TI 2026-097/III
Tinbergen Institute Discussion Paper

Economic Tail Risk Dynamics: Using Panel Data and Double-Pooling

Sicco Kooiker [1]
Janneke van Brummelen [2]
Julia Schaumburg [3]

1 Vrije Universiteit Amsterdam
2 Vrije Universiteit Amsterdam
3 Vrije Universiteit Amsterdam

Predicting Inflation Using Observation-Driven Neural Network Ensembles

Sicco Kooiker[(a)*] *Janneke van Brummelen*[(a,b)] *Julia Schaumburg*[(a,b)]
[(a)] Vrije Universiteit Amsterdam [(b)] Tinbergen Institute

November 2026

Abstract

We propose an observation-driven neural network framework in which the parameters change over time using an updating scheme based on past forecast errors. The framework combines the flexibility of neural networks with observation-driven parameter dynamics, allowing for online learning without repeated re-estimation. To address sensitivity to hyperparameter choices, we combine neural networks with different configurations using weights estimated by recursive least squares. We derive sufficient conditions for invertibility of a restricted neural network filter and establish consistency and asymptotic forecast optimality of the recursively estimated combination weights. We apply the framework to multi-horizon U.S. inflation forecasting using predictors from FRED-MD and evaluate its performance against a broad set of machine learning benchmarks. Our method improves out-of-sample forecast accuracy relative to static neural networks and competing methods. Additional analyses examine the importance of hyperparameter configurations and macroeconomic predictors over time.

Keywords: time-varying neural networks, observation-driven dynamics, ensembling, inflation prediction.

JEL Classification: C45, E37, C22.

*Email address corresponding author: s.h.kooiker@vu.nl (Sicco Kooiker). Schaumburg and Kooiker thank the Dutch Science Foundation (NWO, grant VI.VIDI.191.169) for financial support. We thank the Dutch National supercomputer Snellius at SURF and the Ada cluster (formerly Bazis) at the Vrije Universiteit Amsterdam for providing computational resources.

1 Introduction

Forecasting macroeconomic variables is challenging because the relationships between predictors and outcomes may be nonlinear and change over time. This is particularly relevant for inflation, where parameter instability is well documented (Stock and Watson, 1996, 2007). Recent evidence suggests that machine learning methods can improve inflation and other macroeconomic forecasts, particularly in data-rich environments (Medeiros et al., 2021; Naghi et al., 2024; Giannone et al., 2021; Babii et al., 2022; Goulet Coulombe et al., 2022; Goulet Coulombe, 2024, 2025). Most machine learning forecasting models in this literature are, however, estimated as static mappings that adapt only through re-estimation on rolling or expanding windows. At the same time, time-varying parameter models have been effective in macroeconomic prediction (Primiceri, 2005; D'Agostino et al., 2013; Koop and Korobilis, 2012). The score-driven framework of Creal et al. (2013) and Harvey (2013) provides an observation-driven approach in which parameters are updated recursively as new observations arrive. Applications have demonstrated the good forecasting performance of score-driven models for inflation (Delle Monache and Petrella, 2017; de Castro and Aiube, 2023) and macroeconomic downside risk (Delle Monache et al., 2024).

This paper proposes a framework for adaptive machine learning for macroeconomic forecasting based on observation-driven neural networks (ODNNs). Neural networks are natural candidates because they can approximate complex nonlinear functions and accommodate high-dimensional predictors (Cybenko, 1989; Hornik et al., 1989; Barron, 1993). In our framework, the weights and biases of a neural network evolve over time using an observation-driven update based on the gradient of a loss function, following the general approach of Creal et al. (2024). Unlike standard neural network training, which processes the full dataset in repeated passes, our approach operates online: parameters are updated sequentially as each new observation arrives. We consider both a restricted specification in which only the output layer parameters are time-varying and a more flexible specification in which all neural network parameters are updated using Adam (Adaptive Moment Estimation) (Kingma and Ba, 2015).

Neural networks are notoriously hard to optimize and require careful selection of hyperparameters and initialization. As Kalfa et al. (2025) document, the out-of-sample forecasting performance of machine learning models, neural networks in particular, depends critically on hyperparameter choices, which are typically selected using a validation sample or fixed in advance. In macroeconomic applications, where the number of available observations is often already limited, splitting the data into training and validation samples further reduces the information available for estimation. We therefore propose an ensemble approach that includes hyperparameter learning in the estimation procedure. Instead of selecting a single configuration, we combine the forecasts of a large pool of ODNNs with different hyperparameters and estimate their combination weights recursively using least squares. This approach builds on the forecast combination literature (Bates and Granger, 1969; Timmermann, 2006; Raftery et al., 2010; Koop and Korobilis, 2012; Billio et al., 2013; Del Negro et al., 2016; Stock and Watson, 2004) but is different as both the candidate neural network parameters and the estimated combination weights are updated sequentially with the arrival of new observations. This means that hyperparameter configurations are evaluated on an expanding set of out-of-sample forecasts, without requiring a separate validation sample or repeated re-estimation.

We establish two key theoretical properties of the proposed framework. First, following Straumann and Mikosch (2006) and Blasques et al. (2018), we derive sufficient conditions for invertibility of the observation-driven filter of a restricted ODNN specification with fixed hidden layers. Second, we establish consistency of the recursively estimated combination weights and show that, under suitable regularity conditions, the ensemble asymptotically achieves the forecast accuracy of the optimal linear combination of candidates. For practical implementation, we consider both restricted and unrestricted ODNN specifications as well as a lower-dimensional

parametrization of the hyperparameter weights, which improves computational efficiency.

Our empirical application considers U.S. inflation forecasting using predictors from FRED-MD. We evaluate multi-horizon forecasts over the most recent 15 years and assess performance with relative RMSE and MAE against a random walk benchmark, complemented by model confidence set procedures to evaluate statistical significance. We compare our approach with a selection of the best performing machine learning forecast methods from Naghi et al. (2024), including tree-based, kernel-based, and penalized methods. Our implementation is computationally fast: for each forecast horizon, an ensemble with 10,000 ODNN candidates produces 170 forecasts within 10 seconds of GPU time, despite comprising close to 200 million parameters. Our method outperforms competing benchmarks in out-of-sample forecasting and is the only method that enters every model confidence set for CPI inflation. To provide more insight into our results, we also analyze the dispersion in candidate performance, the allocation of the combination weights, and the importance of different hyperparameter configurations. Finally, we use Shapley values to examine how the contributions of different groups of macroeconomic predictors to the forecasts evolve over time.

The remainder of the paper is organized as follows. Section 2 introduces the proposed framework and Section 3 establishes its theoretical properties. Section 4 discusses the implementation, and Section 5 presents the empirical inflation forecasting application. Section 6 analyzes the candidate pool, including the dispersion in performance, the allocation and inference of the hyperparameter weights, hyperparameter importance and the distance to the oracle combination, and Section 7 studies the predictor contributions. Section 8 concludes.

2 Model

2.1 Forecasting problem

We state the forecasting problem as follows.

$$\pi_{t+h} = \mathcal{G}_{t,h}(x_t) + \varepsilon_{t,h}, \qquad t = 1, \dots, T, \quad h = 1, \dots, H, \tag{1}$$

where π_{t+h} is the endogenous variable at horizon h, $x_t \in \mathbb{R}^n$ is a high-dimensional vector of stationary predictors, i.e. $n \in \mathbb{N}$ may be large, $\mathcal{G}_{t,h}(\cdot)$ is a potentially time-varying nonlinear function and $\varepsilon_{t,h}$ is an error term. We denote by $\mathcal{D}_t = \{\pi_s, x_s\}_{s \le t}$ the information set available at time t and impose that $\mathcal{G}_{t,h}(\cdot)$ is deterministic conditional on $\mathcal{D}_t$. The error has mean zero given the information set, $\mathbb{E}[\varepsilon_{t,h} \mid \mathcal{D}_t] = 0$, such that $\mathbb{E}[\pi_{t+h}|\mathcal{D}_t] = \mathcal{G}_{t,h}(x_t)$, and $\varepsilon_{t,h}$ is assumed to have a finite variance. We do not assume that the errors are independent over time, as for $h > 1$, the forecast errors may be serially correlated.

As our core model, we define $\mathcal{G}_{t,h}(\cdot)$ as a time-varying mixture of time-varying neural network candidates. The forecast of π_{t+h} at time t is

$$\mathcal{G}_{t,h}(x_t) = \sum_{k=1}^{K} w_{k,t,h}\, f(x_t; \theta_{k,t,h}, \eta_k), \tag{2}$$

where $f(\cdot; \theta, \eta)$ is the neural network forecast function and $K \in \mathbb{N}$ is the number of candidate neural networks. Candidate k uses time-varying neural network parameters $\theta_{k,t,h}$ and fixed hyperparameters η_k. The weight $w_{k,t,h}$ denotes the combination weight assigned to candidate k at time t for forecast horizon h. Since each candidate corresponds to one hyperparameter configuration, we refer to these as hyperparameter weights. The parameters and the weights are functions of $\mathcal{D}_t$ only, so the forecast is available in real time.

2.2 Observation-driven neural networks

We specify the forecast function $f(\cdot;\, \theta_{t,h}, \eta)$[1] as a feedforward neural network with time-varying parameters $\theta_{t,h}$ and an architecture determined by the hyperparameters η. The network consists of d hidden layers with widths $n_1, \ldots, n_d$. Layer outputs are defined recursively as

$$a_{i,t,h} = \rho(W_{i,t,h}\, a_{i-1,t,h} + b_{i,t,h})\,, \quad i = 1, \ldots, d, \tag{3}$$

with $a_{0,t,h} = \tilde{x}_t$, where $\tilde{x}_t$ is the predictor vector x_t standardised to zero mean and unit standard deviation using the in-sample mean and standard deviation of each predictor. Here $a_{i,t,h} \in \mathbb{R}^{n_i}$, and $W_{i,t,h} \in \mathbb{R}^{n_i \times n_{i-1}}$ and $b_{i,t,h} \in \mathbb{R}^{n_i}$ are the weight matrix and bias vector of layer i at time t for horizon h, with $n_0 = n$, and $\rho(\cdot)$ is a nonlinear activation function applied elementwise. The output layer produces a scalar forecast $f(x_t;\, \theta_{t,h}, \eta) = W_{d+1,t,h} a_{d,t,h} + b_{d+1,t,h}$, with $W_{d+1,t,h} \in \mathbb{R}^{1 \times n_d}$ and $b_{d+1,t,h} \in \mathbb{R}$. The full parameter vector stacks all weights and biases at time t for horizon h layer by layer,

$$\theta_{t,h} = \big(\mathrm{vec}(W_{1,t,h})^\top,\; b_{1,t,h}^\top,\; \ldots,\; \mathrm{vec}(W_{d+1,t,h})^\top,\; b_{d+1,t,h}\big)^\top \in \mathbb{R}^p, \tag{4}$$

where $\mathrm{vec}(\cdot)$ stacks the columns of a matrix and $p = \sum_{i=1}^{d+1} n_i(n_{i-1} + 1)$ with $n_{d+1} = 1$. The flattened weights of the output layer followed by the bias term of the output layer form the last elements of $\theta_{t,h}$.

For a parameter vector θ, the error of the ODNN on the outcome π_t and the predictor vector x_{t-h} is $e_{t,h}(\theta) := \pi_t - f(x_{t-h}; \theta, \eta)$. We write the loss explicitly as a function of the outcome, predictors, and parameters, $\ell(\pi_t, x_{t-h}; \theta) = \ell_{\mathrm{err}}(e_{t,h}(\theta))$, where ℓ_{err} is the loss function on the forecast error. The conditional distribution of π_{t+h} given $\mathcal{D}_t$ may be time-varying, and therefore so may the parameter vector that minimizes the conditional expected loss. The goal is to track the unknown time-varying optimal parameter vector $\theta^*_{t,h} = \arg\min_\theta \mathbb{E}[\ell(\pi_{t+h}, x_t; \theta) \,|\, \mathcal{D}_t]$.

We update the parameters recursively as new observations arrive. At time t, the latest observed outcome π_t is paired with x_{t-h}, the predictor vector used to forecast π_t at horizon h. We call a pair realized when both the outcome and the paired predictor vector are observable at time t. Each realized pair enters the update once, in chronological order. The general observation-driven update of the parameters takes the form

$$\theta_{t,h} = B\, \theta_{t-1,h} - A\, S_{t,h}\, g_{t,h}, \qquad g_{t,h} = \nabla_\theta \ell(\pi_t, x_{t-h}; \theta)|_{\theta = \theta_{t-1,h}}\,. \tag{5}$$

Here $g_{t,h} \in \mathbb{R}^p$ is the gradient of the loss on the latest realized pair. The learning-rate matrix $A \in \mathbb{R}^{p \times p}$ determines the size of the update, and the scaling matrix $S_{t,h} \in \mathbb{R}^{p \times p}$ rescales the gradient, and may depend on current and past realized pairs $\{(x_{s-h}, \pi_s)\}_{s=h}^{t}$. The persistence matrix $B \in \mathbb{R}^{p \times p}$ determines how much of the previous parameter estimate carries forward. At each $t > h$, we first update the parameters with the pair (x_{t-h}, π_t) and then forecast π_{t+h} with $f(x_t; \theta_{t,h}, \eta)$. Note that for $h > 1$, we intentionally compute the gradient in the update in Equation (5) using regressors and neural network weights with different time indices, $(x_{t-h}, \theta_{t-1,h})$, whereas the candidate forecasts use $(x_t, \theta_{t,h})$. We lag the regressors to ensure that there is no look-ahead bias.

The recursion in Equation (5) has the form of the score-driven update of Creal et al. (2013) and Harvey (2013), with the negative gradient of the loss in place of the score. Since in our framework the loss does not need to be a log-likelihood, the recursion fits the observation-driven framework of Creal et al. (2024), which allows updates based on general moment conditions rather than only the score. Score-driven updates typically also contain an intercept. These intercepts need to be estimated by optimizing over a training sample, which requires repeated passes over the data. This is computationally demanding. We leave it out, which does not cause

[1] For notational convenience, we drop the subscript k here and reintroduce it in Section 2.3.

problems because the gradient term makes up for this by pulling the parameters towards values that fit the data.

