Erasmus School of Economics

Reducing the Dimensionality of House Price Indices with LSTM-Autoencoders for Cluster Analysis

Master Thesis Econometrics & Management Science

Business Analytics & Quantitative Marketing

Author:

Thomas Waltmans

Student ID number:

620515

Supervisor:

Dr. M. Khismatullina

Second assessor:

TBD

Abstract

The focus of this paper is to find clusters in house price indices using a unsupervised deep learning technique for dimensionality reduction and centroid-based and hierarchical clustering methods on the resulting lower dimensional space. The effects of fundamental macroeconomic variables will be canceled out by applying a panel vector autoregression to predict the residual time series, which will be used as the input for the cluster analysis. The panels will be split in accordance with the cluster assignments and the regression model will be reapplied to the respective groups. The performance with respect to the original full-panel regression model will be evaluated. We conclude that our proposed method does not improve upon the original results of the vector autoregression model without including the time series group structure.

Keywords: time-series clustering, high-dimensional data, deep learning, autoencoders, long short-term memory, vector autoregression, house price indices

18th January 2023

Contents

1 Introduction

House prices are continuously rising worldwide, as seen in a study by Everett-Allen (2022), hence research to their underlying behaviour remains an interesting topic to understand the market. Modelling house prices is important in uncovering their movements and reactions. House price data is often given as a time series for a specific country or region. Time series are sequences of data points collected over time. Time series can be found in numerous fields, such as economics, finance, biology, physics and speech recognition; simply in any field that requires collecting data over a certain time span. Often, researchers are interested in modelling time series to understand and predict their movement patterns. As objects within the field of research can exhibit different behaviour, constructing a single one-fits-all model can provide inaccurate results. Therefore, one would consider making multiple models for subgroups that behave similarly. Alternatively, one would like to distinguish between different groups to find the true nature or source of the data within a large database. Ideally, this underlying group structure would be extracted from the data itself, instead of exogenous assignment of objects to specific clusters. This can be done by means of time series cluster analysis, which will be the focus of this paper.

For house prices, the rising level varies between regions as they might react differently to certain external shocks. Therefore, it is interesting to construct multiple models for countries or regions that behave equally. To do this it is necessary find homogeneous subgroups within the data, where individual groups of countries show similar house price dynamics. Subgroups can be assigned manually, such as based on geographical location or economic characteristics, but ideally the clusters are extracted from the data itself. Using the resulting clusters, the group structure can be included in a macroeconomic model or separate models can be constructed for each group respectively. This would result in better fitting models that can be used to better understand and predict the market.

Cluster analysis, or clustering, is a technique that aims to find homogeneous subgroups in a data set. Different types of clustering methods have been proposed, such as k-means clustering, EM algorithm on Gaussian mixture models and hierarchical clustering. These types of clustering are often applied to low-dimensional data. The problem with time series clustering is that the dimensions are high and similarities are hard to be determined. Also, with high-dimensional data, where the size of the data is small compared to the numbers of parameters, we run into the curse of dimensionality, as mentioned in Bouveyron et al. (2007), a phenomenon introduced by Bellman (1966). This means that distances between different objects are hard to define and therefore common distance metrics lose their meaning. All objects appear to be sparse and dissimilar. To adjust for this problem, some dimensionality reduction technique can to be

applied to the data first, before moving to the clustering itself. As we are dealing with time series data, it is also crucial to incorporate its sequential nature in the representation method.

This paper aims to find clusters in U.S. state house price indices to try to improve house price modelling by applying a novel dimensionality reduction technique for time series cluster analysis. As Phillips and Sul (2007) proposed as an extension of their house price clustering research, the effects of macroeconomic variables on the house prices will be considered by first modelling the house prices and using the residual time series for the cluster analysis. Iacoviello and Neri (2010) previously established 10 fundamental economic variables for structural house price modelling. Gupta et al. (2011) compared the structural model to statistical methods, to find that the statistical methods with the 10 fundamental variables outperformed the structural model. We will build upon these findings by moving towards including a latent group structure in a statistical model. We will investigate if this inclusion brings any improvement of the model.

Finally, the main focus of this paper is to investigate whether it is possible to reduce the dimensionality of the time series for cluster analysis by training an unsupervised neural network. This method applies a deep learning algorithm called an autoencoder, which extracts latent features whilst preserving the commonality of the data. On this lower-dimensional data, regular clustering techniques are be performed to uncover hidden group structures amongst time series. The autoencoder will consist of cells that retain the sequential input of the data, and is therefore suitable for time series analysis. The resulting clusters will be analyzed by standard clustering algorithm performance measures and by re-applying the house price regression model to see if the performance increases with respect to the original complete data model. The method has been applied to time series data before by Malhotra et al. (2016) and Nguyen et al. (2021) for anomaly detection and forecasting. In this paper, we investigate whether it can be applied for time series clustering purposes specifically. Overall, the methods can be used to construct better house price models, which can later be used for forecasting and investment decisions. They can also be extended to other time series clustering applications.

2 Related work

Liao (2005) classifies three approaches to time series clustering: raw-data-based, feature-based and model-based. According to the survey, the first approach handles the entire time series clustering by only modifying the similarity measure to be appropriate for time series. Degras et al. (2011) and Zhang (2013) have constructed raw-data-based algorithms that merge clusters based on parallelism. Feature-based and model-based methods convert the time series into a feature vector of lower dimension, or a number of model parameters. Nieto-Barajas and Contreras-Cristán (2014) takes a model-based Bayesian approach and proposes a general Poisson-Dirichlet process mixture model. According to the survey of Aghabozorgi et al. (2015), time series clustering consists of four aspects, as seen in Figure 1. This is closely related to the three approaches described before, where the choice of method within the aspects, or optionally disregarding it all completely, determines the approach classification.

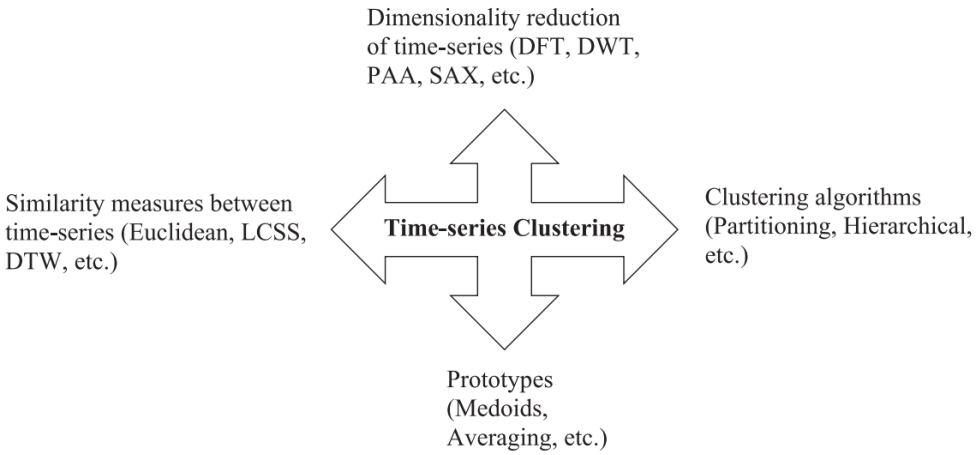


Figure 1: Four aspects of time series clustering. From: Aghabozorgi et al. (2015)

First, dimensionality reduction can be applied to capture the high dimensional data in a lower dimensional space, then a clustering technique can be applied to the resulting distances between the respective trends. For this clustering, a similarity measure between time series objects has to be defined. Lastly, a clustering prototype can be found, which is a representation of each cluster respectively. For example, the mean of the time-series at each point can be used if the time series are of equal length. This is called an averaging prototype.

For this paper we will focus on the dimensionality reduction of time series data for clustering. We will use a specific setup of this machine learning technique that preserves the sequential nature of the data. It is a feature-based time series clustering method. Section 2.1 will cover the relevant literature in this field, as well as the machine learning technique itself. Then, in Section 2.2 common clustering techniques from the literature are explained. We will use two of these to

evaluate the performance of the dimensionality reduction technique for clustering. Section 2.3 briefly touches upon the similarity measure. Machine learning methods require the input data to be of equal length, hence an averaging prototype as described before can be used and we will not expand further on this aspect.

Dimensionality reduction and subsequent clustering have been combined in the field of deep clustering, which also makes use of neural network structures. Although this is similar to the proposed method in this paper, research to deep clustering is very limited as it is essentially a black-box model. This means that there is an input and an output, but no information about the internal workings of the algorithm. In Section 2.4 deep clustering is briefly explained, as well as its downsides for research and application. In this paper, only the dimensionality reduction technique will be performed by the use of neural networks, and its output can be evaluated independently.

2.1 Dimensionality reduction

Dimensionality reduction, also known as time series representations, is a common method within the field of time series clustering. It transforms the series by projecting it on a lower dimensional space, which can then be clustered. According to Keogh and Pazzani (2000) and Lin et al. (2003), computing distances between whole time series objects can be very computationally expensive. Moreover, they find that time series databases can be extremely large, and that much research has been done to speeding up the search process. Keogh and Pazzani (2000) finds that representation methods reduce memory requirements and make the distance calculation overall less computationally expensive, as opposed to methods that have to calculate distances between whole time series. Besides, Ratanamahatana and Keogh (2005) state that it is much easier to make the reduced time series invariant to outliers in the data. Finally, as shown by Ding et al. (2002), dimensionality reduction also reduces the probability of a clustering algorithm being trapped in a local minimum.

With dimensionality reduction, high-dimensional data is transformed into a low-dimensional space that retains the commonality of the data. For time series specifically, it is important to retain the temporal structure of the data when reducing the dimensionality. In an extensive research by Ding et al. (2008), the effectiveness of 8 different representation methods was tested on 38 time series data sets. This is measured by comparing the tightness of the lower bounds. They conclude that there is very little difference in pruning power between the tested representation methods. Generally, there are four representation types: data adaptive, non-data adaptive, model-based and data dictated.

Data adaptive representations, such as singular value decomposition (SVD) by Faloutsos et al. (1994) and expanded further by Korn et al. (1997), can approximate each time series, but do not work well for comparing several time series simultaneously. As seen in the two papers, SVD was originally designed for ad hoc queries in time series data mining.

Non-data adaptive representations, such as discrete wavelet transform (DWT) by Chan and Fu (1999) and piecewise aggregate approximation (PAA) by Keogh and Pazzani (2000), are suited for time series of equal length and have a straightforward comparison of representations of several time-series. Although, according to Chaovalit et al. (2011), DWT might fail to be superior to other methods when dealing with stationary data. Also, Lkhagva et al. (2006) points out that PAA can miss important patterns in the data because of its averaging method.

Model based representations use a stochastic way to represent time series, such as Hidden Markov Models in Panuccio et al. (2002) or Auto-Regressive Moving Average in Corduas and Piccolo (2008). These methods test whether two time series originated from the same generating process and time series model structures have to be known a priori.

Finally, data dictated representations, such as the clipped representation by Ratanamahatana et al. (2005), define ratios automatically based on raw time series. Their representations are extremely compact and only suitable for clustering based on similarity in change, not for similarity in shape. Bagnall and Janacek (2005) find that if the time series are long enough, the information discarded by clipping does not decrease clustering accuracy.

Next, an unsupervised neural network technique is introduced that will be used for time series dimensionality reduction. This representation method learns from the data how to project time series of equal length onto an arbitrary lower dimensional space. It can deal with stochastic data, without any prior model assumptions on the time series. Moreover, it can compare an arbitrary amount of time series simultaneously of arbitrary length. Therefore, it has interesting properties over the methods described above.

2.1.1 LSTM-autoencoders

Long short-term memory (LSTM), as proposed by Hochreiter and Schmidhuber (1997), is a type of neural network that saves the sequential nature of the data, instead of treating each input completely independent of the others. This is particularly interesting for time series, as each input depends on the previous one. Because of the use of both short- and long-term memory, cell states can be communicated to the next input, as well as later inputs in the sequence.

LSTM-autoencoders have been proposed by Sutskever et al. (2014) and Cho et al. (2014) for sequence-to-sequence learning tasks. The LSTM-encoder is used to represent a sequence input

as a latent feature vector of fixed length, while the LSTM-decoder attempts to reconstruct the original sequence from the extracted feature vector. If the size this feature vector is smaller than the input sequence, dimensionality reduction is achieved. Training of the encoder and decoder occurs at the same time, and they are subsequently split up for encoding the latent features. As the output of the autoencoder is directly compared to its input, the learning method is unsupervised and does not need any labelling to learn data structures.

LSTM-autoencoders have been used for time series anomaly detection by Malhotra et al. (2016) and for time series forecasting by Nguyen et al. (2021). We extend this to time series cluster analysis. In other fields, it has been proposed for language generation and reconstruction by Li et al. (2015) and for sequence prediction in image captioning by Bengio et al. (2015).

2.2 Clustering algorithms

Different types of clustering methods exist, which can broadly be divided into six groups: centroid-based, hierarchical, density-based, distribution-based, model-based and grid-based clustering. The focus for this paper will be centroid-based and hierarchical clustering, as these methods do not need a prespecified number of clusters or some distribution assumption on the data. Also, hyperparameters for the other methods are not trivial to be determined in a time series feature representation setting. The two clustering methods will be used to evaluate the appropriateness of the dimensionality reduction technique for clustering and to find the group structure in the house price data.

2.2.1 Centroid-based clustering

Tarpey (2007) uses several transformations of the trend data and applies k-means clustering. k-means clustering is not invariant to linear transformations of the data. The clustering method, proposed by Forgey (1965), aims to partition the n observations into k sets in which each observation is assigned to the cluster with the nearest mean, so as to minimize the within-cluster sum of squares (WCSS). Objects are assigned to one cluster only. Proposed by Bezdek (1981), fuzzy clustering, also known as soft k-means, can be used for cluster assignments to a probability mixture of clusters. Similar to k-means, k-medoids is another partitional algorithm that chooses actual data points as centers, instead of cluster centers. Contrarily, actual cluster centres as used by regular k-means clustering do not necessarily correspond to one of the input data points. The disadvantage of k-means clustering is that the number of clusters k needs to be predefined. To determine this value, elbow plots and silhouette scores can be constructed. Also, the method does not allow for clusters to be of varying shapes.

2.2.2 Hierarchical clustering

Hierarchical clustering is often used as a visualisation of the hierarchy structure within the data, such as by Nieto-Barajas and Contreras-Cristán (2014). The structure is represented in a so-called dendrogram. Hierarchical clustering can be done agglomerative (bottom-up) or divisive (top-down), where you either take one all observations and iteratively merge them into clusters or start with one big cluster and split the data recursively. Moreover, the linkage criterion determines the distance between clusters of objects. Complete-linkage measures the maximum distance between the objects of different clusters, whereas single-linkage takes the minimum distance between the clusters. Other popular linkage criteria are group average, median and centroid. A common algorithm that performs hierarchical clustering is BIRCH, proposed by Zhang et al. (1996). Similar to k-means clustering, a silhouette score can be calculated to determine the quality of the cluster assignment. In the dendrogram, the optimal number of clusters can be observed, as further explained in Section 5.

2.3 Similarity measure

The extensive research of Ding et al. (2008), as mentioned in Section 2.1, also compared 9 similarity measures on the 38 time series databases. It concluded that the accuracy of elastic measures converge with that of Euclidean distance as the training set increases. On small data sets, where there are less then 400 time series in the training set, elastic measures are more accurate. Dynamic time warping (DTW), proposed by Berndt and Clifford (1994), and distance of the longest common subsequence (LCSS), proposed by Vlachos et al. (2002) are examples of elastic distance measures. Ding et al. (2008) finds that DTW and LCSS reduce the computation cost and enable effective lower-bounding, while yielding the same or even better accuracy, compared to other elastic measures. According to Aghabozorgi et al. (2015), Euclidean distance and DTW are the most commonly used similarity measures in time-series clustering. Therefore, we apply Euclidean distance as our similarity measure in this paper.

2.4 Deep clustering

The use of neural networks for clustering has been researched in the field of deep clustering. Deep clustering combines dimensionality reduction and clustering into a single model using deep neural networks. By optimizing these task simultaneously, the performance of both can be improved. Deep clustering makes use of a loss function in the autoencoder between the encoder en decoder part, to iteratively optimize a clustering objective.

Deep time series clustering (DTSC) has been applied in some previous studies, although

Alqahtani et al. (2021) finds that there is no ultimate architecture that solves the problem. Current methods use simple deep autoencoders for the network structure, which do not capture the sequential nature of time series data. In Li and Kameoka (2018), they propose to unconventionally use gated convolutional networks to capture long-term dependencies of the data, as an alternative to LSTM. LSTM is a recurrent neural network structure, which is a more natural choice for modeling long-term dependencies of time series data than a convolutional network. In this paper we will investigate the use of LSTM for time series dimensionality reduction.

Although deep clustering is a promising technique, according to Min et al. (2018) there is a lack of theoretical framework backing it up, as well as concerns about its interpretability and translation to real-world data sets. It relies on benchmark data sets to evaluate hyperparameters, which questions the plausibility of real-world application. Also, the entire method is a black-box, where input and output are generated without knowledge into the internal workings. Therefore, we will treat the dimensionality reduction and clustering methods separately. This way we can properly evaluate the appropriateness of the dimensionality reduction method, using LSTBM neural networks, by comparing it to commonly used time series representation methods. Also, this leads to more control in tuning the hyperparameters for the representation task specifically, rather than trying to find the optimal set for both tasks simultaneously. This should yield more optimal results for the dimensionality reduction.

2.5 House price modelling

House price dynamics can be modelled by use of a dynamic stochastic general equilibrium (DSGE) model, such as by Iacoviello and Neri (2010). This is a dynamic structural method. In Gupta et al. (2011), they compare the DSGE model for U.S. house prices by state, using the proposed 10 fundamental economic variables of Iacoviello and Neri (2010), to variations on a vector autoregression (VAR) model for forecasting with these variables. The VAR model is a statistical method. They conclude that DSGE performs poorly compared to the various VAR models, and that the utilization of fundamental economic variables improves the forecasting performance, which holds more strongly for the 10 fundamental economic variables, compared to a data set of 120 macroeconomic variables.

