
            if(t > 0.25){
                int = c(0,0,DD[j],DD[j-1],SS[j-1],0,0,0,0);
                DHH[j] =  c%*%(FIX+int);
            }
            HH[j] = DHH[j]+HH[j-1];
        }

        H[i] = HH[Ti];
        DH[i] = DHH[Ti];
        Comp[[i]] = cbind("Ti"=tt,"DH"=DHH,"H"=HH,"SS"=SS,"DD"=DD)

    }

    Mat = list();
    Mat$DSR     = D;
    Mat$DH      = DH;
    Mat$H       = H;

    return(Mat);                                # Return list Mat.
}


HPDI = function(n,T,Rates,I=100)
```

```
{
# This function is very similar to the fuction used in C#.
# Calculating the NyKredit Relationship for House prices.
#     n : Number indicating branch number, n type of tree.
#     T : Number of periods in the tree, T.
# Rates : Tree of interest rates.
#
#   Mat : List including HP the house price tree, DH changes in house price,
#         DSR Delta Rates and H house prices without compounding.

    SR = Rates;                              # Interest Rates are SR.
    NodePlusOnePeriod = GeoSum(n,T);         # Number of nodes. T+1.
    H = numeric(NodePlusOnePeriod)
    HP = numeric(NodePlusOnePeriod)
    D1= numeric(NodePlusOnePeriod)
    D2= numeric(NodePlusOnePeriod)
    DH= numeric(NodePlusOnePeriod)
    D = numeric(NodePlusOnePeriod)
    DeltaRates = numeric(NodePlusOnePeriod)

    H[1] = I;
    HP[1]= I;

    Ti = 5;
    Comp = list();

    HH = numeric(Ti);
    DHH = numeric(Ti);
    tt = numeric(Ti);


    for(i in 2:NodePlusOnePeriod){           # Loop over 2:n^(T+1) nodes.

        t  = WhatPeriod(n,T,i);              # Returns the period t of node i.
        P  = Parent(n,T,i);                  # P is the index of the parent node.
        GP = Parent(n,T,P);                  # GP index of the Parent(Parent).

                                             # As long as t < (T+1).
        DeltaRates[i] = SR[i] - SR[P];       # Difference in Current Rate and
                                             # Parent rate.
        D1[i] = -5*DeltaRates[i];            # One year change at i.
        D2[i] = -11*DeltaRates[i];           # Two year change at i.

        DD = c(0,rep(D[i]/4,Ti-1))
        tt[1] = t-1;
        DD[1] = D[P]/4
        HH[1] = H[P]
        DHH[1] = DH[P]

        for(j in 2:Ti){
            tt[j] = t - 1 + 1/(Ti-1)*(j-1)
            if(t == 1){
                DH[i] = DH[P] + D1[i];
                HP[i] = HP[P]*(1 + D1[i]);
            }
            if(t > 1){
                HP[i] =  HP[P] * (1 + D1[i]) + HP[GP]* D2[P];
                DH[i] =  DH[P]+ D1[i] + D2[P];
            }
            H[i] = I*(1 + DH[i]);
        }
        H[i] = HH[Ti];
        DH[i] = DHH[Ti];
        Comp[[i]] = cbind("Ti"=tt,"DH"=DHH,"H"=HH,"DD"=DD)
    }
    Mat         = list();
    Mat$DSR     = DeltaRates;              # Delta Short Rates.
    Mat$H       = H;
    Mat$HP      = HP;
    Mat$DH      = DH;

    return(Mat)                                          # Return list Mat.
}


HPDINT = function(n,T,Rates,Ti=4)
{
# This function is very similar to the fuction used in C#.
# Calculating the Interest Only Regression for House prices.
#     n : Number indicating branch number, n type of tree.
#     T : Number of periods in the tree, T.
# Rates : Tree of interest rates.
#
#   Mat : Struc including HP the house price tree, HP_1 house price tree
#         lagged one period and DSR the Delta Rates.

    SR = Rates;                                      # Interest Rates are SR.
    NodePlusOnePeriod = GeoSum(n,T);                 # Number of nodes. T+1.
    I = 100;
    H = numeric(NodePlusOnePeriod)
    DH= numeric(NodePlusOnePeriod)
    D = numeric(NodePlusOnePeriod)
    Dtemp.Old = numeric(NodePlusOnePeriod)
    H[1] = 0;
    DH[1] = 0;
    Comp = list();
    Ti = Ti + 1;

    HH = numeric(Ti);
    DHH = numeric(Ti);
    tt = numeric(Ti);

    CC=c(0.01254567,-3.65385018,-1.69341039);

    for(i in 2:NodePlusOnePeriod){               #  Loop over 2:n^(T+1) nodes.

        t  = WhatPeriod(n,T,i);            # Returns the period t of node i.
        P  = Parent(n,T,i);                # P is the index of the parent node.
        GP = Parent(n,T,P);                # GP index of the Parent(Parent).

        D[i] = SR[i] - SR[P];                  # Difference in Current Rate and.
                                               # Parent rate.

        DD = c(0,rep(D[i]/4,Ti-1))
        tt = seq(t-1,t,length.out=Ti);
        DD[1] = D[P]/4
        HH[1] = H[P]
        DHH[1] = DH[P]

        for(j in 2:Ti){
            int = c(1,DD[j],DD[j-1]);
            DHH[j] =  CC%*%(int);

            HH[j] = DHH[j]+HH[j-1]
        }
        H[i] = HH[Ti];
        DH[i] = DHH[Ti];
        Comp[[i]] = cbind("Ti"=tt,"DH"=DHH,"H"=HH,"DD"=DD)
 }
```

```
    Mat = list();
    Mat$L       = Comp;
    Mat$DSR     = D;
    Mat$DH      = DH;
    Mat$H       = H;

    return(Mat);                                                # Return list Mat.
}



HPDEcm = function(n,T,Rates,I=100,H1=0,DH1=0,Ti=4)
{
# This function is very similar to the fuction used in C#.
# Calculating the NyKredit Relationship for House prices.
#     n : Number indicating branch number, n type of tree.
#     T : Number of periods in the tree, T.
# Rates : Tree of interest rates.
#
#   Mat : List including HP the house price tree, DH changes in house price,
#         DSR Delta Rates and H house prices without compounding.