The forecast performance of the proposed framework depends on the configuration of its hyperparameters. In the observation-driven literature, static parameters of the filter dynamics are typically estimated by minimizing a criterion function with gradient-based optimizers, e.g. Creal et al. (2024). This is not feasible here, because several hyperparameters, such as the architecture and the activation function, are discrete, and the criterion is highly non-convex in the continuous ones.

In the neural network literature, the standard approach is to search over configurations and select the one that performs best on a held-out validation sample, for example with Optuna (Akiba et al., 2019). This approach has two problems in a time series setting. First, it is computationally intensive, since every configuration is retrained from scratch. In rolling-window exercises, the configuration is therefore often fixed early, the grid is kept small, or the hyperparameters are chosen ad hoc (Kalfa et al., 2025). Second, splitting the data creates a trade-off between noisier parameter estimates and noisier comparisons between configurations, and a short validation sample is easily overfitted (Cawley and Talbot, 2010). Because the validation sample follows the training sample in time, configurations are also compared using models that are never trained on the most recent observations. For these reasons, we weight hyperparameter configurations rather than select one.

2.3 Hyperparameter weights

Each ODNN candidate is specified by a hyperparameter configuration η_k. We randomly draw K distinct hyperparameter configurations. Each configuration η_k defines one candidate ODNN and remains fixed over time. At each time t, candidate k produces the point forecast $f_{k,t,h} := f(x_t; \theta_{k,t,h}, \eta_k)$ for horizon h. We stack the forecasts of all candidates in the vector $\mathbf{f}_{t,h} = (f_{1,t,h}, \dots, f_{K,t,h})^\top \in \mathbb{R}^K$ and the hyperparameter weights of Equation (2) in $\hat{\mathbf{w}}_{t,h} \in \mathbb{R}^K$. We estimate the weights with a recursive least squares (RLS) filter.

The filter targets the best linear combination of the candidate forecasts. We define the optimal hyperparameter weights as the weights that minimize the expected squared forecast error,

$$\mathbf{w}_h^* = \arg\min_{\mathbf{w}\in\mathbb{R}^K} \mathbb{E}\Big[\big(\pi_{t+h} - \mathbf{f}_{t,h}^\top \mathbf{w}\big)^2\Big]. \tag{6}$$

The squared error loss function in combination with the linear relationship of the weights at the candidates allows for an analytical estimate of the minimization problem. This solution can be expressed as a recursive filter.

To estimate $\mathbf{w}_h^*$ recursively, we use the following state-space model ,

$$\pi_{t+h} = \mathbf{f}_{t,h}^\top \mathbf{w}_{t,h} + \varepsilon_{t,h}, \tag{7}$$

$$\mathbf{w}_{t,h} = \mathbf{w}_{t-1,h} + \boldsymbol{\zeta}_{t,h}, \qquad \boldsymbol{\zeta}_{t,h} \sim (\mathbf{0}, \mathbf{Q}), \tag{8}$$

where $\mathbf{w}_{t,h} \in \mathbb{R}^K$ is the state vector of weights. We set $\mathbf{Q} = \mathbf{0}$, so the state is constant over time and equal to $\mathbf{w}_h^*$, and only its estimate changes as new observations arrive. The state-space representation allows us to define a recursive filter for Equations (7) and (8). The filter has updating equations similar to a Kalman filter.

At any given time point, we can only update the weights based on the available pair of forecast and target at that time point. At time t, the latest realized pair is $(\mathbf{f}_{t-h,h}, \pi_t)$, which is the observation of Equation (7) with forecast origin $t-h$. We call a pair realized when both the target and the paired forecasts are observable at time t. For example, the pair $(\mathbf{f}_{t,h}, \pi_{t+h})$ is not yet realized at time t, because π_{t+h} is still unknown. Each pair enters the recursion h periods after its forecast origin, but the filter still updates every period. For $h = 1$ this is the usual one-step update, and for $h > 1$ the forecasts enter the recursion h steps after they are

produced. The timing is the same as for the candidates in Section 2.2. For $t \leq h$ the filter stays at its initial values $\hat{\mathbf{w}}_{0,h}$ and $P_{0,h}$. At each $t > h$, we update with the latest realized pair,

$$\boldsymbol{\kappa}_{t,h} = \frac{P_{t-1,h}\,\mathbf{f}_{t-h,h}}{\sigma_h^2 + \mathbf{f}_{t-h,h}^\top P_{t-1,h}\,\mathbf{f}_{t-h,h}}, \tag{9}$$

$$\hat{\mathbf{w}}_{t,h} = \hat{\mathbf{w}}_{t-1,h} + \boldsymbol{\kappa}_{t,h}\big(\pi_t - \mathbf{f}_{t-h,h}^\top \hat{\mathbf{w}}_{t-1,h}\big), \tag{10}$$

$$P_{t,h} = \big(I - \boldsymbol{\kappa}_{t,h}\mathbf{f}_{t-h,h}^\top\big)P_{t-1,h}, \tag{11}$$

where $\boldsymbol{\kappa}_{t,h} \in \mathbb{R}^K$ denotes the Kalman gain, which determines how the residual updates each weight, $\hat{\mathbf{w}}_{t,h}$ denotes the filtered value of $\mathbf{w}_{t,h}$, and $P_{t,h} \in \mathbb{R}^{K\times K}$ denotes the conditional variance of $\mathbf{w}_{t,h}$. We initialize $\hat{\mathbf{w}}_{0,h}$ and $P_{0,h}$ with an in-sample ridge regression. Section 3.2 shows that $P_{0,h}^{-1}$ acts as a ridge penalty that shrinks the weights towards $\hat{\mathbf{w}}_{0,h}$. Since the gain depends only on the ratio $P_{t-1,h}/\sigma_h^2$, jointly rescaling $P_{0,h}$ and σ_h^2 leaves the weight path unchanged. We therefore normalize the filter by setting $\sigma_h^2 = 1$ and let $P_{0,h}$ determine the scale. By doing this, the Kalman filter reduces to an RLS. After each update, the published forecast is $\hat{\pi}_{t+h|t} = \mathbf{f}_{t,h}^\top \hat{\mathbf{w}}_{t,h}$. We use the published forecast for out-of-sample comparison. Note that for $h > 1$, the update in Equation (10) pairs candidate forecasts and weights with different time indices, $\mathbf{f}_{t-h,h}^\top \hat{\mathbf{w}}_{t-1,h}$, whereas the published forecast uses $\mathbf{f}_{t,h}^\top \hat{\mathbf{w}}_{t,h}$. As in the ODNN update, lagging the inputs by h periods ensures no look-ahead bias.

The algorithm we propose can be interpreted as a forecast combination framework, see for example also Bates and Granger (1969); Granger and Ramanathan (1984); de Menezes et al. (2000); Diebold (1988). Opposed to what some of these references suggest, we do not restrict the weights to be positive or to sum to one and neither do we include an intercept.[2] Recursive estimation of forecast combination weights is considered by Sessions and Chatterjee (1989). We differ from them by assuming static underlying weights and using a different initialization.

The proposed RLS algorithm addresses the limitations mentioned in Section 2.2. Rather than selecting a single configuration, we generate candidate models over a large grid of hyperparameter configurations and combine their forecasts. RLS can be interpreted as an expanding window regression of the target on the candidate forecasts. This reformulates the hyperparameter selection problem as a convex optimization problem where the initialization of $P_{0,h}$ allows us to controls the risk of overfitting. The method is particularly well suited to time series settings when combined with our specification of filtered candidates. Candidate models are updated recursively as new observations arrive, which avoids repeated re-estimation on rolling windows. The performance of each candidate is evaluated on the expanding history of its forecasts and the corresponding results. Each new observation is first used to evaluate the previously generated forecasts and then to update both the candidate models and the hyperparameter weights. As a result, every observation contributes to both training and hyperparameter validation, without the need for separate training and validation subsamples.

Restricted hyperparameter weights With many candidates, RLS estimates one weight per candidate. We propose an adaptation of the RLS algorithm that reduces the number of estimated weight parameters by representing each weight assigned to candidate k as a linear function of a static feature vector $\mathbf{z}_k \in \mathbb{R}^q$ with coefficient vector $\boldsymbol{\psi}_{t,h} \in \mathbb{R}^q$,

$$w_{k,t,h} \;=\; \mathbf{z}_k^\top \boldsymbol{\psi}_{t,h}. \tag{12}$$

[2]We ran additional benchmark tests to compare this method with adaptations that impose these restrictions and include an intercept, but these adaptations did not improve forecasting performance.

Substituting this into (7) gives

$$\pi_{t+h} \;=\; \boldsymbol{\phi}_{t,h}^{\top}\boldsymbol{\psi}_{t,h} + \varepsilon_{t,h}, \qquad \boldsymbol{\phi}_{t,h} \;=\; \sum_{k=1}^{K} \mathbf{z}_k \, f_{k,t,h}, \tag{13}$$

where $\boldsymbol{\phi}_{t,h} \in \mathbb{R}^q$ is the vector of feature forecasts, and $\boldsymbol{\psi}_{t,h} \in \mathbb{R}^q$ is a static unknown coefficient vector following a random walk with zero state covariance similar to (8). We obtain its estimate $\hat{\boldsymbol{\psi}}_{t,h}$ from the completed pairs $(\boldsymbol{\phi}_{s,h}, \pi_{s+h})$ for $s \leq t-h$, using the same recursion as RLS. The forecast at t is then $\boldsymbol{\phi}_{t,h}^{\top}\hat{\boldsymbol{\psi}}_{t,h}$.

In our application, the feature vector $\mathbf{z}_k$ describes the hyperparameter configuration of candidate k. Candidates with similar hyperparameters then receive similar weights, and $\boldsymbol{\psi}_{t,h}$ measures how each feature shifts the weight of a candidate. We refer to this method as RLS-hyp. In Section 3.2, we show that RLS-hyp requires milder assumptions to deliver consistent weights than RLS and in Section 4.6, we show that RLS-hyp can be computationally much faster than RLS when the number of candidates is large compared to the feature vector.

3 Theoretical Properties

The framework of Section 2 updates the candidate ODNN parameters with observation-driven filters and estimates the hyperparameter weights with RLS on the resulting forecasts. In this section, we establish two key properties of this framework. First, we derive sufficient conditions for invertibility of the ODNN filter, ensuring that its time-varying parameters converge to a unique stationary and ergodic process that is independent of its initialization. We establish this result for a restricted filter with fixed hidden layers. Second, we show that, for candidates whose forecasts converge to stationary and ergodic limit forecasts, the recursively estimated hyperparameter weights are consistent and the ensemble asymptotically achieves the forecast accuracy of the optimal linear combination of candidates. We then extend these results to the restricted hyperparameter weights.

3.1 Invertibility of the filter

First, we study the invertibility of the ODNN filter. Following Straumann and Mikosch (2006) and Blasques et al. (2018), we call the filter invertible if it asymptotically forgets its initialization. For any initialization, under additional stationarity and ergodicity assumptions on the data, the filtered time-varying parameters then converge to a unique stationary and ergodic path that does not depend on the initialization. This property can be used to establish the limit stationarity of the candidate forecasts, which is used to establish consistency of the hyperparameter weight estimates in Section 3.2.

We provide this theoretical justification for a restricted version of the filter in Equation (5), in which the hidden-layer parameters $\{W_{i,t,h}, b_{i,t,h}\}_{i=1}^{d}$ stay fixed and only the output-layer weights and bias vary over time. We also set $S_{t,h} = I$. These simplifying restrictions keep the analysis tractable by avoiding identification issues and complex interactions among the many time-varying parameters of the unrestricted neural network. We provide conditions that allow for multiple hidden layers and for both bounded and unbounded continuous activation functions.

We start by defining the neural network with fixed hidden layers as a restricted version of Equation (3). We denote the restricted parameter vector by $\theta_{t,h}^{R}$. It has the same dimension as $\theta_{t,h}$ but restricted dynamics. Specifically, we keep the hidden-layer parameters constant over

time, such that $\theta^R_{t,h} = (\theta^{c\top}_h, \tilde{\theta}^\top_{t,h})^\top$, where

$$\theta^c_h := \big(\mathrm{vec}(W_{1,h})^\top,\, b^\top_{1,h},\, \ldots,\, \mathrm{vec}(W_{d,h})^\top,\, b^\top_{d,h}\big)^\top, \qquad \tilde{\theta}_{t,h} := \big(\mathrm{vec}(W_{d+1,t,h})^\top,\, b_{d+1,t,h}\big)^\top \in \mathbb{R}^{n_d+1},$$

so that only the output-layer parameters $\tilde{\theta}_{t,h}$ vary over time. Since θ^c_h is constant, the last hidden layer of Equation (3) is a fixed map of the standardized predictors, $a_{d,t,h} = \tau_h(\tilde{x}_t)$ with $\tau_h : \mathbb{R}^n \to \mathbb{R}^{n_d}$, and depends on t only through $\tilde{x}_t$. To analyze the resulting network, we use that it is linear in the output-layer parameters $\tilde{\theta}_{t,h}$. The network can be represented as

$$f(x_t; \theta^R_{t,h}, \eta) = \tilde{\theta}^\top_{t,h} \bar{a}_{t,h},$$

where $\bar{a}_{t,h} = (\tau_h(\tilde{x}_t)^\top, 1)^\top$ is determined by θ^c_h and the inputs $\tilde{x}_t$. We define the filter dynamics only for $\tilde{\theta}_{t,h}$, and we write $\tilde{A}$ and $\tilde{B}$ for the corresponding entries of A and B. We take $\tilde{A} = \alpha I$ with $\alpha > 0$ and $\tilde{B} = \gamma I$ with $\gamma \in (0, 1)$ such that all output-layer parameters share one step size and persistence parameter.