Spatial effects are also important in house price analysis, as mentioned in Cohen et al. (2016). Pijnenburg (2017) concludes that house prices often spill over to neighbouring regions when prices increase, but also in case of a price decrease. Although spatial spillover is an important side of house price modelling, we will not include their effects as the focus of this paper is not to construct the best fitting pricing model, but to see if existing models are improved

by including group structures through LSTM-autoencoder dimensionality reduction clustering. For this purpose, we will use the more straightforward panel VAR model as in Gupta et al. (2011) to analyze model improvement.

2.6 House price clustering

In Van Dijk et al. (2011), they attempt to use multidimensional scaling (MDS) on the correlations of first differences of regional house prices to find clusters in the house prices. MDS is a linear dimension reduction technique that visualizes the distances between data objects by mapping them onto a 2-dimensional space. In their paper, they do not find any apparent clusters resulting from the MDS graphs. They conclude that dividing the regions into different groups based only on the cross-correlations of the regional house prices is not a meaningful possibility and move to conditional clustering analysis using latent class techniques as proposed by Paap et al. (2005).

Similar to this paper, Apergis and Payne (2012) also investigates the group structure in quarterly U.S. house prices by state and finds three groups, here called convergence clubs as they apply the convergence and clustering procedure proposed by Phillips and Sul (2007). States in these groups can broadly be categorized in three entire BEA regions with some states in other regions, states primarily in the Southeast and Plains regions with some states in other regions and two separate states in the Southeast. Phillips and Sul (2007) propose to extend their method in the case of multiple variable panel regression, by working with panel regression residuals or through panel VAR and error correction formulations. In this paper, we will use the panel VAR to construct residual time series for subsequent clustering techniques.

Vatansever et al. (2020) takes a model-based autoregressive (AR) fuzzy-clustering approach to find homogenous housing market areas in the Turkish housing market. They find three distinct groups among 196 districts of 5 major cities of Turkey where house sale price moves together. Their model improves forecasting compared to regular AR modelling. Although, as mentioned in Section 2.1, these methods need model structures to be known a priori. The methodology proposed by this paper does not need any time series model structure presumptions.

3 Data

The time series clustering method will be applied to house price indices (HPI) for the 50 U.S. states. The quarter on quarter differences of the log price index will be used from the Bank for International Settlements. The original HPI before transformation can be seen in figure 2.

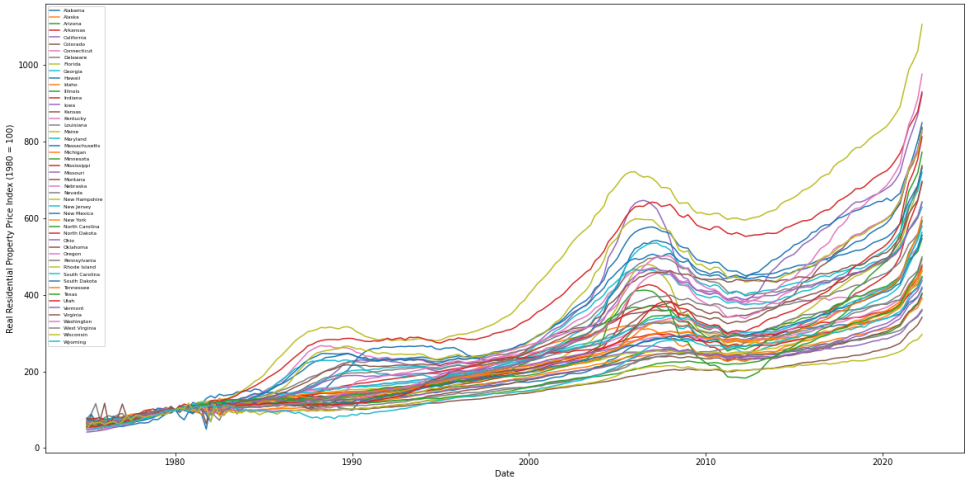


Figure 2: Real Residential Property Price Index by U.S. State

3.1 Fundamental variables

For modelling these indices, nine other macroeconomic variables will be used in accordance with Iacoviello and Neri (2010). These variables are Aggregate Consumption (AC), Business Fixed Investment (BFI), Residential Investment (RI), Inflation (INF), Nominal Short-Term Interest Rate (IR), Hours in Consumption Sector (HX), Hours in Housing Sector (HH), Wage Inflation in Consumption Sector (WIC) and Wage Inflation in Housing Sector (WIH). A more in-depth description of the construction and retrieval of these variables can be found in Appendix A. In Appendix B, plots can be found for each of the ten variables described above.

AC, BFI, RI, INF, IR, WIC and WIH are reported on a national level, and are therefore similar over all panel members. The unique data for each state of these variables are either too short or not reported on a quarterly basis. HPI, HX and HH are reported on a state level, therefore each panel member as a unique time series for these variables.

As we use the data for constructing a VAR model to predict errors and reduce their dimensionality using a neural network technique, the time series need to be of equal length. Therefore, the full panel data set only runs from 2002 Q1 to 2021 Q1, although some variables can be found for longer periods of time. The transformed variables as mentioned in Appendix A, adjusted to be of equal length, are plotted in Appendix B Figure B.1.

3.2 Preliminary testing

For VAR modelling, we first check the stationarity of the time series using the augmented Dickey-Fuller (ADF) test to test the presence of a unit root in the time series. If this null hypothesis can be rejected, we assume the time series to be stationary, which is one of the key assumptions for consistent estimates of the VAR model. This test is applied to all time series of each panel member respectively. If some time series are found to be non-stationary, the difference of the time series ($\Delta x = x_t - x_{t-1}$) will be used to attempt to make the time series stationary and to construct the VAR model on. The results of the tests for each variable at a 5% significance level can be found in Table 1a. Corresponding p-values for the ADF test results on the three data sets can be found in Appendix B Tables B.1, B.2 and B.3.

Table 1: ADF tests: number of non-stationary results

(a) 5% critical value

Variable	x	$\Delta_1(x)$	$\Delta_2(x)$
HPI	45	11	0
AC	50	50	0
BFI	50	0	0
RI	50	50	0
INF	50	0	0
IR	0	0	0
HC	22	2	0
HH	49	11	0
WIC	0	0	0
WIC	50	0	0

(b) 0.1% critical value

Variable	x	$\Delta_1(x)$	$\Delta_2(x)$
HPI	50	13	0
AC	50	50	0
BFI	50	0	0
RI	50	50	0
INF	50	0	0
IR	50	50	0
HC	38	22	0
HH	50	19	0
WIC	0	0	0
WIH	50	0	0

From these results we conclude that we need to use the second differences of the time series for proper VAR modelling. This also means that the length of the time series will be slightly shorter. Note that some of the variables were already differenced, as mentioned in Section 3.1 and summarized in Appendix A, meaning that we take a third difference of some variable in these instances.

Furthermore, we conduct cointegration tests tests to establish the presence of a statistically significant connection between time series. When two or more time series are cointegrated, they have a statistically significant relationship, which is needed for VAR models. We use the Johansen test to test cointegrating relationships between several non-stationary time series data. The test can take two approaches, the trace test or the maximum eigenvalue test. One example of a trace test result for the first panel member (Alabama) can be found in Table 2a. In Table 2b the sum of all significant trace test results are reported for each variable respectively, at a 5% significance level. We can see that WIH is not cointegrated with the other time series in 45 out of 50 panel members. As all other variables are siginificant and therefore cointegrated for

most or all of the panels, we only exclude WIH from these results.

Table 2: Johansen cointegration trace test results

(a) Results for $i = 1$ (Alabama)

Variable	Test Stat	>C(95%)	Signif.
HPI	875.77	>219.4051	True
AC	608.8	>179.5199	True
BFI	444.59	>143.6691	True
RI	300.84	>111.7797	True
INF	209.64	>83.9383	True
IR	127.15	>60.0627	True
HC	78.39	>40.1749	True
HH	38.03	>24.2761	True
WIC	14.43	>12.3212	True
WIH	0.14	>4.1296	False

(b) Number of significant test results

Variable	Significant
HPI	50
AC	50
BFI	50
RI	50
INF	50
IR	50
HC	50
HH	46
WIC	37
WIH	5

Lastly, we perform Granger causality Wald tests to see if the time series are useful for prediction. It tests the null hypothesis that the variable does not Granger-cause equation variable, in this case HPI. The results can be found in Table 3. From these results we can conclude that HHOUS is not a good predictor of HPI, and will be left out of the final VAR model.

Table 3: Panel VAR(3)-Granger causality Wald test

HPI	χ^2	d.f.	P > χ^2
AC	62.877	3	0.000
INF	222.274	3	0.000
IR	263.616	3	0.000
HC	67.058	3	0.000
HH	2.359	3	0.501
WIC	18.793	3	0.000
ALL	621.943	18	0.000

3.2.1 Multiple testing correction

As we run tests for each time series separately, it is also necessary to look at the multiple testing problem. Basically, the more tests, the higher the chance of one or more false discoveries (i.e. falsely rejecting the null hypothesis). We will adjust for this by using the Bonferroni correction for the critical value, that rejects the null hypothesis if

$$p \leqslant \frac{\alpha}{m} \tag{1}$$

where α is the critical value, we use 5%, and m is the total number of hypothesis tested. As we run the ADF, cointegration and Granger tests 50 times, we use $m = 50$, which results in $p = 0.001$. The Bonferroni correction controls for the family-wise error rate (FWER), which is

the probability of making one or more false discoveries. In Table 1b, we can see the results of the ADF tests at a 0.1% significance level. All second difference variables are still stationary. In Appendix B Table B.4, we can see the maximum eigenvalue cointegration test p-values for the second differenced data. We can see that the p-value is always smaller than 0.001 for all variables. Hence, we can reject the null hypothesis and assume all variables are cointegrated. For the Ganger causality Wald test, our findings stay the same as all variables other than HH have a p-value smaller than the Bonferroni corrected critical value.

The final VAR model, resulting from these tests, will be provided in Section 4. It will also expand upon the dimensionality reduction of the residual time series, the latent feature clustering methods and performance measures and the re-application of the VAR model with its performance measures.

4 Methodology

In this section, the proposed method for time series dimensionality reduction and clustering is explained. First, a VAR model will be fitted on the second difference of HPI and the predictors to cancel the underlying macroeconomic effects and find meaningful clusters in the predicted residuals. Then, the dimensionality of the residual data will be reduced by the use of LSTM-autoencoders, after which k-means clustering is performed on the extracted latent features of the residuals. Then, we evaluate the clusters by re-applying the VAR model for each cluster respectively and compare the fit of the models to the original full-panel VAR model.

4.1 Vector autoregression

As mentioned in Section 3, we run ADF tests to check for stationarity, cointegration tests for statistically significant relationships between the variables and a post-model Granger causality test for testing the predictive power of the variables on the HPI. Furthermore, BFI and RI were already left out of the Granger causality test results, as incorporating these variables did not lead to a converging VAR model, and therefore did not lead to any meaningful results.

We choose three lags, in order not to over-specify the model and have some information still in the residuals to perform clustering on. This is in contrast with previous findings that the ideal number of lags in the model is eight, which also corresponds with the paper of Gupta et al. (2011). Besides, with eight lags the model does not perform well due to the relatively short length of the time dimension. With eight lags the model will be overspecified, where only a few of the many parameters are significant. This can lead to inflated standard errors due to the presence of many redundant predictors and cannot be used to describe the effect of a predictor on the response. As we will use the errors in further clustering analysis, we need to circumvent these issues. With three lags, we do not run into these problems, whilst still maintaining some lagged structure in the model. The resulting VAR(3) model can be seen in Equations 2 to 7, where β, γ, ζ, η, θ and λ are the coefficients corresponding to the variables and ε_{it} are the unobserved residuals for panel member i at time t. Again, a detailed description of all variables can be found in Appendix A.

$$
\begin{aligned}
\Delta_2\mathrm{HPI}_{it} = {} & \beta_1\Delta_2\mathrm{HPI}_{i,t-1} + \beta_2\Delta_2\mathrm{HPI}_{i,t-2} + \beta_3\Delta_2\mathrm{HPI}_{i,t-3} \\
& + \beta_4\Delta_2\mathrm{AC}_{t-1} + \beta_5\Delta_2\mathrm{AC}_{t-2} + \beta_6\Delta_2\mathrm{AC}_{t-3} \\
& + \beta_7\Delta_2\mathrm{INF}_{t-1} + \beta_8\Delta_2\mathrm{INF}_{t-2} + \beta_9\Delta_2\mathrm{INF}_{t-3} \\
& + \beta_{10}\Delta_2\mathrm{IR}_{t-1} + \beta_{11}\Delta_2\mathrm{IR}_{t-2} + \beta_{12}\Delta_2\mathrm{IR}_{t-3} \\
& + \beta_{13}\Delta_2\mathrm{HC}_{i,t-1} + \beta_{14}\Delta_2\mathrm{HC}_{i,t-2} + \beta_{15}\Delta_2\mathrm{HC}_{i,t-3} \\
& + \beta_{16}\Delta_2\mathrm{WIC}_{t-1} + \beta_{17}\Delta_2\mathrm{WIC}_{t-2} + \beta_{18}\Delta_2\mathrm{WIC}_{t-3} \; + \varepsilon_{it}^{(\mathrm{HPI})}
\end{aligned} \tag{2}
$$

$$
\begin{aligned}
\Delta_2\mathrm{AC}_{t} = {} & \gamma_1\Delta_2\mathrm{HPI}_{i,t-1} + \gamma_2\Delta_2\mathrm{HPI}_{i,t-2} + \gamma_3\Delta_2\mathrm{HPI}_{i,t-3} \\
& + \gamma_4\Delta_2\mathrm{AC}_{t-1} + \gamma_5\Delta_2\mathrm{AC}_{t-2} + \gamma_6\Delta_2\mathrm{AC}_{t-3} \\
& + \gamma_7\Delta_2\mathrm{INF}_{t-1} + \gamma_8\Delta_2\mathrm{INF}_{t-2} + \gamma_9\Delta_2\mathrm{INF}_{t-3} \\
& + \gamma_{10}\Delta_2\mathrm{IR}_{t-1} + \gamma_{11}\Delta_2\mathrm{IR}_{t-2} + \gamma_{12}\Delta_2\mathrm{IR}_{t-3} \\
& + \gamma_{13}\Delta_2\mathrm{HC}_{i,t-1} + \gamma_{14}\Delta_2\mathrm{HC}_{i,t-2} + \gamma_{15}\Delta_2\mathrm{HC}_{i,t-3} \\
& + \gamma_{16}\Delta_2\mathrm{WIC}_{t-1} + \gamma_{17}\Delta_2\mathrm{WIC}_{t-2} + \gamma_{18}\Delta_2\mathrm{WIC}_{t-3} \; + \varepsilon_{it}^{(\mathrm{AC})}
\end{aligned} \tag{3}
$$

$$
\begin{aligned}
\Delta_2\mathrm{INF}_{t} = {} & \zeta_1\Delta_2\mathrm{HPI}_{i,t-1} + \zeta_2\Delta_2\mathrm{HPI}_{i,t-2} + \zeta_3\Delta_2\mathrm{HPI}_{i,t-3} \\
& + \zeta_4\Delta_2\mathrm{AC}_{t-1} + \zeta_5\Delta_2\mathrm{AC}_{t-2} + \zeta_6\Delta_2\mathrm{AC}_{t-3} \\
& + \zeta_7\Delta_2\mathrm{INF}_{t-1} + \zeta_8\Delta_2\mathrm{INF}_{t-2} + \zeta_9\Delta_2\mathrm{INF}_{t-3} \\
& + \zeta_{10}\Delta_2\mathrm{IR}_{t-1} + \zeta_{11}\Delta_2\mathrm{IR}_{t-2} + \zeta_{12}\Delta_2\mathrm{IR}_{t-3} \\
& + \zeta_{13}\Delta_2\mathrm{HC}_{i,t-1} + \zeta_{14}\Delta_2\mathrm{HC}_{i,t-2} + \zeta_{15}\Delta_2\mathrm{HC}_{i,t-3} \\
& + \zeta_{16}\Delta_2\mathrm{WIC}_{t-1} + \zeta_{17}\Delta_2\mathrm{WIC}_{t-2} + \zeta_{18}\Delta_2\mathrm{WIC}_{t-3} \; + \varepsilon_{it}^{(\mathrm{INF})}
\end{aligned} \tag{4}
$$

$$
\begin{aligned}
\Delta_2\mathrm{IR}_{t} = {} & \eta_1\Delta_2\mathrm{HPI}_{i,t-1} + \eta_2\Delta_2\mathrm{HPI}_{i,t-2} + \eta_3\Delta_2\mathrm{HPI}_{i,t-3} \\
& + \eta_4\Delta_2\mathrm{AC}_{t-1} + \eta_5\Delta_2\mathrm{AC}_{t-2} + \eta_6\Delta_2\mathrm{AC}_{t-3} \\
& + \eta_7\Delta_2\mathrm{INF}_{t-1} + \eta_8\Delta_2\mathrm{INF}_{t-2} + \eta_9\Delta_2\mathrm{INF}_{t-3} \\
& + \eta_{10}\Delta_2\mathrm{IR}_{t-1} + \eta_{11}\Delta_2\mathrm{IR}_{t-2} + \eta_{12}\Delta_2\mathrm{IR}_{t-3} \\
& + \eta_{13}\Delta_2\mathrm{HC}_{i,t-1} + \eta_{14}\Delta_2\mathrm{HC}_{i,t-2} + \eta_{15}\Delta_2\mathrm{HC}_{i,t-3} \\
& + \eta_{16}\Delta_2\mathrm{WIC}_{t-1} + \eta_{17}\Delta_2\mathrm{WIC}_{t-2} + \eta_{18}\Delta_2\mathrm{WIC}_{t-3} \; + \varepsilon_{it}^{(\mathrm{IR})}
\end{aligned} \tag{5}
$$