    SR = Rates;                             # Interest Rates are SR.
    NodePlusOnePeriod = GeoSum(n,T);        # Number of nodes. T+1.
    I = 100;
    H = numeric(NodePlusOnePeriod)
    DH = numeric(NodePlusOnePeriod)
    D = numeric(NodePlusOnePeriod)
    H[1] = H1
    DH[1] = DH1;

    Comp = list();
    Ti = Ti + 1

    HH = numeric(Ti);
    DHH = numeric(Ti);
    tt = numeric(Ti);
    CC = c(0.03837,-4.04156,0.34215,-0.03431,-0.63258)


    for(i in 2:NodePlusOnePeriod){      #  Loop over 2:n^(T+1) nodes.

        t  = WhatPeriod(n,T,i);         # Returns the period t of node i.
        P  = Parent(n,T,i);             # P is the index of the parent node.
        GP = Parent(n,T,P);             # GP index of the Parent(Parent).
                                        # As long as t < (T+1).
        D[i] = SR[i] - SR[P];           # Difference in Current Rate and
                                        # Parent rate.

        DD = c(0,rep(D[i]/4,Ti-1))
        SS = cumsum(DD)+SR[P]
        tt = seq(t-1,t,length.out=Ti);
        DD[1] = D[P]/4
        HH[1] = H[P]
        DHH[1] = DH[P]

        for(j in 2:Ti){
            int = c(1,DD[j],DHH[j-1],HH[j-1],SS[j-1]);
            DHH[j] =  CC%*%(int);
            HH[j] = DHH[j]+HH[j-1]
        }
        H[i] = HH[Ti];
        DH[i] = DHH[Ti];
        Comp[[i]] = cbind("Ti"=tt,"DH"=DHH,"H"=HH,"SS"=SS,"DD"=DD)
    }
    Mat         = list();             # Delta Short Rates.
    Mat$H       = H;
    Mat$DH      = DH;
    Mat$L       = Comp;


    return(Mat)                                          # Return list Mat.
}

Error.Cal = function(Tree,EB,I=1)
{
    H = MMM(Tree)
    p1 = Int.Pol(H[1,])
    p2 = Int.Pol(H[3,])
    U = cbind('d' = I*100*(1-exp(EB))+p2, 'u' = I*-100*(1-exp(EB))+p1)
    return(U)
}



Int.Pol = function(X,leng=3)
{
    le = length(X)
    T2 =c()

    for(i in 1:(le-1)){
        temp = seq(X[i],X[i+1],length.out=leng+2)
        T2 = c(T2,temp[1:(leng+1)])
    }
    T2 = c(T2,X[le])
    return(T2)
}

Read.IntTree = function(STRING)
{
# A function to import interest rate trees.
#
    header = scan(STRING,nlines=1,what=character(), quiet = TRUE)
    SS = read.table(STRING,skip=1)
    names(SS) = c("Year","Node",header)

    return(SS)
}
```

C.2 Modeling Functions.R

```
###############################################################################
#
#   Functions for modeling, estimation and data handeling for time series
#   models. In the following order:
#
#   R.square, R.adj.sqr, Nominal.Dev, Int.Only, ECM.Model, ECM.Model,
#   MONA.Model, TimePeriod, ECM.4.lag, MONA.ROLS, Pred.ROLS, Pred.OLS
#
###############################################################################


R.square = function(Y,Y.hat)
```

```
{
# Calculates the R square or Goodness of fit statistic between to series Y
# and the fitted serise Y.hat.

    N = length(Y);
    R.above.1 = (t((Y.hat-Y)^2)%*%matrix(1,nrow=N))  # Matrix %*% operation.
    R.below.1 = sum((Y-mean(Y))^2)
    R.2 =1 - (R.above.1/R.below.1)

    # Return Goodness Of Fit.
    return(R.2)
}



R.adj.sqr = function(Y,Y.hat,p)
{
# Calculates the adjusted R square or Goodness of fit statistic between two
# series  Y and the fitted serise Y.hat.

    N = length(Y);
    R = R.square(Y,Y.hat)
    R.adj = 1 - ((N-1)/(N-p))%*%(1-R)

    # Return Adjusted Goodness Of Fit.
    return(R.adj)
}


Nominal.Dev = function(KP,Y.hat,st=1974.25)
{
# Calculates the aggregate house price for a multivariate series element
# Y.hat which are changes. KP is the house price time series object, st is
# the start of acumulation for the house price. There are two versions of
# this function Nominal.Dev2 is used for the valdiation of point estimates.

    temp = dim(Y.hat)
    if (is.null(temp)){                                        # If vector.
    N = length(Y.hat);
    M = 1;
    }else{                         # If not vector, i.e. if Y.hat is matrix.
    N = temp[1];
    M = temp[2];
    }

    Y.0 = window(log(KP),st-0.25,st-0.25)      # Set KP to the correct house
    Y.tilde.R = matrix(0,nrow=N,ncol=M)        # price at time st to use in
                                               # update.
        if (!is.null(temp)){                              # If Y.hat matrix.
            for(j in 1:M){
                Y.tilde.R[1,j] = Y.hat[1,j] + Y.0
                for(i in 2:N){
                    Y.tilde.R[i,j] = Y.hat[i,j] + Y.tilde.R[i-1,j]
                }

            }
        Y.tilde.R = ts(as.data.frame(Y.tilde.R),frequency=4,start=st)
        }else{                                            # If Y.hat vector.
            Y.tilde.R[1] = Y.hat[1] + Y.0
            for(i in 2:N){
                Y.tilde.R[i] = Y.hat[i] + Y.tilde.R[i-1]
            }
        Y.tilde.R = ts(Y.tilde.R,frequency=4,start=st)   # Set as ts object.
        }
    # Returns a aggregate timeseries object from st.
    return(Y.tilde.R)
}

Int.Only = function(Data,Times)
{
# Calculates the Interest Only Regression Model. Input is Data a list with
# all time series data and Times also a list with the start of in-sample
# period end of in-sample and end of all data.