We let the loss be a differentiable function of the forecast error e and denote its derivative with respect to the forecast error by $\psi(e)$. On the output layer, the forecast error on the latest realized pair is $e_{t,h}(\tilde{\theta}) = \pi_t - \tilde{\theta}^\top \bar{a}_{t-h,h}$. We have that $\partial e_{t,h}(\tilde{\theta})/\partial \tilde{\theta} = -\bar{a}_{t-h,h}$, such that the gradient with respect to the output layer is

$$g_{t,h} = -\psi\big(e_{t,h}(\tilde{\theta}_{t-1,h})\big)\, \bar{a}_{t-h,h}.$$

This gives the filter updating equation,

$$\tilde{\theta}_{t,h} = \gamma\, \tilde{\theta}_{t-1,h} + \alpha\, \psi\big(e_{t,h}(\tilde{\theta}_{t-1,h})\big)\, \bar{a}_{t-h,h}. \tag{14}$$

The state of the filter is $\tilde{\theta}_{t,h}$.

Assumption 3.1:

(i) *The data $\{(\pi_t, \tilde{x}_t)\}_{t\in\mathbb{Z}}$ are stationary and ergodic, and $\mathbb{E}\log^+ |\pi_t| < \infty$.*

(ii) *The last hidden layer is bounded, i.e. $\|\bar{a}_{t,h}\|_2 \le V$ almost surely for all t, for some constant $V < \infty$.*

(iii) *The derivative of the loss is nondecreasing and 1-Lipschitz, i.e. $0 \le \psi(e) - \psi(e^\dagger) \le e - e^\dagger$ for all $e \ge e^\dagger$.*

(iv) *The persistence and the step size parameters γ and α are such that $\gamma \in (0, 1)$ and $\alpha V^2 < 1 + \gamma$.*

Proposition 3.1: *Let Assumption 3.1 hold and let the activation function be continuous. Then the filter in Equation (14) is invertible. There is a unique stationary and ergodic solution $\{\tilde{\theta}^{\mathrm{s}}_{t,h}\}_{t\in\mathbb{Z}}$, which is a measurable function of $\{(\pi_s, x_{s-h})\}_{s\le t}$, and a contraction factor $\varphi < 1$ such that for any initialization $\tilde{\theta}_{0,h}$,*

$$\|\tilde{\theta}_{t,h} - \tilde{\theta}^{\mathrm{s}}_{t,h}\|_2 \le \varphi^t\, \|\tilde{\theta}_{0,h} - \tilde{\theta}^{\mathrm{s}}_{0,h}\|_2.$$

Also, the forecasts $\tilde{\theta}^{\mathrm{s}\top}_{t,h}\bar{a}_{t,h}$ are stationary and ergodic, and for any initialization $\tilde{\theta}_{0,h}$

$$\big|\tilde{\theta}^\top_{t,h}\bar{a}_{t,h} - \tilde{\theta}^{\mathrm{s}\top}_{t,h}\bar{a}_{t,h}\big| \le V\, \varphi^t\, \|\tilde{\theta}_{0,h} - \tilde{\theta}^{\mathrm{s}}_{0,h}\|_2.$$

The proof of Proposition 3.1 is in Appendix A. The assumptions give sufficient conditions under which the filter is invertible. Hyperparameter configurations that satisfy these conditions result

in invertible filters. We therefore discuss what each assumption implies for hyperparameter values that lead to invertible filters.

Assumption 3.1 (i) is a standard assumption necessary to show that the filter converges to a stationary and ergodic sequence. The logarithmic moment of the target is a weak condition, which holds whenever the target has a bounded moment of arbitrarily small order. Lemmas A.1 and A.2 in Appendix A.1 show that Assumption 3.1 (ii) is satisfied for bounded activation functions, such as Tanh, Sigmoid, ReLU6 and Leaky ReLU6, and for unbounded continuous activation functions with compact input, such as ReLU and Leaky ReLU. For bounded activation functions, the assumption holds for any predictors. The bound V then only depends on the bound of the activation function and the width of the last layer. For unbounded continuous activation functions, we need the input predictors of the first layer of the neural network to lie in a compact set. For ReLU and Leaky ReLU, predictors that are clipped at $\pm c$ satisfy the assumption. Alternatively, we can clip the ReLU and Leaky ReLU functions themselves, which gives the bounded ReLU6 and Leaky ReLU6 functions.

To show there is a contraction between two filtered paths, we use Assumption 3.1 (iv). This condition states that, for a given persistence γ, the squared bound V^2 determines how large the step size α can be for the filter to remain invertible. For Tanh and Sigmoid, the condition reduces to $(n_d + 1)\alpha < 1 + \gamma$, so a wider last layer requires a smaller step size. For ReLU6 and Leaky ReLU6, it reduces to $(36\, n_d + 1)\alpha < 1 + \gamma$. For ReLU and Leaky ReLU, the clipping value c and the size of the hidden-layer weights and biases determine V, and therefore the largest admissible step size. Assumption 3.1 (iii) requires a convex loss with a 1-Lipschitz derivative. Examples of loss functions that satisfy this condition are the Huber loss with threshold $\delta > 0$, for which $\psi(e) = \max\{-\delta, \min\{e, \delta\}\}$, and half the squared error loss, for which $\psi(e) = e$.

We have established invertibility for the restricted filter. For the unrestricted filter, the hidden-layer weights are also time-varying. The forecast is a nonlinear function of the time-varying parameters through the activation functions, whereas in the restricted filter the forecast is linear in the time-varying output-layer parameters. Invertibility requires a contraction for a filter in which the parameters of each layer enter the updates of the other layers nonlinearly, which is hard to show for neural networks. The Adam scaling adds to this, because it depends on past gradients and therefore on the path of the filter. With fixed hidden layers and $S_{t,h} = I$, the contraction only depends on the bound on the last hidden layer, the step size and the persistence, which gives the condition in Assumption 3.1 (iv).

In Section 3.2, we combine the forecasts of many candidates. The result there applies to candidate filters that produce forecasts that converge exponentially fast almost surely (e.a.s.) to stationary and ergodic limit forecasts with finite second moments. We stack the forecasts of the candidates in $\mathbf{f}_{t,h}$ and their limit forecasts in $\mathbf{f}^{\mathrm{s}}_{t,h}$. The following corollary shows that restricted filters satisfy these requirements.

COROLLARY 3.1: *Let every candidate be a restricted filter that satisfies the conditions of Proposition 3.1, Then* $\|\mathbf{f}_{t,h} - \mathbf{f}^{\mathrm{s}}_{t,h}\|_2$ *converges to zero e.a.s., the limit forecasts* $\mathbf{f}^{\mathrm{s}}_{t,h}$ *and the target* π_{t+h} *are jointly stationary and ergodic. If additionally* $\mathbb{E}[\pi_t^2] < \infty$*, then* $\mathbb{E}\|\mathbf{f}^{\mathrm{s}}_{t,h}\|_2^2 < \infty$.

The convergence and the stationarity follow from Proposition 3.1. For the moments, we follow Blasques et al. (2022, Technical Appendix, Proposition TA.3), who derive the moments of the limit of the filter from a moment condition on the update and a uniform contraction condition for a scalar time-varying parameter. We derive the same result for a vector of time-varying parameters. We obtain the moment condition on the update from the finite second moment of the target and the boundedness of the output of the hidden layers in Assumption 3.1 (ii). Further details on the proof of Corollary 3.1 can be found in Appendix A.4.

3.2 Consistency and optimality of the hyperparameter weights

We show that the RLS filter described in Section 2.3 has an equivalent ridge regression representation and use this representation to show that, under Assumption 3.2, the hyperparameter weights converge to the optimal weights in Equation (6). The proofs are in Appendix B.

Unwinding the recursion gives the ridge representation. Let $\mathbf{F}_{1:t-h,h} \in \mathbb{R}^{(t-h)\times K}$ denote the matrix with rows $\mathbf{f}_{s,h}^\top$ for $s = 1, \dots, t-h$, and let $\boldsymbol{\pi}_{1+h:t}$ collect the corresponding realized targets π_{s+h}.

LEMMA 3.1: *Let $P_{0,h}$ be positive definite. Then, for every $t > h$ the filter in Equations (9) to (11) satisfies*

$$\hat{\mathbf{w}}_{t,h} = \left(P_{0,h}^{-1} + \mathbf{F}_{1:t-h,h}^\top \mathbf{F}_{1:t-h,h}\right)^{-1} \left(P_{0,h}^{-1}\hat{\mathbf{w}}_{0,h} + \mathbf{F}_{1:t-h,h}^\top \boldsymbol{\pi}_{1+h:t}\right). \tag{15}$$

The estimate $\hat{\mathbf{w}}_{t,h}$ is the unique minimizer over $\mathbf{w} \in \mathbb{R}^K$ of the penalized least squares criterion

$$\sum_{s=1}^{t-h} \left(\pi_{s+h} - \mathbf{f}_{s,h}^\top \mathbf{w}\right)^2 + \left(\mathbf{w} - \hat{\mathbf{w}}_{0,h}\right)^\top P_{0,h}^{-1} \left(\mathbf{w} - \hat{\mathbf{w}}_{0,h}\right). \tag{16}$$

Lemma 3.1 shows that RLS is a ridge regression on an expanding sample with a fixed penalty. The ridge penalty enters through the initialization and stays in the equation due to $\mathbf{Q} = \mathbf{0}$. The penalty matrix is $P_{0,h}^{-1}$, and it shrinks the weights towards $\hat{\mathbf{w}}_{0,h}$.

To show consistency, we hold K and h fixed and let the number of observations T grow. Since the candidate forecasts start from an arbitrary initialization, they are not stationary, but they converge to stationary sequences. We state our conditions on these limits. These conditions do not require the candidates to be the restricted filters discussed in Section 3.1, allowing any candidates whose forecasts converge e.a.s. to stationary and ergodic limit forecasts with finite second moments.

ASSUMPTION 3.2: *There are limit forecasts $\mathbf{f}_{t,h}^{\mathrm{s}} = (f_{1,t,h}^{\mathrm{s}}, \dots, f_{K,t,h}^{\mathrm{s}})^\top$ such that*

(i) the limit forecasts $\mathbf{f}_{t,h}^{\mathrm{s}}$ and the target π_{t+h} are jointly stationary and ergodic and have finite second moments,

(ii) the gap $\|\mathbf{f}_{t,h} - \mathbf{f}_{t,h}^{\mathrm{s}}\|_2$ between the forecasts and the limit forecasts converges to zero e.a.s. as $t \to \infty$,

(iii) the matrix $M_h := \mathbb{E}\left[\mathbf{f}_{t,h}^{\mathrm{s}} \mathbf{f}_{t,h}^{\mathrm{s}\top}\right]$ is positive definite.

Restricted filters that satisfy the conditions of Proposition 3.1 meet Assumption 3.2 (i) and (ii) when the target has a finite second moment, as Corollary 3.1 shows. We define the optimal weights in Equation (6) with the limit forecasts, so that the expectation does not depend on t. Under Assumption 3.2, the minimization has the unique solution

$$\mathbf{w}_h^* = M_h^{-1}\, \mathbb{E}\left[\mathbf{f}_{t,h}^{\mathrm{s}} \pi_{t+h}\right]. \tag{17}$$

PROPOSITION 3.2: *Let Assumption 3.2 hold and let $P_{0,h}$ be positive definite. Then, for fixed K and h, the hyperparameter weights of the filter in Equations (9) to (11) satisfy*

$$\hat{\mathbf{w}}_{t,h} \to \mathbf{w}_h^* \quad \textit{almost surely as } t \to \infty.$$

The filter forgets its initialization and the ridge penalty, and it converges to the best linear combination of the candidate forecasts. Consistency does not require correct specification, and it does not require the forecast errors to be independent. Assumption 3.2 (i) and (iii) contain

the two conditions of a least squares regression, finite second moments and a rank condition, while Assumption 3.2 (ii) allows us to replace the filtered forecasts by their limits.

The proof of Proposition 3.2 follows steps similar to the consistency argument of Blasques et al. (2022). The proof has three steps. First, the sample moments of the limit forecasts converge by the ergodic theorem. Second, the exponential convergence in Assumption 3.2 (ii) carries these limits over to the filtered forecasts. Third, we substitute these limits into the ridge representation of Lemma 3.1. Because the hyperparameters of the candidates are fixed and the RLS weights have a closed form, pointwise convergence is enough and we do not need a compactness argument.

Consistency of the weights also implies that the combined forecast is asymptotically as accurate as the best linear combination of the candidates. Let $\mathcal{L}_h(\mathbf{w}) := \mathbb{E}\big[(\pi_{t+h} - \mathbf{f}_{t,h}^{s\top}\mathbf{w})^2\big]$ denote the expected squared forecast error of a combination with weights $\mathbf{w}$, which $\mathbf{w}_h^*$ minimizes.

COROLLARY 3.2: *Let the conditions of Proposition 3.2 hold. Then, for fixed K and h, the average squared forecast error of the combined forecast satisfies*

$$\frac{1}{T}\sum_{s=1}^{T}\big(\pi_{s+h} - \mathbf{f}_{s,h}^{\top}\hat{\mathbf{w}}_{s,h}\big)^2 \to \mathcal{L}_h(\mathbf{w}_h^*) \quad \textit{almost surely as } T \to \infty.$$

In particular, $\mathcal{L}_h(\mathbf{w}_h^*) \leq \mathbb{E}\big[(\pi_{t+h} - f_{k,t,h}^{s})^2\big]$ for every candidate k.

Corollary 3.2 shows that the weights are also optimal in terms of squared forecast error. The forecast at time s only uses target and paired stacked candidate forecast vector that are realized at time s, so the average squared error is already an out-of-sample measure. The best combination is at least as accurate as each of the candidates individually, because weight one on one candidate and zero on the others is an admissible choice of $\mathbf{w}$, and $\mathbf{w}_h^*$ minimizes $\mathcal{L}_h$. By Corollary 3.2, the combined forecast is therefore asymptotically at least as accurate as the best single hyperparameter configuration. The proof is in Appendix B.