$$
\begin{aligned}
\Delta_2\mathrm{HC}_{it} = {} & \theta_1\Delta_2\mathrm{HPI}_{i,t-1} + \theta_2\Delta_2\mathrm{HPI}_{i,t-2} + \theta_3\Delta_2\mathrm{HPI}_{i,t-3} \\
& + \theta_4\Delta_2\mathrm{AC}_{t-1} + \theta_5\Delta_2\mathrm{AC}_{t-2} + \theta_6\Delta_2\mathrm{AC}_{t-3} \\
& + \theta_7\Delta_2\mathrm{INF}_{t-1} + \theta_8\Delta_2\mathrm{INF}_{t-2} + \theta_9\Delta_2\mathrm{INF}_{t-3} \\
& + \theta_{10}\Delta_2\mathrm{IR}_{t-1} + \theta_{11}\Delta_2\mathrm{IR}_{t-2} + \theta_{12}\Delta_2\mathrm{IR}_{t-3} \\
& + \theta_{13}\Delta_2\mathrm{HC}_{i,t-1} + \theta_{14}\Delta_2\mathrm{HC}_{i,t-2} + \theta_{15}\Delta_2\mathrm{HC}_{i,t-3} \\
& + \theta_{16}\Delta_2\mathrm{WIC}_{t-1} + \theta_{17}\Delta_2\mathrm{WIC}_{t-2} + \theta_{18}\Delta_2\mathrm{WIC}_{t-3} \; + \varepsilon_{it}^{(\mathrm{HC})}
\end{aligned} \tag{6}
$$

$$
\begin{aligned}
\Delta_2\mathrm{WIC}_{t} = {} & \lambda_1\Delta_2\mathrm{HPI}_{i,t-1} + \lambda_2\Delta_2\mathrm{HPI}_{i,t-2} + \lambda_3\Delta_2\mathrm{HPI}_{i,t-3} \\
& + \lambda_4\Delta_2\mathrm{AC}_{t-1} + \lambda_5\Delta_2\mathrm{AC}_{t-2} + \lambda_6\Delta_2\mathrm{AC}_{t-3} \\
& + \lambda_7\Delta_2\mathrm{INF}_{t-1} + \lambda_8\Delta_2\mathrm{INF}_{t-2} + \lambda_9\Delta_2\mathrm{INF}_{t-3} \\
& + \lambda_{10}\Delta_2\mathrm{IR}_{t-1} + \lambda_{11}\Delta_2\mathrm{IR}_{t-2} + \lambda_{12}\Delta_2\mathrm{IR}_{t-3} \\
& + \lambda_{13}\Delta_2\mathrm{HC}_{i,t-1} + \lambda_{14}\Delta_2\mathrm{HC}_{i,t-2} + \lambda_{15}\Delta_2\mathrm{HC}_{i,t-3} \\
& + \lambda_{16}\Delta_2\mathrm{WIC}_{t-1} + \lambda_{17}\Delta_2\mathrm{WIC}_{t-2} + \lambda_{18}\Delta_2\mathrm{WIC}_{t-3} \; + \varepsilon_{it}^{(\mathrm{WIC})}
\end{aligned} \tag{7}
$$

From this model, the residuals $\hat{\varepsilon}_{it}^{(\mathrm{HPI})}$ for predicting the HPI will be calculated by fitting the estimated parameters of the panel VAR(3) model results to the actual data. For future notation we use

$$\hat{\varepsilon}_{it}^{(\mathrm{HPI})} = \epsilon_{it} \tag{8}$$

to help distinguish between the input ϵ_{it} and output $\hat{\epsilon}_{it}$ of the autoencoder in Section 4.2. We construct a series of HPI residuals over time for each panel member. As these residuals describe how much the expected HPI differs from the actual HPI, clustering these residuals would find groups of panel members with similar external reactions on the HPI, i.e. groups for which the HPI behaves similarly with the effects of the fundamental macroeconomic variables canceled out by the VAR model. Next, latent features will be extracted from these residual time series before performing cluster analysis.

4.2 Latent feature extraction

An autoencoder, as seen in Figure 3, is a neural network architecture that learns a feature representation of the input by comparing the output to the input and adjusting the network weights according to the deviation of the two using backpropagation. Therefore, the normally supervised neural network technique is now unsupervised, as it uses the original data as the labels to check the prediction accuracy and update the network weights.

To extract the learned feature set, the autoencoder will be split up in an encoder and

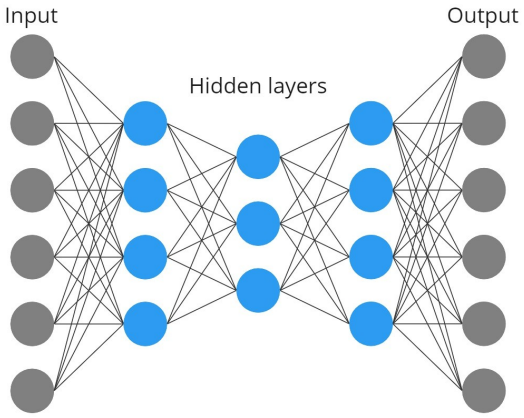


Figure 3: An autoencoder

decoder and the data will be ran through the encoder. If this feature set, found in the middle hidden layer of the autoencoder, is smaller than the size of the input (and therefore also output), the dimensionality of the data is reduced.

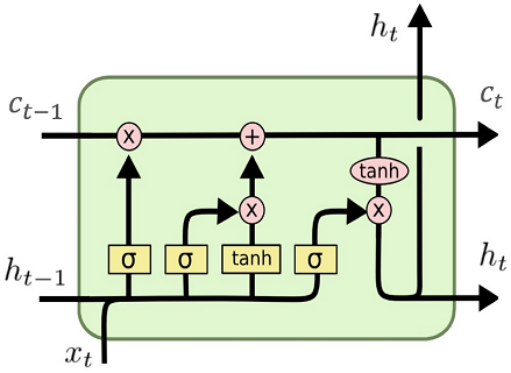


Figure 4: An LSTM cell

An LSTM-autoencoder is a type of autoencoder that makes use of LSTM cells, as seen in Figure 4, to preserve the sequential nature of the input data, in this case time series data. These cells consist of three gates: an input gate, an output gate and a forget gate. Equations 10 to 15 show how the network passes the cell states on from one LSTM cell to the next, where

$$\sigma_s = \frac{1}{1+e^{-x}}, \quad \sigma_h = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{9}$$

are the sigmoid and hyperbolic tangent activation function respectively. The plots of these activation functions can be seen in Figure 5 from Rao and Gudivada (2018).

First, the forget gate f_t uses the sigmoid activation function, which forces its input to a value between 0 and 1, over the current state and the previous hidden state. The input gate i_t updates the cell state similarly. Also in the input gate, the hyperbolic tangent activation

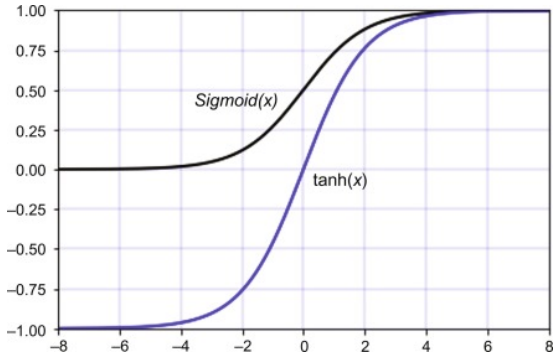


Figure 5: Sigmoid and hyperbolic tangent functions

function, which forces its input to a value between -1 and 1, is applied to the current input and the previous hidden state. The output of these two input parts are multiplied and added to the forget gate. Finally, the output gate passes the hidden state and current inputs through the sigmoid activation function and determines the hidden state will be passed on to the next cell by multiplying the activation outputs. Because of this structure, LSTM is capable of capturing short-term, as well as long-term dependencies within the data, which is important for time series analysis. Thus, the LSTM network passes on information from one cell to the next by

$$f_t = \sigma_s(W_f x_t + U_f h_{t-1} + b_f) \tag{10}$$

$$i_t = \sigma_s(W_i x_t + U_i h_{t-1} + b_i) \tag{11}$$

$$\tilde{c}_t = \sigma_h(W_c x_t + U_c h_{t-1} + b_c) \tag{12}$$

$$o_t = \sigma_s(W_o x_t + U_o h_{t-1} + b_o) \tag{13}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{14}$$

$$h_t = o_t \odot \sigma_h(c_t) \tag{15}$$

The LSTM-autoencoder will be used to reduce the dimensionality of the residual time series. We will use a two- and three-layer LSTM-encoder-decoder setup to compare the performance of the two setups and their results. In Figure 6, a simplified LSTM-autoencoder setup is depicted. The decoder part is the exact mirror image of the encoder, which we can also see in Figure 3.

For the 3-layer LSTM-encoder setup, the input time series of length 72 will be transformed in the hidden layers to a 128 length vector, then to a 32 length vector and finally the compressed feature size will be 12. Hence, we will reduce the dimensionality of a time series of length 72 to a feature set of size 12. The 2-layer LSTM-encoder is similar to the 3-layer setup, but then the only hidden layer between the input and the final compressed feature is of size 64. The autoencoder in Figure 3 corresponds to a 2-layer encoder. Furthermore, a 3-layer encoder would have 2 extra hidden layers in the autoencoder. Note here the difference between the entire

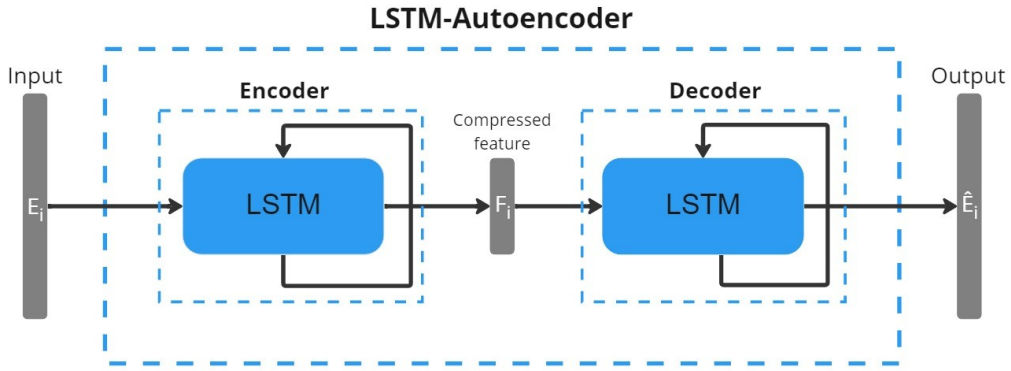


Figure 6: An LSTM-autoencoder

autoencoder and the encoder only.

For this paper specifically, the input of the LSTM-autoencoder are the fitted panel VAR(3) HPI residual time series

$$E_i = \left(\epsilon'_{i,1}, \epsilon'_{i,2}, \ldots, \epsilon'_{i,72}\right) \tag{16}$$

and the output is a predicted HPI residual time series

$$\hat{E}_i = \left(\hat{\epsilon}_{i,1}, \hat{\epsilon}_{i,2}, \ldots, \hat{\epsilon}_{i,72}\right) \tag{17}$$

where the compressed feature vector to be extracted from the encoder is

$$F_i = \left(f_{i,1}, f_{i,2}, \ldots, f_{i,12}\right) \tag{18}$$

for panel member $i \in [\text{Alabama}, \ldots, \text{Wyoming}]$. The predicted output $\hat{E}_i$ will be compared to the input E_i for network learning. The notation can also be seen in Figure 6. As we are working with neural network structures, the residual time series data will be normalized before it can be used as input. We normalize the data by taking

$$\epsilon'_{i,t} = \frac{\epsilon_{i,t} - \min \epsilon_i}{\max \epsilon_i - \min \epsilon_i} \tag{19}$$

By normalizing the data, the network will generally train faster and the probability of getting stuck in local optima is reduced. Note that the ϵ here are the residuals resulting from fitting the model to the data, which is not the same as the unobserved residuals ε in Equation 2, as seen in Equation 8.

The data is split into a train and validation set. The training data is used to learn the feature representation and the validation set will be used to evaluate the trained model in each epoch. For both sets, a loss can be calculated. The loss function quantifies the difference between the

output and the expected outcome. In the case of autoencoders, as the neural network attempts to reconstruct the original data, the expected outcome is equal to the input data. The loss is also used to derive the gradients and update the network weights. Hence, the autoencoder learns the feature representation F_i by feeding the network E_i in batches, comparing $\hat{E}$ to E with a loss function and adjust the weights after each batch accordingly. The amount of times the entire data set is passed through the neural network is the epoch count. As seen in the next Section, we will use 200 epochs as this ensures minimum loss convergence. Finally, the feature set can be calculated after training by breaking the autoencoder up into an encoder and decoder and passing each E_i through the encoder individually to get the corresponding F_i.

As the method reconstructs the input data, cross validation for evaluating model performance is not possible. Hence, the validation set will not be split up further in a validation and test set. Cross validation requires labelled data for outcome comparison and measures the accuracy of exactly hitting the target. In our case, the autoencoder approximately reconstructs the data, and the probability of exactly reproducing the input is infinitesimal. Therefore, actual performance, besides loss minimization convergence, is measured after clustering and subset VAR modelling (i.e. complete model performance) as explained in Sections 4.3 and 4.4.

4.3 Cluster analysis

Next, k-means clustering will be applied to the LSTM-autoencoded compressed residual features to find clusters of panel members. This method assigns individuals to the cluster with the nearest mean. This algorithm needs prespecification of the number of clusters k. Elbow plots will be constructed to find the optimal number of clusters. These plots show the within-cluster sum of square (WCSS), the sum of distances between each point and the centroid of its cluster, over the number of clusters. WCSS uses Euclidian distance and can be calculated by

$$\text{WCSS} = \sum_{j=1}^{k} \sum_{x \in S_j} ||x - \mu_j||^2 = \sum_{j=1}^{k} \sum_{x \in S_j} \left|\left| \begin{bmatrix} f_{x,1} \\ \vdots \\ f_{x,12} \end{bmatrix} - \begin{bmatrix} \bar{f}_1 \\ \vdots \\ \overline{f_{12}} \end{bmatrix}_j \right|\right|^2 \tag{20}$$

for clusters $S_1, \ldots, S_k$ with cluster means $\mu_1, \ldots, \mu_k$, where $||a - b||$ is the Euclidian distance between a point a and b. As the number of clusters increases, WCSS decreases. Ideally, there is a point in this graph where the WCSS decreases less and forms a kink or so-called "elbow", which is the optimal number of clusters to be used in k-means clustering. Also, the silhouette score will be computed for these varying number of clusters. The silhouette score measures how

similar an object is to its own cluster compared to other clusters by

$$S = \frac{b - a}{\max(a, b)} \tag{21}$$

where a is the mean intra-cluster distance and b is the mean nearest-cluster distance. Its value is between -1 and 1 and a high value is desirable, which means that the clusters are well apart and clearly distinguished. Conversely, a value around 0 would mean that the distance between clusters is not significant and that clusters are not distinguished. Negative values indicate that the clusters are not correctly assigned.

Agglomerative clustering will also be done to evaluate the optimal number of clusters and better visualize the cluster assignments in a dendrogram. From this dendrogram, we can see the clustering assignments and sequence order up until the maximum number of clusters (i.e. the sample size). Also, distances between cluster splits in this plot show the optimal number of clusters. From the resulting agglomerative cluster labels, the silhouette scores can again be calculated and plotted over the number of clusters.

Finally, k-means clustering will be applied to the data with k equal to some optimal number of clusters as found by the WCSS plot, the dendrogram and the two silhouette plots. The data panels are split according to the resulting cluster allocations.

4.4 Model analysis

For each subset of the data as found by the clustering, a panel VAR(3) model, as in Equation 2, will be run again to see if these models are an improvement compared to the original whole-data panel VAR(3) model. Hence, if three clusters are found, there will be three seperate panel VAR(3) models on each cluster subset. The results of the models will be compared by the use of Akaike Information Criterion (AIC), R^2 and Root Mean Square Error (RMSE). AIC is calculated by

$$\text{AIC} = 2p - 2\ln(\hat{L}) \tag{22}$$

where p is the number of model parameters and $\hat{L}$ is the maximized value of the likelihood function of the model. As the AIC is calculated for a time series model of each respective panel member, we take the mean of all these AIC's as the criterion of the entire panel VAR(3) model. As the likelihood function for a VAR model is not trivial, the AIC will be calculated using the STATSMODELS package in PYTHON. The numeric outcome of this equation does not give us any information on the model fit, but models with a lower AIC are preferred over other models.

For R^2, we calculate the residual sum of squares (RSS) and total sum of squares (TSS) for the panel VAR(3) model for each panel member respectively and take the average fraction of the two statistics to get

$$R^2 = 1 - \frac{1}{n}\sum_i \left(\frac{\text{RSS}_i}{\text{TSS}_i}\right) = 1 - \frac{1}{n}\sum_i \left(\frac{\sum_t \epsilon_{it}^2}{\sum_t \left(\text{HPI}_{it} - \overline{\text{HPI}_i}\right)^2}\right) \tag{23}$$

where $\overline{\text{HPI}_i}$ is the average of HPI over time for panel member i. The resulting numeric should be between 0 and 1 and models with an R^2 closer to 1 are preferred over others. Finally, the RMSE can simply be calculated by taking, as the name suggests, the square root of the average squared errors as

$$\text{RMSE} = \sqrt{\frac{\sum_i \sum_t \left(\text{HPI}_{it} - \widehat{\text{HPI}}_{it}\right)^2}{n}} = \sqrt{\frac{\sum_i \sum_t \epsilon_{it}^2}{n}} \tag{24}$$

where $\widehat{\text{HPI}}_{it}$ is the predicted value for HPI for panel member i at time t. Similar to the AIC, the outcome does not give any information by itself, but the preferred model is the one with the lowest RMSE value, as this means that the deviation of the data compared to the model is smaller and the model fits better.