    Sta = Times$Sta;                        # Start of in-sample or Offline.
    End = Times$End;                  # End of in-sample or start of Online.
    Clo = Times$Clo;                                # End of all or Offline.

    diff.off       =c(Sta,End)
    diff.on        =c(diff.off[2],Clo)

    # Offline
    Off = TimePeriod(Data,diff.off[1],diff.off[2])
    HouseP.Int <- lm(Off$Y ~ Off$I2 + Off$I3)               # OLS performed.
    Y.hat.off = ts(fitted(HouseP.Int),frequency=4,start=diff.off[1])
    Off$X = as.matrix(data.frame(rep(1,length(Off$I2)),Off$I2,Off$I3))
    Beta = matrix(coef(HouseP.Int))                          # Coefficients.

    # Online
    On = TimePeriod(Data,diff.on[1],diff.on[2])            # Function below.
    On$X = as.matrix(data.frame(rep(1,length(On$I2)),On$I2,On$I3))
    Y.hat.on = ts(On$X%*%Beta,frequency=4,start=diff.on[1])

    # All
    All = TimePeriod(Data,diff.off[1],diff.on[2])          # Function below.
    All$X = as.matrix(data.frame(rep(1,length(All$I2)),All$I2,All$I3))
    Y.hat.all = ts(All$X%*%Beta,frequency=4,start=diff.off[1])

    # Fits
    sig = (t(resid(HouseP.Int))%*%resid(HouseP.Int))/(dim(Off$X)[1]-dim(Off$X)[2])
    Hat = list(’Off’=Y.hat.off,’On’=Y.hat.on,’All’=Y.hat.all,’sigma.hat.sq’=sig)

    # Returns four sublist in the output list object.
    return(list(’OLS’=HouseP.Int,’Off’=Off,’On’=On,’All’=All,’Hat’=Hat))
}



ECM.Model = function(Data,Times)
{
# Calculates the Error-Correction Model using only lagged kp and rente, levels
# and differenced series. Input is Data a list with
# all time series data and Times also a list with the start of in-sample
# period end of in-sample and end of all data.

    Sta = Times$Sta;                        # Start of in-sample or Offline.
    End = Times$End;                  # End of in-sample or start of Online.
    Clo = Times$Clo;                                # End of all or Offline.

    diff.off       =c(Sta,End)
    diff.on        =c(diff.off[2],Clo)

    # Offline
    Off = TimePeriod(Data,diff.off[1],diff.off[2])
    MODEL.ECM = lm(Off$ECM$DKP ~ Off$ECM$DRE + Off$ECM$DKP.1 + Off$ECM$KP.1 + Off$ECM$RE.1)
    Y.hat.off = ts(fitted(MODEL.ECM),frequency=4,start=diff.off[1])
    Off$X = ts.union(’I’=rep(1,length(Off$ECM$DRE)),’DRE’=Off$ECM$DRE,
                     ’DKP.1’=Off$ECM$DKP.1, ’KP.1’=Off$ECM$KP.1,’RE.1’=Off$ECM$RE.1)
    Beta = matrix(coef(MODEL.ECM))                          # Coefficients.

    # Online
    On = TimePeriod(Data,diff.on[1],diff.on[2])            # Function below.
    On$X = ts.union(’I’=rep(1,length(On$ECM$DRE)),’DRE’=On$ECM$DRE,
                    ’DKP.1’=On$ECM$DKP.1,’KP.1’=On$ECM$KP.1,’RE.1’=On$ECM$RE.1)
    Y.hat.on = ts(On$X%*%Beta,frequency=4,start=diff.on[1])

    # All
    All = TimePeriod(Data,diff.off[1],diff.on[2])          # Function below.
    All$X = ts.union(’I’=rep(1,length(All$ECM$DRE)),’DRE’=All$ECM$DRE,
                     ’DKP.1’=All$ECM$DKP.1, ’KP.1’=All$ECM$KP.1,’RE.1’=All$ECM$RE.1)
    Y.hat.all = ts(All$X%*%Beta,frequency=4,start=diff.off[1])


    # Fits
    sig = (t(resid(MODEL.ECM))%*%resid(MODEL.ECM))/(dim(Off$X)[1]-dim(Off$X)[2])
    Hat = list(’Off’=Y.hat.off,’On’=Y.hat.on,’All’=Y.hat.all,’sigma.hat.sq’=sig)

    # Returns four sublist in the output list object.
    return(list(’OLS’=MODEL.ECM,’Off’=Off,’On’=On,’All’=All,’Hat’=Hat))
}



MONA.Model = function(Data,Times)
{
# Calculates Restricted Ordinary Least Squares (ROLS). Input as before Data
# with  time series objects and Times with start of in-sample, end of
# in-sample and end of all data.

    Sta = Times$Sta;                        # Start of in-sample or Offline.
    End = Times$End;                  # End of in-sample or start of Online.
    Clo = Times$Clo;                                # End of all or Offline.

    diff.off       =c(Sta,End)
    diff.on        =c(diff.off[2],Clo)

    # Offline
    Off = TimePeriod(Data,diff.off[1],diff.off[2])
    # OLS
    HouseP.lm = lm(Off$Y~Off$X1+Off$X2+Off$X3+Off$X4+Off$X5+Off$X6+Off$X7+Off$X8)
    # ROLS
    R = MONA.ROLS(Off)                        # The ROLS function see below.
    Beta_R = R$Beta_R
    Y.hat.off = ts(Off$X%*%Beta_R,frequency=4,start=diff.off[1])

    # Online
    On = TimePeriod(Data,diff.on[1],diff.on[2])
    Y.hat.on = ts(On$X%*%Beta_R,frequency=4,start=diff.on[1])

    # All
    All = TimePeriod(Data,diff.off[1],diff.on[2])
    Y.hat.all = ts(All$X%*%Beta_R,frequency=4,start=diff.off[1])

    # Fits
    Hat = list(’Off’=Y.hat.off,’On’=Y.hat.on,’All’=Y.hat.all,
               ’sigma.hat.sq’ = R$sigma.hat.sq)

    # Returns five sublists ’ROLS’ has the ROLS coefficients.
    return(list(’OLS’=HouseP.lm,’ROLS’=R,’Off’ = Off,’On’=On, ’All’ = All,
                ’Hat’=Hat))
}
```

```
TimePeriod = function(Data,From,To)
{
# A data cutting function. Input is Data object with time series objects
# From and To mark the time window which is sought. Uses the ts function
# window.