The restricted hyperparameter weights of Section 2.3 have the same properties as the hyperparameter weights when they satisfy Assumption 3.2. We show that for the restrictions we propose in Section 2.3 these assumptions are satisfied. Let $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_K)^{\top} \in \mathbb{R}^{K\times q}$ denote the feature matrix. The feature forecasts of Equation (13) are then $\boldsymbol{\phi}_{t,h} = \mathbf{Z}^{\top}\mathbf{f}_{t,h}$, and the coefficient estimate $\hat{\boldsymbol{\psi}}_{t,h}$ implies the weights $\mathbf{Z}\hat{\boldsymbol{\psi}}_{t,h}$ on the candidates. RLS-hyp applies the filter in Equations (9) to (11) to the feature forecasts, with initial values $\hat{\boldsymbol{\psi}}_{0,h}$ and $\tilde{P}_{0,h}$. When $\mathbf{Z}^{\top}M_h\mathbf{Z}$ is positive definite, the expected squared forecast error over the restricted weights has the unique minimizer

$$\boldsymbol{\psi}_h^* = \arg\min_{\boldsymbol{\psi}\in\mathbb{R}^q} \mathcal{L}_h(\mathbf{Z}\boldsymbol{\psi}) = \big(\mathbf{Z}^{\top}M_h\mathbf{Z}\big)^{-1}\mathbf{Z}^{\top}\,\mathbb{E}\big[\mathbf{f}_{t,h}^{s}\pi_{t+h}\big]. \tag{18}$$

COROLLARY 3.3: *Let Assumption 3.2 (i) and (ii) hold, and let $\mathbf{Z}^{\top}M_h\mathbf{Z}$ and $\tilde{P}_{0,h}$ be positive definite. Then, for fixed K, q and h, the coefficients of RLS-hyp satisfy:*

(i) $\hat{\boldsymbol{\psi}}_{t,h} \to \boldsymbol{\psi}_h^*$ *almost surely as* $t \to \infty$

(ii)

$$\frac{1}{T}\sum_{s=1}^{T}\big(\pi_{s+h} - \boldsymbol{\phi}_{s,h}^{\top}\hat{\boldsymbol{\psi}}_{s,h}\big)^2 \to \mathcal{L}_h(\mathbf{Z}\boldsymbol{\psi}_h^*) \quad \textit{almost surely as } T \to \infty.$$

(iii) the function $\mathcal{L}_h$ has a minimizer over $\mathbb{R}^K$, and every such minimizer $\mathbf{w}_h^*$ satisfies

$$\mathcal{L}_h(\mathbf{Z}\boldsymbol{\psi}_h^*) = \mathcal{L}_h(\mathbf{w}_h^*) + \big(\mathbf{w}_h^* - \mathbf{Z}\boldsymbol{\psi}_h^*\big)^{\top}M_h\big(\mathbf{w}_h^* - \mathbf{Z}\boldsymbol{\psi}_h^*\big), \tag{19}$$

where the second term is zero if and only if $\mathcal{L}_h$ has a minimizer of the form $\mathbf{Z}\boldsymbol{\psi}$ with $\boldsymbol{\psi} \in \mathbb{R}^q$.

(iv) if, in addition, $\frac{1}{\sqrt{T}} \sum_{s=1}^{T} \boldsymbol{\phi}^{s}_{s,h} r_{s,h}$ converges in distribution to $\mathcal{N}(\mathbf{0}, \Omega_h)$ as $T \to \infty$, where $r_{s,h} = \pi_{s+h} - \boldsymbol{\phi}^{s\top}_{s,h} \boldsymbol{\psi}^*_h$ is the projection error and Ω_h is positive definite, then

$$\sqrt{T}\big(\hat{\boldsymbol{\psi}}_{T+h,h} - \boldsymbol{\psi}^*_h\big) \to \mathcal{N}(\mathbf{0}, \Sigma_h), \qquad \Sigma_h = \big(\mathbf{Z}^\top M_h \mathbf{Z}\big)^{-1} \Omega_h \big(\mathbf{Z}^\top M_h \mathbf{Z}\big)^{-1},$$

in distribution as $T \to \infty$.

Corollary 3.3 shows that RLS-hyp converges to the best combination among the weights that are linear in the hyperparameter features. The corollary does not require Assumption 3.2 (iii). Its rank condition is about the $q \times q$ matrix $\mathbf{Z}^\top M_h \mathbf{Z}$ rather than the $K \times K$ matrix M_h, and can therefore hold when some candidate forecasts are linear combinations of others. The price of the restriction is the second term in Equation (19), which measures the distance between the optimal weights and the best restricted weights. RLS-hyp is asymptotically as accurate as RLS when the optimal weights are a linear function of the features. Unlike in Corollary 3.2, the restricted combination is not guaranteed to be at least as accurate as each candidate. The weight of one on a single candidate is in general not of the form $\mathbf{Z}\boldsymbol{\psi}$. The proof is in Appendix B.

Part (iv) of Corollary 3.3 allows inference on the coefficients of RLS-hyp. When the restricted combination is correctly specified, the product of the limit feature forecasts and the projection error is correlated only up to lag $h-1$, and the central limit theorem follows from stationarity, ergodicity and finite second moments (Hansen, 1982). Without correct specification, it holds when the limit forecasts and the target are near-epoch dependent on a mixing process and have sufficiently many moments (Davidson, 1994). In both cases, we use the estimator of Newey and West (1987) to consitenlty estimate the long-run variance.

As discussed in Section 3.1, invertibility of the unrestricted filter requires a different approach. The consistency results do not rely on this restriction. Proposition 3.2 and Corollary 3.2 hold for any candidate that satisfies Assumptions 3.2 (i) and (ii). They extend to these filters once their invertibility is established.

The consistency results do rely on the rank condition in Assumption 3.2 (iii). With many candidates, some limit forecasts may be close to linear combinations of others. The RLS weights are then not unique in the limit, but the combined forecast can still converge to the optimal forecast. A unique limit can be obtained by letting the ridge penalty grow with the sample size, which in the RLS filter requires reinitializing $P_{0,h}$ (Moroshko et al., 2015; Lai and Bernstein, 2024).

Finally, we treat the number of candidates as fixed, whereas the application uses far more candidates than observations. Kelly et al. (2024) and He (2024) show that ridge regression can retain out-of-sample predictive power when the number of regressors grows as fast as or faster than the sample size. These results provide further motivation for combining a large number of candidates, and a formal treatment of K growing with T is outside the scope of this research.

4 Implementation

Implementing the algorithm requires a choice of candidates and their filters, which we specify in this section. We consider two versions of the candidate filter. The first version uses the restricted filter in Equation (14), which falls within the assumptions of Section 3, and we refer to it as R-ODNN. The second version uses an unrestricted version of the filter in Equation (5), and we refer to it as U-ODNN. The U-ODNN updates all parameters of the network. Because it also updates the hidden-layer parameters, the invertibility results of Section 3.1 do not apply directly. The consistency of the hyperparameter weights in Section 3.2 still applies to the U-ODNN model, as long as the forecasts converge to stationary and ergodic limits. We include the U-ODNN to

demonstrate how common methods used in neural network modeling and optimization carry over to our framework, despite not having worked out the theory for this model. In particular, we show how the popular Adam optimization algorithm translates to an ODNN filter. We also discuss how we implement the RLS and RLS-hyp filters and the computational advantages of both.

4.1 Loss function

For both implementations and all candidates we use the same class of loss function. For the loss function ℓ_{err} on the forecast error in Section 2.2, we use the Huber loss, $\ell_{\text{err}} = \ell^{\text{Huber}}$, such that the loss on the latest realized pair is

$$\ell(\pi_t, x_{t-h}; \theta) = \ell^{\text{Huber}}\big(e_{t,h}(\theta)\big). \tag{20}$$

For $\delta > 0$, the Huber loss (Huber, 1964) of a scalar forecast error e is

$$\ell^{\text{Huber}}(e) = \begin{cases} \frac{1}{2}e^2 & \text{if } |e| \le \delta, \\ \delta\,|e| - \frac{1}{2}\delta^2 & \text{otherwise,} \end{cases} \tag{21}$$

where the threshold δ controls the transition between the quadratic and the linear part. For $\delta \to \infty$, the Huber loss converges to half the squared error. For small δ, the Huber loss is approximately δ times the absolute error. We only consider $\delta > 0$, since the absolute error has a discontinuous derivative at zero and therefore violates Assumption 3.1 (iii).

We choose the Huber loss because macroeconomic time series such as inflation sometimes contain large shocks, to which the squared error loss is very sensitive, as the gradient is proportional to the forecast error. Instead, for the Huber loss, beyond δ, the derivative of the loss with respect to the forecast is bounded by δ, which makes the updates more robust to large shocks.

We set the threshold δ as the q-th quantile of the absolute target $|\pi_t|$ over the period of in-sample observations before the start of the out-of-sample evaluation period, with $q \in (0, 1)$, where q is treated as a hyperparameter. Defining δ through a quantile makes the hyperparameter independent of the scale of the data.

4.2 Filter specification

Restricted filter The R-ODNN uses the restricted filter in Equation (14). The filter only updates the output-layer parameters $\tilde{\theta}_{t,h}$, while the hidden-layer parameters θ_h^c stay fixed at the estimates of Section 4.4. It uses unit scaling, $S_{t,h} = I$, with one step size α and one persistence parameter γ for all output-layer parameters, including the output bias. In the neural network literature, gradient updates with unit scaling based on a single observation are referred to as stochastic gradient descent (SGD). An important difference is that our filter passes through the data only once, while SGD typically passes through the same data multiple times.

Proposition 3.1 only covers the R-ODNN for hyperparameters that satisfy Assumption 3.1, and we restrict the candidates accordingly. The Huber loss with $\delta > 0$ satisfies Assumption 3.1 (iii). Assumption 3.1 (iv) requires $\gamma \in (0, 1)$ and $\alpha V^2 < 1 + \gamma$. The persistence of the R-ODNN is $\gamma = 1 - \lambda_{\text{wd}}\alpha$, so we only consider a strictly positive weight decay, $\lambda_{\text{wd}} > 0$. In our grid $\lambda_{\text{wd}}\alpha \le 5 \cdot 10^{-4}$, such that $\gamma \in (0, 1)$ holds for every R-ODNN candidate. To make the condition on the step size easy to verify, we only consider bounded activation functions.

Unrestricted filter For the U-ODNN, we specify the filter of Equation (5) with techniques that are common in neural network optimization. The neural network literature offers many alternatives to SGD that do apply scaling. One of the most popular is Adam (Kingma and Ba,

2015), which forms the scaling matrix from exponentially weighted moving averages (EWMA) of the gradient and of its squared elements. We use this scaling in our filter by keeping track of two moment estimates

$$m_{t,h} = \beta_1\, m_{t-1,h} + (1-\beta_1)\, g_{t,h}, \qquad v_{t,h} = \beta_2\, v_{t-1,h} + (1-\beta_2)\, g_{t,h}^2, \tag{22}$$

where $\beta_1, \beta_2 \in [0, 1)$ are decay rates and $g_{t,h}^2 = g_{t,h} \odot g_{t,h}$ is the elementwise squared gradient. Since both averages start at zero, they are biased towards zero in the first periods of the filter. Adam corrects this bias with $\hat{m}_{t,h} = m_{t,h}/(1-\beta_1^{t-h})$ and $\hat{v}_{t,h} = v_{t,h}/(1-\beta_2^{t-h})$.

Adam fits Equation (5) with $S_{t,h} = \text{diag}\big(1/(\sqrt{\hat{v}_{t,h}}+\epsilon_A)\big)$, and with the gradient $g_{t,h}$ replaced by $\hat{m}_{t,h}$. We set $A = \text{diag}(\alpha, \ldots, \alpha, \alpha_b)$. Here α_b is a separate step size for the output bias $b_{d+1,t,h}$, which we order last in $\theta_{t,h}$, and α applies to all other parameters. A separate step size allows the level of the forecast, the bias of the output layer, to adapt at a different speed than the network weights, which are more likely to vary at a common rate than the level. This gives the update

$$\theta_{t,h} = B\, \theta_{t-1,h} - A\, \frac{\hat{m}_{t,h}}{\sqrt{\hat{v}_{t,h}} + \epsilon_A}, \tag{23}$$

where $\epsilon_A > 0$ is a small constant for numerical stability, and the square root and the division apply elementwise.

Adam offers two advantages over SGD in our setting. First, $\hat{m}_{t,h}$ smooths the gradient over recent observations. This can be beneficial, because SGD can sometimes exhibit oscillating behavior by overshooting its optimum. In Adam, gradients that oscillate between updates partly cancel each other in the average, such that the resulting path of $\theta_{t,h}$ is more stable.[3] Second, the scaling matrix $S_{t,h}$ gives each parameter its own effective step size. Parameters that receive large gradients accumulate a large $\hat{v}_{t,h}$, which reduces their effective step size. The decay rates β_1 and β_2 set the adaptation speed of both moment estimates, and they are treated as hyperparameters.

The scaling in Adam has a counterpart in score-driven models, where the scaling matrix is typically based on the conditional information matrix (Creal et al., 2013). For neural network optimization, second-order derivative methods are undesirable because second-order derivatives scale quadratically with the number of parameters and neural networks are known for having many parameters. Adam has the advantage of using only first-order information, through an EWMA of the diagonal elements of the gradient outer product.

4.3 Regularization

Machine learning models, and in particular neural networks, have a tendency to overfit the data due to their flexibility. That is, in-sample performance does not translate into out-of-sample performance. Observation-driven filters have the advantage that they are self-regularizing.