4.5 Flowchart

The flowchart of the complete methodology can be seen in Figure 7.

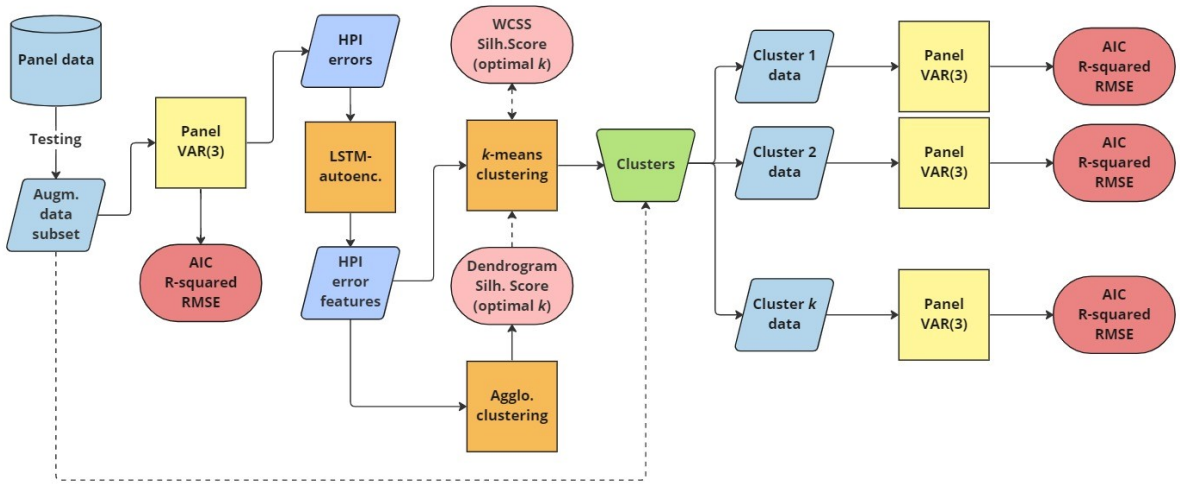


Figure 7: Flowchart of the complete methodology

To summarize, the augmented data subset as explained in Section 3 will be fitted to the panel VAR(3) model of Equation 2 to extract errors and to calculate performance measures. The errors will be used to construct residual time series and used as the LSTM-autoencoder

input, with which the latent features will be extracted. The resulting feature set will be clustered using k-means clustering and agglomerative clustering to find the optimal number of k. Using the clusters found by k-means clustering, the augmented data will be split up accordingly and these subsets will be fitted to the panel VAR(3) model again. The performance will be evaluated with respect to the original full-sample panel VAR(3) model to see if it has improved by incorporating the clusters. In the next Section, the results from this method will be presented.

5 Results

In this Section, the results of the described methods in Section 4 are reported. First, we shortly look at the panel VAR model results to start with, from which we calculate the residuals. Then, we report the results for the cluster analysis of these residual time series, specifically we look at the dimensionality reduction autoencoder performance, the quality of the clusters and the optimal number of clusters from the two clustering methods. Finally, we try to fit the original panel VAR(3) model to the subsets of clustered data to see if the fit improves by comparing the three model performance metrics.

5.1 Panel VAR(3)

First, we fit a panel VAR(3) model, as described in Equation 2, to the second difference transformed data and variables, as described in Section 3. In Table 4, the results of the panel VAR(3)

Table 4: Panel VAR(3) GMM estimation on second difference full-panel data

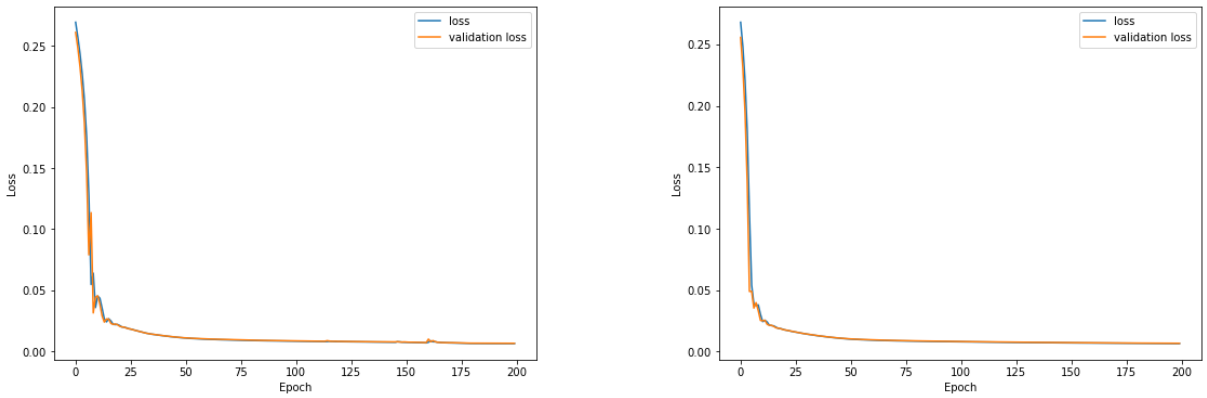
Final GMM Criterion Q(b) = 1.31e-28				No. of obs = 3550		
Initial weight matrix: Identity				No. of panels = 50		
GMM weight matrix: Robust				Ave. no. of T = 71.000		
	Coefficient	Std. err.	z	P>z	[95% conf.	int.]
hpi						
hpi						
L1.	-0.961	0.021	-45.79	0.000	-1.003	-0.92
L2.	-0.928	0.022	-42.64	0.000	-0.971	-0.886
L3.	-0.478	0.021	-22.88	0.000	-0.519	-0.437
ac						
L1.	0.000	0.000	-0.74	0.461	0.000	0.000
L2.	0.000	0.000	-5.29	0.000	0.000	0.000
L3.	0.000	0.000	-6.45	0.000	0.000	0.000
inf						
L1.	0.007	0.000	14.63	0.000	0.006	0.008
L2.	0.005	0.001	9.07	0.000	0.004	0.006
L3.	0.001	0.000	2.55	0.011	0.000	0.002
ir						
L1.	-0.008	0.001	-7.63	0.000	-0.01	-0.006
L2.	0.012	0.001	10.41	0.000	0.009	0.014
L3.	0.007	0.001	7.33	0.000	0.005	0.009
hc						
L1.	-0.004	0.001	-5.21	0.000	-0.006	-0.003
L2.	-0.007	0.001	-7.75	0.000	-0.009	-0.005
L3.	-0.006	0.001	-6.69	0.000	-0.008	-0.004
wic						
L1.	0.000	0.000	-0.23	0.819	0.000	0.000
L2.	0.001	0.000	2.4	0.016	0.000	0.001
L3.	0.001	0.000	3.45	0.001	0.000	0.001

for the HPI specifically are summarized, as this is the variable we want to predict using the fundamental economic variables and to construct the residual time series. The full panel VAR(3) GMM estimation includes these results for all variables, but we omit them as we do not make use of them in the estimation of the residuals and further analysis.

From Table 4 we can see that most coefficient are significant on a 5% level. The panel is fully balanced, hence the average number of timesteps T is exactly 71 and the number of panels are the 50 U.S. states. As the last lag is still significant for each variable, we do not need to adjust (reduce) this number. Also, all variables contain significant coefficients, so we do not need to exclude any variables from the model, based on these results. Post-model Granger causality tests were already performed in Section 3. Using this model, the residuals are calculated in STATA. The residual time series is used for time series cluster analysis in PYTHON.

5.2 Dimensionality reduction and clustering

The dimensionality of the predicted residual time series are reduced by 2- and 3-layer LSTM-autoencoders down to a dimensionality of 12. The data is split up into a train and validation set. In Figures 8a and 8b, the reconstruction losses of the LSTM-autoencoders are plotted for the two sets. The loss functions describe how well the model is learning. From the loss, gradients are derived to update the neural network weights. It measures the difference between expected output, in the case of autoencoders the input, and the produced output. We want to minimize the loss during training. Clearly, the loss function approaches 0, so convergence is achieved.

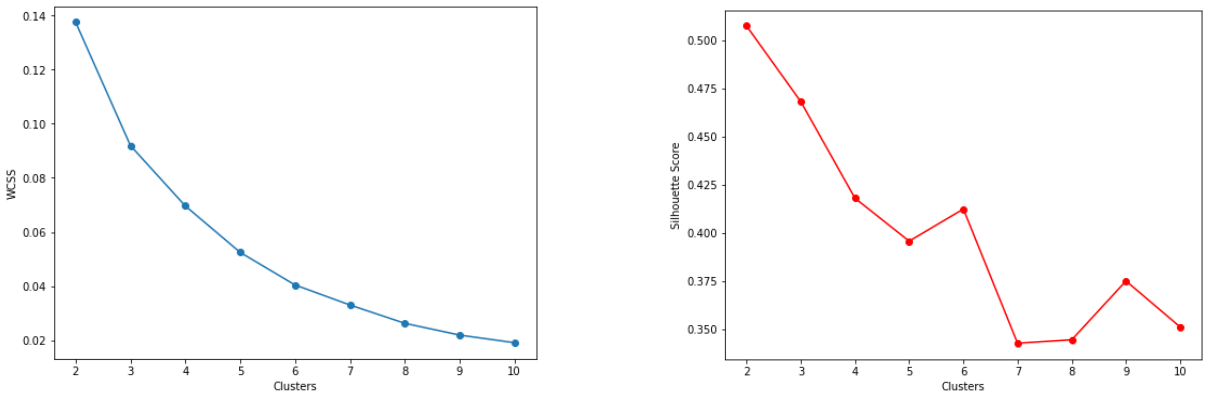


(a) 2-layer LSTM-autoencoder

(b) 3-layer LSTM-autoencoder

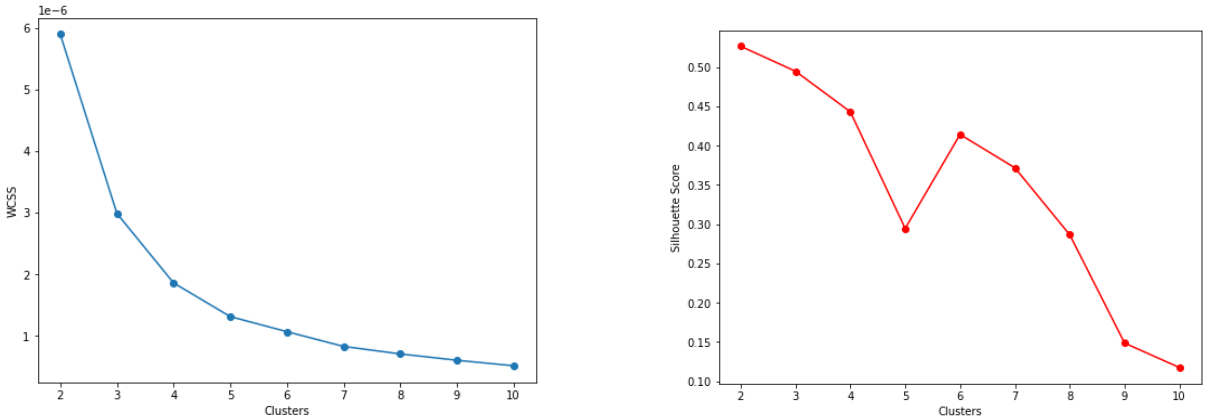
Figure 8: Train and validation loss

In Figures 9 and 10 we can find the elbow plots and silhouette scores for the k-means clustered residual time series features. The elbow plot shows how the WCSS decreases over the number of clusters, where we are looking for the "elbow". The silhouette plot gives information about the quality of the clusters, where we want to maximize its value.



(a) Elbow plot (b) Silhouette plot

Figure 9: 2-layer LSTM-autoencoded feature k-means clustering results



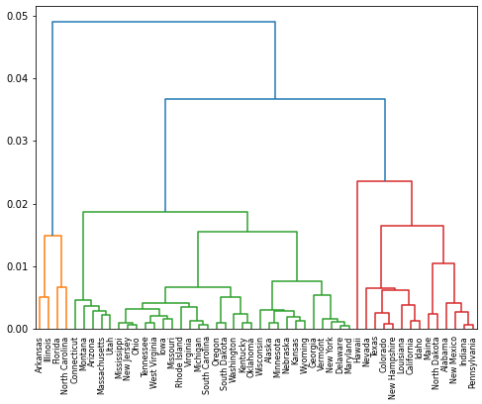
(a) Elbow plot (b) Silhouette plot

Figure 10: 3-layer LSTM-autoencoded feature k-means clustering results

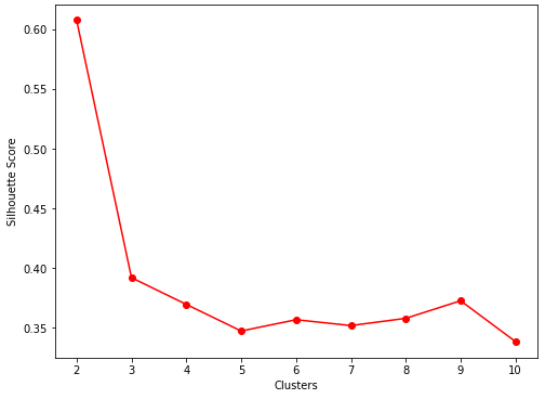
Both elbow plots are ambiguous, as the elbow shape is not necessarily evident. In Figure 9a, a very slight kink can be seen at 3 and 5 clusters. If we then look at the silhouette scores in Figure 9b, we see that the clusters are best defined at 2 and decrease as the number of clusters increase up until 5 clusters. As the elbow is not evident and the silhouette scores are over a relatively small and positive range, we will choose 2, 3 and 5 clusters for k-means clustering of the 2-layer LSTM-autoencoder extracted residual features.

In Figure 10a, the elbow shape is more present, but its exact location is not obvious. Rather, there seems to be a more fluid elbow shape at 3, 4 and 5 clusters. If we look at the silhouette scores, we see that 3 and 4 clusters are quite well defined, but there is a dip at 5 clusters. Therefore, we assume that 3 or 4 is the optimal number of clusters, but we will also consider to run k-means clustering for 5 clusters, keeping the low silhouette score in mind.

To further analyze the optimal number of clusters, we use agglomerative clustering to construct dendrograms and, again, the resulting silhouette scores. The results can be found in Figures 11 and 12.

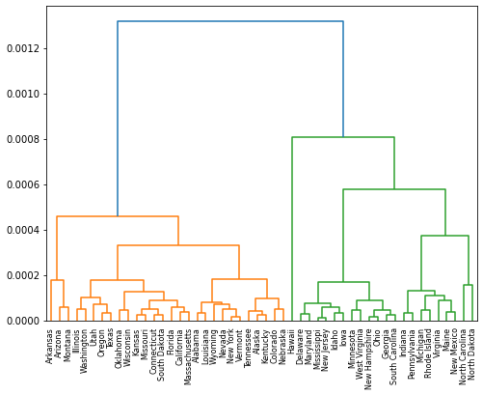


(a) Dendrogram

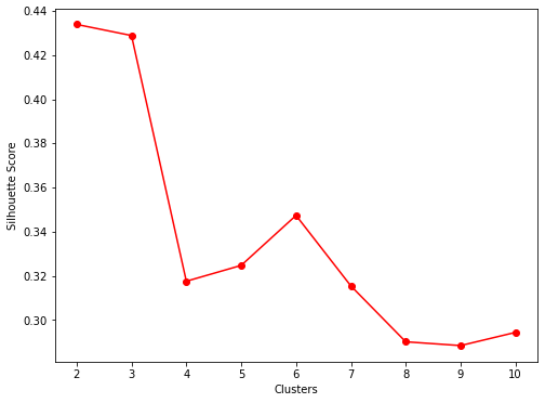


(b) Silhouette plot

Figure 11: 2-layer LSTM-autoencoded feature agglomerative clustering results



(a) Dendrogram



(b) Silhouette plot

Figure 12: 3-layer LSTM-autoencoded feature agglomerative clustering results

From the dendrograms, we try to find the largest horizontal slice between two splits. In Figure 11a, we can see that the preferred number of clusters according to the dendrogram is 3, and that 2 clusters is a close second. The accompanying silhouette plot in Figure 11b shows that 2 clusters are much better defined than 3 clusters. This can explain why the WCSS plot in Figure 9a does not show a clear elbow shape. Combining these findings, 2 seems to be the optimal number of clusters for the residual time series features.

In Figure 12a, 2 is clearly the optimal number of clusters. From the silhouette plot in Figure 12b, we can see that 2 and 3 clusters are approximately equally well defined, and much better defined than clustering on a larger number of clusters. This is in contrast with our findings in Figure 10, but can also explain why the elbow is not at one single point. We will assume that there is no optimal number of clusters, and inspect all data splits between 2 and 5 clusters.

In Table 5 we can see the summary of the optimal k for cluster analysis. Values within brackets are considered due to the ambiguity of the results, but are undesirable with respect to the other values, as described above.