    # Model Variables.
    Y  = window(diff(log(Data$KP)),From,To)
    X1 = window(diff(log(Data$PCP)),From,To)
    X2 = window(diff(Data$RENTE.SSATS),From,To)
    X3 = window(diff(Data$RENTE.SSATS),From-0.25,To-0.25)
    X4 = window(Data$RENTE.SSATS,From-0.25,To-0.25)
    X5 = window(Data$DPCPE,From-0.25,To-0.25)
    X6 = window(Data$DKPE,From-0.25,To-0.25)
    X7 = window(log(Data$KP/Data$PCP),From-0.25,To-0.25)
    X8 = window(log((Data$YDP-Data$IPV)/Data$PCP)-log(Data$FWH),
                From-0.25,To-0.25)

    KP = Data$KP;
    # Time vector.
    ts = time(Y);

    # Tax with out Interest.
    SSATS.X2 = window(diff(Data$SSATS),From,To)
    SSATS.X3 = window(diff(Data$SSATS),From-0.25,To-0.25)
    SSATS.X4 = window(Data$SSATS,From-0.25,To-0.25)

    # Interest with out Tax.
    INT.X2 = window(diff(Data$RENTE+0.01),From,To)
    INT.X3 = window(diff(Data$RENTE+0.01),From-0.25,To-0.25)
    INT.X4 = window(Data$RENTE+0.01,From-0.25,To-0.25)

    X0 = ts(rep(1,length(Y)),frequency=4,start=From);
    Zip = ts(rep(0,length(Y)),frequency=4,start=From);

    # The Fixed vector.
    FA = ts.union(X0,X1,"S2"=SSATS.X2,"S3"=lag(SSATS.X3,-1),
                "S3"=lag(SSATS.X4,-1),"X5"=lag(X5,-1),
                "X6"=lag(X6,-1),"X7"=lag(X7,-1),"X8"=lag(X8,-1))
    FA = window(FA,From,To);

    # Interest Only Vector.
    AA = ts.union(Zip,Zip,"I2"=INT.X2,"I3"=lag(INT.X3,-1),"I4"=lag(INT.X4,-1),
                 Zip,Zip,Zip,Zip)
    AA = window(AA,From,To);

    ECM = ECM.4.lag(Data,From,To);

    # Design or Explanatory Matrix.
    X = as.matrix(data.frame("X0"=rep(1,length(Y)),X1,X2,X3,X4,X5,X6,X7,X8))

    # Returns a list with numerous sublist including all the data needed for
    # analysis and forecasting.
    return(list('Y' = Y,'X'=X, 'X1'=X1,'X2'=X2,'X3'=X3,'X4'=X4,'X5'=X5,
                'X6'=X6,'X7'=X7,'X8'=X8,'S2'=SSATS.X2,'S3'=SSATS.X3,
                'S4'=SSATS.X4,'I2'=INT.X2,'I3'=INT.X3,'I4'=INT.X4,
                't'=ts,'KP'=KP,'FA'=FA,'AA'=AA,'ECM'=ECM) )

}

ECM.4.lag = function(Data,st,en)
{
# A data cutting function. Input is Data object with time series objects
# From and To mark the time window which is sought. Uses the ts function
# window.

    RE = Data$RENTE+0.01
    KP = log(Data$KP)
    DKP = diff(KP)
    DRE = diff(RE)

    DRE.1 = window(lag(DRE,-1),st,en);
    DRE.2 = window(lag(DRE,-2),st,en);
    DRE.3 = window(lag(DRE,-3),st,en);
    DRE.4 = window(lag(DRE,-4),st,en);

    DKP.1 = window(lag(DKP,-1),st,en);
    DKP.2 = window(lag(DKP,-2),st,en);
    DKP.3 = window(lag(DKP,-3),st,en);
    DKP.4 = window(lag(DKP,-4),st,en);

    KP.1 = window(lag(KP,-1),st,en)
    RE.1 = window(lag(RE,-1),st,en)
    DKP = window(DKP,st,en)
    DRE = window(DRE,st,en);
    RE = window(RE,st,en)


    # Design or Explanatory Matrix.
    #X = as.matrix(data.frame("X0"=rep(1,length(DKP)),DRE,DKP.1,KP.1,RE.1))

    # Returns a list with numerous sublist including all the data needed for
    # analysis and forecasting.
    return(list('DKP' = DKP, 'DRE' = DRE, 'DRE.1'=DRE.1,'DRE.2'=DRE.2,'DRE.3'=DRE.3,
                'DRE.4'=DRE.4, 'DKP.1'=DKP.1, 'DKP.2'=DKP.2, 'DKP.3'=DKP.3, 'DKP.4'=DKP.4,
                'KP.1'=KP.1, 'RE'=RE, 'RE.1' = RE.1,'KP' = KP, 'DRE' = DRE))#, 'X' = X))

}



MONA.ROLS = function(Data)
{
# The actual Resticted Oridnary Least Squares is calculated for the MONA house
# price model. Returning all the same values as OLS with lm does. Input is Data
# list of the format as TimePeriod outputs.

