Deep Quantile Regression

Ilias Chronopoulos[*] Aristeidis Raftapostolos[†] George Kapetanios[‡]

August 29, 2022

Abstract

In this paper we propose a *deep quantile* estimator, using neural networks and their universal approximation property to examine a non-linear association between the conditional quantiles of a dependent variable and predictors. The proposed methodology is versatile and allows both the use of different penalty functions, as well as high dimensional covariates. We present a Monte Carlo exercise where we examine the finite sample properties of the proposed estimator and show that our approach delivers good finite sample performance. We use the *deep quantile* estimator to forecast Value-at-Risk and find significant gains over linear quantile regression alternatives and other models, supported by various testing schemes. We consider also an alternative architecture that allows the use of mixed frequency data in neural networks. The paper also contributes to the interpretability of neural networks output by making comparisons between the commonly used SHAP values and an alternative method based on partial derivatives.

Keywords: Quantile regression, machine learning, neural networks, value-at-risk, forecasting.
JEL Classification: C45, C58, G17.

[*] University of Essex. Email: ilias.chronopoulos@essex.ac.uk.
[†] King's College London and University of Strathclyde. Email: aristeidis.1.raftapostolos@kcl.ac.uk.
[‡] King's College London. Email: george.kapetanios@kcl.ac.uk.

1 Introduction

Since the seminal work of Koenker and Bassett Jr (1978) and Koenker and Hallock (2001), quantile regression has grown in popularity and has found applications in several disciplines both in academia and industry, see e.g. Chernozhukov and Umantsev (2001), Adams, Adrian, Boyarchenko, and Giannone (2021) and Koenker, Chernozhukov, He, and Peng (2017). They generalize ordinary sample quantiles to the regression setting, that give more extensive information on the conditional distribution of a dependent variable, given the covariates, relative to the classical regression setting; i.e. estimation of the conditional mean. This extension can be of great importance under extreme events, where the conditional distribution of variables such as asset returns tends to exhibit skewness, or under the presence of outliers and/or asymmetries, see e.g. Baur and Schulze (2005).

An assumption made in the early literature, was the linear association between the conditional quantile of the target variable and predictors. This was predominately an assumption that allowed for streamlined computation and theoretical inference, but was clearly restrictive. A more recent strand of the literature, relaxed the linearity assumption and considered non-parametric estimators for the conditional quantile, that is based on different methods, see e.g. Belloni, Chernozhukov, Chetverikov, and Fernández-Val (2019) and references therein. Recent advances in Machine Learning (ML) literature, which is the focus of this paper, show how modelling frameworks such as neural networks can be used to estimate general, non-linear and potentially highly complicated associations.

Specifically, a large number of studies have shown that *feed-forward* neural networks can approximate arbitrarily well any continuous function of several real variables, see e.g. Hornik (1991), Hornik, Stinchcombe, and White (1989), Galant and White (1992) and Park and Sandberg (1991). Recent work by Liang and Srikant (2016) and Yarotsky (2017), extends this result for *feed-forward* neural networks with multiple layers, provided sufficiently many hidden neurons and layers are available. Notice that, besides neural networks, other non-parametric approaches, e.g. splines, wavelets, the Fourier basis, as well as simple polynomial approximations, do have the universal approximation property, based on the Stone-Weierstrass theorem.

There is considerable empirical work identifying non-linearities and asymmetries in financial variables, see e.g. Gu, Kelly, and Xiu (2020a), Gu, Kelly, and Xiu (2020b), He and Krishnamurthy (2013) and Pohl, Schmedders, and Wilms (2018), where they illustrate that ML offers richer functional form specifications that can capture potential non-linearities between dependent and independent variables. Some examples include Gu, Kelly, and Xiu (2020b) in which, they evaluate the forecast accuracy of machine learning methods in measuring equity risk premia, and find that neural networks give substantial forecasting gains in asset pricing compared to linear models, and Bucci (2020), where a recurrent neural network is proposed, that approximates realised volatility well and outperforms other classic non-linear estimators in forecasting. In a similar fashion, Smalter Hall and Cook (2017) use several neural network architectures to predict unemployment in the US and find that neural networks outperform forecasts from a linear benchmark model at short horizons. In addition, Gu, Kelly, and Xiu (2020a) propose the use of a conditional Autoencoder[1], and illustrate its superior performance relative to linear unsupervised learning methods.

Before we discuss the contributions of this paper, we provide a succinct summary of the current machine learning literature on non-linear quantile and Value-at-Risk (*VaR*) estimation, but we note that the majority of this work, was not available during the writing of this paper. Keilbar and Wang (2021) use neural networks to estimate a non-linear conditional *VaR* model introduced by Tobias and Brunnermeier (2016) and find that, it gives significant gains in modelling systemic risk. In addition, Tambwekar, Maiya, Dhavala, and Saha (2021) estimate a non-linear binary quantile regression and develop confidence scores to assess the reliability of prediction. Padilla, Tansey, and Chen (2020) examine the performance of a quantile neural network using Rectified Linear Unit (ReLU) as activation function. They derive a theoretical upper bound for the mean squared error of a ReLU network and show that their non-linear quantile estimator has strong performance of ReLU neural networks for quantile regression across a broad range of function classes and error distributions. Chen, Liu, Ma, and Zhang (2020) propose a unified non-linear framework, based on *feed-forward* neural networks, that allows the estimation of treatment effects, for which they establish consistency and asymptotic normality. Their framework includes the quantile estimator and allows for high-dimensional covariates. ML based estimators for quantiles have been proposed in other fields, see e.g. Meinshausen (2006), where quantile random forests are introduced, and Zhang, Quan, and Srinivasan (2018) that propose a quantile neural network estimator.

In this paper, we contribute to the expanding literature on the use of ML in Finance and propose a novel *deep quantile* estimator that can capture non-linear associations between asset returns and predictors and that also allows for high dimensional data. We further consider an alternative architecture that allows the use of mixed frequency data. We also contribute towards the explainable machine learning literature, by proposing the use of partial derivatives as a means to "peeking" inside the black box.

We first explore the small sample properties of the proposed estimator via Monte Carlo experiments, which show that the estimator delivers good finite sample performance. Then we examine the performance of the proposed estimator, in the context of one of the most widely examined problems in finance: that of measuring and subsequently forecasting the risk of a portfolio adequately, via *VaR* modelling. *VaR* is a popular model that was first introduced in the late 80s and since then, has become a standard toolkit in measuring market risk. It measures how much value a portfolio can lose within a given time period with some small probability, τ. *VaR* and quantiles are related in the following manner, let $\boldsymbol{r} = (r_1, \ldots, r_T)'$ denote the returns of a portfolio, then, the τ^{th} *VaR* is equivalent of computing the negative value of the τ^{th} quantile of $\boldsymbol{r}$, $-q_{\tau}(\boldsymbol{r})$.

In this paper, we argue, following the non-parametric literature, that the linear relationship between *VaR* and predictors can be restrictive and propose a quantile neural network estimator that allows a non-linear association between covariates and *VaR*. This method appears particularly suitable for developing sound predictions for the past stock return losses in the US over the sample period from September 1985 up to August 2020, the importance of which has been brought to the forefront by the recent COVID-19 pandemic. Specifically, our aim is to forecast ten-day ahead *VaR* produced from daily *VaR* forecasts. We use daily frequency returns in a fixed forecasting framework that is outlined below. Under this forecasting framework, mixed frequency models become relevant benchmarks to the non-linear quantile estimator, see e.g. Ghysels, Plazzi, and Valkanov (2016). Hence, we also include a linear MIxed DAta Sampling (MIDAS) model as a competitor and also a non-linear MIDAS model, which is an extension to the *deep quantile* estimator. Further, we consider ten-day compounded *VaR* forecasts that exhibit similar patterns, which we relegate to the Online Appendix.

We are not the first to use ML methods for *VaR* forecasting, see e.g. Du, Wang, and Xu (2019), where they propose a recurrent neural network, as a novel forecasting methodology for the *VaR* model and exhibit an improved forecast performance relative to traditional methods. To the best of our knowledge though, there has been no application that uses a neural network quantile estimator in finance for forecasting *VaR*. Note that in this paper we also consider a large set of neural networks that also allow for mixed frequency estimation.

Our empirical analysis shows that the proposed *deep quantile* estimator outperforms the linear , MIDAS and other non-parametric quantile models, in forecasting *VaR*. We assess the forecasting accuracy between models based on two statistical tests. The first is the Diebold and Mariano (1995) test with the Harvey, Leybourne, and Newbold (1997) adjustment, and the second is the Giacomini and White (2006) test. Results from both tests suggest that our neural network estimator has higher accuracy in forecasting *VaR*. We use the linear quantile method as a benchmark to assess whether our proposed estimator has predictive gains or not. This measure illustrate gains up to 74% relative to the linear one, for the *deep quantile* estimator and up to 76% for the non-linear MIDAS model. Further, we use the quantile score test that provides further evidence in favour of our neural network estimator.

We further examine whether our proposed estimator nests forecasts produced from the linear and other non-parametric models, using the encompassing test of Giacomini and Komunjer (2005). Overall, we find that forecasts from the *deep quantile* estimator encompass forecasts from competing models more times than vice versa. There are some cases where the test is inconclusive, suggesting that a forecast combination from a different pair of models would provide a better result, which is in line with the result of Bates and Granger (1969).

While ML methods show a great capacity at both approximating highly complicated non-linear functions and forecasting, they are routinely criticized as they lack interpretability and are considered a "black box"; in the sense that they do not offer simple summaries of relationships in the data. Recently though, there has been a number of studies that try to make ML output interpretable, see e.g. Athey and Imbens (2017), Wager and Athey (2018), Belloni, Chernozhukov, and Hansen (2014), Joseph (2019). In this paper we also try to understand in a semi-structural fashion, which variables impact the forecasting performance of the *deep quantile* estimator more. To this end, we first use Shapley Additive Explanation Values (SHAP) as proposed by Lundberg and Lee (2017) and further developed in Joseph (2019), that have started to become a standard tool for interpretability in ML methods. Further we use partial derivatives, as a means of investigating the marginal contribution/influence of each variable to the output. We compare the partial derivatives and SHAP values over time, and our results can be summarised as follows. First, partial derivatives overall are more stable than SHAP values, and are able to produce interpretable results, at a fraction of the computational time of SHAP. Second, the partial derivatives of the *deep quantile* estimator fluctuate around the estimate of the conditional linear quantile and i) exhibit time variation and ii) can capture stressful events in the U.S. economy for instance the COVID-19 pandemic and the 2008 financial crisis.

The remainder of the paper is organised as follows. Section 1 introduces the *deep quantile* estimator. Section 2 contains the Monte Carlo exercise. Section 3 presents our empirical application. Section 4 presents the semi-structural analysis. Conclusions are set out in Section 5. We relegate to the Online Appendix the specifications of the competing models, empirical results from one-step ahead *VaR* forecast, ten-day compounded *VaR* forecasts and results from the quantile score test and predictive gains.

2 Theory

In this section we start by summarising the underlying theory of a quantile regression as outlined by Koenker and Bassett Jr (1978) and Koenker (2005) and argue that the linear relationship of the conditional quantile between a dependent variable given the covariates, can be restrictive. We illustrate how some fundamental results on the universal approximation property of neural networks can be used to approximate a non-linear relationship instead, and propose a *deep quantile* estimator. We conclude with a discussion on how different penalisation schemes can be used and further how hyper-parameters can be selected via Cross Validation (CV).

2.1 Linear Quantile Regression

The standard goal in econometric analysis is to infer a relationship between a dependent variable and one or more covariates. Let $\{y_t, x_t\}_{t=1}^{T}$ be a random sample from the following linear regression model

$$y_t = \boldsymbol{x}_t'\boldsymbol{\beta} + u_t, \tag{1}$$

where y_t is the dependent variable at time t, $\boldsymbol{\beta} = \left(\beta_1, \ldots, \beta_p\right)'$ is a vector of unobserved slope parameters, $\boldsymbol{x}_t = \left(x_{t1}, \ldots, x_{tp}\right)'$ is a vector of known covariates, and u_t is the random error of the regression which satisfies $E(u_t|\boldsymbol{x}_t) = 0$. Standard regression analysis tries to come up with an estimate of the conditional mean of y_t given $\boldsymbol{x}_t$, that minimises the expected squared error loss:

$$\widehat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \frac{1}{T} \sum_{t=1}^{T} \left(y_t - \boldsymbol{x}_t'\boldsymbol{\beta}\right)^2. \tag{2}$$

This can be restrictive though, when i) non-linearities and outliers exist and ii) since it provides just an aspect of the conditional distribution of y_t, given $\boldsymbol{x}_t$ by construction. These potential limitations led to the development of quantile regression. In their seminal work, Koenker and Bassett Jr (1978) generalise ordinary sample quantiles to the regression setting, that give more complete information on the conditional distribution of y_t given $\boldsymbol{x}_t$, for which we now provide a succinct description.

The quantile regression model can be defined as

$$Q_y\left(\tau|\boldsymbol{x}_t\right) = \boldsymbol{x}_t'\boldsymbol{\beta}\left(\tau\right), \quad \tau \in (0,1), \tag{3}$$

such that y_t satisfies the quantile constraint $Pr[y_t \leq \boldsymbol{x}_t'\boldsymbol{\beta}\left(\tau\right)|\boldsymbol{x}_t] = \tau$, where $\boldsymbol{\beta}\left(\tau\right)$ are regression coefficients that depend on τ. Quantile regression tries to come up with an estimate for the τ^{th} conditional quantile, $\widehat{Q}_y\left(\tau, \boldsymbol{x}_t\right) := \widehat{\boldsymbol{\beta}}(\tau)$, by minimizing the following function

$$\widehat{\boldsymbol{\beta}}\left(\tau\right) = \arg\min_{\boldsymbol{\beta}} \frac{1}{T} \sum_{t=1}^{T} \rho_\tau\left(y_t - \boldsymbol{x}_t'\boldsymbol{\beta}\left(\tau\right)\right), \tag{4}$$

where $\rho_\tau\left(\cdot\right)$ is the quantile loss function defined as

$$\rho_\tau\left(u_t\right) = \begin{cases} \tau u_t\left(\tau\right), & \text{if } u_t\left(\tau\right) \geq 0 \\ \left(1-\tau\right) u_t\left(\tau\right), & \text{if } u_t\left(\tau\right) < 0 \end{cases}$$

and $u_t\left(\tau\right) = y_t - \boldsymbol{x}_t'\boldsymbol{\beta}\left(\tau\right)$. The quantile estimator in eq. 4, provides i) much richer information on the whole conditional distribution of y_t as function of the $\boldsymbol{x}_t$, and ii) more robust estimates under the presence of outliers and non-linearities, when compared to the ordinary least squares estimator.