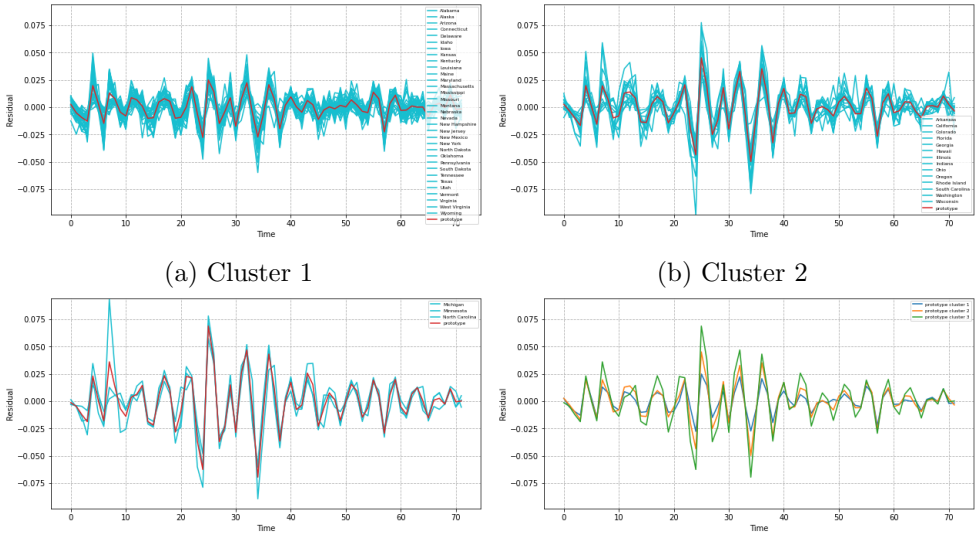
Table 5: Optimal number of clusters results

Setup	Optimal k		
	k-means	agglom.	final
2-layer LSTM	2, 3, 5	2, 3	2, 3(, 5)
3-layer LSTM	3, 4(, 5)	2, 3	2, 3, 4(, 5)

In Appendix B Figures B.2 and B.3, the cluster assignments are visualized for the first two principal components of the normalized and standardized feature data for varying number of clusters.

5.3 Panel data subsets

Next, the data was split up with respect to the number of clusters as described above. Cluster assignments will be the result of k-means clustering. In Figure B.5, the cluster residual plots for the LSTM(2)-K(3) model are depicted, as well as the resulting average cluster prototypes. In the cluster plots of Figures 13a to 13c, the prototype is the red time series and in Figure 13d we can see all three prototypes in one plot.



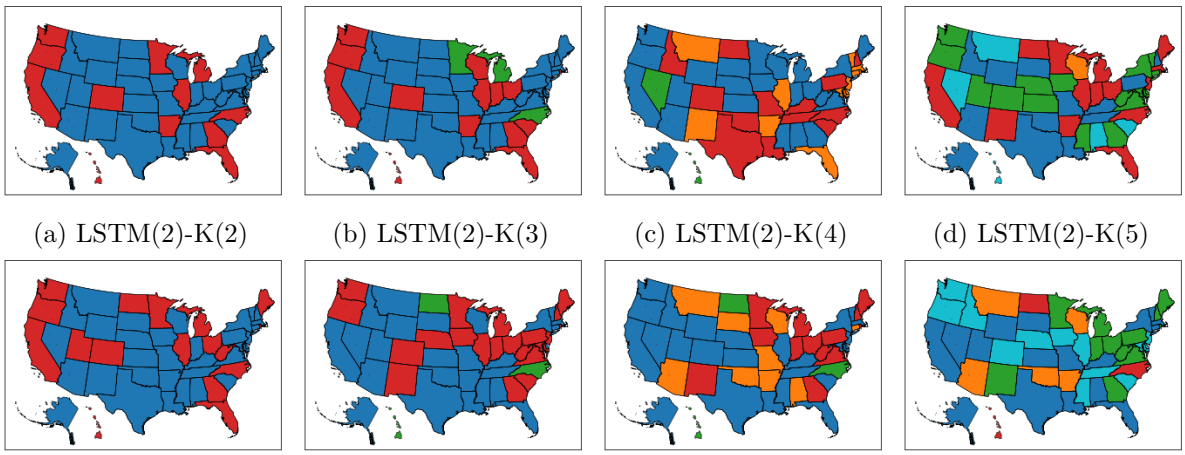
(a) Cluster 1 (b) Cluster 2

(c) Cluster 3 (d) Cluster prototypes

Figure 13: Residual cluster plots for LSTM(2)-K(3)

Some diversity between residual clusters and their prototypes can be seen from these plots. Also, as the silhouette scores of the clusters are strictly positive, these clusters should be apart and distinguished to some degree, assuming that the extracted features describe the residual time series well. Clusters 1 and 2 seem to distinguish between time series with low and high amplitude. Cluster 3 might incorporate some outliers that cannot be assigned to either previous group. Cluster residual plots for the other models can be found in Appendix B. Similar conclusions can be drawn from these Figures, but it is too hard to tell from visually analyzing these plots

only as the residuals behave similarly over time in general. The LSTM(2) models seem to be clustering the residuals slightly better, as the time series within the groups seem more equal and there is a more clear distinguish between low- and high-amplitude residuals, where LSTM(3) model groups appear to be less dense. In Figure 14, the cluster assignments of each model are geographically represented on a U.S. map. For comparing the two autoencoders, we have included the geographical cluster map of LSTM(2)-K(4) in these Figures. Even though, we do not consider these results as mentioned in Section 5.2 due to the optimal number of clusters.



(a) LSTM(2)-K(2) (b) LSTM(2)-K(3) (c) LSTM(2)-K(4) (d) LSTM(2)-K(5)

(e) LSTM(3)-K(2) (f) LSTM(3)-K(3) (g) LSTM(3)-K(4) (h) LSTM(3)-K(5)

Figure 14: Geographical representation of clustering results

From these maps we can see the difference in the cluster assignments of LSTM(2)-K(k) on top and LSTM(3)-K(k) on the bottom. For 2 and 3 clusters, the models produce similar results, but as the number of clusters increase, the cluster assignments of the two models diverge.

Next, the panel VAR(3) model that was used for constructing the residual time series and resulting clustering assignments is reapplied to the clustered subsets and their performance is evaluated with respect to the original full panel model.

5.4 Cluster subset panel VAR(3)

In Table 6 we can find the statistics for model evaluation. LSTM(l)-K(k) describes the representation clustering model of an l-layer LSTM-autoencoder and k-means clustering combination. For each of these combinations, the panel VAR(3) is applied to each cluster data subset respectively and the average performance statistics were calculated to get the values as presented in the Table. In bold we can see the optimal value for the performance statistic for each autoencoder method respectively. Individual results regarding these models for each cluster respectively can be found in Appendix B Table B.7. Here we can see that the bad performance of LSTM(2)-K(3) is mainly due to one cluster performing significantly worse than the full panel model. Moreover,

there are no extreme outliers in the three performance measures of all cluster models.

Table 6: Average performance results of the cluster data subsets

Model	AIC	R2	RMSE
PVAR(3)	-3.76097	0.346066	0.01556
LSTM(2)-K(2)	-3.5534	**0.335163**	0.017063
LSTM(2)-K(3)	-3.37954	0.302861	0.01831
LSTM(2)-K(5)[1]	**-3.73257**	0.289981	**0.016272**
LSTM(3)-K(2)	**-3.6644**	**0.328495**	**0.016277**
LSTM(3)-K(3)	-3.38658	0.320251	0.017583
LSTM(3)-K(4)	-3.50123	0.320302	0.017132
LSTM(3)-K(5)[1]	-3.5642	0.317468	0.016778

In Table B.7, we can see that for some specific clusters, the panel VAR(3) model does perform better, but that they also do not deviate much from the original model in Table B.6. Also in Appendix B, we can see the results for the Granger causality Wald tests for each individual data cluster model. Generally, the variables are good predictors for HPI, but in some cases the test is not significant. In Table 6 we can see that all cluster subset panel VAR(3) models perform worse than the original model on the complete data set. To compare these performance statistics this more easily, we can look at Figure 15. There, the performance measures are scaled between zero and one for comparison. The AIC and RMSE are inverted, so that we are looking for the maximum value for each measure for the best performing model. It is important to note that because of the scaling, the worst performing model is always at level 0 and the best performing model is always scaled at 1. These numeric values have no other significance but to commonly compare the performance results.

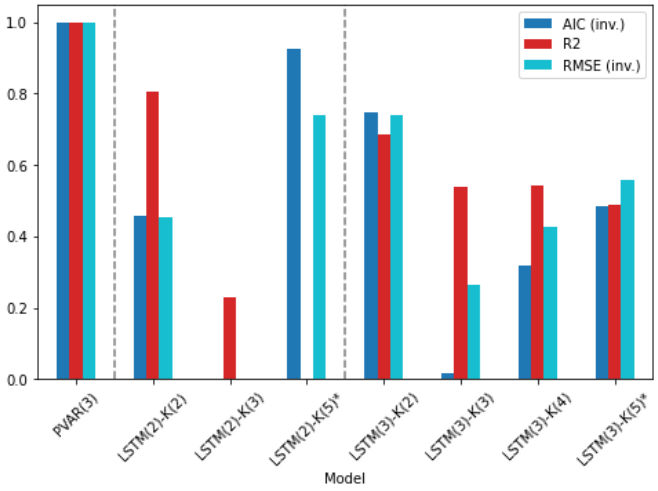


Figure 15: Scaled bar plots of performance of LSTM(l)-K(k) models

From this figure we can see that the average performance of the clustered models does not

[1]Results considered, but less likely the optimal clustering model as seen in Section 5.2.

improve when compared to the original panel VAR(3) model of the full panel. This is true for all three performance measures. Interesting is the LSTM(2)-K(5) model, which comes second to the PVAR(3) model in terms of AIC and RMSE, but has the worst R^2 of all the models. The overall best performing clustered model is the LSTM(3)-K(2), where 2 was also the clear optimal number of clusters in the corresponding dendrogram in Figure 12a. LSTM(2)-K(3) is one of the worst performing models, whilst 3 clusters was the optimal number of clusters in the corresponding dendrogram in Figure 11a. Another good candidate for the 2-layer network was 2 clusters, which clearly performs better on all three performance measures. Surprisingly, LSTM(l)-K(5) models perform relatively well compared to the other models for the same number of LSTM layers l, whilst these models were only considered due to the ambiguity of the results in Section 5.2 and were considered less preferable in terms of optimal number of clusters. Bar plots of the unscaled performance measures can be found in Appendix B Table B.10. We will expand more upon these findings in Sections 6 and 7.

6 Conclusion

This thesis presents a novel approach for analyzing house price movements by extracting homogeneous subgroups from the data and constructing models for each cluster respectively. The effects of fundamental macroeconomic variables were accounted for by the panel VAR(3) model. This model was used to predict residuals of the house prices to cancel out these effects. From the residuals, time series were constructed and used for cluster analysis. The resulting group structure was used to attempt to improve the initial full-panel regression.

Autoencoders were used to reduce the dimensionality of the residual time series. Two setups were trained: a 2-layer and a 3-layer LSTM-autoencoder. The autoencoders reduce the dimensionality of the residual time series from a size of 72 to 12 latent features. The loss approached 0 over the epochs, so convergence of the algorithms is achieved. Further evaluation of the autoencoders can only be done through clustering the results and analyzing model fits.

Centroid-based and hierarchical clustering were used to extract a group structure of the latent residual features. The optimal number of clusters was studied in detail, which did not result in one apparent solution. Instead, a range of possible optimal cluster candidates was used for futher evaluation. The algorithms provided clusters that were relatively well defined, as they have positive silhouette scores ranging from 0.3 to 0.6 for the cluster number candidate range. Although higher values closer to 1 would be preferred, we never find negative values that would indicate incorrect cluster specification. The augmented macroeconomic data set was split up according to the groups resulting from the latent residual feature clustering.

Comparing the cluster sets found by the two autoencoders respectively on a geographical map, we can see that as the number of clusters increase, there seems to be more discrepancy in the results. Whilst for a smaller number of clusters, the results seem to approximately coincide. It is also not trivial to find a logical explanation and real-life translation for the resulting clusters, such as political, geographical location, population density or Human Development Index. They also clearly do not correspond to any BEA regional grouping as in the paper by Apergis and Payne (2012). If we examine the clusters on the house price residual time series, there seems to be some group structure within and between the different clusters, especially a separation between high- and low-amplitude time series. These results could indicate that the dimensionality reduction and clustering does perform well for the residual time series themselves, but that these groups do not reflect the same clusters in the macroeconomic data and real-life samples.

The panel VAR(3) model is reapplied to each panel subset cluster respectively and compared to the full-panel model that was constructed previously. On all three performance metrics (i.e. AIC, R^2 and RMSE), the full-panel model outperformed the average cluster subset models for

both autoencoders and number of clusters within the optimal range, although some individual panel VAR(3) clusters performed better. There is no apparent connection between the number of clusters and the average performance of the resulting regression models. This result coincides with the incoherent clusters, observed on a geographical map, as we mentioned previously.

Generally, from the results we see that the 3-layer LSTM methods perform more stable results between the different number of clusters. Especially when we look at the R^2 statistic, and also at the RMSE, we see that there is much less variance when compared to the results of the 2-layer LSTM. However, this does not mean that the model performs better for clustering purposes, but that the optimal number of clusters appears to be more ambiguous for actual model performance. If we look at the AIC, both LSTM methods produce volatile results. As the results do not steer clearly in favour of one of the cluster models, there is no connection between these results and cluster performance statistics, i.e. the silhouette score.

The fact that the cluster models do not outperform the primary panel VAR model, could indicate that the residuals found by the model do not reflect the group structure as we argued they would. The clustering method itself seems to perform relatively well, as we see in the low- and high-amplitude visual split in Figure 13. This would indicate that the dimensionality reduction and subsequent clustering of the residuals does work as intended. The resulting clusters do not translate to any logical geographical group structure, as we have seen in Figure 14. This would also indicate some information gap between the panel VAR residuals and the actual data. Therefore, and because of the fluctuating model performance, it is not trivial to find an argument for the better performance of some LSTM(l)-K(k) model over the other.

Overall, we cannot say that the model is improved by incorporating the cluster structure as found by this paper. The results are also not always consistent between the two autoencoder methods and do not translate well to the actual samples. We will discuss the shortcomings of this paper and propose further research in the next Section.

7 Discussion and limitations

Unfortunately, the cluster analysis and price model methodology did not improve the overall performance with respect to the full-panel price model. One reason for this could be the proposed structure of the LSTM-autoencoders for dimensionality reduction. In this paper, we use the autoencoders to extract latent features of a size six times smaller than the original time series. This might be stretching the capabilities of the LSTM-autoencoders in reducing time series dimensionality. Often, this method is applied to trim noise in the data and reduce the dimensionality by a small factor only. Besides, we have only tested two specific LSTM-autoencoder setups. More setups with varying number of hidden layers can be tested to try and optimize this methodology. Also, for each setup it is possible to use a different number of nodes in the hidden layers. An extensive research, with performing cluster analysis and VAR(3) modelling on each autoencoded feature set respectively, could yield different results.

Other clustering methods can also be used in combination with the dimensionality reduction technique. In this paper, we only applied centroid-based and hierarchical clustering, but other approaches can be tested to see if the results are different. k-means is a very common method, but more advanced techniques exist for identifying clusters in many different ways. Distribution-based clustering algorithms such as DBSCAN are popular methods that can be looked at, but they do require different parameters for identifying the minimum size to form a dense region and the cluster neighbourhood size, which are not trivial. Moreover, different similarity measures other than Euclidean distance can be explored. DTW is an interesting and commonly used alternative for time series up until a size of 400, as mentioned in Section 2.3.

As we find that the clusters of residuals seem to have some group structure, but this does not translate to similar and logical clusters in the macroeconomic data, further evaluation of the panel VAR model in this methodology should be done. This could be an indication of some information being lost between the model residuals and the actual data. When we look at the Granger causality tests, it is possible to adapt individual models more to include only the statistically significant variables. Although this would exclude even more of the fundamental macroeconomic variables, it might have a positive effect on the model performance and further research would be interesting. Besides, other regression methods can be found in Gupta et al. (2011). They propose variations of the VAR for house price modelling, such as Bayesian (factor augmented) VAR. It would be interesting to investigate if the residuals from these models better describe clusters in the data, where a connection can be made to some geographical characteristic or other measurable regional difference. Besides, we can expand the model to include spatial effects, as mentioned in Section 2.5. It is interesting to see if including these effects alters the

model in a way that combining it with the group structure, does improve upon the original model with spatial effects without the group structure.

From a data perspective there were also some limitations. Most variables did not include variations in the panel, as seen in Appendix B Figure B.1. Some are extracted on a U.S. national level, instead of a state level. This is because these variables either are not publicly available on a state level, the state panel data is too short to perform meaningful VAR modelling and clustering or it is not available quarterly, but only yearly. For the latter, in experimenting we have tried to adjust for this by extending the yearly data to quarterly by taking the same level for each quarter in one year respectively or by a linear function between the years. However, as we try to capture the quarterly responses of the data and relations between the macroeconomic variables that can change in different ways between quarters and often is not constant or linear, this approach was considered counterintuitive. Besides, by taking these approaches, the first difference of the time series lost most of its information. Further research can focus on collecting the macroeconomic variables on a quarterly level for each panel member.

Finally, as mentioned in Section 2.4, deep clustering can be performed with similar autoencoder structures and residual time series data to see if results are similar. As this is a fully black-box method, we chose to break up the feature extraction and clustering to have more evaluation methods, but the combination in one neural network might yield more consistent results and improve the initial regression model.

References

Aghabozorgi, S., Shirkhorshidi, A. S., and Wah, T. Y. (2015). Time-series clustering–a decade review. *Information Systems*, 53:16–38.

Alqahtani, A., Ali, M., Xie, X., and Jones, M. W. (2021). Deep time-series clustering: A review. *Electronics*, 10(23):3001.

Apergis, N. and Payne, J. E. (2012). Convergence in us house prices by state: evidence from the club convergence and clustering procedure. *Letters in Spatial and Resource Sciences*, 5(2):103–111.

Bagnall, A. and Janacek, G. (2005). Clustering time series with clipped data. *Machine learning*, 58(2):151–178.

Bellman, R. (1966). Dynamic programming. *Science*, 153(3731):34–37.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. (2015). Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28.

Berndt, D. J. and Clifford, J. (1994). Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:.

Bezdek, J. C. (1981). *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media.

Bouveyron, C., Girard, S., and Schmid, C. (2007). High-dimensional data clustering. *Computational statistics & data analysis*, 52(1):502–519.

Chan, K.-P. and Fu, A. W.-C. (1999). Efficient time series matching by wavelets. In *Proceedings 15th International Conference on Data Engineering (Cat. No. 99CB36337)*, pages 126–133. IEEE.