The filter regularizes itself in two ways. First, the persistence matrix can be specified as $B = I - \lambda_{\text{wd}} A$, with hyperparameter $\lambda_{\text{wd}} \geq 0$. For $\lambda_{\text{wd}} > 0$, the diagonal elements of B lie between zero and one. This makes the parameters decay towards zero when the gradients equal zero. For $\lambda_{\text{wd}} = 0$, the parameters have no tendency to decay towards zero. In the deep learning literature, this persistence is known as decoupled weight decay (Loshchilov and Hutter, 2019). Second, the learning-rate matrix A acts as a regularizer. Lange et al. (2026) directly associate the inverse of the learning-rate matrix in score-driven filters with a penalized

[3] Adam does not guarantee that an update reduces the loss on the current observation, since a single gradient step can overshoot the minimum. We also considered the stochastic proximal point algorithm of Asi and Duchi (2019), which replaces the gradient step by an implicit update. This approach has also been proposed for score-driven models as the implicit score-driven filter of Lange et al. (2026). The proximal update guards against overshooting, but it requires solving an inner optimization problem at every observation. In our application, this did not translate into improvements in forecasting performance.

optimization problem, where smaller values of A imply slower adaptation to new information and therefore stronger penalization.

The R-ODNN only uses part of these techniques. Its persistence parameter is $\gamma = 1 - \lambda_{\text{wd}}\alpha$, and the restricted filter of Section 3.1 requires $\gamma \in (0, 1)$. The R-ODNN therefore needs a strictly positive weight decay with $\lambda_{\text{wd}}\alpha < 1$, since without weight decay the persistence equals one and Proposition 3.1 does not apply. The step size also keeps its role as a regularizer in the R-ODNN, although we restrict it to be equal for all filtered parameters, as in the restricted filter of Equation (14). The contraction argument of Lemma A.3 relies on this single step size.

Besides the self-regularization of the filter, we also add the ℓ_1 or Lasso penalty (Tibshirani, 1996) to the loss of the U-ODNN in Equation (20), which becomes $\ell^{\text{Huber}}\big(e_{t,h}(\theta)\big) + \lambda_1\|\theta\|_1$, where $\lambda_1 \geq 0$ sets its strength and $\|\theta\|_1$ sums the absolute values of all weights and biases. The penalty adds $\lambda_1 \operatorname{sign}(\theta)$ to the gradient, elementwise for all non-zero parameters, and thus pulls each parameter towards zero. Due to the functional form of the penalty, this form of regularization penalizes sparse solutions less than dense solutions. For the R-ODNN, we set $\lambda_1 = 0$, since the derivative of the ℓ_1 penalty is discontinuous at zero, which violates Assumption 3.1 (iii). In practice, the discontinuity occurs at a single point, and it is unlikely that the gradient is evaluated exactly at this point. The penalty therefore works well in practice for the U-ODNN.

4.4 Initialization

The filter starts from $\theta_{0,h}$, which we draw once at the start of the sample. We set all biases to zero, $b_{i,0,h} = 0$, and draw each element of the weight matrix $W_{i,0,h}$ independently from $N(0, s_0^2/n_{i-1})$ for $i = 1, \ldots, d+1$, where $s_0 > 0$ is the initialization scale. Scaling the standard deviation by $1/\sqrt{n_{i-1}}$ keeps the variance of the layer outputs of similar size across layers of different widths (LeCun et al., 1998). The hyperparameter s_0 rescales this standard choice.

Especially for the U-ODNN, the starting value matters more in our setting than in standard neural network training. With a single pass over the data, the filter takes far fewer steps than training over many epochs. With a small step size α, the effect of $\theta_{0,h}$ can therefore last longer. A small s_0 starts the network close to a flat function. The early forecasts are then close to zero, and the gradients with respect to the weights are small. A large s_0 produces large initial forecasts and large gradients, which can lead to unstable updates. The initialization scale therefore determines, in combination with the step size α, how long the filter needs to move away from its starting value.

The R-ODNN requires values for the hidden-layer parameters θ_h^c before the restricted filter can start. We obtain them by full-batch optimization on the first half of the in-sample period. Starting from the draw $\theta_{0,h}$, we minimize the average loss in Equation (20) over this subsample with respect to all parameters of the network. We then fix the hidden layers at their estimates and start the restricted filter at the midpoint of the in-sample period, with the output-layer parameters initialized at their batch estimates. From the midpoint onward, the R-ODNN only updates the output layer with every new observation.

This sample split has a cost. The R-ODNN does not produce candidate forecasts for the first half of the in-sample period, so its hyperparameter weights are estimated on fewer observations than those of the U-ODNN. We cannot use the R-ODNN forecasts on the first half, since the hidden layers are estimated on these observations. These forecasts contain future information, which would distort the RLS weights in favor of candidates that overfit the first half. The batch step also makes the R-ODNN more costly than the U-ODNN, because it requires repeated passes over the data for every candidate. In exchange for the repeated passes, the sample split aligns the implementation with the theory.

For the RLS filter, we have to specify the initialization. We split the training sample into ten blocks of equal size and run a blocked cross-validation over a grid of penalty values. To avoid serial dependence between the training and validation blocks, we remove the first $h-1$ forecasts of each validation block. We initialize the RLS filter at the weight vector and ridge

covariance matrix of the penalty with the lowest cross-validation loss. For RLS-hyp, we first specify the feature vector $\mathbf{z}_k$, a fixed representation of the hyperparameters of candidate k. It contains the numerical hyperparameter values, one-hot indicators of the activation function and the hidden-layer architecture, and an intercept. We drop one reference category per group of indicators, so that the indicators are not collinear with the intercept. We rescale each column of the feature matrix $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_K)^\top$ to unit ℓ_2-norm. We initialize RLS-hyp in the same way as RLS.

4.5 Hyperparameters and candidate draws

The ODNN model introduced in Section 2.2 requires choosing a set of hyperparameters that decide on the network architecture and the observation-driven dynamics. Table 1 lists all hyperparameters together with the values of the grid for the U-ODNN and the R-ODNN. The hyperparameters include the typical hyperparameters considered in the literature on neural network applications in macroeconomics (Kalfa et al., 2025). We do not consider dropout or batch sizes larger than one, since these do not directly fit our filtered setup.

TABLE 1: Hyperparameter grid for the candidate ODNNs. The third column gives the values for the U-ODNN and the fourth column the values for the R-ODNN, where "same" means that the values are unchanged.

Symbol	Description	U-ODNN values	R-ODNN values
$\mathbf{n}$	Hidden layer widths	(8), $(16, 8)$, $(32, 16, 8)$, $(64, 32, 16, 8)$	same
ρ	Activation function[a]	ReLU6, Leaky ReLU6, Tanh, Sigmoid	same
s_0	Weight initialization scale	$\{0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$	same
q	Huber loss quantile	$\{0, 0.75, 0.9, 0.95, 0.98, 0.99\}$	$\{0.75, 0.9, 0.95, 0.98, 0.99\}$
α	Step size (learning rate)	$\{10^{-5}, 5{\cdot}10^{-5}, 10^{-4}, 5{\cdot}10^{-4}, 10^{-3}, 5{\cdot}10^{-3}\}$	same
α_b	Step size, terminal bias[b]	$\{10^{-5}, 5{\cdot}10^{-5}, 10^{-4}, 5{\cdot}10^{-4}, 10^{-3}, 5{\cdot}10^{-3}\}$	$\alpha_b = \alpha$
β_1	First-moment decay rate	$\{0.0, 0.5, 0.9, 0.95\}$	not used
β_2	Second-moment decay rate	$\{0.95, 0.98, 0.99\}$	not used
λ_{wd}	Weight decay (AdamW)	$\{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$	$\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$
λ_1	ℓ_1 penalty coefficient	$\{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$	0

[a] ReLU6 is the ReLU function clipped at 6, and Leaky ReLU6 is the Leaky ReLU function clipped at -6 and 6.
[b] Separate step size applied to the bias of the output layer; all other parameters use α.

4.6 Computational complexity

Using filtering instead of classic estimation makes our method fast, and this speed is what allows a large candidate pool. We discuss the computational cost of the candidates and of RLS. We show that for large candidate pools the computational cost of estimating the hyperparameter weights becomes nontrivial. We show that RLS estimates the weights faster than a ridge regression that is re-estimated at every forecast origin, also when the ridge regression is solved in its so-called dual form. The algorithm can be made even faster when RLS-hyp is used.

The cost of updating the candidates grows linearly in K, the number of candidates. Most of this cost lies in the gradient of the network output, which backpropagation computes in $\mathcal{O}(p)$ operations, where p is the number of network parameters (Bishop, 2006). The matrices A and B and the scaling matrix are diagonal, so the remaining steps of the U-ODNN and R-ODNN filters also require $\mathcal{O}(p)$ operations. Updating all candidates thus requires $\mathcal{O}(p_{\max}K)$ operations per forecast origin and $\mathcal{O}(p_{\max}KT)$ operations over the full sample, where $p_{\max}$ is the number of parameters of the largest network in the pool of architectures described in Section 4.5.

The cost of estimating the hyperparameter weights grows faster than linearly in K. A ridge regression of the realized targets on the candidate forecasts in its primal form requires the

expensive inverse of an $K \times K$ matrix. The order of computational cost becomes $\mathcal{O}(K^2t + K^3)$ operations per forecast origin (Shawe-Taylor and Cristianini, 2004). When the number of realized pairs t is smaller than K, the ridge regression can be solved in its dual form. The dual form inverts a $t \times t$ matrix instead of an $K \times K$ matrix (Saunders et al., 1998). This reduces the order of computational cost to $\mathcal{O}(t^2K + t^3)$ operations per forecast origin. For $t \ll K$, the dominating cost becomes t^2K. The dual form is then much cheaper than the primal form.

RLS avoids the large matrix inversions. Because the observation is a scalar, the inverse in Equation (9) reduces to a division by a scalar, although the state has dimension K. The product $P_{t-1,h}\mathbf{f}_{t-h,h}$ in Equation (9) and the covariance update in Equation (11) both require $\mathcal{O}(K^2)$ operations (Durbin and Koopman, 2012). RLS is therefore faster than the dual ridge regression as soon as $K \leq t^2$. Its cost per forecast origin does not grow with t. Ridge regression also requires selecting the penalty, and RLS requires selecting an initialization. Since we obtain both with a ridge regression, they have the same initial cost. If the penalty of the ridge regression is reselected at every forecast origin, this cost recurs at every forecast origin, while the initialization of RLS is computed only once.

RLS-hyp is even faster than RLS. It replaces the K-dimensional weight vector by the q-dimensional vector $\boldsymbol{\psi}_{t,h}$, with $q \ll K$. The Kalman filter stays the same, and the product $\tilde{P}_{t-1,h}\boldsymbol{\phi}_{t-h,h}$ and the covariance update require $\mathcal{O}(q^2)$ operations for each update of $\hat{\boldsymbol{\psi}}_{t,h}$ and $\tilde{P}_{t,h}$. Computing $\boldsymbol{\phi}_{t,h} = \mathbf{Z}^\top \mathbf{f}_{t,h}$ in Equation (13) adds $\mathcal{O}(Kq)$ operations. The total cost is $\mathcal{O}(Kq+q^2)$ operations per forecast origin, which can be far below the $\mathcal{O}(K^2)$ operations of RLS when $q \ll K$.

5 Forecasting U.S. Inflation

We evaluate the forecasting performance of the ODNN against a broad set of machine learning benchmarks for U.S. inflation forecasting. The evaluation follows the framework of Medeiros et al. (2021) and Naghi et al. (2024), allowing direct comparison with established results in the literature. Section 5.1 describes the data and the forecast design, and Section 5.2 introduces the benchmark methods and the alternative combination schemes. In Section 5.3, we compare the recursive least squares combinations of the unrestricted and the restricted ODNN candidates (U-ODNN-RLS and R-ODNN-RLS) with the best performing machine learning models in Naghi et al. (2024). In Section 5.4, we compare all the combination schemes described in Section 5.2 against each other. Finally, Section 5.5 reports the computation time of our method.

5.1 Data and forecast design

We consider U.S. headline inflation measured as the month-on-month log-difference of the Consumer Price Index (CPI) as the forecast target (see Appendix C.1 for results on U.S. Personal Consumption Expenditures (PCE)). Predictors are drawn from the FRED-MD database (McCracken and Ng, 2016), a panel of macroeconomic and financial indicators at monthly frequency. We apply the transformation codes recommended by McCracken and Ng (2016) except for the inflation variables where we take log differences.

The full sample runs from February 1960 through August 2025. We evaluate out-of-sample forecast performance between July 2011 and August 2025. We report results in- and excluding March 2020 till June 2022. We exclude this period, the COVID-19 pandemic, from the evaluation since the errors of the models in this period are unprecedentedly high. Including this period would give its forecast errors disproportionate weight in the summary statistics, obscuring the models' performance over the rest of the sample.

We produce direct multi-horizon forecasts at horizons $h \in \{1, \ldots, 24\}$ months. For each horizon h, a separate model is estimated. Benchmark models are estimated in an expanding window, re-estimating parameters as each new observation arrives. The candidate ODNNs are

estimated in a single pass over the sample, with parameters subsequently updated observation-by-observation. Our framework naturally allows for expanding time series.

The candidate ODNNs are specified by their hyperparameter choices. We use the grid as specified in in Table 1 and randomly draw without replacement unique hyperparameters until $K = 10{,}000$ candidates are drawn.

Occasionally, a hyperparameter combination can result in unrealistically extreme forecasts. Across the combination schemes of the U-ODNN candidates, we apply *trimming* to omit these candidates. Over the in-sample period we calculate the 1% and 99% quantiles of the target. We multiply both quantiles by three to obtain a lower and an upper bound. If an prediction of a candidate lies outside these bounds, we remove the candidate from the pool. Out-of-sample, a candidate whose prediction lies outside these bounds receives zero weight at that time. The R-ODNN candidates satisfy the conditions of Proposition 3.1, which hold given the assumption that the forecasts are stationary and do not need to be trimmed.