Notice that the linear association assumption, $Q_y\left(\tau|\boldsymbol{x}_t\right) = \boldsymbol{x}_t'\boldsymbol{\beta}\left(\tau\right)$, can be generally restrictive. Instead, we consider the case of the following non-linear association,

$$Q_y\left(\tau|\boldsymbol{x}_t\right) = h_\tau\left(\boldsymbol{x}_t\right),$$

where $h_\tau(\cdot)$ is some unknown, (potentially highly) non-linear function. In this paper we propose an estimation strategy to approximate $h_\tau(\boldsymbol{x}_t)$ with neural networks using their universal approximation property. Specifically, we assume that there exists a neural network with a function $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$, to be defined below, that can approximate $h_\tau(\boldsymbol{x}_t)$ well. Before we illustrate how our methodology is implemented, we provide a discussion on how neural networks can approximate $h_\tau(\boldsymbol{x}_t)$.

2.2 Neural Networks

In this paper, we limit our attention to *feed-forward* neural networks, to approximate $h_\tau(\boldsymbol{x}_t)$. This architecture consists of an input layer of covariates, the hidden layer(s) where non-linear transformations of the covariates occur, and the output layer that gives the final prediction. Each hidden layer has several interconnected neurons relating it to both the previous and next ones. Specifically, information flows from one layer to the other, via neurons only in one direction, and the connections correspond to weights. Optimising a loss function *w.r.t* these weights makes neural networks capable of learning.

Throughout our exposition, L denotes the total number of hidden layers, a measure for the depth of a neural network, and $J^{(l)}$ denotes the total number of neurons at layer l, a measure for its width. We start by presenting a general definition of a deep (multi-layer) *feed-forward* neural network. Let $\sigma_l(\cdot)$, $l = 0, \dots, L$ be the activation function used at the l^{th} layer, that is applied element-wise and induces non-linearity. We use the ReLU activation function, $\sigma_l(\cdot) = \max(\cdot, 0)$, for $l = 1, \dots, L-1$ and a linear one for the output layer, $l = L$. We denote by $g^{(l)}$ the output of the l^{th} layer which is a vector of length equal to the number of the $J^{(l)}$ neurons in that layer, such that $g^{(0)} = \boldsymbol{x}_t$. Then, the overall structure of the network is equal to:

$$G_\tau(\boldsymbol{x}_t, \boldsymbol{w}) = g^{(L)}\left(g^{(L-1)}\left(\dots\left(g^{(1)}(\cdot)\right)\right)\right), \tag{5}$$

where

$$g^{(l)}(\boldsymbol{x}_t) = \sigma_l\left(\boldsymbol{W}^{(l-1)} g^{(l-1)} + \mathbf{b}^{(l)}\right), \qquad l = 1, \dots, L, \tag{6}$$

$\boldsymbol{W}^{(l)}$ is a $J^{(l)} \times J^{(l-1)}$ matrix of weights, $\boldsymbol{b}^{(l)}$ is a $J^{(l)} \times 1$ vector of biases giving an overall vector $\boldsymbol{w} = \left(vec(\boldsymbol{W}^{(0)})', \dots, vec(\boldsymbol{W}^{(L)})', \boldsymbol{b}^{(1)'}, \dots, \boldsymbol{b}^{(L)'}\right)'$ of trainable parameters of dimensions $J^{(l)}(1 + J^{(l-1)})$ total number of parameters in each hidden layer l, $J^{(0)} = p$ and $J^{(L)} = 1$.

According to various universal approximation theorems (see e.g. the theoretical results in Hornik (1991), Hornik, Stinchcombe, and White (1989), Galant and White (1992), Kapetanios and Blake (2010), Liang and Srikant (2016) and Yarotsky (2017)), $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$ can approximate arbitrarily well $h_\tau(\boldsymbol{x}_t)$, such that, for any $\epsilon > 0$,

$$\sup_t |G_\tau(\boldsymbol{x}_t, \boldsymbol{w}) - h_\tau(\boldsymbol{x}_t)| < \epsilon. \tag{7}$$

In this sense, the above (ϵ)-approximation can be seen as a sieve type non-parametric estimation bound, where ϵ can become arbitrarily small by increasing the complexity of $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$.

The increase in complexity can occur, either by letting $L \to \infty$, which stands for *deep learning*, or by letting $J^{(l)} \to \infty$. While asymptotically, both ways deliver the same results (see e.g. Farrell, Liang, and Misra (2021) and references therein), the approximation error has been shown to decline exponentially with L, see e.g. Babii, Chen, Ghysels, and Kumar (2020) but only polynomially with $J^{(l)}$, providing some evidence for the prevalent use of deep learning. Notice that there also exists an alternative approximation theory for sparse deep learning, see e.g. the work of Schmidt-Hieber (2020). As an illustration, in the Online Appendix we depict a simple *feed-forward* neural network with two inputs, two hidden layers, a total of five neurons and one output layer.

2.3 Non-linear Quantile Regression

We assume that the conditional quantile follows a non-linear relationship $Q_y(\tau|\boldsymbol{x}_t) = h_\tau(\boldsymbol{x}_t)$ and there exists a function $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$, that can (ϵ)-approximate $h_\tau(\boldsymbol{x}_t)$, see the bound in eq. 7. Using this assumption, we can formally define the conditional quantile function as the following approximation

$$Q_y(\tau|\boldsymbol{x}_t) = G_\tau(\boldsymbol{x}_t, \boldsymbol{w}) + O(\varepsilon),$$

where $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$ is the unknown non-linear function we want to estimate in order to approximate $h_\tau(\boldsymbol{x}_t)$. We obtain the deep neural network conditional quantile estimate from the solution of the following minimization problem:

$$Q_y(\tau|\boldsymbol{x}_t) = \arg\min_{\boldsymbol{w}} \frac{1}{T} \sum_{t=1}^{T} \rho_\tau(y_t - G_\tau(\boldsymbol{x}_t, \boldsymbol{w})), \tag{8}$$

where $\boldsymbol{w} = (\text{vec}(\boldsymbol{W}^{(0)})', \ldots, \text{vec}(\boldsymbol{W}^{(L)})', \boldsymbol{b}^{(1)'}, \ldots, \boldsymbol{b}^{(L)'})'$ contains all model parameters, and $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$ denotes the overall non-linear mapping, described in eq. 5 and 6. Notice that the choice of $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$ will govern whether the model is parametric or non-parametric. If the number of neurons and layers is small, then the model is parametric, if the above number becomes large, then the model becomes non-parametric, since the number of estimated parameters increases with the sample size, similar to sieve non-parametric approximations.

To allow the use of mixed frequency data, we can make the following changes to the structure of the network $G_\tau(\boldsymbol{x}_t, \boldsymbol{w})$:

In the input layer, we implement frequency alignment on each input variable $\boldsymbol{x}_t$ according to the corresponding maximum lag order K. Thus, each high frequency predictor $\boldsymbol{x}_t$ is transformed into a low frequency vector $x_t^\star = B(L_\varphi; \boldsymbol{\vartheta})\boldsymbol{x}_t$,

$$B(L_\varphi; \boldsymbol{\vartheta}) = \sum_{k=0}^{K} B(k; \boldsymbol{\vartheta}) L_\varphi^k, \quad B\left(k; \boldsymbol{\vartheta}\right) = \frac{\exp(\vartheta_1 k + \vartheta_2 k^2)}{\sum_{k=1}^{K} \exp(\vartheta_1 k + \vartheta_2 k^2)}, \tag{9}$$

where $B\left(k; \boldsymbol{\vartheta}\right)$ is the normalised Almon polynomial, L_φ^k is a lag operator such that $L_\varphi^k x_t^\varphi = x_{t-k}^\varphi$; the lag coefficients in $B(k; \boldsymbol{\vartheta})$ of the corresponding lag operator L^k are parameterised as a function of a small dimensional vector of parameters $\boldsymbol{\vartheta}$. We use this weight function on the frequency alignment vector to reduce the number of parameters and ensure a parsimonious specification. As a consequence, the low frequency variable $\boldsymbol{x}_t^\star$ which has the same frequency as the output y_t is obtained. The rest of the architecture of the *deep MIDAS* follows the architecture of the *deep quantile* estimator, but instead of using $\boldsymbol{x}_t$ in eq. 6, we use $\boldsymbol{x}_t^\star$.

2.4 Regularized Non-Linear Quantile Regression

Neural networks have a great capacity to estimate non-linear relationships from the data, but this comes at a cost, since they are prone to over-fitting. This can lead to a severe drop in their forecasting performance, especially in small samples. There is a variety of commonly used techniques in ML, see e.g. Gu, Kelly, and Xiu (2020a) for a good summary, that can be used to ease this impact, originally coming from the high-dimensional statistical literature. The reader is also referred to Goodfellow, Bengio, and Courville (2016) for an excellent summary of different topics about the implementation of neural networks, including regularization.

2.4.1 Regularization

A common solution to this caveat is regularization, where a penalty term is imposed on the weights of the neural network and is appended in the loss function. Regularization, generally improves the out-of-sample performance of the network by decreasing the in-sample noise from over-parameterization, utilising the bias-variance trade-off. Further, another benefit of regularization is that it provides computational gains in the optimization algorithm. The penalised loss function, for a given quantile τ, can be written as:

$$L(G_\tau(\boldsymbol{x}_t, \boldsymbol{w}), y_t) = \frac{1}{T} \sum_{t=1}^{T} \rho_\tau(y_t - \widehat{G}_\tau(\boldsymbol{x}_t, \boldsymbol{w})) + \phi(\boldsymbol{w}), \tag{10}$$

where the penalty term is

$$\phi(\boldsymbol{w}) = \begin{cases} \lambda \left\| \boldsymbol{w} \right\|_1, & \text{Lasso} \\ \lambda \left\| \boldsymbol{w} \right\|_2^2, & \text{Ridge} \\ \lambda(1-\alpha)\|\boldsymbol{w}\|_1 + \lambda\alpha\|\boldsymbol{w}\|_2^2, & \text{Elastic Net} \\ 0, & \text{otherwise} \end{cases},$$

and λ and α are tuning parameters, for which we discuss their selection below. Generally, there is a plethora of loss functions, and the choice among them, depends mainly on the task at hand. In this paper we use the quantile loss function. The different penalisation schemes on $\phi(\boldsymbol{w})$ work as follows: *deep LASSO* or l_1-norm penalisation, is a regularization method that shrinks uniformly all the weights to zero, and some at exactly zero. The latter is referred to as the variable selection property of the *deep LASSO*. *Deep Ridge* works in a similar manner to the *deep LASSO*, by shrinking the weights, uniformly to zero, but not at exactly zero. Finally, the *deep Elnet*[2] is a combination of *deep LASSO* and *deep Ridge*, that has been shown to retain good features from both methods, see e.g. Zou and Hastie (2005).

2.4.2 Cross Validation

We use Cross Validation (CV) to calibrate all the different (hyper)-parameters outlined above, and aim to maximise the out-of-sample (forecasting) performance of the network.

Our CV scheme consists of choices on: i) the total number of layers (L) and neurons (J), ii) the learning rate (γ) for the Stochastic Gradient Decent (SGD), iii) the batch size, dropout rate and the level of regularization. Regarding the choice on the activation function, we use ReLU for the hidden layers and a linear function for the output layer. Overall, our aim is to build a neural network that has the best pseudo-out-of-sample (POOS) performance. To achieve this, we need to evaluate the model, select the optimal parameters and hyper-parameters and test its POOS behaviour. It is clear that tuning all these different architectures, parameters and hyper-parameters increases the computational cost a lot.

For this reason we tune the learning rate for the optimiser, γ, from five discrete values in the interval $[0.01, 0.001]$. For the width and depth of the neural network we tune the hyper-parameters from the following grids $[1, 5, 10]$ and $[10, 30, 50]$, respectively. The batch size is selected via the following grid $[10, 20]$.[3] Furthermore, we tune the regularization parameter, λ, from five discrete values in the interval $[0.01, 0.001]$, both for *deep LASSO* and *deep Ridge*, and for the case of the elastic net we choose α from a grid $[0.1, 0.5, 0.9]$. We also use dropout regularization, where the dropout probability is up to 20%, see e.g. Gu, Kelly, and Xiu (2020b).

For the non-linear MIDAS , we also cross validate ϑ_1 from eight discrete values in the interval $[-1, 0.5]$ and for ϑ_2, we use six discrete values in $[-0.5, 0.5]$.

We split the whole sample into three distinct subsamples, the training, validation and test subsamples. These subsamples are consequential to maintain the time series structure of the data. The training subsample consists of the first 60% of the sample, the validation is the next 20% of sample and the test is final 20% of sample. First, we use the training sample to estimate (i.e. train) the network parameters. Then, the second subsample or validation is used to tune hyper-parameters by constructing the fitted/forecasted values given the parameters from the training sample. We proceed with the calculation of the quantile loss function as in eq. 10 and evaluate the models' POOS performance on this subset. We repeat the same process Δ number of times, where Δ is the number of all possible combinations of points across quantiles. We store the quantile loss values and select the parameters and hyper-parameters that minimise the quantile loss based on the POOS forecasts. In the validation step we wish to find the optimal parameters and hyper-parameters that capture complex non-linear relations and produce reliable POOS forecasts.

Finally, in the test subsample we use the optimal parameters and hyper-parameters from the validation step and evaluate the out-of-sample performance of the network.

2.4.3 Optimisation

The estimation of neural networks is generally a computational cumbersome optimization problem due to non-linearities and non-convexities. The most commonly used solution utilises stochastic gradient descent (SGD) to train a neural network. SGD uses a batch of a specific size, that is, a small subset of the data at each iteration of the optimization to evaluate the gradient, to alleviate the computation hurdle. The step of the derivative at each epoch is controlled by the learning rate, γ. We use the adaptive moment estimation algorithm (ADAM) proposed by Kingma and Ba (2014)[4], which is a more efficient version of SGD.

3 Monte Carlo

3.1 Setup

In this section we present Monte Carlo (MC) experiments, in order to study the finite sample performance of the *deep quantile* estimator proposed in Section 2, for the different penalisation schemes. We generate artificial data $\{y_t\}$ using a single predictor $\{x_t\}$, according to the following model

$$y_t = h_\tau(x_t) + u_t, \tag{11}$$

where u_t is the realisation of a random variable u distributed as, $u_t \sim iidN(-\sigma\Phi^{-1}(\tau), \sigma^2)$, $\sigma = 0.1$ and Φ^{-1} is the quantile function of the standard normal distribution. $h_\tau(\cdot)$ is the general non-linear function that we wish to approximate via the *deep quantile* estimator.

All the experiments are based on the following values: $\tau \in (1\%, 2.5\%, 5\%, 10\%, 20\%)$, $T \in (100, 300, 500, 1000, 2000, 5000)$ and the number of MC replications is 100. We consider the following four *data generating mechanisms* (DGM) to assess the finite sample properties of the *deep quantile* estimator:

Case I: We consider the case of a $N(0,1)$ simulated single predictor that is generated as

$$y_t = h_\tau(x_t) + u_t, \quad h_\tau(x_t) = \sin(2\pi x_t), \quad x_t \sim N(0,1).$$

This is the simplest design in our Monte Carlo experiments. We use this simple case to showcase that linear methods, as expected, cannot produce reasonable performance under a sigmoid type of a non-linear function $h_\tau(\cdot)$.