      Y = Data$Y
      n = length(Y)

      # OLS
      X = Data$X
      XX.1 = solve(t(X) %*% X)
      Beta = XX.1%*%t(X)%*%Y

      # Constraint R%*%Beta_R = r
      a = c(Int=0,X1=0.25,X2=0,X3=0,X4=1,X5=1,X6=1,X7=0,X8=0);
      R = t(as.matrix(a));
      r = 0.25

      # Coefficient for ROLS, Beta_R.
      b = t(R)%*%solve(R%*%XX.1%*%t(R))
      c = (r-R%*%Beta);
      Beta_R = Beta + XX.1%*%b%*%c;

      # Y.hat, fit with Beta_R.
      Y.hat <- X %*% Beta_R
      # Estimated variance of residuals.
      sigma.hat.sq <- sum((Y - Y.hat)^2) / (n - ncol(X)+1)

      # Covariance matrix, V, for Beta_R.
      M = diag(1,9) - XX.1%*%b%*%R
      C =  M %*% XX.1 %*% t(M)
      V = sigma.hat.sq * C
      se = sqrt(diag(V))

      # t - values
      t = Beta_R/se

      # p - value
      p.value = 2*pt(-abs(t),df=n-ncol(X)+1)
      All=data.frame('Estimate'=round(Beta_R,5),'Std.Error'=round(as.matrix(se),5)
                     ,'t.value'=round(t,3),'p.value'=p.value)

      # Returns many values in a list 'Summary' returns a comprihensive description
      # similar to a summary(lm-object).
      return(list('Beta_R'=Beta_R, 'Beta'=Beta, 'XX.1'=XX.1, 'Cov.ROLS'=V,
                  'sigma.hat.sq'= sigma.hat.sq, 'Std.Error.Beta_R' = se,
                  't.value.R' = t, 'p.value.R' = p.value, 'Y.hat' = Y.hat, 'M'=M,
                  'Summary'=All))

}



Pred.ROLS = function(List,alpha=0.05)
{
# Calculates prediction intervals for the MONA ROLS model. The covariance matrix is
# different and the prediction therefor also. alpha sets the prediction intervals
# confidence interval by conf.int = 1-(alpha/2). alpha is set to 0.05 by default.

    yOFF=List$Hat$Off
    yON =List$Hat$On                      #Out of sample, or Online Point Estimate.
    xOFF=List$Off$X
    xON =List$On$X

    sigma = List$Hat$sigma.hat.sq
    M = List$ROLS$M

    XX = solve(t(xOFF)%*%xOFF);
    n = length(yOFF);
    p = dim(XX)[1] - 1;
    tt = qt(1-alpha/2,n-p)

    tmp=c();

    # For each out of sampe point calculate the prediction interval.
    for(i in 1:length(yON)){
        TEM = sqrt( sigma * (1 + xON[i,]%*%M%*%XX%*%t(M)%*%as.matrix(t(xON)[,i])));
        tmp[i] = tt * TEM
    }
    predict = cbind(yON-tmp,yON,yON+tmp,tmp)

    # Returns a time series object with four series, point estimat - variance, point
    # estimate, point estimate + variance, variance.
    return(predict)
}



Pred.OLS = function(List,alpha=0.05)
{
```

```
# Calculates prediction intervals for the OLS model. alpha sets the prediction intervals
# confidence interval by conf.int = 1-(alpha/2). alpha is set to 0.05 by default. List is
# a list of type as output from TimePeriod.

    yOFF=List$Hat$Off
    yON =List$Hat$On                            #Out of sample, or Online Point Estimate.
    xOFF=List$Off$X
    xON =List$On$X

    sigma = List$Hat$sigma.hat.sq

    XX = solve(t(xOFF)%*%xOFF);
    n = length(yOFF);
    p = dim(XX)[1];
    tt = qt(1-alpha/2,n-p)

    tmp=c();
    # For each out of sampe point calculate the prediction interval.
    for(i in 1:length(yON)){
        tmp[i] = tt * sqrt( sigma * (1 + xON[i,]%*%XX%*%as.matrix(t(xON)[,i])));
    }
    predict = cbind(yON-tmp,yON,yON+tmp,tmp)

    # Returns a time series object with four series, point estimat - variance, point
    # estimate, point estimate + variance, variance.
    return(predict)
}


##############################################################################
#
#   Functions for simulating error in change and levels for time series, along
#   with many sub functions. In the following order:
#
#   BOOT, GenerateCoefficients, GenerateEstimatChange, Erro.Cal, AggHPsim, MS,
#   Print.Boot, Plot.C, Lines.Boot, PredictInt, YLIM
#
##############################################################################


BOOT = function(ROLS,INT,ECM,k,N=10000,t.st=1997.75,Coeff=F)
{
# BOOT is a simulation of the error when bootstrapping three different models, it
# estimates the change in house prices error for MONA full, MONA fixed and INT only.
# The inputs are: ROLS object which is the output from MONA.Model.
#                 INT obeject which is the output from Int.Only.
#                 k the prediction horizon.
#                 N repetitions for each simulation, default set to N=10000.
#                 t.st the start of prediction. Default set to last Offline, 1997.75.
#                 Coeff a logical variable, see below default set to FALSE.

    # Data
    All = ROLS$All
    X = All$X;  S2 = All$S2;  S3 = All$S3;  S4 = All$S4
    Y = All$Y;  I2 = All$I2;  I3 = All$I3;  I4 = All$I4

    # Setting up for the ROLS.
    V.R  = ROLS$ROLS$Cov.ROLS
    Beta.R  = ROLS$ROLS$Beta_R
    sig.R  = ROLS$ROLS$sigma.hat.sq
    B.A = Beta.R
    B.F = Beta.R

    # Setting up for the Interest Only Regression.
    X.In = INT$All$X
```

```
sig.I = as.numeric(INT$Hat$sigma.hat.sq)
Beta.I = as.matrix(coefficients(INT$OLS))
V.I = sig.I*solve(t(INT$Off$X)%*%INT$Off$X)
B.I = Beta.I

# Setting up for the Error-correction Model.
X.Ecm = ECM$All$X
sig.E = as.numeric(ECM$Hat$sigma.hat.sq)
Beta.E = as.numeric(coefficients(ECM$OLS))
V.E = sig.E * solve(t(ECM$Off$X)%*%ECM$Off$X)
B.E = Beta.E

# Initializing variables.
k = k + 1;                             # Add one to k to add last In-sample point.
Y.tF = matrix(0,ncol=k,nrow=N);
Y.tA = matrix(0,ncol=k,nrow=N);
Y.tI = matrix(0,ncol=k,nrow=N);
Y.tE = matrix(0,ncol=k,nrow=N);

ind.F = which(time(Y)==t.st);                                   # Index of Fixing.
ind = ind.F;                 # Index without fixing. Initially set to fixed index.