We compare the root mean squared error (RMSE) and mean absolute error (MAE) relative to a random walk (RW) benchmark. Ratios below one indicate improvement over the benchmark. Statistical significance of differences in forecast performance across methods is assessed using multi-horizon model confidence set (MCS) procedures (Quaedvlieg, 2021), which identify the set of models that cannot be statistically distinguished from the best performer at a given significance level. We report uniform and average MCS sets across forecast horizons 1–12 and 1–24. We separate the short and long horizons to isolate performance at each and to align with Medeiros et al. (2021) and Naghi et al. (2024), who forecast up to twelve months ahead. We use $B = 499$ bootstrap replications and a significance level of 0.10.

5.2 Benchmarks

The machine learning benchmark methods are implemented as in Naghi et al. (2024). For the implementation details, we refer to their paper. The benchmarks we consider are the unobserved components stochastic volatility (UCSV) as in Medeiros et al. (2021), the extremely randomized trees (ERT) (Geurts et al., 2006), light gradient boosting machine (LGBM) (Ke et al., 2017), random forest (RF) (Breiman, 2001), heteroskedasticity LASSO with data-driven penalty (RLASSO) (Belloni et al., 2012), adaptive LASSO (AdaLasso) (Zou, 2006). The feed-forward neural network (NN) has three layers with hidden layer widths 32, 16, and 8 similar to Medeiros et al. (2021), but we take an equally weighted average over 100 neural networks with different initializations as in Gu et al. (2020). Additionally, we add a univariate autoregressive model (AR) with automatic lag selection.

We also compare RLS with the combination schemes of Stock and Watson (2004), applied to the same candidate pool. The equal-weight combination (ODNN-Mean) and the median combination (Median) do not depend on the past performance of the candidates. The inverse-error combinations weight each candidate by the inverse of its cumulative squared error (InvMSE), following Bates and Granger (1969), or by the inverse of its cumulative absolute error (InvMAE). The recent best combinations place all weight on the candidate with the lowest cumulative squared error (RB-SE) or absolute error (RB-AE). Unlike Stock and Watson (2004), we do not discount past errors. All combination schemes therefore target fixed hyperparameter weights, in line with RLS.[4]

5.3 Forecasting results

Table 2 reports relative root mean squared error (RRMSE) with and without the COVID period for CPI inflation. Appendix C.1 reports relative mean absolute error (RMAE) for CPI inflation and both RRMSE and RMAE for PCE inflation. RRMSE is computed as model RMSE divided

[4]We also considered time-varying hyperparameter weights, but these worsened forecast performance in our application.

by random walk RMSE over the out-of-sample period. Values below one indicate improvement over the random walk.

TABLE 2: Relative RMSE by forecast horizon for CPI inflation.

	Direct horizon					Cumulative horizon				Multi-horizon MCS			
Model	$h=1$	$h=3$	$h=6$	$h=12$	$h=24$	$\Sigma_{1\text{–}3}$	$\Sigma_{1\text{–}6}$	$\Sigma_{1\text{–}12}$	$\Sigma_{1\text{–}24}$	unif.$_{12}$	avg.$_{12}$	unif.$_{24}$	avg.$_{24}$
Panel A: Full sample													
RW	0.0026	0.0034	0.0033	0.0035	0.0040	0.0079	0.0150	0.0298	0.0656	—	—	—	—
AR	0.88*	0.77*	0.80*	0.79*	0.75*	0.72*	0.64*	0.61*	0.58*	0.14†	0.01	0.14†	0.11†
U-ODNN-RLS	**0.81***	0.71*	0.73*	**0.69***	**0.67***	0.65*	0.54*	0.46*	**0.40***	**1.00†**	**1.00†**	**1.00†**	**1.00†**
R-ODNN-RLS	0.82*	0.71*	0.73*	0.71*	0.71*	0.64*	0.52*	0.47*	0.43*	0.99†	0.07	1.00†	0.11†
UCSV	0.93	0.83*	0.85*	0.89*	0.89*	0.79*	0.75*	0.75*	0.78*	0.14†	0.01	0.14†	0.07
ERT	0.88	0.72*	0.75*	0.73*	0.68*	0.67*	0.57*	0.52*	0.45*	0.14†	0.02	0.75†	0.11†
LGBM	0.81	**0.70***	0.75*	0.71*	0.69*	**0.62***	**0.52***	**0.46***	0.42*	0.99†	0.30†	1.00†	0.11†
CatBoost	0.84*	0.72*	**0.72***	0.79*	0.72*	0.65*	0.54*	0.47*	0.44*	0.46†	0.02	0.59†	0.11†
NN	0.85	0.83*	0.76*	0.70*	0.70*	0.72*	0.63*	0.53*	0.47*	0.14†	0.02	0.14†	0.11†
RF	0.83*	0.79*	0.78*	0.78*	0.72*	0.72*	0.66*	0.59*	0.56*	0.14†	0.01	0.14†	0.11†
RLasso	1.43*	0.95	0.99	0.88	1.00	1.07	1.01	0.93	0.81	0.14†	0.01	0.08	0.11†
AdaLasso	0.85*	0.75*	0.77*	0.82*	0.94	0.73*	0.58*	0.50*	0.57*	0.14†	0.02	0.08	0.11†
Panel B: COVID excluded (2020-03–2022-06)													
RW	0.0024	0.0030	0.0030	0.0030	0.0034	0.0073	0.0137	0.0276	0.0602	—	—	—	—
AR	0.84*	0.74*	0.76*	0.74	0.70*	0.68*	0.60*	0.55*	0.59*	0.08	0.03	0.09	0.00
U-ODNN-RLS	0.73*	**0.63***	**0.64***	**0.62***	**0.56***	0.55*	0.42*	**0.35***	**0.38***	**1.00†**	**1.00†**	**1.00†**	**1.00†**
R-ODNN-RLS	0.73*	0.66*	0.65*	0.63*	0.57*	**0.54***	**0.42***	0.37*	0.40*	0.70†	0.03	0.98†	0.01
UCSV	0.90	0.83*	0.84*	0.85*	0.84*	0.77*	0.74*	0.71*	0.78*	0.08	0.03	0.09	0.00
ERT	0.75*	0.65*	0.68*	0.66*	0.58*	0.55*	0.45*	0.43*	0.46*	0.22†	0.03	0.50†	0.00
LGBM	**0.73***	0.66*	0.71*	0.64*	0.58*	0.55*	0.46*	0.41*	0.43*	0.22†	0.03	0.50†	0.00
CatBoost	0.78*	0.67*	0.68*	0.73*	0.60*	0.57*	0.46*	0.39*	0.44*	0.08	0.03	0.15†	0.00
NN	0.75*	0.68*	0.71*	0.69*	0.67*	0.56*	0.48*	0.47*	0.49*	0.20†	0.03	0.09	0.00
RF	0.76*	0.66*	0.71*	0.66*	0.59*	0.58*	0.48*	0.43*	0.54*	0.20†	0.03	0.16†	0.00
RLasso	1.24	0.84	0.88	0.81	0.99	0.93	0.91	0.90	0.89	0.08	0.03	0.09	0.00
AdaLasso	0.75*	0.67*	0.73*	0.71*	1.00	0.58*	0.49*	0.40*	0.45*	0.09	0.03	0.09	0.00

Notes: The first data row of each panel gives the absolute RMSE of the random walk (RW); all other entries are relative to RW. Multi-horizon MCS p-values are computed over per-horizon squared losses (subscripts denote the maximum horizon). Bold = best and underlined = second-best competitor per column (RW excluded). Stars denote Diebold–Mariano equal-predictive-accuracy tests against the random-walk benchmark (two-sided, squared-loss), where specifically a * indicates $p < 0.05$, Dagger, †, denotes inclusion in the model confidence set.

For the full sample, the two best-performing models are U-ODNN-RLS and LGBM. Some other models are competitive or better at some horizons, but perform much worse at others. When the COVID period is excluded, the advantage of U-ODNN-RLS is larger. It obtains the lowest RRMSE at the direct horizon forecasts of 3, 6, 12, and 24 and the second-lowest RRMSE at the first horizon. It improves on the widely used random forest in RRMSE at every horizon for both samples. Some competing methods occasionally attain a lower forecast error, but no other model matches the performance of U-ODNN-RLS and LGBM across horizons. The U-ODNN-RLS significantly beats the random walk at the 5% level at every horizon in both samples. U-ODNN-RLS also delivers competitive long-horizon forecasts.

We also examine cumulative-horizon forecasts. Accumulating the direct forecasts over horizons 1–3, 1–6, 1–12 and 1–24 gives a forecast of the CPI change over three months, six months, one year and two years relative to the current month. The U-ODNN-RLS always ranks among the top four models. For the full sample, LGBM obtains the lowest RRMSE over three months, six months and one year, while U-ODNN-RLS stays close and obtains the lowest RRMSE over two years. When the COVID period is excluded, U-ODNN-RLS obtains the lowest RRMSE over one year and two years, and R-ODNN-RLS over three months and six months. Both ODNN combinations improve on the random forest at every cumulative horizon in both samples, and significantly beat the random walk at the 5% level throughout. The R-ODNN-RLS is slightly more accurate than the U-ODNN-RLS over three months and six months in both samples.

Additionally, we report multi-horizon model confidence set (MCS) p-values, using both the uniform and the average loss criteria over horizons 1 to 12 and 1 to 24. Models included in the MCS at the 10% level are marked by a dagger (†) beside the corresponding p-value. We separate the short and long horizons to isolate performance at each and to align with Medeiros et al. (2021) and Naghi et al. (2024), who forecast up to twelve months ahead. The U-ODNN-RLS obtains a p-value of one in each of the four confidence sets, in both samples, and is the only model included in every set. For the full sample, the average-loss confidence set over horizons 1–12 keeps only U-ODNN-RLS and LGBM. Over horizons 1–24, it keeps all models except UCSV, most with a p-value just above the 10% level. Under the uniform criterion, no model is excluded over horizons 1–12, and only RLasso and AdaLasso are excluded over horizons 1–24. When the COVID period is excluded, U-ODNN-RLS is the only model in both average-loss confidence sets. The uniform confidence sets also keep R-ODNN-RLS, ERT, LGBM and the random forest, together with the NN over horizons 1–12 and CatBoost over horizons 1–24.

Figure 1 plots the forecasts and the cumulative squared forecast errors minus those of the random walk of U-ODNN-RLS, R-ODNN-RLS, ERT and LGBM over the out-of-sample period. A declining curve means the model accumulates less error than the benchmark over time. U-ODNN-RLS and LGBM achieve similar cumulative error reductions relative to the random walk, with substantial gains around mid-2022. Although U-ODNN-RLS has lower cumulative error than LGBM for most of the sample, the gap does not consistently grow. The machine learning errors accumulate in a broadly similar way, but they diverge during large inflation shocks. The R-ODNN-RLS follows the U-ODNN-RLS closely over the whole period: at the first horizon the two curves never differ by more than 0.34×10^{-4}, and the U-ODNN-RLS ends slightly lower. ERT performs much worse in early 2020, when prices dropped at the onset of the pandemic. The two ODNN combinations and LGBM earn most of the advantage over the other models in mid-2022, during the large CPI increase and the subsequent drop and stabilization. The U-ODNN-RLS performs well across the full out-of-sample period and particularly during sharp changes in the inflation rate at the first horizon.

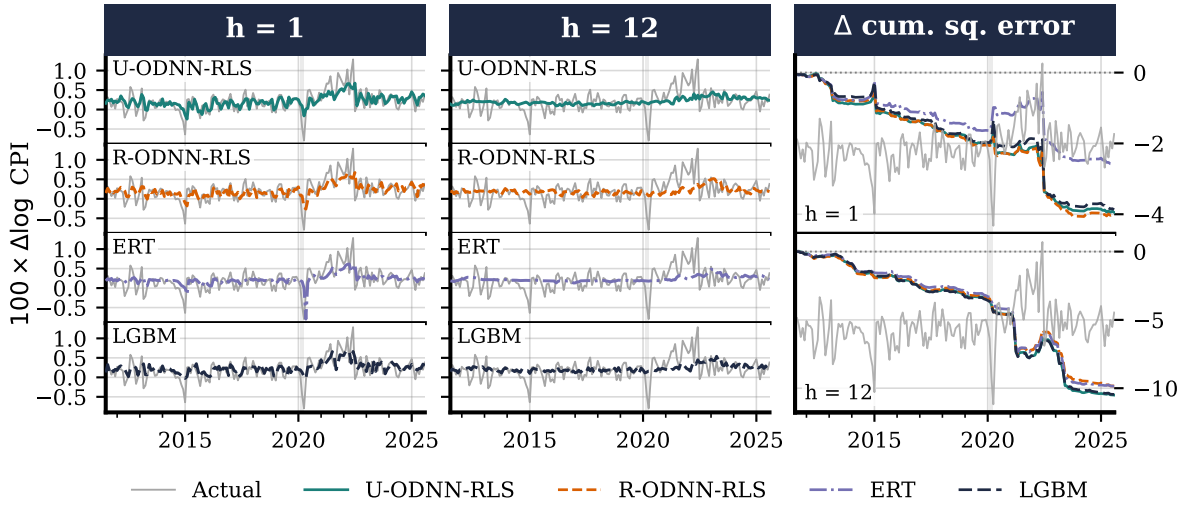

FIGURE 1: Forecasts of monthly CPI inflation and cumulative squared forecast minus that of the random walk at July 2011–August 2025, at horizons $h = 1$ and $h = 12$.

5.4 Alternative combination schemes

We have shown that the U-ODNN-RLS performs well and outperforms the equal-weight combination at the shorter horizons. To validate this choice, we compare it against five further

combination schemes from Section 5.2. We measure relative MSE and relative MAE against the random walk at horizons 1, 6, 12 and 24, together with the average across the four horizons, and we split the evaluation into two parts. Panel A of Table 3 reports in-sample performance over the final part of the in-sample period, from December 1985 to June 2011. Panel B reports the corresponding out-of-sample results.

TABLE 3: CPI forecast accuracy across horizons. Metrics are relative to the random-walk benchmark.