Case II: We consider an AR(1) simulated single predictor as follows

$$y_t = h_\tau(x_t) + u_t, \quad h_\tau(x_t) = \sin(2\pi x_t),$$

where x_t is simulated as

$$x_t = 0.8x_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim N(0,1).$$

In this design we increase the complexity by introducing a correlated predictor.

Case III: We consider the case of a single predictor generated via a GARCH(1,1) model

$$y_t = h_\tau(x_t) + u_t, \quad h_\tau(x_t) = \sin(2\pi x_t),$$

where x_t is simulated as:

$$x_t = \sigma_t \varepsilon_t, \quad \sigma_t^2 = 1 + 0.7x_{t-1}^2 + 0.2\sigma_{t-1}^2.$$

In this design, we wish to examine, how the proposed estimator fares, when the regressor is conditionally heteroskedastic, following a $GARCH(1,1)$ model. A $GARCH$ type of assumption on the distribution of asset returns is one commonly used in the literature.

Case IV: We consider the case of a single predictor that is generated as follows:

$$y_t = h_\tau(x_t) + u_t, \quad h_\tau(x_t) = G_\tau(\boldsymbol{x}_t, \boldsymbol{w}), \quad x_t \sim N(0,1).$$

In this case we simulate $h_\tau(x_t)$ to reflect a function composition, commonly used in neural networks. We simulate it with 3 hidden layers and a specific number of neurons, such as

$$G_\tau(\boldsymbol{x}_t, \boldsymbol{w}) = \left(\boldsymbol{W}^{(3)} \left(\sin\left(\boldsymbol{W}^{(2)} \left(\sin\left(\boldsymbol{W}^{(1)} \left(\sin\left(\boldsymbol{W}^{(0)} x_t' + \boldsymbol{b}^{(1)}\right)\right) + \boldsymbol{b}^{(2)}\right)\right) + \boldsymbol{b}^{(3)}\right)\right)\right)',$$

where $\boldsymbol{w} = (\text{vec}(\boldsymbol{W}^{(0)})', \ldots, \text{vec}(\boldsymbol{W}^{(3)})', \boldsymbol{b}^{(1)'}, \ldots, \boldsymbol{b}^{(3)'})'$, $\boldsymbol{W}^{(0)}$ is 50×1, $\boldsymbol{W}^{(1)}$ is 10×50, $\boldsymbol{W}^{(2)}$ is 8×10 and $\boldsymbol{W}^{(3)}$ is 1×8. Further, we simulate the weights, $\boldsymbol{w}$, so that, every entry $w_{i,j}$ is simulated as, $w_{i,j} = \delta_{i,j} 1(\delta_{i,j} > 0.5)$, where $\delta_{i,j} \sim U(0,1)$, allowing for some sparsity.

Across all cases, we estimate $h_\tau(x_t)$ using our proposed estimator with different penalisation schemes. Let $\widehat{h}_{\tau, pen} = \widehat{G}_{\tau, pen}(x_t, \boldsymbol{w})$ denotes the estimate, where *pen* corresponds to no regularization, *deep LASSO*, *deep Ridge* and *deep Elnet*. We use the following metrics in order to evaluate the small sample properties, of our *deep quantile* estimator across $R = 100$, MC replications: i) the average mean squared error of the true residuals, $AMSE_{u_t} = \frac{1}{R}\frac{1}{T}\sum_{i=1}^{R}\left(\sum_{t=1}^{T} u_t^2\right)_i$, ii) the average mean squared error of the estimated residuals, $AMSE_{\widehat{u}_t, pen} = \frac{1}{R}\frac{1}{T}\sum_{i=1}^{R}\left(\sum_{t=1}^{T}(y_t - \widehat{y}_{t,pen})^2\right)_i$ and finally, iii) the average absolute bias $ABIAS_{\widehat{h}_{\tau,pen}} = \frac{1}{R}\frac{1}{T}\sum_{i=1}^{R}\left(\sum_{t=1}^{T}|(h_\tau(x_t) - \widehat{G}_{\tau,pen}(x_t, \boldsymbol{w}))|\right)_i$. We report results only for $AMSE_{\widehat{u}_t, pen}$ below, since results for the alternative metrics exhibit similar patterns and are available upon request.

Figures 1 – 4 about here

3.2 Results

We present our Monte Carlo results for Cases I – IV in Figures 1 – 4 respectively. In Figure 1, we can see that the linear quantile estimator, under a non-linear setup doesn't work as expected and the MSE remains constant as the sample size increases. Next we present the asymptotic properties for our proposed estimator across different penalization schemes, namely *deep quantile*, *deep LASSO*, *deep Ridge* and *deep Elastic Net*, and find that the proposed non-linear estimators have good finite sample properties.

When $\tau = 1\%$ it appear that our estimator works well for sample sizes larger than $T = 300$, but in comparison with the linear one it generally works better. In Case II our non-linear estimators depict fine finite sample properties and their performance is better than the linear one. In this case the non-regularized estimator performs better than the regularized ones. Next, similar behaviour appears in Case III. In Case IV, where we allow for some sparsity in the weights, we find, as expected, that the linear quantile regression estimator, does not work under non-linearity, while the non-linear one works as expected.

Overall, our Monte Carlo results suggest that the *deep quantile* estimator has good finite sample properties, and can approximate non-linear functions. We further find, as expected, that the linear quantile regression estimator, does not work under non-linearity. Finally, we find evidence in favour of the penalisation schemes proposed in Section 2. Specifically, the penalised *deep quantile* estimators also have good finite sample properties, and in some cases, perform better that the non-regularized one; a finding in favour of weight regularization.

4 Empirical Setup

In this section we outline our empirical application setup, where we use the proposed *deep quantile* estimator to forecast *VaR*. We examine the predictive ability of the proposed estimator and other non-parametric models, relative to the linear one, using the quantile encompassing test of Giacomini and Komunjer (2005). We further examine the predictive performance of the different methods by testing their forecasting accuracy, using the Diebold and Mariano (1995), Giacomini and White (2006) and quantile score tests.

4.1 Deep Quantile *VaR* forecasting

The data used in our empirical application consist of around 36 years of daily prices on the S&P500 index (source: Bloomberg), from September 1985 to August 2020 (T = 9,053 observations). We use daily log returns, defined as $r_t = \log(P_t / P_{t-1})$ for our forecasting analysis. We use four different classes of *VaR* models and produce forecasts for $\tau = (1\%, 5\%, 10\%)$ empirical conditional quantiles, using the *deep quantile* estimator.

The first *VaR* specification we consider is the GARCH(1,1) model that has been proposed by Bollerslev (1986), in which $\sigma^2_{1,t} = \omega_0 + \omega_1 \sigma^2_{1,t-1} + \omega_2 r^2_{t-1}$, see eq. 12. The second *VaR* specification we consider, is RiskMetrics, proposed by J.P. Morgan (1996), which assumes $\sigma^2_{2,t} = \lambda \sigma^2_{2,t-1} + (1-\lambda) r^2_{t-1}$, where for daily returns, $\lambda = 0.94$, see eq. 13.

The last two specifications we consider follow the *Conditional Autoregressive Value-at-Risk* model (CAViaR), proposed by Engle and Manganelli (2004), where a specific quantile is analysed, rather than the whole distribution. Specifically, the CAViaR model corrects the past $VaR_{j,t-1}$ estimates in the following way: it increases $VaR_{j,t}$ when $VaR_{j,t-1}$ is above the τ^{th} quantile, while, when the $VaR_{j,t-1}$ is less than the τ^{th} quantile, it reduces $VaR_{j,t}$. Thus, the third *VaR* we examine is the Symmetric absolute value (SV) that responds symmetrically to past returns, see eq. 14 and lastly, we consider the Asymmetric slope value (ASV) as it offers a different response to positive and negative returns, see eq. 15. For ease of exposition, we refer to the above specification as $VaR_{1,t}, \dots, VaR_{4,t}$, respectively. Below we summarise their specifications:

$$VaR_{1,t} = \beta_0 + \beta_1 \sigma_{1,t} \tag{12}$$

$$VaR_{2,t} = \beta_0 + \beta_1 \sigma_{2,t} \tag{13}$$

$$VaR_{3,t} = \beta_0 + \beta_1 VaR_{3,t-1} + \beta_2 |r_{t-1}| \tag{14}$$

$$VaR_{4,t} = \beta_0 + \beta_1 VaR_{4,t-1} + \beta_2 r^{+}_{t-1} - \beta_3 r^{-}_{t-1}, \tag{15}$$

where β_i, $i = 0, \dots, 3$ are parameters to be estimated. We use these specifications following Giacomini and Komunjer (2005). Under the mixed frequency setup, we consider the following equation

$$VaR_{i,t}^{(\text{MIDAS})} = B(L_{\varphi}; \boldsymbol{\vartheta}) VaR_{i,t}, \tag{16}$$

where $B(L_{\varphi}; \boldsymbol{\vartheta})$ is defined in eq. 9 , i= $1, \dots, 4$ and $\boldsymbol{\vartheta}$ are parameters to be estimated. For a more detailed summary of MIDAS we refer the reader to Ghysels, Santa-Clara, and Valkanov (2004). As discussed in Section 2, the linear association between *VaR* and the covariates can be restrictive. Instead we assume that the relationship between the response variable, *VaR*, and the covariates has an unknown non-linear form for a given τ, that we wish to approximate with our proposed *deep quantile* estimator as

$$VaR_{1,t} = G_{\tau}\left(\sigma_{1,t}, \boldsymbol{w}\right) \tag{17}$$

$$VaR_{2,t} = G_{\tau}\left(\sigma_{2,t}, \boldsymbol{w}\right) \tag{18}$$

$$VaR_{3,t} = G_{\tau}\left(VaR_{3,t-1}, |r_{t-1}|, \boldsymbol{w}\right) \tag{19}$$

$$VaR_{4,t} = G_{\tau}\left(VaR_{4,t-1}, r_{t-1}^{+}, r_{t-1}^{-}, \boldsymbol{w}\right), \tag{20}$$

where $\text{VaR}_{j,t}$, $j = 1, \dots, 4$ is indexed at (day) $t = 1, \dots, T$. The dimension p of covariates that we use in our analysis depends on the specification chosen for *VaR*. Specifically, if $j = 1, 2$ then $p = 1$, if $j = 3$, $p = 2$ and finally if $j = 4$ then $p = 3$.

In the Online Appendix, we briefly delineate the model specifications for the quantile B-splines, quantile polynomial and quantile MIDAS estimators.

4.2 Forecasting Exercise Design

This section presents our forecasting exercise design. First we split our sample in three distinct parts; the training sample, which is used for the estimation of the weights, the validation sample which is used for tuning the hyperparamenters of the models and the test sample which is used for the evaluation of different models. We use a $60\%, 20\%, 20\%$ split[5], which corresponds to $5,053$ observations in the training sample, $2,000$ in the validation and $2,000$ in the test sample.

This specific split is used because we follow Giacomini and Komunjer (2005) and want the power of the Conditional Quantile Forecast Encompassing (CQFE) test to be comparable with her exercise. We use the CV scheme described in Section 2 and tune the width and depth of the neural network, the batch size, the learning rate, the dropout rate and the regularization of hyper-parameters. Generally, a forecasting exercise is performed either via a recursive or rolling window, see e.g. Ghysels, Plazzi, Valkanov, Rubia, and Dossani (2019), yet in either setting to produce all one step ahead forecasts for the last 2,000 observations and to tune the hyper-parameters can be computationally challenging. Instead, we follow

Giacomini and Komunjer (2005) and perform a fixed forecast window exercise, in which we estimate our models once.

For our forecasting design we use a fixed forecast window exercise and predict the ten-day-ahead *VaR* as:

$$\widehat{VaR}_{1,t+10}|\mathcal{F}_t = G_\tau(\sigma_{1,t}, \boldsymbol{w}^*), \tag{21}$$

where $\mathcal{F}_t$ denotes the information set up to time t, $\boldsymbol{w}^*$ denotes the optimal weights obtained from the CV. Eq. 21 illustrates how forecasts for the first *VaR* specification were obtained via the *deep quantile* estimator. In a similar manner forecasts can be obtained for other *VaR* specifications and alternative models, using eq. 12 – 20.

We evaluate the forecasting performance of *VaR* models with the proposed *deep quantile* estimator as in Section 2. Further, we consider ten-day compounded *VaR* forecasts, which we relegate to the Online Appendix.

4.3 Forecast Evaluation

In this section we discuss the various tests we have considered, in order to evaluate the predictive ability of the *deep quantile* estimator and present the testing results.

4.3.1 Diebold Mariano Test

We perform a quantitative forecast comparison across different methods and test their statistical significance. To do so, we calculate the *Root Mean Squared Forecast error* (RMSFE) for each method and perform the Diebold and Mariano (1995) (DM) test, with the Harvey, Leybourne, and Newbold (1997) adjustment to gauge the statistical significance of the forecasts. With the DM test, we assess the forecast accuracy of the *deep quantile* estimator relative to the benchmark linear quantile regression model. In this exercise we set τ equal to 1%, 5% and 10%.

In general, RMSFE is used to measure the accuracy of point estimates and is defined as

$$\text{RMSFE} = \sqrt{\frac{\sum_{t=1}^{T}(y_{t+h} - \widehat{G}_\tau\left(\boldsymbol{x}_{t+h}, \boldsymbol{w}\right))^2}{T}},$$

where h denotes the forecasting horizon and $\widehat{G}_\tau\left(\boldsymbol{x}_{t+h}, \boldsymbol{w}\right)$ is the solution to the eq. 8 after selecting the optimal $\boldsymbol{w}$ via CV at the τ^{th} quantile. Results from the DM test are reported in Table 1, where asterisks denote the statistical significance of rejecting the null hypothesis of the test at 1%, 5% and 10% level of significance, for all quantiles and models we consider. These results suggest that forecasts produced from the non-linear estimator outperform, for the majority of cases, forecasts obtained from the linear and non-parametric quantile regression estimators.

4.3.2 Giacomini White Test

In a similar manner and to complement the DM test, we follow Carriero, Kapetanios, and Marcellino (2009) and further calculate the Giacomini and White (2006) test of equal forecasting accuracy, that can handle forecasts based on both nested and non-nested models, regardless of the estimation procedures used for the derivation of the forecasts, including our proposed *deep quantile* estimator. Table 1 illustrates the results for Giacomini and White (2006) test, where daggers denote the statistical significance of rejecting the null hypothesis of the test at 1%, 5% and 10% level of significance, for all quantiles and different models we consider. Similarly to the DM forecasting accuracy test, the Giacomini and White (2006) test is again significant at 1% in most cases, with the following exceptions.

Quantile polynomial regression forecasts are only significant at the 10% level of significance for SV model. In quantile splines, forecasts for the GARCH specification at $\tau = 5\%$ and RM at $\tau = 10\%$ are not significant. Forecasts from the linear MIDAS, under the GARCH specification, at $\tau = 1\%$ are insignificant and under the ASV specification, at $\tau = 5\%$, are significant at the 5% significance level. Results for the ASV with *Deep Ridge* estimator at $\tau = 1\%$ are significant only for the Giacomini and White (2006) test. For the ASV *deep MIDAS Ridge* estimator and at $\tau = 1\%$, the forecasts are significant only based on the DM test. Forecasts from *deep Elnet* model under SV specification and at $\tau = 5\%$ are significant at 5% level of significance. Finally, forecasts from *deep Elnet* under SV specification and $\tau = 5\%$ are significant at the 5% level of significance.