Chaovalit, P., Gangopadhyay, A., Karabatis, G., and Chen, Z. (2011). Discrete wavelet transform-based time series analysis and mining. *ACM Computing Surveys (CSUR)*, 43(2):1–37.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Cohen, J. P., Ioannides, Y. M., and Thanapisitikul, W. W. (2016). Spatial effects and house price dynamics in the usa. *Journal of Housing Economics*, 31:1–13.

Corduas, M. and Piccolo, D. (2008). Time series clustering and classification by the autoregressive metric. *Computational statistics & data analysis*, 52(4):1860–1872.

Degras, D., Xu, Z., Zhang, T., and Wu, W. B. (2011). Testing for parallelism among trends in multiple time series. *IEEE Transactions on Signal Processing*, 60(3):1087–1097.

Ding, C., He, X., Zha, H., and Simon, H. D. (2002). Adaptive dimension reduction for clustering high dimensional data. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 147–154. IEEE.

Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552.

Everett-Allen, K. (2022). What happened to global house prices in 2021?

Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. *ACM Sigmod Record*, 23(2):419–429.

Forgey, E. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classification. *Biometrics*, 21(3):768–769.

Gupta, R., Kabundi, A., and Miller, S. M. (2011). Forecasting the us real house price index: Structural and non-structural models with and without fundamentals. *Economic Modelling*, 28(4):2013–2021.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Iacoviello, M. and Neri, S. (2010). Housing market spillovers: evidence from an estimated dsge model. *American Economic Journal: Macroeconomics*, 2(2):125–64.

Keogh, E. J. and Pazzani, M. J. (2000). A simple dimensionality reduction technique for fast similarity search in large time series databases. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 122–133. Springer.

Korn, F., Jagadish, H. V., and Faloutsos, C. (1997). Efficiently supporting ad hoc queries in large datasets of time sequences. *Acm Sigmod Record*, 26(2):289–300.

Li, J., Luong, M.-T., and Jurafsky, D. (2015). A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Li, L. and Kameoka, H. (2018). Deep clustering with gated convolutional networks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 16–20. IEEE.

Liao, T. W. (2005). Clustering of time series data—a survey. *Pattern recognition*, 38(11):1857–1874.

Lin, J., Keogh, E., Lonardi, S., and Chiu, B. (2003). A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11.

Lkhagva, B., Suzuki, Y., and Kawagoe, K. (2006). New time series data representation esax for financial applications. In *22nd International Conference on Data Engineering Workshops (ICDEW'06)*, pages x115–x115. IEEE.

Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., and Shroff, G. (2016). Lstm-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148*.

Min, E., Guo, X., Liu, Q., Zhang, G., Cui, J., and Long, J. (2018). A survey of clustering with deep learning: From the perspective of network architecture. *IEEE Access*, 6:39501–39514.

Nguyen, H., Tran, K. P., Thomassey, S., and Hamad, M. (2021). Forecasting and anomaly detection approaches using lstm and lstm autoencoder techniques with the applications in supply chain management. *International Journal of Information Management*, 57:102282.

Nieto-Barajas, L. E. and Contreras-Cristán, A. (2014). A bayesian nonparametric approach for time series clustering. *Bayesian Analysis*, 9(1):147–170.

Paap, R., Franses, P. H., and Van Dijk, D. (2005). Does africa grow slower than asia, latin america and the middle east? evidence from a new data-based classification method. *Journal of Development Economics*, 77(2):553–570.

Panuccio, A., Bicego, M., and Murino, V. (2002). A hidden markov model-based approach to sequential data clustering. In *Joint IAPR international workshops on statistical techniques in pattern recognition (SPR) and structural and syntactic pattern recognition (SSPR)*, pages 734–743. Springer.

Phillips, P. C. and Sul, D. (2007). Transition modeling and econometric convergence tests. *Econometrica*, 75(6):1771–1855.

Pijnenburg, K. (2017). The spatial dimension of us house prices. *Urban Studies*, 54(2):466–481.

Rao, C. and Gudivada, V. N. (2018). *Computational analysis and understanding of natural languages: principles, methods and applications*. Elsevier.

Ratanamahatana, C., Keogh, E., Bagnall, A. J., and Lonardi, S. (2005). A novel bit level time series representation with implication of similarity search and clustering. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 771–777. Springer.

Ratanamahatana, C. A. and Keogh, E. (2005). Multimedia retrieval using time series representation and relevance feedback. In *International Conference on Asian Digital Libraries*, pages 400–405. Springer.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Tarpey, T. (2007). Linear transformations and the k-means clustering algorithm: applications to clustering curves. *the american statistician*, 61(1):34–40.

Van Dijk, B., Franses, P. H., Paap, R., and Van Dijk, D. (2011). Modelling regional house prices. *Applied Economics*, 43(17):2097–2110.

Vatansever, M., Demir, I., and Hepşen, A. (2020). Cluster and forecasting analysis of the residential market in turkey: an autoregressive model-based fuzzy clustering approach. *International Journal of Housing Markets and Analysis*.

Vlachos, M., Kollios, G., and Gunopulos, D. (2002). Discovering similar multidimensional trajectories. In *Proceedings 18th international conference on data engineering*, pages 673–684. IEEE.

Zhang, T. (2013). Clustering high-dimensional time series based on parallelism. *Journal of the American Statistical Association*, 108(502):577–588.

Zhang, T., Ramakrishnan, R., and Livny, M. (1996). Birch: an efficient data clustering method for very large databases. *ACM sigmod record*, 25(2):103–114.

A Data and sources

Aggregate Consumption (AC): Personal Consumption Expenditure per capita (seasonally adjusted, billions of chained 2000 dollars). Source: Bureau of Economic Analysis (BEA).

Business Fixed Investment (BFI): Real Private Nonresidential Fixed Investment (seasonally adjusted, billions of chained 2000 dollars), divided by the Civilian Noninstitutional Population (CNP16OV, source: Bureau of Labor Statistics). Source: BEA.

Residential Investment (RI): Real Private Residential Fixed Investment (seasonally adjusted, billions of chained 2000 dollars), divided by CNP16OV. Source: BEA.

Inflation (INF): Quarter on quarter log differences in the implicit price deflator for the nonfarm business sector, demeaned. Source: Bureau of Labor Statistics (BLS).

Nominal Short-term Interest Rate (IR): 3-month Treasury Bill Rate (Secondary Market Rate), expressed in quarterly units, demeaned. (Series ID: H15/RIFSGFSM03_NM). Source: Board of Governors of the Federal Reserve System.

House Price Index (HPI): Differences of the log transformation of Real Residential Property Price Index by U.S. State (Not Seasonally Adjusted, Series ID: [XX]STHPI in Saint Louis Fed Fred). Source: Bank for International Settlements (BIS).

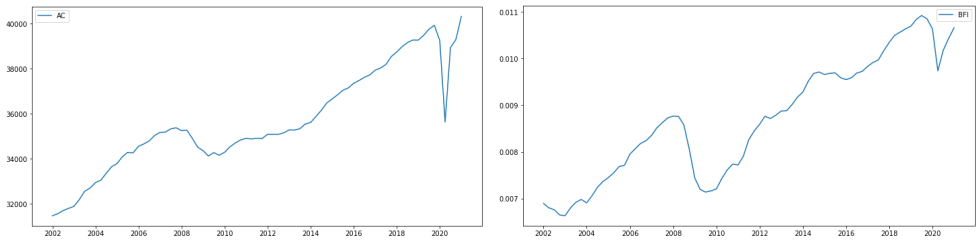
Hours in Consumption Sector (HC): Total Nonfarm Payrolls per U.S. State (Series ID: [XX]NA in Saint Louis Fed Fred) less all employees in the construction sector per U.S. State (Series ID: [XX]CONS), times Average Weekly Hours of Production Workers (Series ID: CES0500000007), divided by Civilian Noninstitutional Population per U.S. State (Source: BLS). Demeaned. Source: BLS.

Hours in Housing Sector (HH): All employees in the construction sector per U.S. State (Series ID: [XX]CONS in Saint Louis Fed Fred), times Average Weekly Hours of Construction Workers (Series ID: CES2000000007), divided by Civilian Noninstitutional Population per U.S. State (Source: BLS). Demeaned.

Wage Inflation in Consumption-good Sector (WIC): Quarterly changes in Average Hourly Earnings of Production/Nonsupervisory Workers on Private Nonfarm Payrolls, Total Private (Series ID: PRSCQ). Demeaned. Source: BLS.

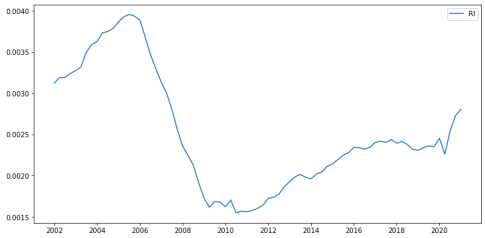
Wage Inflation in Housing Sector (WIH): Quarterly changes in Average Hourly Earnings of Production/Nonsupervisory Workers in the Construction Industry (Series ID: CES2000000008). Demeaned. Source: BLS.

B Tables and figures

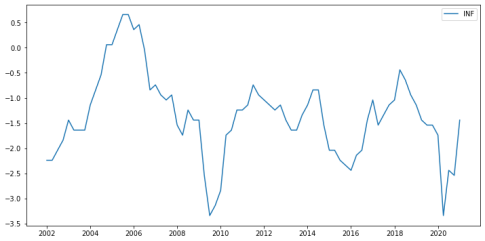


(a) Aggregate Consumption

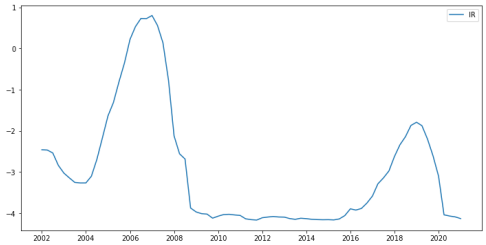
(b) Business Fixed Investment



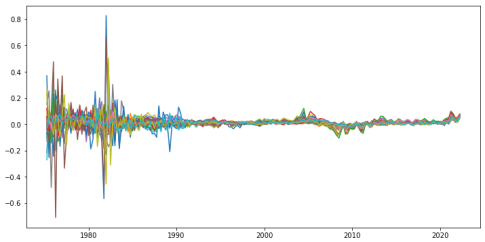
(c) Residential Investment



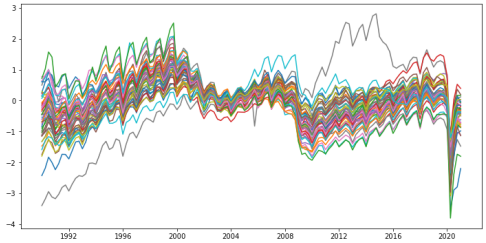
(d) Inflation



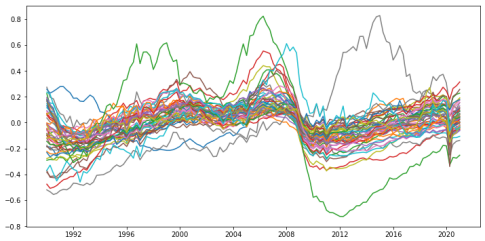
(e) Nominal Short-term Interest Rate



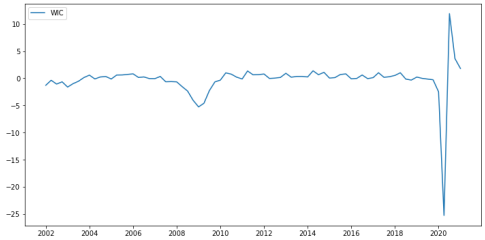
(f) House Price Index



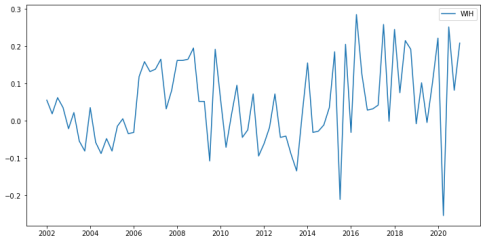
(g) Hours in Consumption Sector



(h) Hours in Housing Sector



(i) Wage Inflation in Cons.-good Sector



(j) Wage Inflation in Housing Sector

Figure B.1: All variables after transformations for equal time period

Table B.1: P-values of ADF test results

	HPI	AC	BFI	RI	INF	IR	HC	HH	WIC	WIH
Alabama	0.0918	0.3961	0.7708	0.1003	0.0511	0.0147	0.0007	0.5191	0.0000	0.4102
Alaska	0.3789	0.3961	0.7708	0.1003	0.0511	0.0147	0.0563	0.6743	0.0000	0.4102
Arizona	0.2060	0.3961	0.7708	0.1003	0.0511	0.0147	0.0013	0.3884	0.0000	0.4102
Arkansas	0.1988	0.3961	0.7708	0.1003	0.0511	0.0147	0.0886	0.2005	0.0000	0.4102
California	0.2399	0.3961	0.7708	0.1003	0.0511	0.0147	0.1920	0.2577	0.0000	0.4102
Colorado	0.4099	0.3961	0.7708	0.1003	0.0511	0.0147	0.0864	0.2849	0.0000	0.4102
Connecticut	0.4772	0.3961	0.7708	0.1003	0.0511	0.0147	0.1245	0.3930	0.0000	0.4102
Delaware	0.0816	0.3961	0.7708	0.1003	0.0511	0.0147	0.0018	0.6751	0.0000	0.4102
Florida	0.0492	0.3961	0.7708	0.1003	0.0511	0.0147	0.0778	0.0700	0.0000	0.4102
Georgia	0.3700	0.3961	0.7708	0.1003	0.0511	0.0147	0.1348	0.4483	0.0000	0.4102
Hawaii	0.0156	0.3961	0.7708	0.1003	0.0511	0.0147	0.2217	0.0970	0.0000	0.4102
Idaho	0.5138	0.3961	0.7708	0.1003	0.0511	0.0147	0.0405	0.1944	0.0000	0.4102
Illinois	0.2626	0.3961	0.7708	0.1003	0.0511	0.0147	0.0062	0.0910	0.0000	0.4102
Indiana	0.2165	0.3961	0.7708	0.1003	0.0511	0.0147	0.0557	0.3931	0.0000	0.4102
Iowa	0.9403	0.3961	0.7708	0.1003	0.0511	0.0147	0.1005	0.4334	0.0000	0.4102
Kansas	0.7361	0.3961	0.7708	0.1003	0.0511	0.0147	0.0146	0.3545	0.0000	0.4102
Kentucky	0.7385	0.3961	0.7708	0.1003	0.0511	0.0147	0.0938	0.3605	0.0000	0.4102
Louisiana	0.2158	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.1383	0.0000	0.4102
Maine	0.2234	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.5383	0.0000	0.4102
Maryland	0.0455	0.3961	0.7708	0.1003	0.0511	0.0147	0.1632	0.4404	0.0000	0.4102
Massachusetts	0.6690	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.6260	0.0000	0.4102
Michigan	0.5919	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.4204	0.0000	0.4102
Minnesota	0.3429	0.3961	0.7708	0.1003	0.0511	0.0147	0.0065	0.5817	0.0000	0.4102
Mississippi	0.1873	0.3961	0.7708	0.1003	0.0511	0.0147	0.1172	0.3043	0.0000	0.4102
Missouri	0.5156	0.3961	0.7708	0.1003	0.0511	0.0147	0.0004	0.4725	0.0000	0.4102
Montana	0.0897	0.3961	0.7708	0.1003	0.0511	0.0147	0.0001	0.0942	0.0000	0.4102
Nebraska	0.6806	0.3961	0.7708	0.1003	0.0511	0.0147	0.0767	0.2646	0.0000	0.4102
Nevada	0.2604	0.3961	0.7708	0.1003	0.0511	0.0147	0.0284	0.5829	0.0000	0.4102
New Hampshire	0.6508	0.3961	0.7708	0.1003	0.0511	0.0147	0.0241	0.4054	0.0000	0.4102
New Jersey	0.4352	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.5267	0.0000	0.4102
New Mexico	0.4469	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.3873	0.0000	0.4102
New York	0.0823	0.3961	0.7708	0.1003	0.0511	0.0147	0.1412	0.4051	0.0000	0.4102
North Carolina	0.4920	0.3961	0.7708	0.1003	0.0511	0.0147	0.0114	0.1550	0.0000	0.4102
North Dakota	0.5031	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.2179	0.0000	0.4102
Ohio	0.9441	0.3961	0.7708	0.1003	0.0511	0.0147	0.0884	0.4098	0.0000	0.4102
Oklahoma	0.6762	0.3961	0.7708	0.1003	0.0511	0.0147	0.0083	0.2358	0.0000	0.4102
Oregon	0.0394	0.3961	0.7708	0.1003	0.0511	0.0147	0.2278	0.1037	0.0000	0.4102
Pennsylvania	0.2686	0.3961	0.7708	0.1003	0.0511	0.0147	0.1084	0.1812	0.0000	0.4102
Rhode Island	0.4090	0.3961	0.7708	0.1003	0.0511	0.0147	0.3565	0.1859	0.0000	0.4102
South Carolina	0.3268	0.3961	0.7708	0.1003	0.0511	0.0147	0.0021	0.2843	0.0000	0.4102
South Dakota	0.5800	0.3961	0.7708	0.1003	0.0511	0.0147	0.0223	0.4699	0.0000	0.4102
Tennessee	0.5570	0.3961	0.7708	0.1003	0.0511	0.0147	0.1451	0.3562	0.0000	0.4102
Texas	0.1546	0.3961	0.7708	0.1003	0.0511	0.0147	0.0247	0.4167	0.0000	0.4102
Utah	0.0442	0.3961	0.7708	0.1003	0.0511	0.0147	0.0385	0.0503	0.0000	0.4102
Vermont	0.0827	0.3961	0.7708	0.1003	0.0511	0.0147	0.0259	0.5324	0.0000	0.4102
Virginia	0.0694	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.1648	0.0000	0.4102
Washington	0.1941	0.3961	0.7708	0.1003	0.0511	0.0147	0.1336	0.1299	0.0000	0.4102
West Virginia	0.6144	0.3961	0.7708	0.1003	0.0511	0.0147	0.0675	0.3862	0.0000	0.4102
Wisconsin	0.3542	0.3961	0.7708	0.1003	0.0511	0.0147	0.0000	0.0162	0.0000	0.4102
Wyoming	0.1137	0.3961	0.7708	0.1003	0.0511	0.0147	0.0163	0.1161	0.0000	0.4102