# For t=0,...,k, since now k = k+1.
for(p in 1:k){
    tp = t.st + (p-1)*0.25;                               # Time period increment.
    ind = ind.F + (p-1);                                        # Index increment.
    X.A = X[ind,];                                # X.A set to correspoding explt.
    X.F = X[ind.F,];                                     # X.F set to fixed explt.
    X.F[3]=S2[ind.F]+I2[ind]                             # Interest elemtents set.
    X.F[4]=S3[ind.F]+I3[ind]                               # eplanitory variables.
    X.F[5]=S4[ind.F]+I4[ind]
    X.I = X.In[ind,];
    X.E = X.Ecm[ind,];

    # Repeat the following process N times.
    for(i in 1:N){

        # Add error to coefficients. If Coeff=T.
        if(Coeff){
           B.F = GenerateCoefficients(B=Beta.R,CVar=V.R); # Subfunction see below.
           #B.I = GenerateCoefficients(B=Beta.I,CVar=V.I);
           #B.A = GenerateCoefficients(B=Beta.R,CVar=V.R);
           #B.F[3]=B.A[3]; B.F[4]=B.A[4]; B.F[5]=B.A[5];
        }


        # Error estimate of Fixed and All explanitory vectors.
        Y.tF[i,p]=GenerateEstimatChange(X=X.F,te=B.F,sdt=sig.R) # Subfunction.
        Y.tA[i,p]=GenerateEstimatChange(X=X.A,te=B.A,sdt=sig.R)
        Y.tI[i,p]=GenerateEstimatChange(X=X.I,te=B.I,sdt=sig.I)
        Y.tE[i,p]=GenerateEstimatChange(X=X.E,te=B.E,sdt=sig.E)

    }

}
Y.F = list(’Y’=Y.tF,’MS’=MS(Y.tF,t.st))    # lists with value, mean and sd.
Y.I = list(’Y’=Y.tI,’MS’=MS(Y.tI,t.st))    # MS subfunction.
Y.A = list(’Y’=Y.tA,’MS’=MS(Y.tA,t.st))
Y.E = list(’Y’=Y.tE,’MS’=MS(Y.tE,t.st))
E.F = Erro.Cal(Y,Y.F,t.st,k)                          # Error.Cal subfunction.
E.I = Erro.Cal(Y,Y.I,t.st,k)
E.A = Erro.Cal(Y,Y.A,t.st,k)
E.E = Erro.Cal(Y,Y.E,t.st,k)
Misc = list(’KP’=All$KP,’Y’ = Y,’k’=k-1,’t.st’=t.st);
Ret = list(’Y.F’=Y.F,’E.F’=E.F,’Y.I’=Y.I,’E.I’=E.I,
               'Y.A'=Y.A,'E.A'=E.A,'Y.E'=Y.E,'E.E'=E.E,'Misc'=Misc);

    # Return a list with many sublist, e.g. one for each model.
    return(Ret)

}


GenerateCoefficients = function(Beta,CVar)
{
# Sub function of BOOT, generates a sample from a normal distribution
# where N(Beta,CVar).

    p = length(Beta);
    B = numeric(p);
    for(i in 1:p){
    B[i] = rnorm(1,mean=Beta[i],sd=sqrt(CVar[i,i]));
    }

    # Returns a vector with a sample from the coefficient distribution.
    return(B)
}

GenerateEstimatChange = function(X,te,sdt,mean=0)
{
# Sub function of BOOT, calculates a sample from a normal distribution
# using the residual variance and adding to the model part.

    Model = X%*%te
    Resid = rnorm(1,mean=mean,sd=sqrt(sdt))
    Y = Model+Resid;

    # Return a sample value of Y with a residual and regression error.
    return(Y)
}



Erro.Cal = function(Y,Y.S,t,k)
{
# A simple function for moving the point estimate to zero, i.e. basing the
# change from 0. Y
    Y.S=Y.S$Y
    p = min(dim(Y.S))
    N = max(dim(Y.S))

    temp = matrix(0,nrow=N,ncol=p)
    D = matrix(0,nrow=N,ncol=p)
    Y.obs = window(Y,t,t+(k-1)*0.25)
    for(i in 1:p){
    temp[,i]=rep(Y.obs[i],N)
    }
    D = temp-Y.S;
    A = list('Y'=D,'MS'=MS(D,t));

    # Returns the Y.S matrix centered around 0.
    return(A)
}



AggHPsim = function(Ret,N=10000)
{
# A simulation for the aggregate effect of the house price model. Three models are
# simulated MONA full, MONA fixed and INT only. The input is a list object from the
# BOOT function above.