Overall, results from both the DM and Giacomini and White (2006) tests suggest that the non-linear estimators outperform, for the majority of times, competing linear and non-parametric estimators in *VaR* forecasting.

Table 1 about here

4.3.3 Conditional Quantile Forecast Encompassing (CQFE)

We present the implementation of the CQFE test as proposed by Giacomini and Komunjer (2005) and the Generalized Method of Moments (GMM) estimation as proposed by Hansen (1982). Let $\widehat{q}_{1,t}$ be a vector of the τ^{th} quantile forecasts produced from model 1 and $\widehat{q}_{2,t}$ be the competing forecasts produced from model 2. The basic principle of CQFE is to test whether $\widehat{q}_{1,t}$ conditionally encompasses $\widehat{q}_{2,t}$. Encompassing occurs when the second set of forecasts fails to add new information to the first set of quantile forecasts (or vice versa) in which case the first (second) quantile forecast is said to encompass the second (first).

The aim of the CQFE test is to test the null hypothesis, that $\widehat{q}_{1,t}$ performs better that any linear combination of $\widehat{q}_{1,t}$ and $\widehat{q}_{2,t}$. Under the null hypothesis, it holds

$$E_t\left(\rho_\tau\left(y_{t+1} - \widehat{q}_{1,t}\right)\right) \leq E_t\left(\rho_\tau\left(y_{t+1} - \theta_0 - \theta_1\widehat{q}_{1,t} - \theta_2\widehat{q}_{2,t}\right)\right), \tag{22}$$

that is satisfied if and only if the weights (θ_1, θ_2) are equal to $(1, 0)$. The objective function of the GMM is:

$$J_T = g_T(\boldsymbol{\theta})' \boldsymbol{W}_T \, g_T(\boldsymbol{\theta}).$$

The optimal weights are computed as:

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} g_T(\boldsymbol{\theta})' \boldsymbol{W}_T \, g_T(\boldsymbol{\theta}), \quad g_T(\boldsymbol{\theta}) = \frac{\sum_{t=1}^{T} \left(\tau - \mathbb{1}_\tau\{y_{t+1} - \boldsymbol{\theta}' \boldsymbol{q}_t < 0\}\right) \boldsymbol{z}_T}{T},$$

where $\boldsymbol{W}_T$ is a positive definite matrix, $g_T(\boldsymbol{\theta})$ is the sample moment condition, $\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2)'$ is a set of weights, $\boldsymbol{\theta}^\star = (\theta_0^\star, \theta_1^\star, \theta_2^\star)'$ denotes the optimal weights, $\widehat{\boldsymbol{q}}_t = (1, \widehat{q}_{1,t}, \widehat{q}_{2,t})'$ is a vector with the forecasted values based on the pairwise models 1, and 2 in the CQFE test, m denotes the out-of-sample size and $\boldsymbol{z}_T$ is a vector of instruments. Hansen (1982) showed that by setting $\boldsymbol{W}_T = \boldsymbol{S}_T^{-1}$ i.e the inverse of an asymptotic covariance matrix, is optimal as it estimates $\boldsymbol{\theta}^\star$ with as small as possible asymptotic variance. $\boldsymbol{S}$ is also known as the spectral density matrix of $\boldsymbol{g}_T$. We follow Newey and West (1987) and use a heteroskedasticity robust estimate $\widehat{\boldsymbol{S}}_T$, of $\boldsymbol{S}$ defined as:

$$\widehat{\boldsymbol{S}}_T = \widehat{\boldsymbol{S}}_0 + \sum_{j=1}^{m} \left(1 - \frac{j}{m+1}\right) \left(\widehat{\boldsymbol{S}}_j + \widehat{\boldsymbol{S}}_j'\right), \quad \text{where} \quad \widehat{\boldsymbol{S}}_j = \frac{1}{T} \sum_{t=j+1}^{T} g_t\left(\widehat{\boldsymbol{\theta}}\right) g_{t-j}\left(\widehat{\boldsymbol{\theta}}\right).$$

$\widehat{S}_0$ is the estimated spectral density matrix evaluated at frequency zero. The GMM estimation is performed recursively, i.e. i) minimize J_T using an identity weighting matrix to get $\boldsymbol{\theta}^\star$, which gives $\boldsymbol{W}_T$ via $\widehat{\boldsymbol{S}}_T$ and ii) minimize J_T using $\boldsymbol{W}_T = \widehat{\boldsymbol{S}}_T^{-1}$ from step i).

Consequently, we consider two separate test $H_{10} : (\theta_1^\star, \theta_2^\star) = (1, 0)$ versus $H_{1a} : (\theta_1^\star, \theta_2^\star) \neq (1, 0)$ and $H_{20} : (\theta_1^\star, \theta_2^\star) = (0, 1)$ versus $H_{2a} : (\theta_1^\star, \theta_2^\star) \neq (0, 1)$, which correspond to testing whether forecast $\widehat{q}_{1,t}$ encompasses $\widehat{q}_{2,t}$ or $\widehat{q}_{2,t}$ encompasses $\widehat{q}_{1,t}$. Then the CQFE statistics are defined as:

$$\begin{aligned}
\text{ENC}_1 &= T\left((\theta_1^\star, \theta_2^\star) - (1, 0)\right) \widehat{\boldsymbol{\Omega}} \left((\theta_1^\star, \theta_2^\star) - (1, 0)\right)' \\
\text{ENC}_2 &= T\left((\theta_1^\star, \theta_2^\star) - (0, 1)\right) \widehat{\boldsymbol{\Omega}} \left((\theta_1^\star, \theta_2^\star) - (0, 1)\right)',
\end{aligned}$$

where $\widehat{\boldsymbol{\Omega}} = g_T(\boldsymbol{\theta})' \boldsymbol{S}^{-1} g_T(\boldsymbol{\theta})$. The asymptotic distribution of the GMM estimates of $\boldsymbol{\theta}$ requires the moment conditions to be once differentiable. To satisfy this requirement, we follow Giacomini and Komunjer (2005) and replace the moment condition with the following smooth approximation:

$$g_\tau(\boldsymbol{\theta}) = \frac{\sum_{t=1}^{T} \left[\tau - (1 - \exp((y_{t+1} - \boldsymbol{\theta}' \widehat{\boldsymbol{q}}_t)/\eta))\right] \mathbb{1}\{y_{t+1} - \boldsymbol{\theta}' \widehat{\boldsymbol{q}}_t < 0\}) \boldsymbol{z}_T}{T},$$

where η is the smoothing parameter. We choose the critical values, c_{crit} of the test from a χ_2^2 distribution, in which $\widehat{q}_{i,t}$ encompasses $\widehat{q}_{j,t}$, if $\text{ENC}_i \leq c_{\text{crit}}$ $\forall i \neq j = 1, 2$. In the empirical application, the vector of instruments, z_T, is $(1,\, r_t,\, VaR_{i,t},\, VaR_{j,t}),\, \forall\, i \neq j = 1, 2$.

We select η to be 0.005, following the CQFE test rejection probabilities in Giacomini and Komunjer (2005), since our POOS size is $2,000$ observations. We consider the following five blocks: i) the non-parametric, ii) the non-linear, iii) the non-linear MIDAS, iv) the linear and v) the linear MIDAS blocks. The non-parametric block consists of the quantile polynomial and quantile splines estimators, the non-linear block consists of the *deep quantile* estimators for the different regularization schemes and the non-linear MIDAS block consists of the *deep MIDAS* estimators for the different regularization schemes. Finally, the linear and linear MIDAS blocks consist of the linear quantile and linear quantile MIDAS estimators, respectively.

We examine each block of models across different quantiles. Specifically, we consider how many times the models within a specific block outperform models from other blocks and present these results in Table 2. Under this setting a *win* denotes that the prevailing model encompasses the competing benchmark model, while a *loss* means that the competing model encompasses the prevailing one. Precisely, we consider a *win* when the computed p-value of the CQFE test fails to reject the null hypothesis, i.e. H_{10} or H_{20}. On the contrary, in the case where the CQFE test suggests that there is no encompassing between the forecasts, we consider this as a *loss*, i.e. the null hypothesis is rejected. Furthermore, the CQFE test has a gray zone in which the test can fail to reject both null hypotheses (H_{10} and H_{20}), hence the test is inconclusive. Below we summarise the CQFE testing results for the different quantiles when $\eta = 0.005$.

For the 10^{th} quantile, the non-linear block encompasses 660 times the competing blocks, in comparison to the linear block, which encompasses the competing blocks 165 times and the non-parametric block that encompasses the others 320 times. The linear block does not encompass other blocks less than 23 times and the non-linear block for 139 times. Additionally, the test is inconclusive 643 times for the non-linear block and 149 times for the linear one. Thus, the non-linear block is ranked first in terms of how many times it encompasses the other blocks and the non-linear MIDAS block is ranked second.

For the 5^{th} quantile, the non-linear block encompasses 711 times other blocks, 333 times the non-parametric and the linear 177 times. Further, the linear block does not encompass the other blocks 11 times and the non-linear 88 times. Finally, for the non-linear block, the CQFE test is inconclusive 702 times and 167 times for the linear block. The ranking of the first two blocks is the same as in the 10^{th} quantile.

Finally, we examine the 1^{st} quantile. In this case, the non-linear block encompasses 723 times the other blocks, 341 times the non-parametric and the linear block 171 times. Furthermore, the linear block does not encompass 17 times the other blocks and the non-linear 76 times. The test is inconclusive 715 times for the non-linear block and 165 times for the linear one. The ranking remains the same as above. Results for different smoothing parameters η suggest similar patterns and are available upon request.

Table 2 about here

5 Semi-Structural analysis

A general issue in ML is the trade-off between accuracy and interpretability; where the output of a highly complicated model, e.g. a deep neural network, can have great accuracy or forecasting performance, but cannot be easily interpreted. In this section we first discuss the details of two methods that can be used to make ML methods interpretable. The first one is the Shapley Additive Explanation Values (SHAP), that has received a lot of attention recently, and the second is partial derivatives. Further we make a formal comparison on the output of both methods, based on the output of the *deep quantile* estimator that illustrates, i) that both methods can be used to make the impact of each covariate in neural networks interpretable and ii) perhaps surprisingly that the use of partial derivatives, offers more stable results at a fraction of the computational cost.

5.1 Shapley values

Shapley values (SHAP) are a general class of additive attribution methods, based on the initial work of Shapley (1953) where the goal was to determine how to fairly split a pay-off among players in a cooperative game. In the context of ML, the goal of SHAP values is to explain the prediction of the dependent variable by estimating the contribution of each covariate to the prediction. SHAP values, following the exposition in Lundberg and Lee (2017) and Lundberg, Erion, and Lee (2018) can be constructed as follows.

Let $f(\boldsymbol{x}_t) = \widehat{G}(\boldsymbol{x}_t, \boldsymbol{w})$ be the output of the estimated model we wish to interpret, given a $p \times 1$ vector of covariates $\boldsymbol{x}_t$, and $\widehat{f}$ the explanation model, to be defined below. Further, let $\boldsymbol{x}_t^{\dagger}$ be the $M \times 1$ subset (vector) of $\boldsymbol{x}_t$ that contains simplified covariates. These simplified covariates, can be mapped to the original through a mapping function $h_{\boldsymbol{x}_t}(\cdot)$, such that $\boldsymbol{x}_t = h_{\boldsymbol{x}_t}(\boldsymbol{x}_t^{\dagger})$. Then under the local accuracy property of Lundberg and Lee (2017), if there exists a vector, $\boldsymbol{z}_t^{\dagger}$, with binary inputs, such that $\boldsymbol{z}_t^{\dagger} \approx \boldsymbol{x}_t^{\dagger}$, then $\widehat{f}(\boldsymbol{z}_t^{\dagger}) \approx f(h_{\boldsymbol{x}_t}(\boldsymbol{z}_t^{\dagger}))$, where the explanation model (i.e. the additive attribution function) is

$$\widehat{f}(\boldsymbol{z}_t^{\dagger}) = \phi_0 + \sum_{i=1}^{M} \phi_i z_{t,i}^{\dagger}, \tag{23}$$

and $\widehat{f}(\boldsymbol{z}_t^{\dagger})$ represents the linear decomposition of the original ML model, where ϕ_0 is the intercept, $\phi_i \in \mathbb{R}$ is the effect to each dependent variable $z_t^{\dagger} \in (0, 1)$, that provides local and global inference at the same time. If $z_{t,i} = 1$ then the covariate is observed, on the contrary, if $z_{t,i} = 0$ then the covariate is unknown. Under the following three properties: i) local accuracy i.e. the explanation function should match the original model, ii) missingness, which ensures that input variable have no attributed effect and iii) consistency, under which, if an input variables is important, then the effect to each dependent variable should not decline, the SHAP value is

$$\phi_i = \sum_{M \subseteq p \backslash \{i\}} \frac{|M|!\,(p - |M| - 1)!}{p!} \left[f_{M \cup \{i\}} \left(x_{M \cup \{i\}} \right) - f_M(x_M) \right], \tag{24}$$

where p is the set of all predictors, $|M|$ is the number of non-zero elements in $\boldsymbol{x}_t^{\dagger}$, $f_M(x_M)$ is the model's output using except from the i^{th} covariate, and $f_{M \cup \{i\}} \left(x_{M \cup \{i\}} \right)$ is the output of the model, when $\{i\}$ is included in the covariate set.

The calculation of SHAP values can be computationally expensive, as it requires 2^N possible permutations of the predictors.For the case of deep neural networks Lundberg and Lee (2017), and Shrikumar, Greenside, and Kundaje (2017), have shown that *DeepLIFT* can be used as an approximation of the deep SHAP that is computationally feasible [6], preserving the three properties above. *DeepLIFT* is a recursive prediction explanation method for deep learning. The Additive feature attribution methods analogy of *DeepLIFT* is called the summation-to-delta property is

$$\sum_{i=1}^{p} C_{\Delta x_{t,i} \Delta o} = \Delta o. \tag{25}$$

Then the SHAP values can be obtained as

$$\phi_i = C_{\Delta x_{t,i} \Delta o},$$

where $C_{\Delta x_{t,i} \Delta o}$, represents the impact of a covariate to a reference value relative to the initial value, is assigned to each $x_{t,i}$ covariate, $o = f(\cdot)$ is the output of the model, $\Delta o = f(x) - f(r)$, $\Delta x_{t,i} = f(x_{t,i}) - r_{t,i}$ and r the reference value. Eq. 25 matches eq. 23, if in Δo we set $\phi_0 = f(r_{t,i})$ and $\phi_i = C_{\Delta x_{t,i} \Delta o}$.