Method	$h = 1$		$h = 6$		$h = 12$		$h = 24$		Average	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
Panel A: Out-of-sample										
RW	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Mean	1.00	0.99	0.80	0.75	0.76	0.72	0.69	0.66	0.81	0.78
Median	1.00	1.00	0.80	0.75	0.76	0.72	0.69	0.67	0.81	0.78
RB-SE	0.95	0.92	0.77	0.74	0.76	0.73	0.69	0.70	0.79	0.77
RB-AE	0.91	0.87	0.77	0.73	0.77	0.74	0.67	0.66	0.78	0.75
InvMSE	0.97	0.97	0.80	0.75	0.76	0.72	0.69	0.66	0.80	0.77
InvMAE	0.98	0.98	0.80	0.75	0.76	0.72	0.69	0.66	0.81	0.78
RLS	**0.81**	**0.81**	**0.73**	**0.70**	0.69	**0.67**	**0.67**	**0.64**	**0.72**	**0.70**
RLS-hyp	0.83	0.84	0.73	0.71	**0.69**	0.68	0.68	0.65	0.73	0.72

Bold = best and underlined = second-best combination scheme in each column, within each panel.

The RLS is the best performing method at all four horizons, although the RLS-hyp is equally accurate in squared error at horizons 6 and 12 and the RB-AE at horizon 24. The advantage of the RLS in both squared and absolute error is most pronounced at the first horizon and more moderate at the longer horizons. The median, inverse-MSE and inverse-MAE combinations perform almost identically to the equal-weight combination, in line with the well-known difficulty of improving on a simple average. The RB-SE is less stable across horizons. It improves on the equal-weight combination at the first horizon, but has the largest absolute error at the two-year horizon. Taken together, the panel shows that recursive least squares is an effective combination scheme that generally outperforms standard benchmarks, including the equal-weight combination.

5.5 Computation time

Computation time is usually disregarded in macroeconomic forecasting, since estimating the models under consideration takes negligible time relative to the forecast horizon. With the introduction of machine learning methods, however, models have grown larger and estimation times longer, so that the computational budget now affects modelling choices and the way practitioners work with these models. Monthly updating becomes costly, and not only because of the complexity of the models themselves, they must also be re-estimated from scratch at every evaluation window. For the machine learning benchmarks we consider, training time scales linearly with the sample size. ODNN breaks this dependence, as its optimization time is fixed and does not grow with the sample. Combined with the parallel structure of the algorithm and the one-step learning setup, this results in highly efficient generation of forecasts.

Figure 2 reports compute time per forecast, expressed as a multiple of the AR(1) benchmark. The timings are based on the total training time per scalar forecast in the out-of-sample experiment for horizons 1–24. R-ODNN and U-ODNN require roughly 5 and 10 times as long as the AR(1) model to produce a forecast, whereas CatBoost requires approximately 430 times

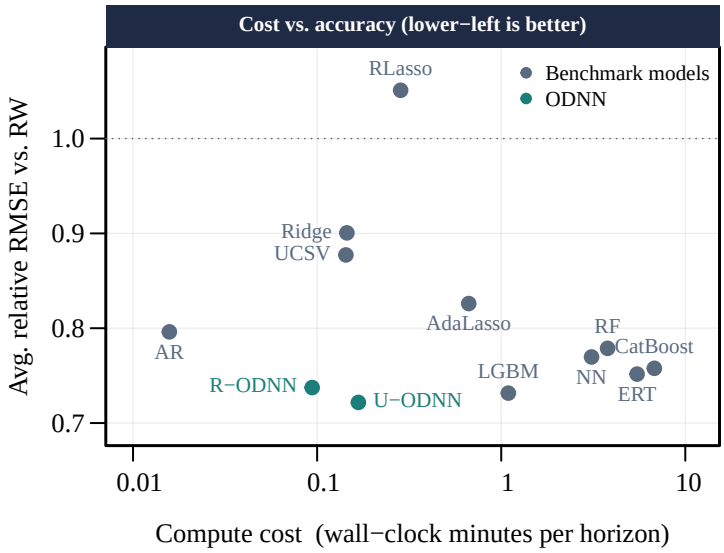

FIGURE 2: Average computation time per forecast against the relative RMSE (RW baseline), averaged over horizons $h = 1, 3, 6, 12, 24$.

as long. LGBM sits in between with around 70 times as long as the AR(1). R-ODNN is a bit faster than U-ODNN on average because for R-ODNN the trained hidden layers are shared among horizons.

It is important to note that the ODNN timings assume execution on a GPU and that the forecast can be generated sequentially. In a real-time setting, data revisions can complicate this. One may instead prefer to restart the filter from an earlier point in time before the date of the revised data. This is possible, but increases the computational cost of the filter by the number of addional filter steps. The traditional benchmarks do not have this problem, because they are re-estimated from scratch at every origin. In addition, we did not attempt to reduce the training times of the competing machine learning methods further than the implementation provided by Naghi et al. (2024). Alternative implementations could speed up the training of these benchmarks.

6 Analysis of the Candidates

We now examine how RLS uses the candidate pool and the underlying hyperparameters contribute to the forecasts. Several questions organize the analysis. First, do the hyperparameters matter, in the sense that good and bad configurations produce a meaningful spread in forecast performance, and what distinguishes a good candidate from a bad one? Second, how are the hyperparameter weights distributed across candidates over the out-of-sample period, and are they sparse or dense? Third, which hyperparameters drive the forecasts of U-ODNN-RLS? Fourth, how close does U-ODNN-RLS come to the best attainable combination, and do we need the full pool of 10,000 candidates? Because the candidates differ through their hyperparameter choices, grouping their forecasts by those choices allows us to examine how different configurations contribute to the performance of the combination.

6.1 Dispersion across candidates

The candidates vary widely in out-of-sample accuracy. Figure 3 plots the distribution of candidate RMSE relative to the random walk for the four horizons, with the random walk and

the RLS combination marked. At $h = 1$ the candidates are widely dispersed and the median candidate is worse than the random walk, at a ratio of 1.13, with more than 2,000 candidates falling beyond 1.5 times the random walk RMSE. At horizons $h = 6$, 12 and 24 the distribution shifts left and most candidates beat the random walk, with median ratios of 0.88, 0.83 and 0.75, though a tail of well over a thousand configurations remains far worse than the benchmark. The wide and horizon-dependent spread shows that the choice of hyperparameters matters for forecast accuracy. The RLS combination attains ratios of 0.81, 0.73, 0.69 and 0.67 across the four horizons and sits in the left tail of the candidate distribution throughout. At $h = 1$ only 1 of the 10,000 candidates beats the RLS combination, while between 26 and 170 beat it at the longer horizons. Many of these superior candidates also outperform every competing static machine learning method in our comparison, which shows that the observation-driven neural network can be very effective when specified with the right hyperparameters.

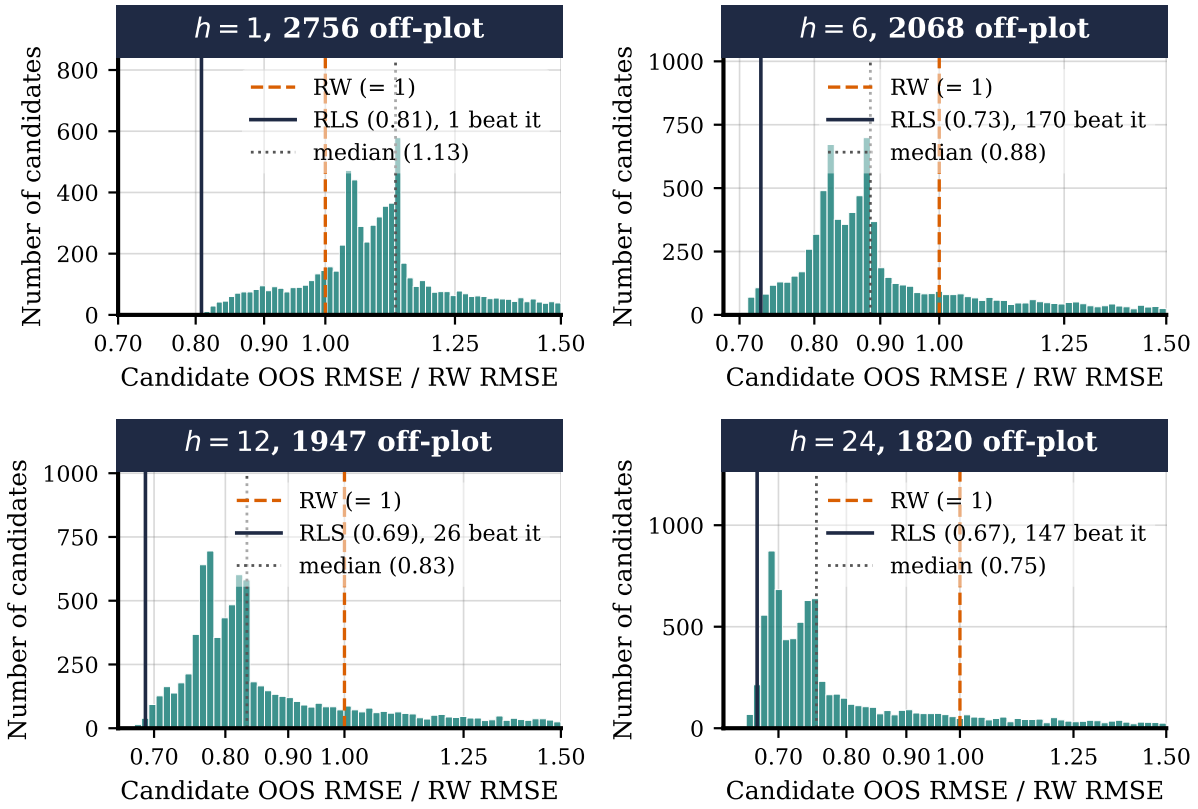

FIGURE 3: Distribution of individual candidate out-of-sample RMSE relative to the random walk benchmark, by forecast horizon. Vertical lines mark the random walk, the candidate median, and the RLS combination. Candidates beyond 1.5 times the random walk RMSE are omitted and counted in each panel title.

Figure 4 illustrates this spread with the forecast paths of one good and one bad candidate at the first horizon, shown over the first and the last ten years of the sample. The bad candidate combines the smallest learning rate in our grid with the largest initialization scale and the deepest network. Because its updates are tiny, it never recovers from its poor initialization. The large initial weights make the forecasts far too volatile: they swing much more than inflation itself, mostly above it, with extreme spikes around 2020. A candidate of this kind does not improve the forecast and should receive little weight. The good candidate combines a moderate learning rate with a small initialization scale and a compact single-layer network. It tracks inflation closely, adapting quickly to the large swings between 2020 and 2022. It captures the joint dynamics of the predictors and inflation and forecasts more accurately than any of the

machine learning methods compared in Section 5.3.

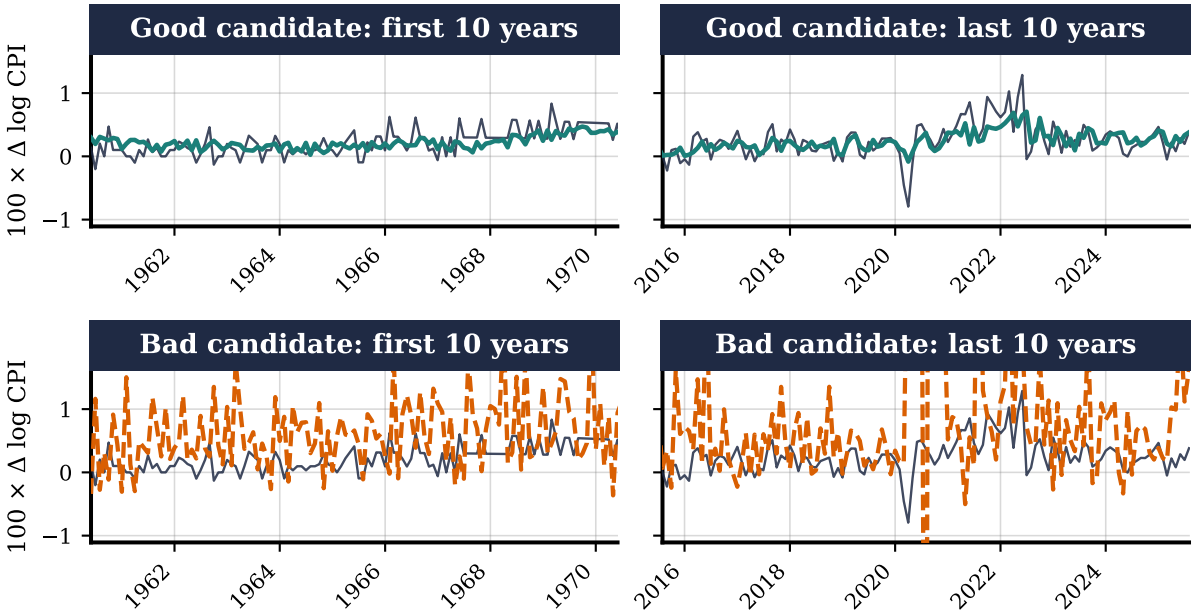

FIGURE 4: Forecast paths of a good (top) and a bad (bottom) candidate at the first horizon, against actual inflation, over the first and the last ten years of the sample. Each panel plots realized inflation (thin grey line) and the candidate's forecast (thick colored line). Values are $100 \times \Delta \log \text{CPI}$. The good candidate uses a sigmoid activation, a learning rate of 0.005, an initialization scale of 1 and a single hidden layer of eight units. The bad candidate uses a ReLU activation, a learning rate of 10^{-5}, an initialization scale of 2 and also a single hidden layer of eight units.