Table B.2: P-values of ADF test results for first difference data

	HPI	AC	BFI	RI	INF	IR	HC	HH	WIC	WIH
Alabama	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Alaska	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0026	0.0000	0.0000	0.0000
Arizona	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0001	0.0000	0.0000	0.0000
Arkansas	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0010	0.1237	0.0000	0.0000
California	0.0001	0.3147	0.0000	0.4165	0.0002	0.0022	0.0629	0.1971	0.0000	0.0000
Colorado	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0163	0.0038	0.0000	0.0000
Connecticut	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0031	0.0000	0.0000	0.0000
Delaware	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0001	0.0001	0.0000	0.0000
Florida	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0008	0.2809	0.0000	0.0000
Georgia	0.0776	0.3147	0.0000	0.4165	0.0002	0.0022	0.0002	0.0029	0.0000	0.0000
Hawaii	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.4546	0.0013	0.0000	0.0000
Idaho	0.0093	0.3147	0.0000	0.4165	0.0002	0.0022	0.0068	0.0000	0.0000	0.0000
Illinois	0.2504	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.1270	0.0000	0.0000
Indiana	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0082	0.0000	0.0000	0.0000
Iowa	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Kansas	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0014	0.0000	0.0000	0.0000
Kentucky	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0001	0.0000	0.0000	0.0000
Louisiana	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0013	0.0000	0.0000
Maine	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0002	0.0000	0.0000	0.0000
Maryland	0.1605	0.3147	0.0000	0.4165	0.0002	0.0022	0.0114	0.0000	0.0000	0.0000
Massachusetts	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Michigan	0.1036	0.3147	0.0000	0.4165	0.0002	0.0022	0.0004	0.0215	0.0000	0.0000
Minnesota	0.1808	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Mississippi	0.0302	0.3147	0.0000	0.4165	0.0002	0.0022	0.0002	0.0313	0.0000	0.0000
Missouri	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Montana	0.2812	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0010	0.0000	0.0000
Nebraska	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0001	0.0946	0.0000	0.0000
Nevada	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0005	0.0000	0.0000	0.0000
New Hampshire	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0030	0.1413	0.0000	0.0000
New Jersey	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
New Mexico	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0342	0.0000	0.0000	0.0000
New York	0.1929	0.3147	0.0000	0.4165	0.0002	0.0022	0.0094	0.0000	0.0000	0.0000
North Carolina	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0009	0.0672	0.0000	0.0000
North Dakota	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0491	0.0000	0.0000	0.0000
Ohio	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0048	0.0000	0.0000	0.0000
Oklahoma	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0027	0.0000	0.0000	0.0000
Oregon	0.1040	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0775	0.0000	0.0000
Pennsylvania	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0113	0.0105	0.0000	0.0000
Rhode Island	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0004	0.0000	0.0000	0.0000
South Carolina	0.2344	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.1690	0.0000	0.0000
South Dakota	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0019	0.0000	0.0000	0.0000
Tennessee	0.1730	0.3147	0.0000	0.4165	0.0002	0.0022	0.0065	0.1424	0.0000	0.0000
Texas	0.1790	0.3147	0.0000	0.4165	0.0002	0.0022	0.0012	0.0000	0.0000	0.0000
Utah	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0073	0.0212	0.0000	0.0000
Vermont	0.0002	0.3147	0.0000	0.4165	0.0002	0.0022	0.0041	0.0004	0.0000	0.0000
Virginia	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Washington	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0002	0.0000	0.0000	0.0000
West Virginia	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0017	0.1743	0.0000	0.0000
Wisconsin	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000
Wyoming	0.0000	0.3147	0.0000	0.4165	0.0002	0.0022	0.0000	0.0000	0.0000	0.0000

Table B.3: P-values of ADF test results for second difference data

	HPI	AC	BFI	RI	INF	IR	HC	HH	WIC	WIH
Alabama	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Alaska	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Arizona	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0008	0.0000	0.0000
Arkansas	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
California	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Colorado	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Connecticut	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Delaware	0.0007	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Florida	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0005	0.0000	0.0000
Georgia	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Hawaii	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Idaho	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Illinois	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Indiana	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Iowa	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Kansas	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Kentucky	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Louisiana	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Maine	0.0003	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Maryland	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Massachusetts	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Michigan	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Minnesota	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Mississippi	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0001	0.0000	0.0000	0.0000
Missouri	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Montana	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Nebraska	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Nevada	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
New Hampshire	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
New Jersey	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
New Mexico	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
New York	0.0010	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
North Carolina	0.0007	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
North Dakota	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Ohio	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Oklahoma	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Oregon	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Pennsylvania	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Rhode Island	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
South Carolina	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
South Dakota	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Tennessee	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Texas	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0001	0.0000	0.0000	0.0000
Utah	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Vermont	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Virginia	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Washington	0.0003	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
West Virginia	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000
Wisconsin	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0001	0.0000	0.0000
Wyoming	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000

Table B.4: P-values of Johansen maximum eigenvalue test results for second difference data

	HPI	AC	BFI	RI	INF	IR	HC	HH	WIC	WIH
Alabama	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Alaska	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Arizona	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Arkansas	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
California	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Colorado	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Connecticut	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Delaware	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Florida	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Georgia	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Hawaii	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Idaho	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Illinois	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Indiana	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Iowa	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Kansas	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Kentucky	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Louisiana	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Maine	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Maryland	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Massachusetts	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Michigan	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Minnesota	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Mississippi	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Missouri	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Montana	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Nebraska	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Nevada	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
New Hampshire	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
New Jersey	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
New Mexico	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
New York	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
North Carolina	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
North Dakota	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Ohio	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Oklahoma	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Oregon	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Pennsylvania	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Rhode Island	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
South Carolina	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
South Dakota	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Tennessee	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Texas	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Utah	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Vermont	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Virginia	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Washington	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
West Virginia	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Wisconsin	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001
Wyoming	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001	<0.001

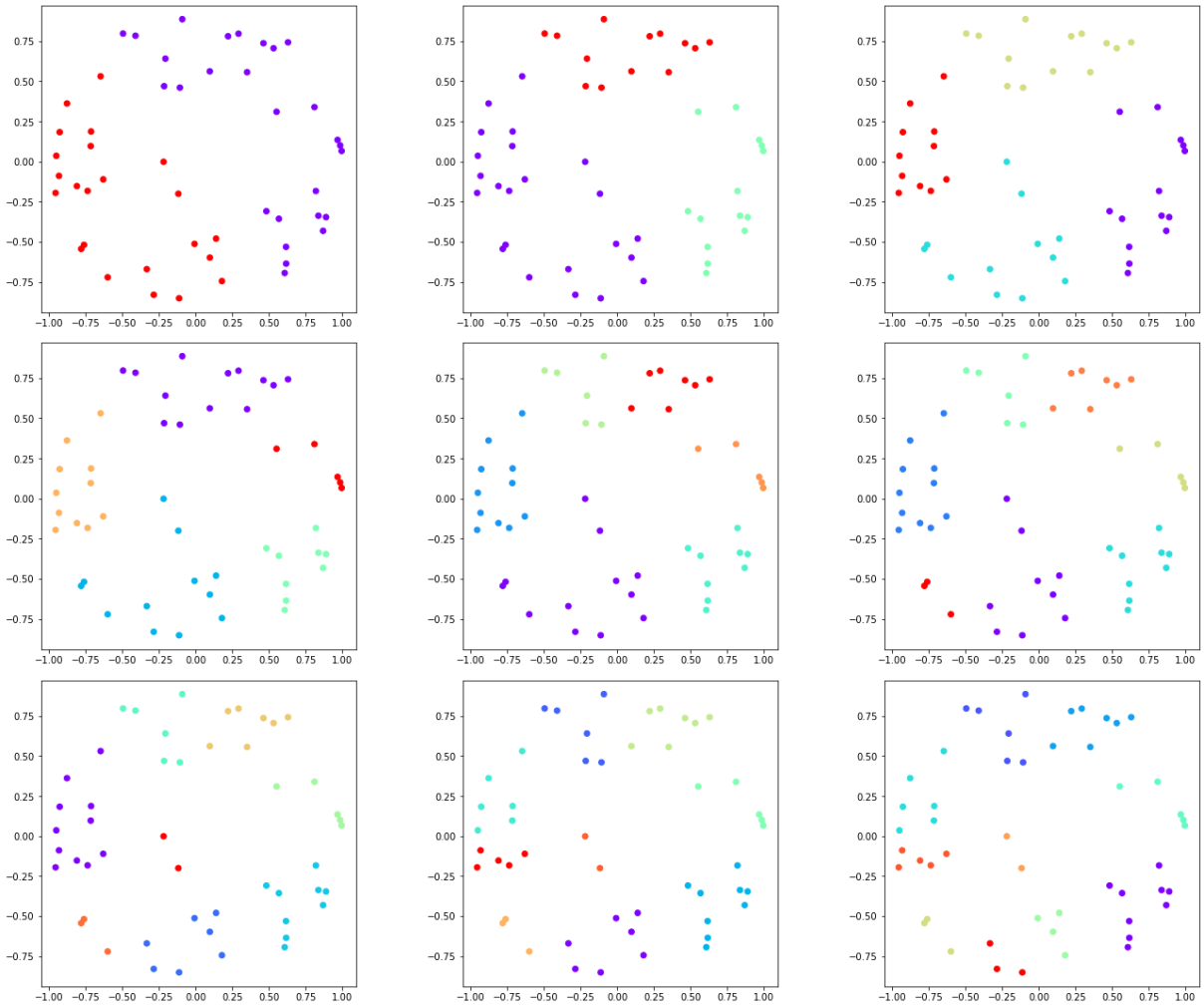


Figure B.2: 2-layer LSTM-autoencoder agglomerative PCA-clustering plots for 2 to 10 number of clusters

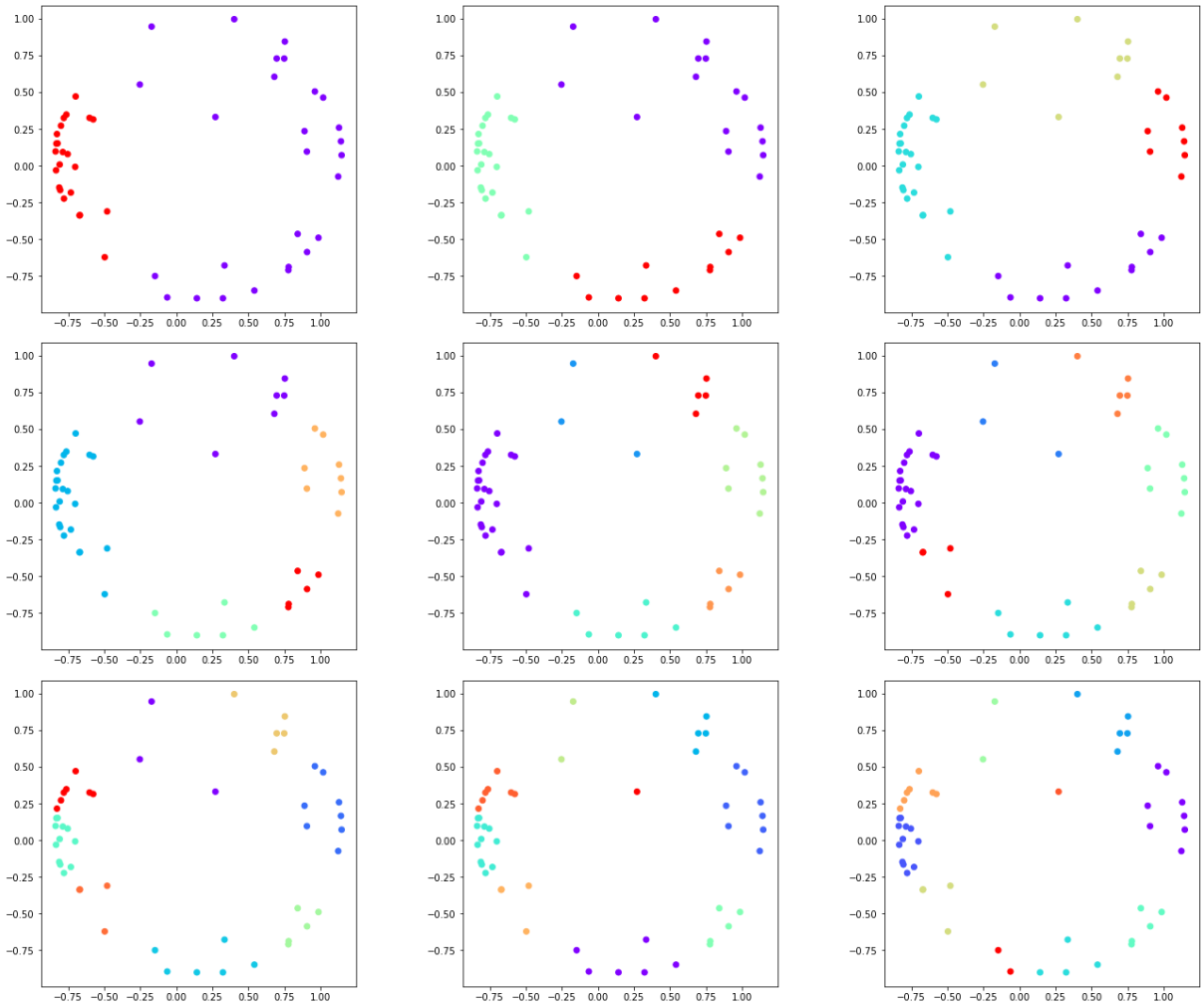
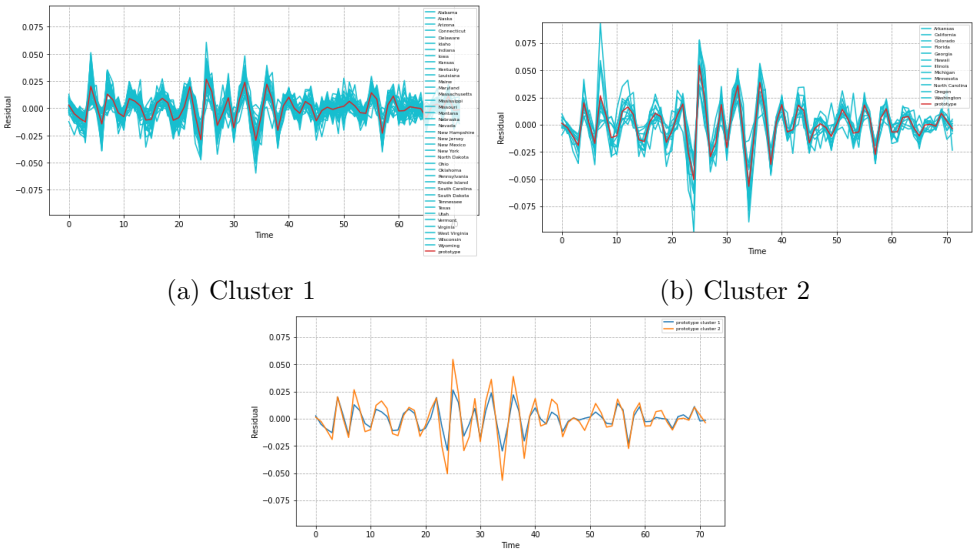


Figure B.3: 3-layer LSTM-autoencoder agglomerative clustering plots for 2 to 10 number of clusters

Table B.5: Cluster assignments

	LSTM(2)			LSTM(3)			
state	K(2)	K(3)	K(4)	K(5)	K(2)	K(3)	K(4)
Alabama	0	0	0	4	0	1	3
Alaska	0	0	0	0	0	1	0
Arizona	0	0	0	0	0	1	3
Arkansas	1	1	3	1	0	1	3
California	1	1	0	1	1	1	0
Colorado	1	1	1	2	1	2	0
Connecticut	0	0	3	2	0	1	3
Delaware	0	0	3	2	0	1	0
Florida	1	1	3	1	1	1	0
Georgia	1	1	0	2	1	2	1
Hawaii	1	1	2	4	1	0	2
Idaho	0	0	1	0	0	1	0
Illinois	1	1	3	1	1	2	0
Indiana	0	1	0	1	0	2	1
Iowa	0	0	0	2	0	2	1
Kansas	0	0	0	2	0	1	0
Kentucky	0	0	1	0	0	1	0
Louisiana	0	0	1	0	0	1	0
Maine	0	0	0	1	1	2	1
Maryland	0	0	3	2	0	2	0
Massachusetts	0	0	3	1	0	1	0
Michigan	1	2	0	1	1	2	1
Minnesota	1	2	0	1	1	2	1
Mississippi	0	0	0	2	0	1	0
Missouri	0	0	1	0	0	1	3
Montana	0	0	3	4	0	1	3
Nebraska	0	0	0	2	0	2	0
Nevada	0	0	2	4	0	1	0
New Hampshire	0	0	1	0	0	2	1
New Jersey	0	0	3	1	0	2	0
New Mexico	0	0	3	1	0	2	1
New York	0	0	0	2	0	1	0
North Carolina	1	2	1	1	1	0	2
North Dakota	0	0	1	1	1	0	2
Ohio	0	1	0	1	1	2	1
Oklahoma	0	0	1	0	0	1	3
Oregon	1	1	0	2	1	2	0
Pennsylvania	0	0	1	0	0	2	1
Rhode Island	0	1	3	1	1	2	1
South Carolina	0	1	1	4	0	2	0
South Dakota	0	0	0	0	0	1	3
Tennessee	0	0	1	0	0	1	0
Texas	0	0	1	0	0	1	0
Utah	0	0	0	2	1	1	0
Vermont	0	0	3	2	0	1	0
Virginia	0	0	0	2	0	2	1
Washington	1	1	0	2	1	2	0
West Virginia	0	0	0	2	1	2	1
Wisconsin	0	1	0	3	0	1	3
Wyoming	0	0	0	0	0	1	0