5.2 Partial Derivatives

The use of partial derivatives for the interpretation of a model is straight forward in econometrics, with various uses, ranging from the simple linear regression model to impulse response analysis. In this section we show how partial derivatives can be used even in highly non-linear deep neural networks. Before we start the analysis, note that while the deep neural networks are highly non-linear, their solution/output via SGD optimization methods, can be treated as differentiable function, as the majority of activation functions are differentiable. Let's consider the case of ReLU, that is not differentiable at 0, whereas it is in every other point. From the point of gradient descent, heuristically, it works well enough to treat it as a differentiable function. Further, Goodfellow, Bengio, and Courville (2016) argue that this issue is negligible and ML softwares are prone to rounding errors, which make it very unlikely to compute the gradient at a singularity point. Note that even in this extreme case, both SGD and $ADAM$, will use the right subgradient at 0.

For a general $\boldsymbol{x}_t \in \mathbb{R}^p$, let

$$d_{j,i,t} = \frac{\partial \widehat{G}_{j,\tau}(\boldsymbol{x}_t, \boldsymbol{w})}{\partial x_{j,i,t-1}}, \tag{26}$$

denote the partial derivative of covariate $x_i = x_{it}$, for $i = 1, \ldots, p$ at time $t = 1, \ldots, T$, $\widehat{G}_{j,\tau}(\boldsymbol{x}_t, \boldsymbol{w})$ is the forecasted $VaR_{j,t}$, across the j different VaR specifications we consider. We assess the partial derivative in time, since, following Kapetanios (2007), we expect it to vary in time, due to the inherent non-linearity of the neural network. Our covariate(s) $\boldsymbol{x}_t$ are the conditional volatility for GARCH and RM, VaR lagged values, the absolute $S\&P500$ daily return and the positive and negative S&P500 daily returns for SV and ASV, respectively. It is evident that under the classic linear regression problem, or linear quantile regression model, the effect of the covariates $\boldsymbol{x}_t$ to the dependent variable y_t is constant, time invariant, and corresponds to $\widehat{\boldsymbol{\beta}}(\tau)$.

5.3 Results

In this application we use the whole sample size i.e. around 36 years of daily returns on the $S\&P500$ index to provide an accurate interpretation of our *deep quantile* estimator. Figures 5 – 8 illustrate the partial derivatives and SHAP values evaluated in time on the output of our *deep quantile*[7] estimator, for a specific quantile τ. Further, we compare the partial derivatives of the *deep quantile* estimator relative to the linear quantile regression partial derivative, i.e. the $\boldsymbol{\beta}(\tau)$ coefficient. Both partial derivatives and SHAP values seem to identify interesting patterns that can be linked to some well known events. Below we discuss our results for all models we have considered in our empirical application.

The results for the first two models, i.e. *GARCH* and *RM* can be summarised together, since in both models there is only one covariate, that is the conditional volatility, but with a different specification. The results from this model are illustrated in Figures 5. We find that the partial derivative appears to be more stable over time, fluctuating around the constant partial derivative, $\boldsymbol{\beta}(\tau)$, of the linear quantile estimator. When there is a crisis or a stressful event in the financial markets, they increase. As an example, we see significant spikes in the partial derivatives, both in March 2020 as well as in 2008, which stand for the onset of the COVID-19 pandemic and the *Great Recession* respectively. We also find that the biggest increase occurs in 1987, the year when *Black Monday* happened, and also significant variation during the U.S. government shutdown in 2019. The values for the partial derivatives generally increase, as τ decreases. SHAP values have a similar behaviour with the partial derivatives, but are more volatile across time. For the first two models, there are some events, e.g. during the 1991, where the values for both SHAP and partial derivatives do not increase a lot. We view this finding as an inability of these two models, to properly account for this crisis.

In the last two models, the merit of SHAP values and partial derivatives becomes clear, since in these models we have more than one covariates and both methods can provide an indication on the effect of each covariate on the final output. Overall, we find that increasing the number of covariates, allow the models to account for all crises within the sample. For the case of the SV model, we find that the important covariate is the lagged values of VaR, rather than the absolute values of $S\&P500$. Similar to the one covariate models, we find that the partial derivatives are more stable than SHAP values, fluctuating closely around $\boldsymbol{\beta}(\tau)$ and picking up when there are crisis or distress in the economy or financial markets. The SHAP values again appear to be more volatile with a wider range. Similar to the findings of the one covariate models, the higher the values for the partial derivative and SHAP, the lower the τ quantile.

For the case of the ASV model, we find that again the lagged values of VaR is the most significant covariate, the negative $S\&P500$ returns have some impact and the positive $S\&P500$ returns are almost insignificant. Similar to the cases above, we find that the partial derivative is more stable than SHAP values, fluctuating closely around $\boldsymbol{\beta}(\tau)$ and picking up when there is a crisis or distress in the economy or financial markets. The SHAP values again appear to be more volatile with a wider range. Again and same as before, lower quantiles have higher partial derivatives. The results for these two models are illustrated in Figures 6, 7 and 8.

Different penalization schemes maintain the aforementioned results, with a lower magnitude. Overall, we observe that the linear quantile regression shows a fixed pattern across time and is evident that this model does not anticipate shocks in the economy. Our analysis suggests that it is higher during stressful events. As Engle and Manganelli (2004) suggest, SV and ASV react more to negative shocks and in stressful events their spike is larger than the $GARCH$ and RM models. Finally, covariates with the minimum contribution on the forecasted values, such as the positive $S\&P500$ returns has negligible impact on both SHAP and partial derivatives values.

Figures 5 – 8 about here

6 Conclusion

In this paper we contribute to the expanding literature on the use of ML in finance and propose a *deep quantile* estimator that has the potential to capture the non-linear association between asset returns and predictors. In Section 2, we lay out the exact workings of our proposed estimator, and illustrate how it generalises linear quantile regression.

In the Monte Carlo exercise in Section 3, we study the finite sample properties of the *deep quantile* estimator, based on a number of data generating processes. We present extensive evidence the estimator gives good finite sample performance, that is a function of T, uniformly across different regularization schemes.

We use the *deep quantile* estimator, with various penalization schemes, to forecast *VaR*. We find that our estimator gives considerable predictive gains, up to 74%, relative to the *VaR* forecasts produced by the linear quantile regression. This result is backed by the forecasting accuracy tests, i.e. the Diebold and Mariano (1995), the Giacomini and White (2006) and the quantile score tests. Further, results from the CQFE test of Giacomini and Komunjer (2005) suggest that forecasts obtained from the non-linear estimators encompass forecasts from the linear and non-parametric models with a higher frequency. These findings are in support of the non-linear association between the conditional quantile of asset returns and covariates, hence suggesting a new avenue in forecasting in finance and in macroeconomics during extreme events.

In addition, we do a semi-structural analysis to examine the contribution of the predictors in *VaR* over time. We consider, following the ML literature, SHAP values and further partial derivatives. Our findings suggests that our non-linear estimator reacts more in stressful events and exhibits time-variation, while the linear quantile estimator presents, as expected, a constant time invariant behaviour. We conclude that financial variables are characterised by non-linearities, that our proposed *deep quantile* estimator can approximate quite well.

Finally, we make a formal comparison between SHAP and partial derivatives, and interestingly find that partial derivatives can be used to make ML methods interpretable, are less volatile, easier to interpret and can be computed at a fraction of time used in the calculation of SHAP values.

References

ADAMS, P. A., T. ADRIAN, N. BOYARCHENKO, AND D. GIANNONE (2021): "Forecasting macroeconomic risks," *International Journal of Forecasting*.

ATHEY, S., AND G. W. IMBENS (2017): "The state of applied econometrics: Causality and policy evaluation," *Journal of Economic Perspectives*, 31(2), 3–32.

BABII, A., X. CHEN, E. GHYSELS, AND R. KUMAR (2020): "Binary choice with asymmetric loss in a data-rich environment: Theory and an application to racial justice," *arXiv preprint arXiv:2010.08463*.

BATES, J. M., AND C. W. GRANGER (1969): "The combination of forecasts," *Journal of the Operational Research Society*, 20(4), 451–468.

BAUR, D., AND N. SCHULZE (2005): "Coexceedances in financial markets—a quantile regression analysis of contagion," *Emerging Markets Review*, 6(1), 21–43.

BELLONI, A., V. CHERNOZHUKOV, D. CHETVERIKOV, AND I. FERNÁNDEZ-VAL (2019): "Conditional quantile processes based on series or many regressors," *Journal of Econometrics*, 213(1), 4–29.

BELLONI, A., V. CHERNOZHUKOV, AND C. HANSEN (2014): "Inference on treatment effects after selection among high-dimensional controls," *The Review of Economic Studies*, 81(2), 608–650.

BOLLERSLEV, T. (1986): "Generalized autoregressive conditional heteroskedasticity," *Journal of econometrics*, 31(3), 307–327.

BUCCI, A. (2020): "Realized Volatility Forecasting with Neural Networks," *Journal of Financial Econometrics*, 18(3), 502–531.

CARRIERO, A., G. KAPETANIOS, AND M. MARCELLINO (2009): "Forecasting exchange rates with a large Bayesian VAR," *International Journal of Forecasting*, 25(2), 400–417.

CHEN, X., Y. LIU, S. MA, AND Z. ZHANG (2020): "Efficient estimation of general treatment effects using neural networks with a diverging number of confounders," *arXiv preprint arXiv:2009.07055*.

CHERNOZHUKOV, V., AND L. UMANTSEV (2001): "Conditional value-at-risk: Aspects of modeling and estimation," *Empirical Economics*, 26(1), 271–292.

DIEBOLD, F. X., AND R. S. MARIANO (1995): "Comparing predictive accuracy," *Journal of Business and Economic Statistics*, 13, 253–263, Reprinted in Mills, T. C. (ed.) (1999), *Economic Forecasting. The International Library of Critical Writings in Economics*. Cheltenham: Edward Elgar.

DU, Z., M. WANG, AND Z. XU (2019): “On Estimation of Value-at-Risk with Recurrent Neural Network,” in *2019 Second International Conference on Artificial Intelligence for Industries (AI4I)*, pp. 103–106. IEEE.

ENGLE, R. F., AND S. MANGANELLI (2004): “CAViaR: Conditional autoregressive value at risk by regression quantiles,” *Journal of Business & Economic Statistics*, 22(4), 367–381.

FARRELL, M. H., T. LIANG, AND S. MISRA (2021): “Deep neural networks for estimation and inference,” *Econometrica*, 89(1), 181–213.

GALANT, A., AND H. WHITE (1992): “On learning the derivatives of an unknown mapping with multilayer feed forward neural network,” *Neural networks*, 5, 129–138.

GHYSELS, E., A. PLAZZI, AND R. VALKANOV (2016): “Why invest in emerging markets? The role of conditional return asymmetry,” *The Journal of Finance*, 71(5), 2145–2192.

GHYSELS, E., A. PLAZZI, R. VALKANOV, A. RUBIA, AND A. DOSSANI (2019): “Direct versus iterated multiperiod volatility forecasts,” *Annual Review of Financial Economics*, 11, 173–195.

GHYSELS, E., P. SANTA-CLARA, AND R. VALKANOV (2004): “The MIDAS touch: Mixed data sampling regression models,” .

GIACOMINI, R., AND I. KOMUNJER (2005): “Evaluation and combination of conditional quantile forecasts,” *Journal of Business & Economic Statistics*, 23(4), 416–431.

GIACOMINI, R., AND H. WHITE (2006): “Tests of conditional predictive ability,” *Econometrica*, 74(6), 1545–1578.

GOODFELLOW, I., Y. BENGIO, AND A. COURVILLE (2016): *Deep learning*. MIT press.

GU, S., B. KELLY, AND D. XIU (2020a): “Autoencoder asset pricing models,” *Journal of Econometrics*.

——— (2020b): “Empirical asset pricing via machine learning,” *The Review of Financial Studies*, 33(5), 2223–2273.

HANSEN, L. P. (1982): “Large sample properties of generalized method of moments estimators,” *Econometrica: Journal of the Econometric Society*, pp. 1029–1054.

HARVEY, D., S. LEYBOURNE, AND P. NEWBOLD (1997): “Testing the equality of prediction mean squared errors,” *International Journal of forecasting*, 13(2), 281–291.

HE, Z., AND A. KRISHNAMURTHY (2013): “Intermediary asset pricing,” *American Economic Review*, 103(2), 732–70.

HORNIK, K. (1991): “Approximation capabilities of multilayer feedforward networks,” *Neural networks*, 4(2), 251–257.

HORNIK, K., M. STINCHCOMBE, AND H. WHITE (1989): "Multi-Layer Feedforward Networks and Universal Approximators," *Neural Network*, 2, 359–366.

JOSEPH, A. (2019): "Shapley regressions: a framework for statistical inference on machine learning models," Discussion paper, Bank of England.

J.P. MORGAN, M. (1996): "Reuters (1996) RiskMetrics-Technical Document," *JP Morgan*.

KAPETANIOS, G. (2007): "Measuring conditional persistence in nonlinear time series," *Oxford Bulletin of Economics and Statistics*, 69(3), 363–386.

KAPETANIOS, G., AND A. P. BLAKE (2010): "Tests of the martingale difference hypothesis using boosting and RBF neural network approximations," *Econometric Theory*, 26(5), 1363–1397.

KEILBAR, G., AND W. WANG (2021): "Modelling systemic risk using neural network quantile regression," *Empirical Economics*, pp. 1–26.

KINGMA, D. P., AND J. BA (2014): "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*.

KOENKER, R. (2005): *Quantile Regression*, Econometric Society Monographs. Cambridge University Press.

KOENKER, R., AND G. BASSETT JR (1978): "Regression quantiles," *Econometrica: journal of the Econometric Society*, pp. 33–50.

KOENKER, R., V. CHERNOZHUKOV, X. HE, AND L. PENG (2017): *Handbook of quantile regression*. CRC press.

KOENKER, R., AND K. F. HALLOCK (2001): "Quantile regression," *Journal of economic perspectives*, 15(4), 143–156.

LIANG, S., AND R. SRIKANT (2016): "Why deep neural networks for function approximation?," *arXiv preprint arXiv:1610.04161*.

LUNDBERG, S. M., G. G. ERION, AND S.-I. LEE (2018): "Consistent individualized feature attribution for tree ensembles," *arXiv preprint arXiv:1802.03888*.

LUNDBERG, S. M., AND S.-I. LEE (2017): "A unified approach to interpreting model predictions," in *Advances in neural information processing systems*, pp. 4765–4774.

MEINSHAUSEN, N. (2006): "Quantile regression forests," *Journal of Machine Learning Research*, 7(Jun), 983–999.

NEWEY, W. K., AND K. D. WEST (1987): "Hypothesis testing with efficient method of moments estimation," *International Economic Review*, pp. 777–787.

PADILLA, O. H. M., W. TANSEY, AND Y. CHEN (2020): "Quantile regression with deep ReLU Networks: Estimators and minimax rates," *arXiv preprint arXiv:2010.08236*.

PARK, J., AND I. W. SANDBERG (1991): "Universal Approximation using Radial-Basis-Function Networks," *Neural Computation*, 3(4), 246–257.

POHL, W., K. SCHMEDDERS, AND O. WILMS (2018): "Higher order effects in asset pricing models with long-run risks," *The Journal of Finance*, 73(3), 1061–1111.