    Misc = Ret$Misc;
    Y.A=Ret$Y.A;      k = Misc$k;
    Y.F=Ret$Y.F;      t.st = Misc$t.st;
    Y.I=Ret$Y.I;      KP = Misc$KP;
    Y.E=Ret$Y.E;
    Y = Misc$Y;
    # Observed Nominal House Price.
    #Y.OBS = Nominal.Dev(KP,Y);
    ln.kp = log(KP);
    A = window(ln.kp,t.st,t.st)                 # Start Value of House Price.
    A.on = window(ln.kp,t.st)

    Y.cA = matrix(0,ncol=k+1,nrow=N);
    Y.cI = matrix(0,ncol=k+1,nrow=N);
    Y.cF = matrix(0,ncol=k+1,nrow=N);
    Y.cE = matrix(0,ncol=k+1,nrow=N);
    # For t=0,...,k, since now k = k+1.
    for(p in 1:(k+1)){
      # Repeate each forcast N times.
      for(i in 1:N){
            if(p==1){
                # First t is known.
                Y.cA[i,p] = A;
                Y.cI[i,p] = A;
                Y.cF[i,p] = A;
                Y.cE[i,p] = A;
            }else{
                # t>0 sample change for from distibutions gotten from the
                # BOOT output.
                RCA = rnorm(1,mean=Y.A$MS[p,1],sd=Y.A$MS[p,2])
                RCI = rnorm(1,mean=Y.I$MS[p,1],sd=Y.I$MS[p,2])
                RCF = rnorm(1,mean=Y.F$MS[p,1],sd=Y.F$MS[p,2])
                RCE = rnorm(1,mean=Y.E$MS[p,1],sd=Y.E$MS[p,2])

                # t>0 aggregate effect by adding the sample change to a
                # sample from a distribution of previous aggregate price.
                Y.cA[i,p] = RCA + rnorm(1,mean=tMA,sd=tSA);
                Y.cI[i,p] = RCI + rnorm(1,mean=tMI,sd=tSI);
                Y.cF[i,p] = RCF + rnorm(1,mean=tMF,sd=tSF);
                Y.cE[i,p] = RCE + rnorm(1,mean=tME,sd=tSE);
            }

      }
         tMA = mean(Y.cA[,p]);   tSA = sd(Y.cA[,p]);
         tMI = mean(Y.cI[,p]);   tSI = sd(Y.cI[,p]);
         tMF = mean(Y.cF[,p]);   tSF = sd(Y.cF[,p]);
         tME = mean(Y.cE[,p]);   tSE = sd(Y.cE[,p]);
    }

    Y.F = list('Y'=Y.cF,'MS'=MS(Y.cF,t.st))    # lists with value, mean and sd.
    Y.I = list('Y'=Y.cI,'MS'=MS(Y.cI,t.st))
    Y.A = list('Y'=Y.cA,'MS'=MS(Y.cA,t.st))
    Y.E = list('Y'=Y.cE,'MS'=MS(Y.cE,t.st))
    Ret = list('Y.F'=Y.F,'Y.I'=Y.I,'Y.A'=Y.A,'Y.E'=Y.E);

    # Return a list with a hierachy of lists.
    return(Ret)
}



MS = function(Y,t=F)
{
```

```
# Calculates the mean and standar deviation of matrix Y returns as time series
# if is.numeric(t). Subfunction of BOOT and AggHPsim.

    p = ncol(Y)
    N = nrow(Y)
    mean = numeric(p)
    sd = numeric(p)
    # Simpler way for this is the function apply. See ?apply.
    for(i in 1:p){
        mean[i] = mean(Y[,i])
        sd[i] = sd(Y[,i])
    }
    if(is.numeric(t)){
    temp = ts(cbind(’Mean’=mean,’Sd’=sd),frequency=4,start=t);
    }else{
    temp = cbind(’Mean’=mean,’Sd’=sd)
    }

    # Return a vector with mean and sd of each column in Y.
    return(temp);
}



Print.Boot = function(List)
{
# A function which prints out the result for the simulation of BOOT, input is list
# of the same format as BOOT or AggHPsim export.

    MS=List$MS
    k=nrow(MS)
    cat("  k \t Mean \t\t Stand Deviation \t\n")
    cat("--------------------------------------------\n")
    for(p in 1:k){
        cat(" ",p-1,"\t",MS[p,1],"\t",MS[p,2],"\t\n")
    }

    # No Value is Returned.
}



Plot.C = function(List,br=20,main="",col=2,add=F,lty=2,lwd=1,type=’l’,xlab="",mu=F)
{
# A home made function for plotting the normal disributions denerated by the data
# from BOOT and AggHPsim. List is a list object from the simulation functions BOOT or
# AggHPsim.
    if(is.list(List)){                                  # If List is a list object.
    Y=List$Y
    A=List$MS
    }else{                                              # If List is numeric.
    Y = List;
    A = MS(Y);
    }

    p = min(dim(Y))
    ylim = numeric(p)
    xlim = range(Y)
    tmp = 0;
    mu.tmp = 0;
    # Used to find a common ylim that has all distributions.
    for(i in 1:p){
        tmp = hist(Y[,i],freq=F,plot=F,br=br)
        mu.tmp = range(mu.tmp,range(Y[,i]-A[i,1]))
        ylim[i]=max(tmp$density)
    }
    # Switch used to get all graphs on one graph.
    if(mu){
        xlim = mu.tmp;
        A[,1]=0;
    }else{
        xlim = range(Y)
    }
    ylim = c(0,max(ylim))
    nd=seq(xlim[1],xlim[2],0.001)
    # If add=F then the plot is set up.
    if(!add){
    plot(Y,type='n',ylim=ylim,xlim=xlim,xlab=xlab,ylab='Density',main=main)
    }

    for(i in 1:p){
        y=dnorm(nd,mean=A[i,1],sd=A[i,2])
        lines(nd,y,type=type,col=col,lwd=lwd,lty=lty)
    }
    abline(h=0)
    abline(v=0)

    # No return value.
}



Lines.Boot = function(List,col=1,lwd=1,lty=2,prod=1,on=T,pp=T)
{
# Plots the simulated prediction intervals and point estimates, prod is the
# t-value of the prediction interval.