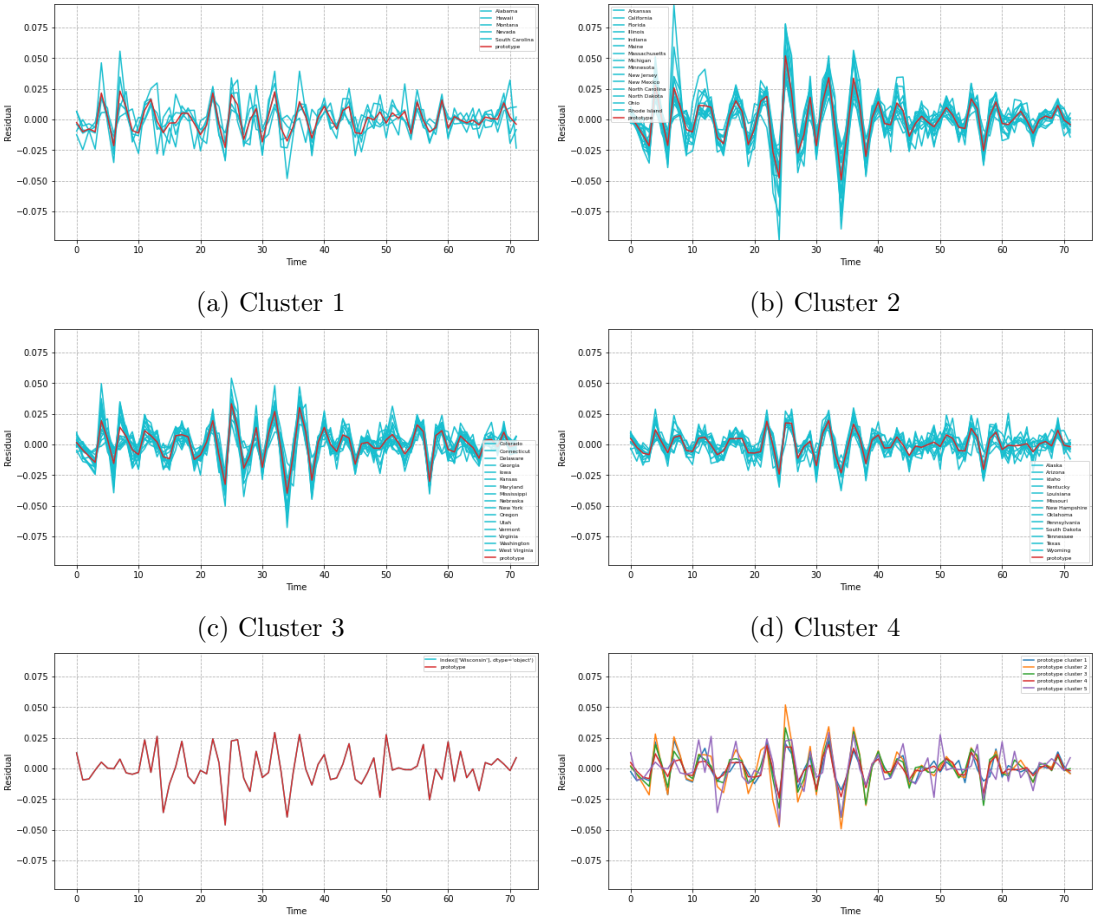


(a) Cluster 1

(b) Cluster 2

(c) Cluster prototypes

Figure B.4: Residual cluster plots for LSTM(2)-K(2)



(a) Cluster 1

(b) Cluster 2

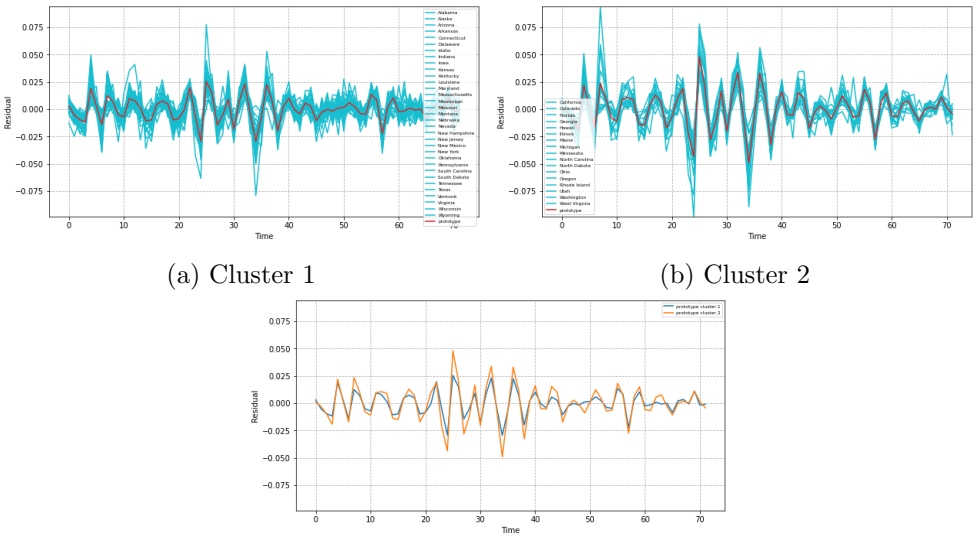
(c) Cluster 3

(d) Cluster 4

(e) Cluster 5

(f) Cluster prototypes

Figure B.5: Residual cluster plots for LSTM(2)-K(5)

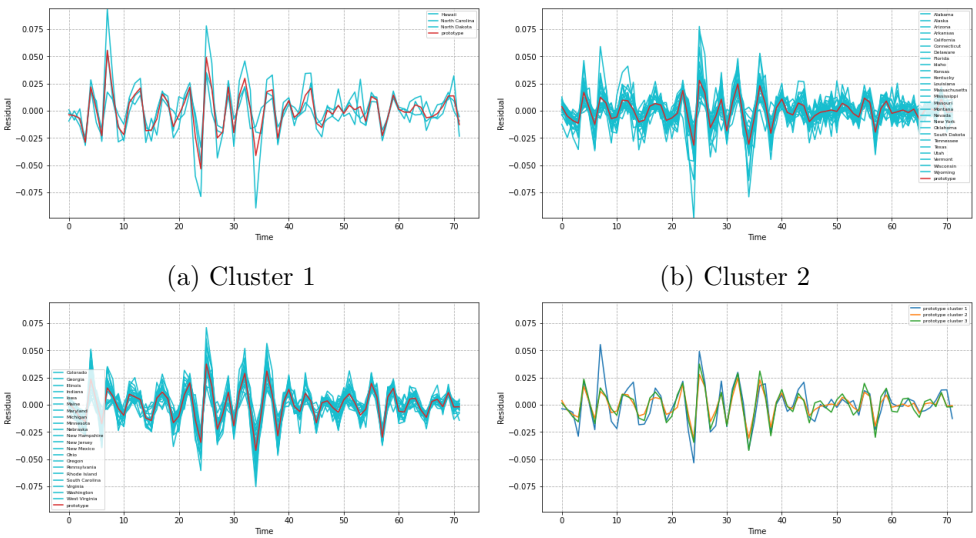


(a) Cluster 1

(b) Cluster 2

(c) Cluster prototypes

Figure B.6: Residual cluster plots for LSTM(3)-K(2)



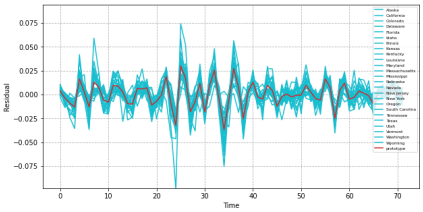
(a) Cluster 1

(b) Cluster 2

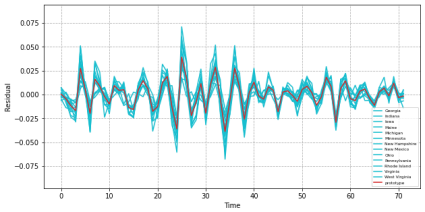
(c) Cluster 3

(d) Cluster prototypes

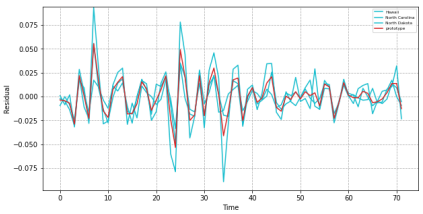
Figure B.7: Residual cluster plots for LSTM(3)-K(3)



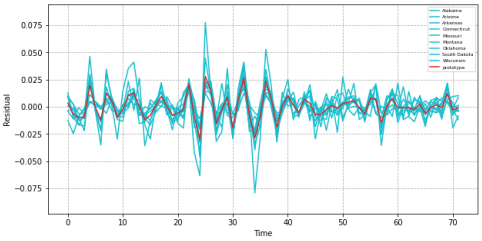
(a) Cluster 1



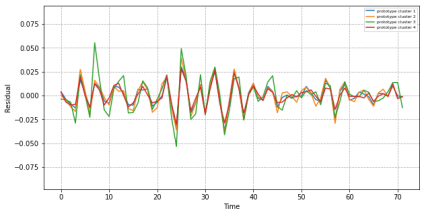
(b) Cluster 2



(c) Cluster 3

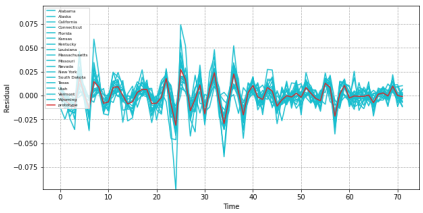


(d) Cluster 4

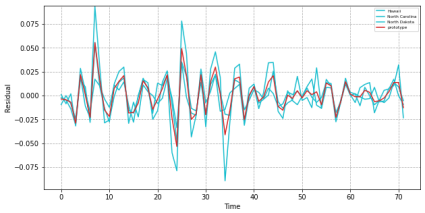


(e) Cluster prototypes

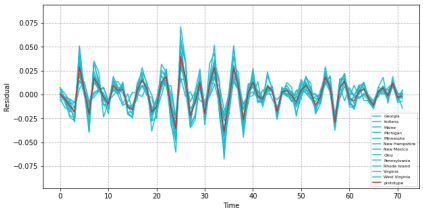
Figure B.8: Residual cluster plots for LSTM(3)-K(4)



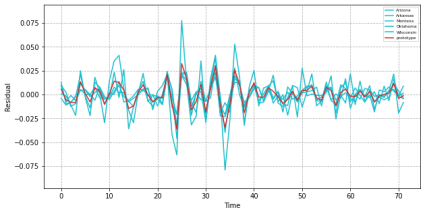
(a) Cluster 1



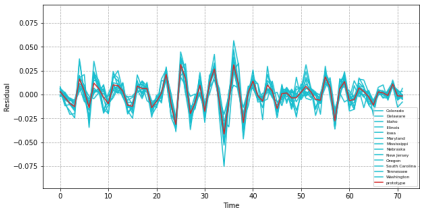
(b) Cluster 2



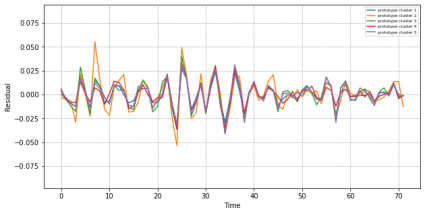
(c) Cluster 3



(d) Cluster 4

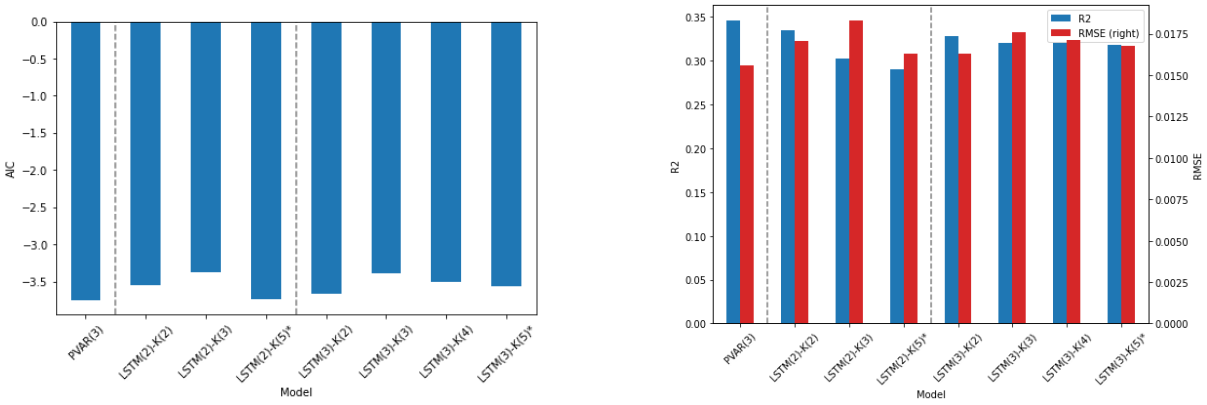


(e) Cluster 5



(f) Cluster prototypes

Figure B.9: Residual cluster plots for LSTM(3)-K(5)



(a) AIC

(b) R^2 and RMSE

Figure B.10: Bar plots of performance of LSTM(l)-K(k) models

Table B.6: Reported P-values of the panel VAR-Granger causality Wald test for each cluster panel VAR model

panel VAR-Granger causality Wald test
Ho: Excluded variable does not Granger-cause Equation variable
Ha: Excluded variable Granger-causes Equation variable
Results Table

model	cluster	ac	inf	ir	hc	wic	ALL
LSTM(2)-K(2)	1	0.000	0.000	0.000	0.000	0.089	0.000
	2	0.000	0.000	0.000	0.000	0.007	0.000
LSTM(2)-K(3)	1	0.000	0.000	0.000	0.000	0.001	0.000
	2	0.008	0.000	0.000	0.001	0.326	0.000
	3	0.002	0.000	0.000	0.114	0.545	0.000
LSTM(2)-K(5)	1	0.034	0.064	0.000	0.000	0.902	0.000
	2	0.000	0.000	0.000	0.007	0.046	0.000
	3	0.000	0.000	0.000	0.000	0.003	0.000
	4	0.045	0.000	0.000	0.000	0.041	0.000
	5	0.234	0.378	0.000	0.009	0.090	0.000
LSTM(3)-K(2)	1	0.000	0.000	0.000	0.000	0.077	0.000
	2	0.000	0.000	0.000	0.000	0.005	0.000
LSTM(3)-K(3)	1	0.000	0.000	0.000	0.000	0.003	0.000
	2	0.000	0.000	0.000	0.000	0.077	0.000
	3	0.014	0.439	0.000	0.024	0.155	0.000
LSTM(3)-K(4)	1	0.000	0.000	0.000	0.000	0.002	0.000
	2	0.000	0.000	0.000	0.000	0.002	0.000
	3	0.026	0.084	0.000	0.150	0.060	0.000
	4	0.010	0.000	0.000	0.005	0.447	0.000
LSTM(3)-K(5)	1	0.002	0.000	0.000	0.000	0.337	0.000
	2	0.026	0.084	0.000	0.150	0.060	0.000
	3	0.000	0.000	0.000	0.000	0.004	0.000
	4	0.156	0.000	0.000	0.281	0.455	0.000
	5	0.000	0.000	0.000	0.000	0.001	0.000

Table B.7: Performance statistics for the tested panel VAR models before and after clustering

Model	Cluster	AIC	R2	RMSE
PVAR(3)		-3.76097	0.346066	0.01556
LSTM(2)-K(2)	1	-3.95258	0.332044	0.014116
	2	-3.15422	0.338281	0.02001
	average	-3.5534	0.335163	0.017063
LSTM(2)-K(3)	1	-3.98269	0.340713	0.013623
	2	-3.46837	0.30528	0.018345
	3	-2.68757	0.26259	0.022963
	average	-3.37954	0.302861	0.01831
LSTM(2)-K(5)	1	-3.367791	0.347381	0.016688
	2	-3.263475	0.327856	0.018948
	3	-3.835004	0.273018	0.015319
	4	-4.391741	0.287061	0.011650
	5	-3.804858	0.214587	0.018757
	average	-3.732574	0.289981	0.016272
LSTM(3)-K(2)	1	-3.966194	0.324261	0.014019
	2	-3.362602	0.332729	0.018534
	average	-3.664398	0.328495	0.016277
LSTM(3)-K(3)	1	-2.50131	0.355121	0.022011
	2	-3.94352	0.344422	0.014578
	3	-3.71492	0.26121	0.01616
	average	-3.38658	0.320251	0.017583
LSTM(3)-K(4)	1	-3.88829	0.330646	0.014691
	2	-3.69994	0.256497	0.016339
	3	-2.50131	0.355121	0.022011
	4	-3.91536	0.338943	0.015487
	average	-3.50123	0.320302	0.017132
LSTM(3)-K(5)	1	-3.88898	0.352909	0.014689
	2	-2.50131	0.355121	0.022011
	3	-3.67690	0.252716	0.016638
	4	-3.81130	0.313015	0.015922
	5	-3.94252	0.313580	0.014632
	average	-3.56420	0.317468	0.016778

C Programming files

Most of the code for this paper has been written in Python. The file "datacompiler.py" merges and transforms all data as mentioned in Appendix A into a single data frame. The file "UShouseprices.py" is the file used for execution of the main clustering algorithms. The file "myfunctions.py" is a supporting file that stores some function blocks which are called in the main file. The panel VAR model has been written in Stata. The file "panelvar.do" contains the code for executing the model, calculating residuals and performing the Granger test. Input data was manually assigned in the data editor. These include the second differenced data set, as well as this data split up according to cluster assignments resulting from the Python files.