SCHMIDT-HIEBER, J. (2020): "Nonparametric regression using deep neural networks with ReLU activation function," *The Annals of Statistics*, 48(4), 1875–1897.

SHAPLEY, L. S. (1953): "A value for n-person games," *Contributions to the Theory of Games*, 2(28), 307–317.

SHRIKUMAR, A., P. GREENSIDE, AND A. KUNDAJE (2017): "Learning important features through propagating activation differences," *arXiv preprint arXiv:1704.02685*.

SMALTER HALL, A., AND T. R. COOK (2017): "Macroeconomic indicator forecasting with deep neural networks," *Federal Reserve Bank of Kansas City Working Paper*, (17-11).

TAMBWEKAR, A., A. MAIYA, S. S. DHAVALA, AND S. SAHA (2021): "Estimation and Applications of Quantiles in Deep Binary Classification," *IEEE Transactions on Artificial Intelligence*.

TOBIAS, A., AND M. K. BRUNNERMEIER (2016): "CoVaR," *The American Economic Review*, 106(7), 1705.

WAGER, S., AND S. ATHEY (2018): "Estimation and inference of heterogeneous treatment effects using random forests," *Journal of the American Statistical Association*, 113(523), 1228–1242.

YAROTSKY, D. (2017): "Error bounds for approximations with deep ReLU networks," *Neural Networks*, 94, 103–114.

ZHANG, W., H. QUAN, AND D. SRINIVASAN (2018): "An improved quantile regression neural network for probabilistic load forecasting," *IEEE Transactions on Smart Grid*, 10(4), 4425–4434.

ZOU, H., AND T. HASTIE (2005): "Regularization and variable selection via the elastic net," *Journal of the royal statistical society: series B (statistical methodology)*, 67(2), 301–320.

Notes

[1] Autoencoders are artificial neural networks that can be used as a dimensionality reduction technique.

[2] Deep Elastic net

[3] We have also considered batch normalisation and find that overall, results exhibit similar pattern with and without it.

[4] ADAM is using estimates for the first and second moments of the gradient to calculate the learning rate.

[5] The split is approximately equal to $60\%, 20\%, 20\%$. We have examined alternative training, validation and test splits, which give similar patterns to the presented empirics and are available upon request.

[6] There are other methods that can be used to achieve this, such as Tree Explainer, Kernel Explainer, Linear Explainer, Gradient Explainer.

[7] In this section we limit our attention in the output of the best performing model, in terms of its forecasting capacity, as reflected by the forecast gains measure in Section 4, for each model, based on the different penalisation schemes. Results from all the different penalisation schemes suggest similar patterns to the ones discussed above and are available upon request.

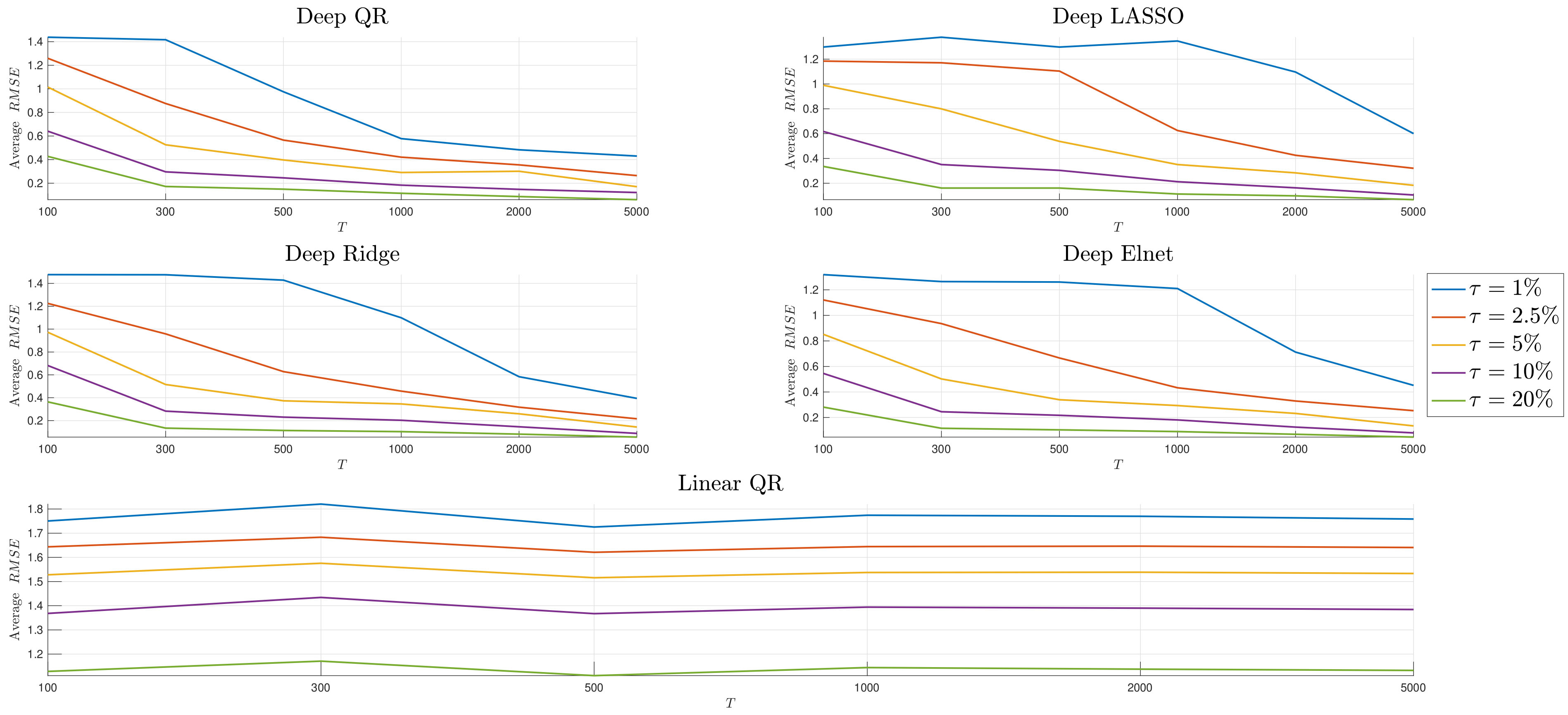


Figure 1: Monte Carlo results for Case I. Model: $y_t = h_\tau(x_t) + u_t,\ h_\tau(x_t) = \sin(2\pi x_t),\ x_t \sim\ N(0,1),\ u_t \sim iidN(-\sigma\Phi^{-1}(\tau), \sigma^2),\ \sigma = 0.1$ and Φ^{-1} is the quantile function of the standard normal distribution. Figure presents the average mean squared error of the estimated residuals, $AMSE_{\hat{u}_t,\,pen}$ for the different penalization schemes (Model), $T = 100, 300, 500, 1000, 2000, 5000$ and different quantiles, $\tau = (1\%, 2.5\%, 5\%, 10\%, 20\%)$.

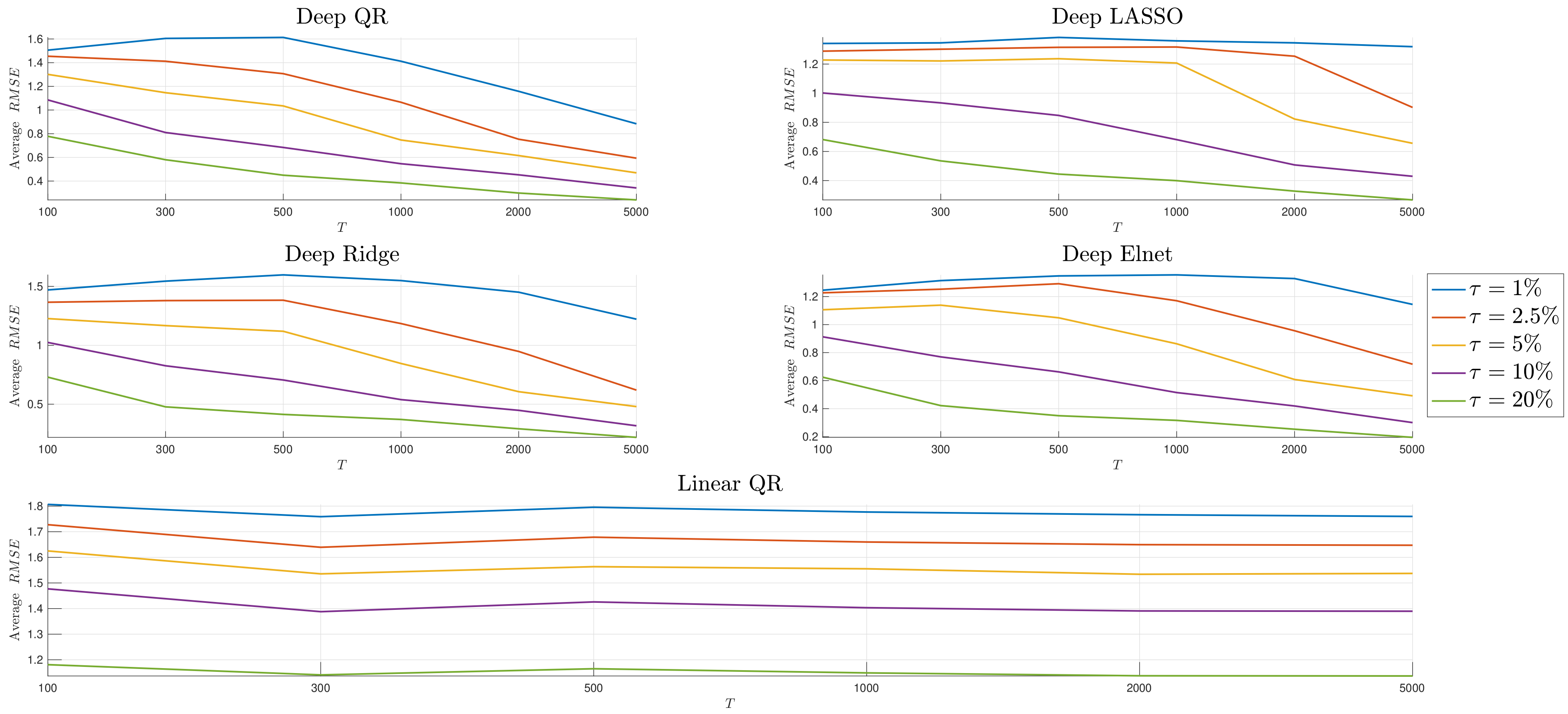


Figure 2: Monte Carlo results for Case II. Model: $y_t = h_\tau(x_t) + u_t,\ h_\tau(x_t) = \sin(2\pi x_t),\ x_t = 0.8x_{t-1} + \varepsilon_t,\ \varepsilon_t \sim N(0,1),\ u_t \sim iidN(-\sigma\Phi^{-1}(\tau), \sigma^2),\ \sigma = 0.1$ and Φ^{-1} is the quantile function of the standard normal distribution. Figure presents the average mean squared error of the estimated residuals, $AMSE_{\hat{u}_t,pen}$ for the different penalization schemes (Model), $T = 100, 300, 500, 1000, 2000, 5000$ and different quantiles, $\tau = (1\%, 2.5\%, 5\%, 10\%, 20\%)$

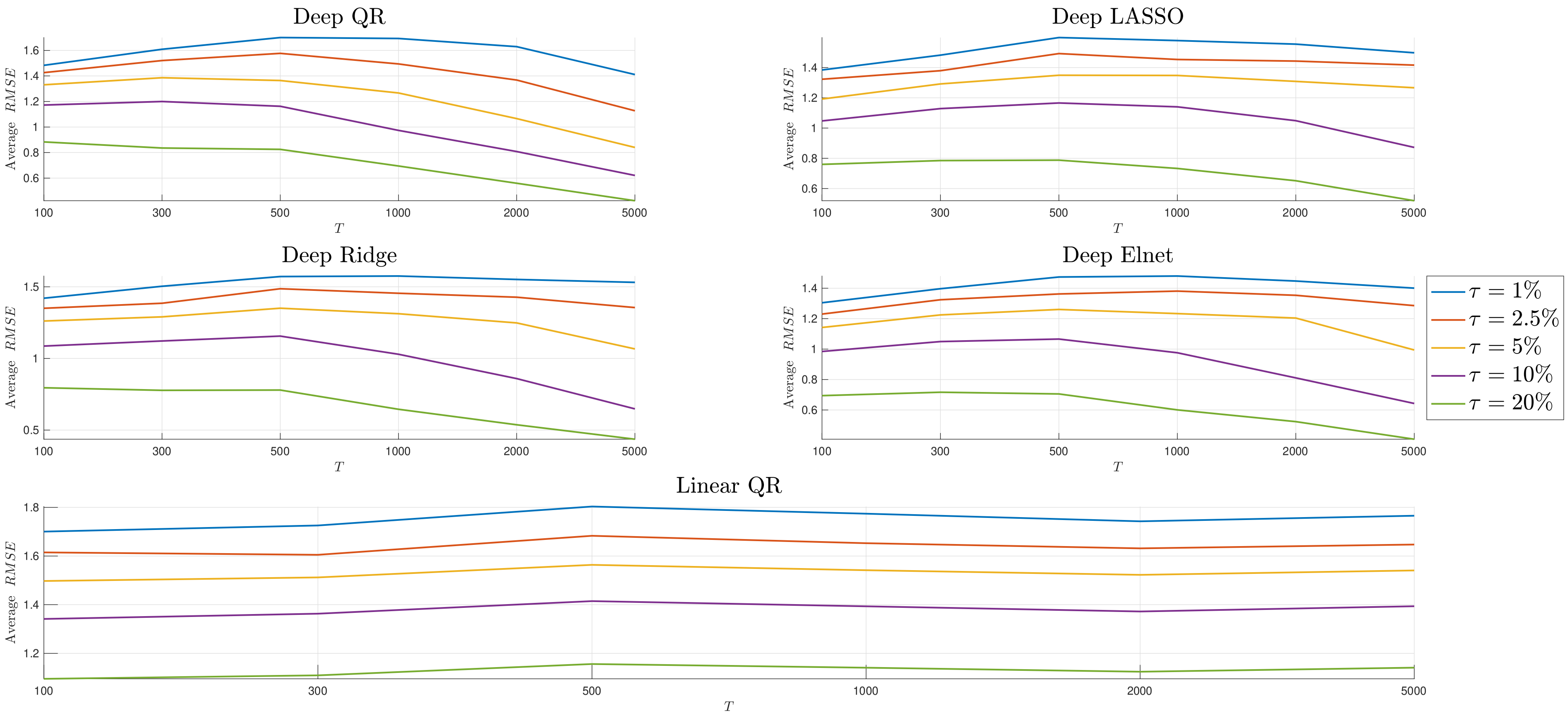


Figure 3: Monte Carlo results for Case III. Model: $y_t = h_\tau(x_t) + u_t,\ h_\tau(x_t) = \sin(2\pi x_t),\ x_t = \sigma_t \varepsilon_t,\ \sigma_t^2 = 1 + 0.7x_{t-1}^2 + 0.2\sigma_{t-1}^2,\ \varepsilon_t \sim N(0,1),\ u_t \sim iidN(-\sigma\Phi^{-1}(\tau), \sigma^2),\ \sigma = 0.1$ and Φ^{-1} is the quantile function of the standard normal distribution. Figure presents the average mean squared error of the estimated residuals, $AMSE_{\hat{u}_t, pen}$ for the different penalization schemes (Model), $T = 100, 300, 500, 1000, 2000, 5000$ and different quantiles, $\tau = (1\%, 2.5\%, 5\%, 10\%, 20\%)$