    A=PredictInt(List,prod=prod)            # Small sub function see below.
    lines(A[,2],col=col,lwd=lwd,lty=lty-1)
    if(pp){
    points(A[,2],col=col,pch=19,cex=0.8)
    }
    if(on){
    lines(A[,1],col=col,lwd=lwd,lty=lty)
    lines(A[,3],col=col,lwd=lwd,lty=lty)
    }

    # No return Value.
}



PredictInt = function(Y,prod=1)
{
# Sub function of Lines.Boot. Y is a matrix with mean values and  standard
# deviations (MS list object). prod is the t-value used for the width of
# the prediction intervals.

    me = Y$MS[,1];  sd = Y$MS[,2];

    # Return a mean-(variance*t-value), mean, mean+(variance*t-value)
    return(cbind('SD.m'=me-prod*sd,'MU'=me,'SD.p'=me+prod*sd))
}



YLIM = function(List,TSer,prod=1)
{
# Small help function for plotting. Finds the range for ylim when setting up plots.
# List is a MS list object. TSer is the observed house price value.
```

```
    a=range(c(List$MS[,1] + prod * List$MS[,2], List$MS[,1] - prod * List$MS[,2]))
    tt=range(range(TSer),a)

    # Return the vector with range tt.
    return(tt)
}
```

Appendix D

C# Code and Class Diagram

For C# code contact me at snorri.pall.sigurdsson@gmail.com.

D.1 C# Class Diagram

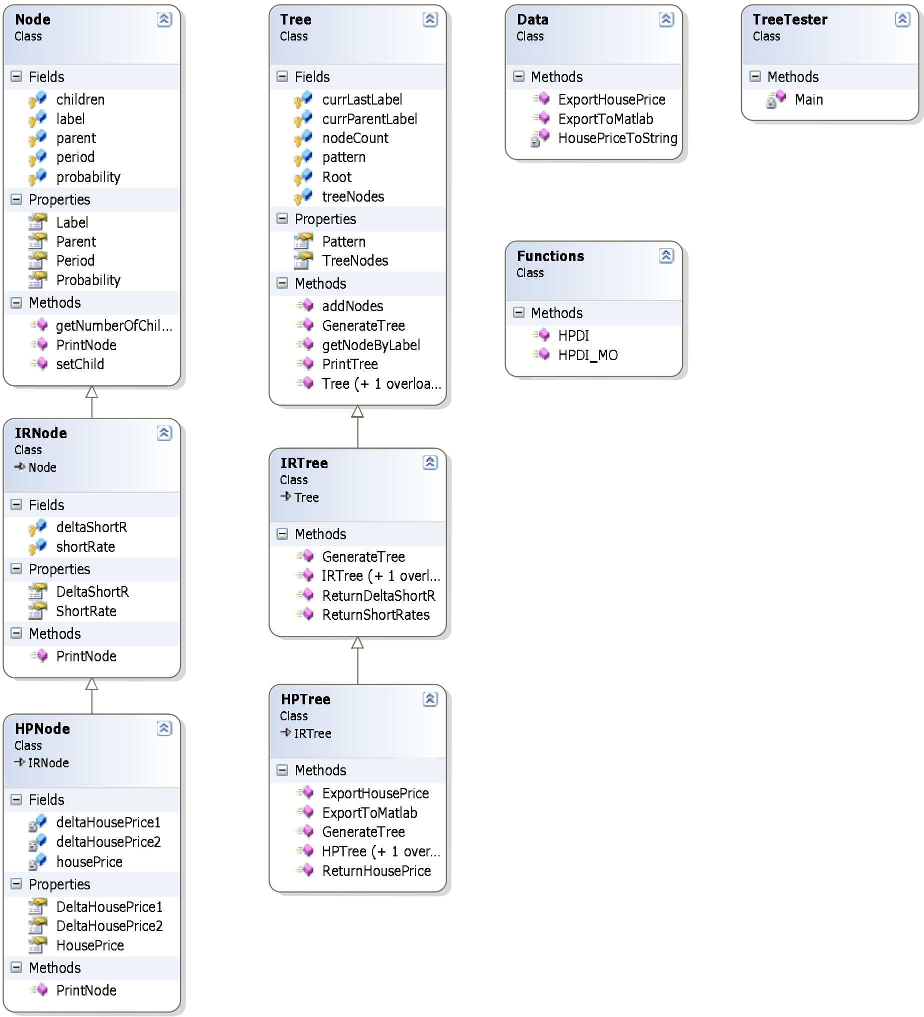

Figure D.1: The class diagram for the scenario tree implementation in C#.

Bibliography

[1] B. Barot and Z. Yang. House prices and housing investment in sweden and the united kingdom, econometric analysis for the period 1970-1998. *Review of Urban & Regional Development studies (RURDS)*, 14:No.2, 2002.

[2] A.C. Davison and D.V. *Bootstrap Methods and their Application, 1st edition*. Cambridge, 1997.

[3] J. D. Hamilton. *Time Series Analysis*. Princeton university press, 1994.

[4] National Bank Of Iceland. Report to the minister of social affairs on the effect of changing the loan system for residential housing. pages 43–66, 2004.

[5] B. Jensen and R. Poulsen. Transition densities of diffusion processes: Numerical comparison of approximation techniques. *Journal of Derivatives*, 9(4):18–32, 2002.

[6] G. Judge, R. Hill, W. Griffiths, H. Lutkepohl, and T. Lee. *Introduction to the theory and practices of econometrics, 2nd edition*. John Wiley & Sons, 1988.

[7] T. Kyhl and J. Nielsen. Økonometriske værktøjer. page 7, 2002.

[8] J. Lunde. Fluctuations and stability in the danish housing market: Background, causes and policy. 2005.

[9] H. Madsen. *Time Series Analysis*. IMM, Technical University of Denmark, 2001.

[10] NyKredit markets. Erhvervsejendomme er rentefølsomme. 2006.

[11] D. Montgomery and G. Runger. *Applied Statistic and Probability for engineers, 3rd edition.* John Wiley & Sons, 2003.

[12] Danmarks Nationalbank. Mona - a quarterly model of the danish economy. pages 41–47, 2003.

[13] K.M Rasmussen and J. Clausen. Mortgage loan portfolio optimization using multi stage stochastic programming. 2005.

[14] S. McRobb S. Bennett and R. Farmer. *Object-Oriented System Analysis And Design Using UML, 2nd edition.* McGraw-Hill, 2002.

[15] R. S. Tsay. *Analysis of Financial Time Series, 2nd edition.* John Wiley & Sons, 2005.

[16] M. Verbeek. *A Guide To Modern Econometrics, 1st edition.* John Wiley & Sons, 2000.