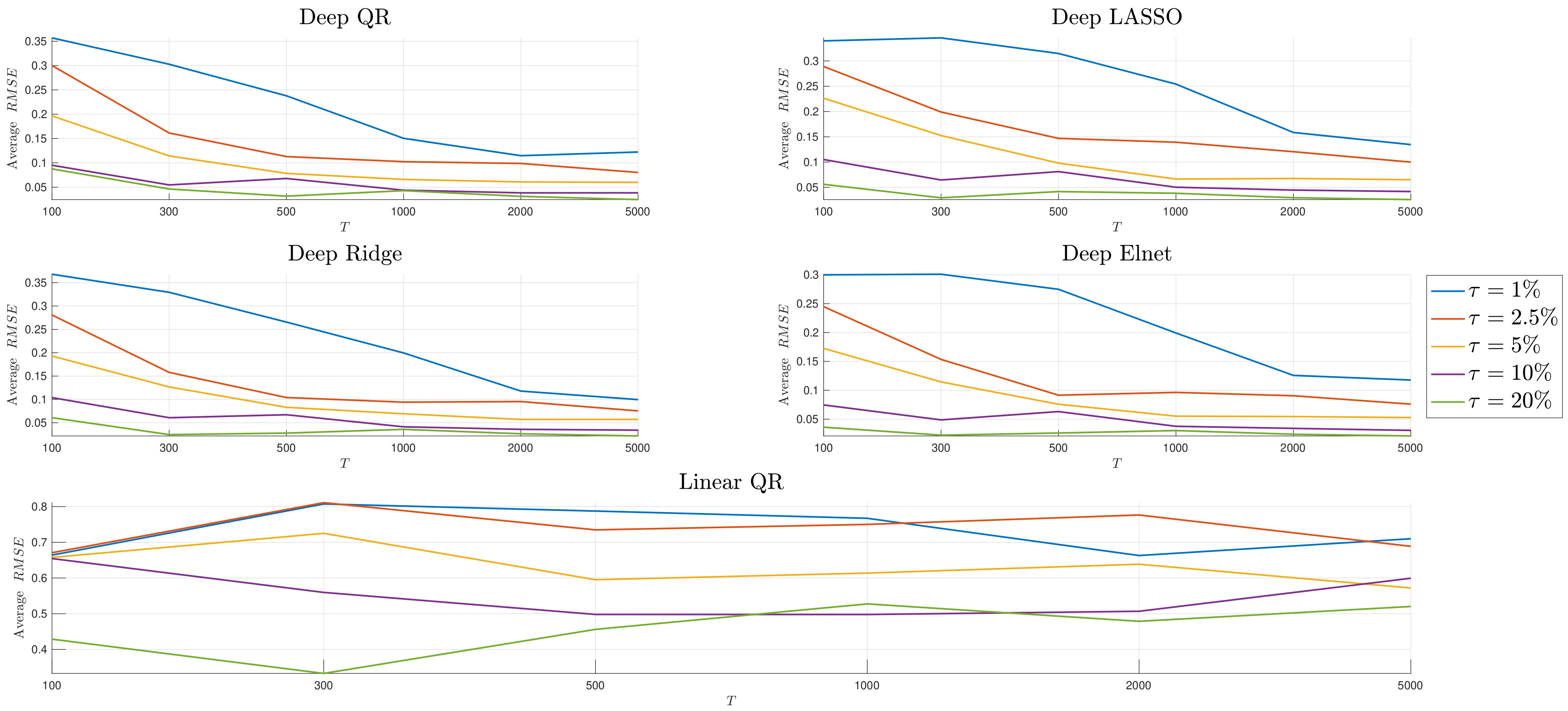


Figure 4: Monte Carlo results for Case IV. Model: $y_t = h_\tau(x_t) + u_t,\ h_\tau(x_t) = G(x_t, \boldsymbol{w}),\ x_t \sim N(0,1),\ w_{i,j} = \delta_{i,j} 1(\delta_{i,j} > 0.1),\ \delta_{i,j} \sim U(0,1),\ u_t \sim iidN(-\sigma\Phi^{-1}(\tau), \sigma^2)$, $\sigma = 0.1$ and Φ^{-1} is the quantile function of the standard normal distribution. Figure presents the average mean squared error of the estimated residuals, $AMSE_{\hat{u}_t,pen}$ for the different penalization schemes (Model), $T = 100, 300, 500, 1000, 2000, 5000$ and different quantiles, $\tau = (1\%, 2.5\%, 5\%, 10\%, 20\%)$

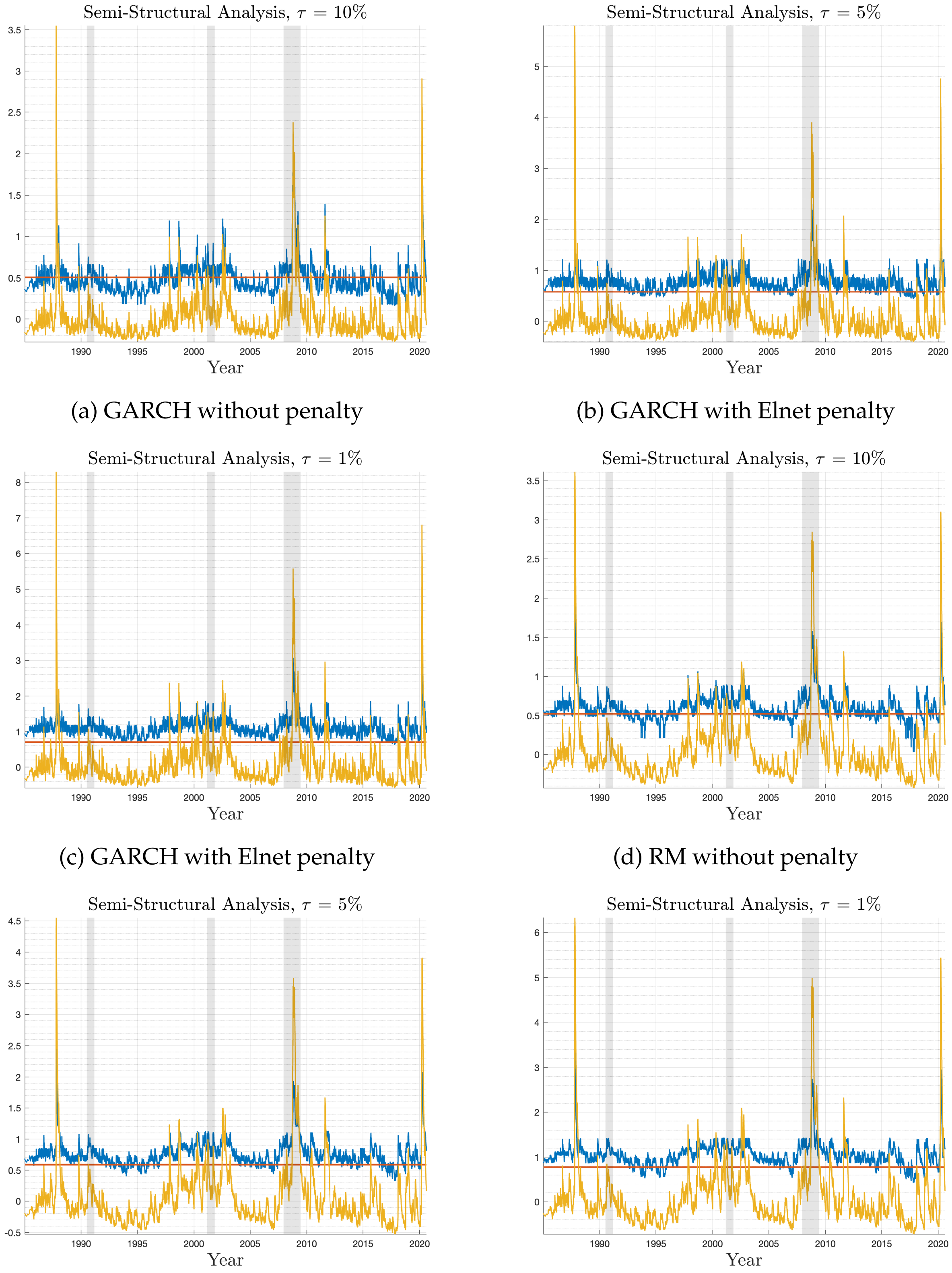
Model	τ	Polynomial	Spline	MIDAS	Deep QR	Deep MIDAS	Deep LASSO	Deep MIDAS LASSO	Deep Ridge	Deep MIDAS Ridge	Deep Elnet	Deep MIDAS Elnet
GARCH	1%	0.808	$0.631^{***/\dagger\dagger\dagger}$	1.015	$0.572^{***/\dagger\dagger\dagger}$	$0.454^{***/\dagger\dagger\dagger}$	$0.723^{***/\dagger\dagger\dagger}$	$0.502^{***/\dagger\dagger}$	$0.550^{***/\dagger\dagger\dagger}$	0.554^{***}	$0.410^{***/\dagger\dagger\dagger}$	$0.731^{***/\dagger\dagger\dagger}$
	5%	1.030	0.919	$1.230^{***/\dagger\dagger\dagger}$	$0.659^{***/\dagger\dagger\dagger}$	$0.504^{***/\dagger\dagger\dagger}$	$0.674^{***/\dagger\dagger\dagger}$	$0.852^{***/\dagger\dagger\dagger}$	$0.793^{***/\dagger\dagger\dagger}$	$0.600^{***/\dagger\dagger\dagger}$	$0.548^{***/\dagger\dagger\dagger}$	$0.541^{***/\dagger\dagger\dagger}$
	10%	1.103	$1.161^{***/\dagger\dagger}$	$1.276^{***/\dagger\dagger\dagger}$	$0.552^{***/\dagger\dagger}$	$0.563^{***/\dagger\dagger\dagger}$	$0.655^{***/\dagger\dagger}$	$0.644^{***/\dagger\dagger\dagger}$	$0.598^{***/\dagger\dagger}$	$0.586^{***/\dagger\dagger\dagger}$	$0.667^{***/\dagger\dagger\dagger}$	$0.598^{***/\dagger\dagger}$
RM	1%	0.771	$0.503^{***/\dagger\dagger\dagger}$	$1.059^{***/\dagger\dagger\dagger}$	$0.366^{***/\dagger\dagger\dagger}$	$0.399^{***/\dagger\dagger\dagger}$	$0.346^{***/\dagger\dagger\dagger}$	$0.399^{***/\dagger\dagger\dagger}$	$0.261^{***/\dagger\dagger\dagger}$	$0.242^{***/\dagger\dagger\dagger}$	$0.386^{***/\dagger\dagger\dagger}$	$0.399^{***/\dagger\dagger\dagger}$
	5%	0.918	$0.704^{***/\dagger\dagger\dagger}$	$1.195^{***/\dagger\dagger\dagger}$	$0.371^{***/\dagger\dagger\dagger}$	$0.302^{***/\dagger\dagger\dagger}$	$0.629^{***/\dagger\dagger\dagger}$	$0.333^{***/\dagger\dagger\dagger}$	$0.415^{***/\dagger\dagger\dagger}$	$0.497^{***/\dagger\dagger\dagger}$	$0.396^{***/\dagger\dagger\dagger}$	$0.818^{***/\dagger\dagger\dagger}$
	10%	1.089	1.011	$1.367^{***/\dagger\dagger\dagger}$	$0.433^{***/\dagger\dagger\dagger}$	$0.422^{***/\dagger\dagger\dagger}$	$0.500^{***/\dagger\dagger\dagger}$	$0.433^{***/\dagger\dagger\dagger}$	$0.511^{***/\dagger\dagger\dagger}$	$0.367^{***/\dagger\dagger\dagger}$	$0.433^{***/\dagger\dagger\dagger}$	$0.422^{***/\dagger\dagger\dagger}$
SV	1%	0.988	$1.308^{***/\dagger\dagger\dagger}$	$0.605^{***/\dagger\dagger\dagger}$	$0.661^{***/\dagger\dagger\dagger}$	$0.631^{***/\dagger\dagger\dagger}$	$0.655^{***/\dagger\dagger\dagger}$	$0.643^{***/\dagger\dagger\dagger}$	$0.732^{***/\dagger\dagger\dagger}$	$0.756^{***/\dagger\dagger\dagger}$	$0.679^{***/\dagger\dagger\dagger}$	$0.637^{***/\dagger\dagger\dagger}$
	5%	0.992	$1.473^{***/\dagger\dagger}$	$0.645^{***/\dagger\dagger\dagger}$	$0.629^{***/\dagger\dagger\dagger}$	$0.702^{***/\dagger\dagger\dagger}$	$0.698^{***/\dagger\dagger\dagger}$	$0.722^{***/\dagger\dagger\dagger}$	$0.645^{***/\dagger\dagger\dagger}$	$0.576^{***/\dagger\dagger\dagger}$	$0.812^{**/\dagger\dagger\dagger}$	$0.669^{***/\dagger\dagger\dagger}$
	10%	0.986^{*}	$1.175^{***/\dagger}$	$0.650^{***/\dagger\dagger\dagger}$	$0.657^{***/\dagger\dagger\dagger}$	$0.664^{***/\dagger\dagger\dagger}$	$0.762^{***/\dagger\dagger\dagger}$	$0.699^{***/\dagger\dagger\dagger}$	$0.720^{***/\dagger\dagger\dagger}$	$0.748^{***/\dagger\dagger\dagger}$	$0.783^{**/\dagger\dagger\dagger}$	$0.734^{***/\dagger\dagger\dagger}$
ASV	1%	0.998	$1.030^{***/\dagger\dagger\dagger}$	$3.524^{***/\dagger}$	$0.671^{***/\dagger\dagger\dagger}$	$0.695^{***/\dagger\dagger\dagger}$	$0.709^{***/\dagger\dagger\dagger}$	$0.651^{***/\dagger\dagger\dagger}$	$0.884^{\dagger\dagger\dagger}$	0.787^{**}	$0.815^{*/\dagger\dagger\dagger}$	$0.647^{***/\dagger\dagger\dagger}$
	5%	0.997	$1.024^{***/\dagger\dagger\dagger}$	$3.313^{**/\dagger}$	$0.648^{***/\dagger\dagger\dagger}$	$0.645^{***/\dagger\dagger\dagger}$	$0.687^{***/\dagger\dagger\dagger}$	$0.648^{***/\dagger\dagger\dagger}$	$0.627^{***/\dagger\dagger\dagger}$	$0.648^{***/\dagger\dagger\dagger}$	$0.687^{***/\dagger\dagger\dagger}$	$0.696^{***/\dagger\dagger\dagger}$
	10%	1.000	$1.236^{***/\dagger\dagger}$	$0.977^{***/\dagger\dagger\dagger}$	$0.644^{***/\dagger\dagger\dagger}$	$0.639^{***/\dagger\dagger\dagger}$	$0.657^{***/\dagger\dagger\dagger}$	$0.648^{***/\dagger\dagger\dagger}$	$0.634^{***/\dagger\dagger\dagger}$	$0.630^{***/\dagger\dagger\dagger}$	$0.639^{***/\dagger\dagger\dagger}$	$0.653^{***/\dagger\dagger\dagger}$

Table 1: Comparison of the forecasting methods. Table reports relative RMSFE. The smaller the entry (< 1) the better the forecast. *, **, and *** denote results from Diebold and Mariano (1995) test with the Harvey, Leybourne, and Newbold (1997) adjustment for predictive accuracy, indicating rejection of the null hypothesis that the models have the same predictive accuracy at the 10%, 5%, and 1% levels of significance, respectively. †, ††, and † † † denote results from the Giacomini and White (2006) test, indicating rejection of the null hypothesis of equal forecasting accuracy of the models at the 10%, 5%, and 1% levels of significance, respectively.

$\eta = 0.005$	Block	wins	losses	inconclusive	wins	losses	inconclusive	wins	losses	inconclusive
			$\tau = 1\%$			$\tau = 5\%$			$\tau = 10\%$	
	linear	171	17	165	177	11	167	165	23	149
	non-parametric	341	35	333	333	43	319	320	56	305
	non-linear	**723**	**76**	**715**	**711**	**88**	**702**	**660**	**139**	**643**
	MIDAS	168	20	164	168	20	166	167	21	165
	non-linear MIDAS	685	67	680	660	92	653	613	139	606

Table 2: Entries of the table present the number of times a block encompasses (wins), does not encompass (losses) and is inconclusive, according the CQFE test for different quantiles τ. Results are reported for $\eta = 0.005$.

Figure 5: Partial Derivative, SHAP and $\widehat{\beta}(\tau)$ for GARCH and RM models.



(a) GARCH without penalty

(b) GARCH with Elnet penalty

(c) GARCH with Elnet penalty

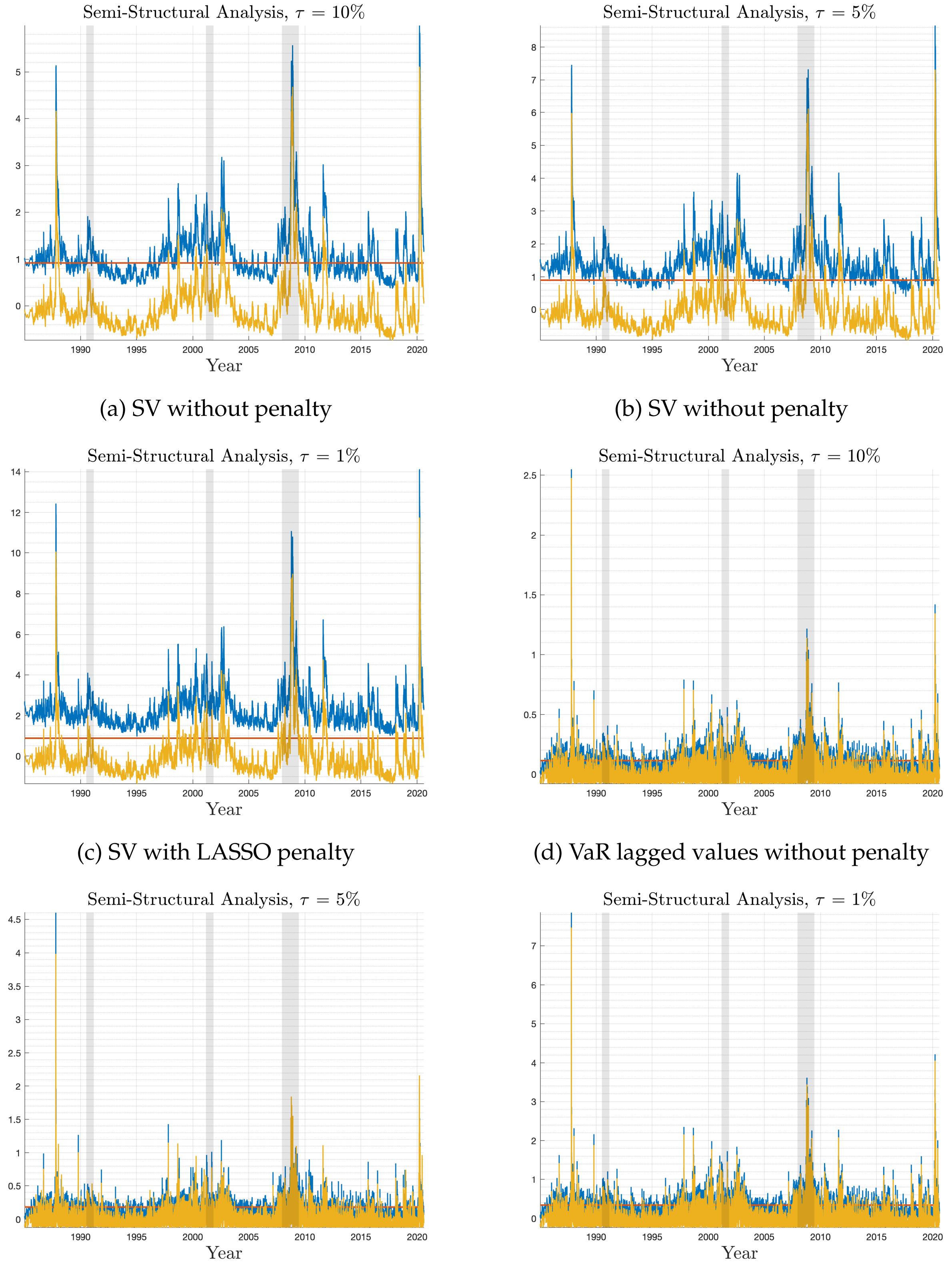
(d) RM without penalty

(e) RM without penalty

(f) RM with Ridge penalty

—:Partial Derivative, —: SHAP values, —: $\widehat{\beta}(\tau)$, shaded area presents NBER recession indicators

Figure 6: Partial Derivative, SHAP and $\widehat{\beta}(\tau)$ for SV model.



(a) SV without penalty

(b) SV without penalty

(c) SV with LASSO penalty

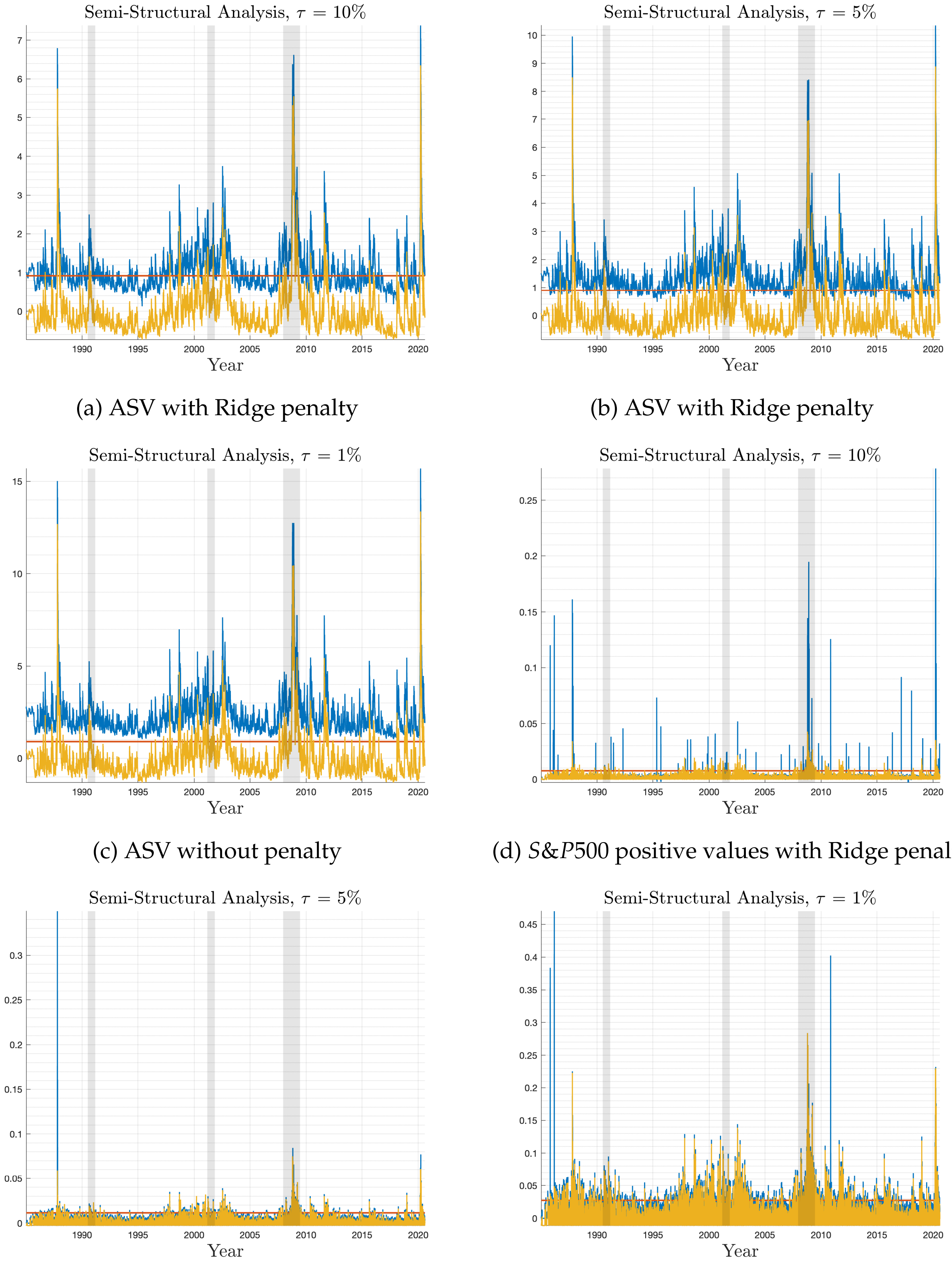
(d) VaR lagged values without penalty

(e) VaR lagged values without penalty

(f) VaR lagged values with LASSO penalty

▬:Partial Derivative, ▬: SHAP values, ▬: $\widehat{\beta}(\tau)$, shaded area presents NBER recession indicators

Figure 7: Partial Derivative, SHAP and $\widehat{\beta}(\tau)$ for ASV model.



(a) ASV with Ridge penalty

(b) ASV with Ridge penalty

(c) ASV without penalty

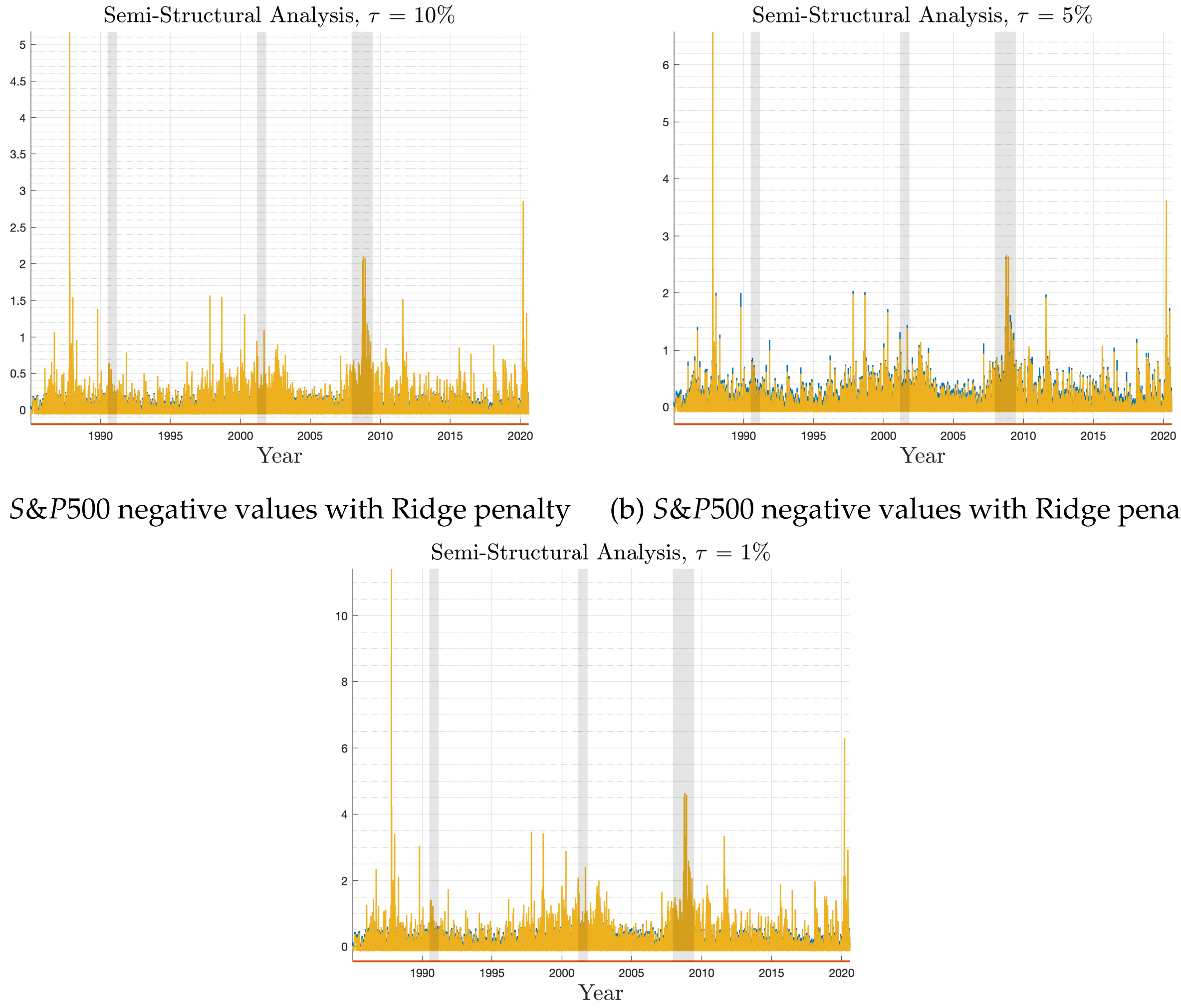
(d) $S\&P500$ positive values with Ridge penalty

(e) $S\&P500$ positive values with Ridge penalty

(f) $S\&P500$ positive values without penalty

▬:Partial Derivative, ▬: SHAP values, ▬: $\widehat{\beta}(\tau)$, shaded area presents NBER recession indicators

Figure 8: Partial Derivative, SHAP and $\widehat{\beta}(\tau)$ for ASV model.



(a) $S\&P500$ negative values with Ridge penalty (b) $S\&P500$ negative values with Ridge penalty

(c) $S\&P500$ negative values without penalty

—:Partial Derivative, —: SHAP values, —: $\widehat{\beta}(\tau)$, shaded area presents NBER recession indicators