6.2 Allocation of weights

We next ask how RLS allocates weight across this pool. Table 4 summarizes the concentration of the weights in two ways, averaged over the out-of-sample period. The first row reports the effective number of candidates, computed as $\left(\sum_k |\hat{w}_{k,t,h}|\right)^2 / \sum_k \hat{w}_{k,t,h}^2$, which equals the number of candidates when weight is spread evenly and becomes one as weight concentrates on a single candidate. The effective number rises with the horizon, from about 2,800 candidates at $h = 1$ to roughly 2,900 at $h = 6$, 3,100 at $h = 12$ and 3,200 at $h = 24$. The one-step-ahead combination is therefore the sparsest, yet it still spreads weight over thousands of candidates. The second row ranks candidates by the absolute value of their weights and reports how many are needed to reach 90% of the total weight. This requires roughly 4,000 candidates at $h = 1$, rising to about 4,500 at $h = 24$. Thus, the estimated hyperparameter weights are densely distributed over the candidates.

TABLE 4: Concentration of the RLS hyperparameter weights, averaged over the out-of-sample period.

	$h = 1$	$h = 6$	$h = 12$	$h = 24$
Effective no. of candidates	2,825	2,889	3,112	3,188
Candidates for 90% of weight	4,029	4,147	4,388	4,533

Effective no. of candidates is $\left(\sum_k |\hat{w}_{k,t,h}|\right)^2 / \sum_k \hat{w}_{k,t,h}^2$, averaged over time. Candidates for 90% of weight is the number of candidates, ranked by $|\hat{w}|$, that hold 90% of the total absolute weight, averaged over time.

6.3 Hyperparameter importance

To analyze which hyperparameters drive the forecasts of U-ODNN-RLS, we can use the hyperparameter weights to inform us. We base our methods on those that are used in the context of explaining the relationship between regressors and inflation variables in Joseph et al. (2024). We decompose each RLS forecast into the contributions of the candidates using Shapley values (Štrumbelj and Kononenko, 2010; Lundberg and Lee, 2017). We then aggregate these contributions over the candidates that share a hyperparameter value.

For a linear model the Shapley value can be obtained easily. The RLS forecast in Equation (2) is linear in the candidate forecasts, with weights $\hat{w}_{k,t,h}$. The Shapley value of candidate k at time t is therefore

$$\mathcal{A}_{k,t,h} = \hat{w}_{k,t,h}\left(f_{k,t,h} - \bar{f}_{k,t,h}\right), \qquad \bar{f}_{k,t,h} = \frac{1}{t-1}\sum_{s=1}^{t-1} f_{k,s,h}. \tag{24}$$

Where we use trailing mean $\bar{f}_{k,t,h}$ of all forecasts of the candidate before time t as the baseline measure. The forecast decomposes exactly into an intercept and the Shapley values of the candidates,

$$\hat{\pi}_{t+h|t} = \mathcal{B}_{0,t,h} + \sum_{k=1}^{K} \mathcal{A}_{k,t,h}, \qquad \mathcal{B}_{0,t,h} = \sum_{k=1}^{K} \hat{w}_{k,t,h}\bar{f}_{k,t,h}. \tag{25}$$

The intercept $\mathcal{B}_{0,t,h}$ is the forecast of U-ODNN-RLS when every candidate forecasts its trailing mean.

We aim to explain how U-ODNN-RLS is affected by the choice of hyperparameters. For each hyperparameter j, we group the candidates into value groups $G_{j,u}$ according to its grid value u. The sum of the Shapley values within a group, $\sum_{k\in G_{j,u}} \mathcal{A}_{k,t,h}$, is the contribution of value u to the forecast. We summarize these contributions with a relative importance measure,

$$\omega_{j,u,h} = \frac{1}{\nu_{j,u}} \frac{\sum_{t=T_0}^{T}\sum_{k\in G_{j,u}} |\mathcal{A}_{k,t,h}|}{\sum_{t=T_0}^{T}\sum_{k=1}^{K} |\mathcal{A}_{k,t,h}|}, \tag{26}$$

where T_0 is the starting time of the out-of-sample period. We compute the relative importance measure only over the out-of-sample period. To obtain the relative importance, we divide the absolute importance by $\nu_{j,u} = |G_{j,u}|/K$, the frequency of each value group among all candidates. The trimmed candidates have a Shapley value equal to 0. The relative importance measure equals one if every candidate has the same non-zero absolute Shapley value. A relative importance above one indicates that U-ODNN-RLS relies more on a value group than its frequency among candidates would suggest.

Figure 5 reports the relative importance at $h = 1$ and $h = 24$. The corresponding figure for $h = 6$ and $h = 12$ is in Appendix C.2. For each of the four horizons, the activation function and the initialization scale show the largest differences in relative importance. The Tanh activation function is the most important, with a relative importance between 1.82 and 2.72. Leaky ReLU and ReLU have a relative importance close to one up to $h = 12$ and much lower importance at $h = 24$, while Sigmoid has low importance at the short horizons and gains importance at longer horizons. Both Leaky ReLU and ReLU are popular activation functions and are often preferred over Tanh and Sigmoid. However, in our model, the Tanh function is the strongest contributor to the forecasts. It is symmetric around zero and keeps the sign of its input. Moreover, the gradient of Tanh is non-zero for all inputs and changes according the size of the input, in contrast to the piecewise constant gradients of Leaky ReLU and ReLU. Therefore, the gradients of Tanh are potentially more informative for the updates in the filter in Equation (5).

Another hyperparameter that shows large differences in importance across its values is the initialization scale. For each horizon, scales larger than 1 are favored over scales smaller than

or equal to 1. LeCun et al. (1998) describe how an initialization scale that is too small, leads to small gradients and slower adaptation. An initialization scale that is too large leads to overly large gradient updates, which also slow down learning.

The λ_1 penalty shows that candidates with the largest penalty are the least important for all horizons. The architecture also shows large differences in importance, with the smallest architecture favored over the larger ones for all horizons. The learning rate, however, shows only modest differences across its values, and only at $h = 24$ is the largest learning rate clearly favored. The bias learning rate shows a more consistent picture, with the largest learning rate clearly favored for all horizons.

For the Adam moment parameters, the differences in importance are relatively small, and no combination of the moment parameters is consistently favored across horizons. The differences are also modest for the Huber quantile and the weight decay, with similar relative importance across values.

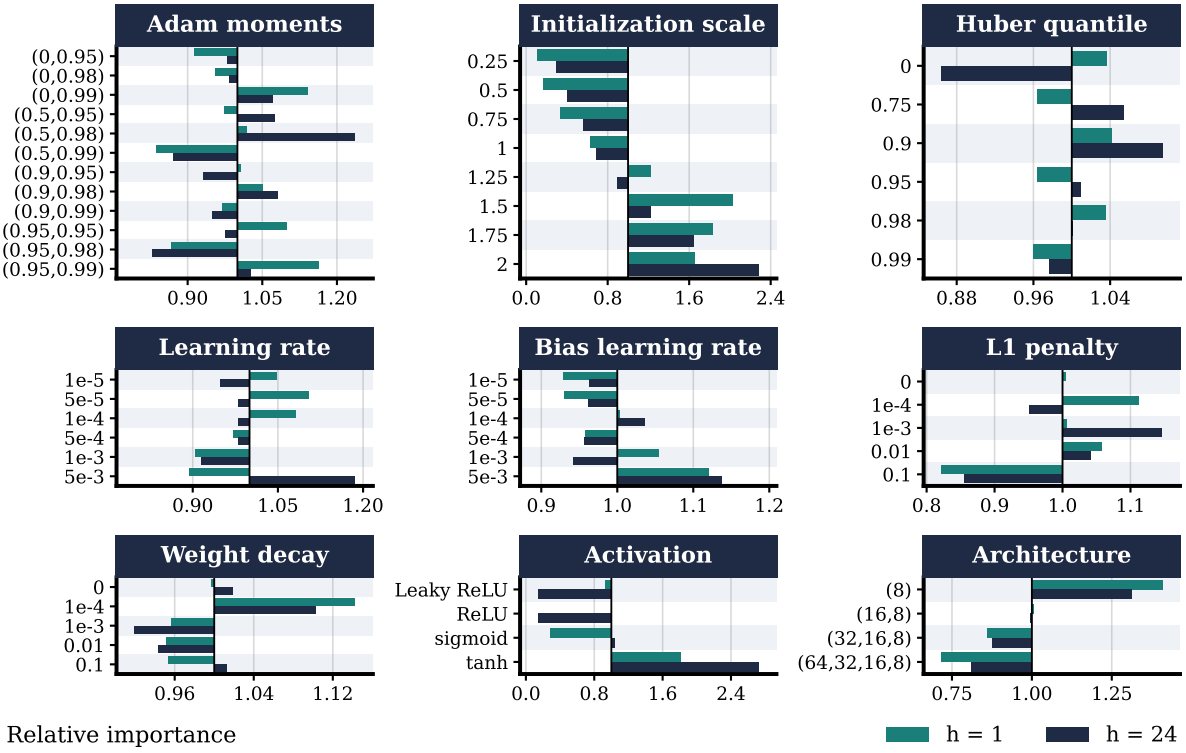

FIGURE 5: Relative importance of the hyperparameter values for U-ODNN-RLS at $h = 1$ and $h = 24$. Larger is more important.

We saw in Section 5.4, that RLS provides substantially better forecasts than equal weighting of candidates. The Shapley values suggest that this forecast improvement originates from RLS being able to differentiate between important and unimportantant hyperparameters.

6.4 Inference on the restricted hyperparameter weights

Besides assessing hyperparameter importance through Shapley values, we can use the RLS-hyp method to assess the coefficient estimates directly. Each regressor defined inside the RLS-hyp has an interpretable coefficient estimate. Larger values assign larger weight to those candidates and smaller values assign smaller weight. Although the regressors are a misspecified representation of the hyperparameters, they give the best linear approximation in the provided regressors. To get an idea of the precision of the estimates, we also compute a 90% Newey-West interval around the coefficient estimates, based on the asymptotic normality in Corollary 3.3 (iv). This

method is much less computationally demanding than computing Shapley values and allows us to quantify the precision of the estimates. Tools like this help us understand which hyperparameters get assigned larger weight, and therefore which hyperparameters drive the forecast results.

The estimated hyperparameter weights and confidence intervals are displayed in Figure 6. We first discuss U-ODNN. There are clear similarities between the forecast horizons. The intercept is estimated at the same level. For $h = 1$, the largest positive coefficient is the weight decay. Stronger decay gets more weight than weaker decay. For the negative coefficients, lower values of the Adam decay rates β_1 and β_2 are preferred. This is to be expected, since stronger momentum comes at the expense of quick adaptation. For longer horizons there is a clear pattern favouring stronger momentum and less weight decay. Interestingly, these patterns are quite different from Figure 5. This might be due to the linearity assumption, which is potentially too restrictive. For the architecture, the largest architecture is favoured less than the smallest reference architecture. We find a mostly similar result for the other horizons, which is clearly in line with what we saw for the Shapley values. For the longer horizons, it is interesting that, similar to the Shapley values, larger initialization scales are assigned more weight and somewhat less weight is assigned to Leaky ReLU6, but we do not find the dominance of the Tanh function. For the ℓ_1 penalty we also see a negative effect. The Shapley values tell us that this possibly originates only from the large value $\lambda_1 = 0.1$.

When we turn to the R-ODNN, we get fairly similar results. Here, the Huber quantile has a large negative coefficient. A lower Huber quantile is preferred, which means more dampening of large errors. Potentially, we do not see this for U-ODNN because Adam already dampens these errors. Strikingly, Leaky ReLU6 has a large positive coefficient here. This is especially striking because its reference is ReLU6. Leaky ReLU6 seems to add value in the R-ODNN.

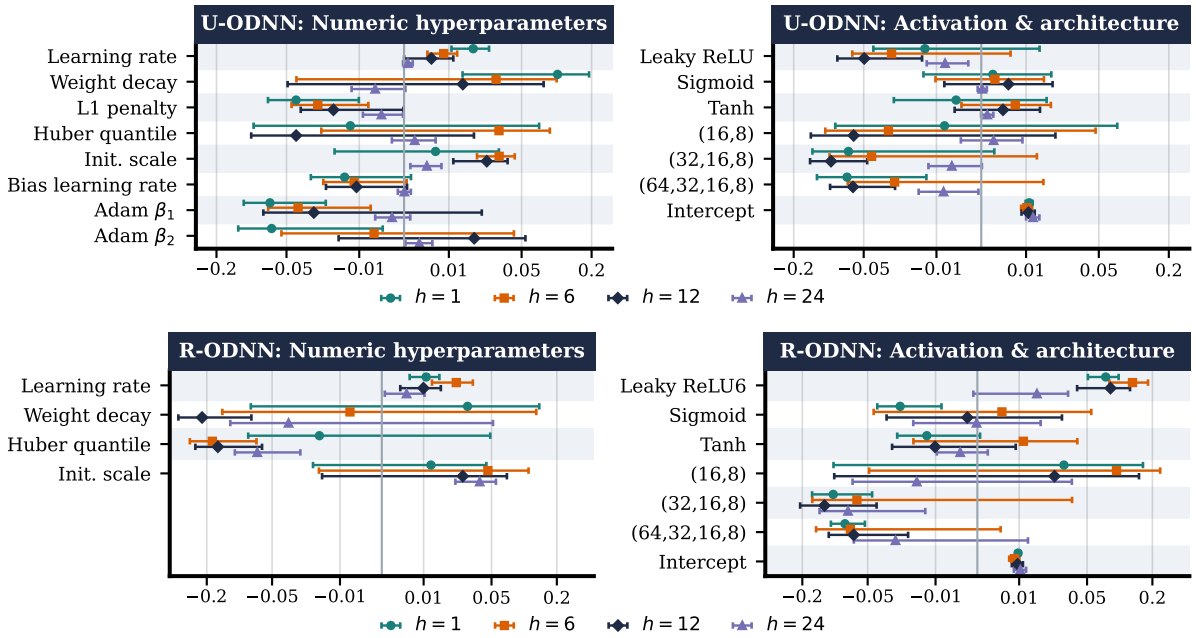

FIGURE 6: End-of-sample coefficients of the restricted hyperparameter weights (RLS-hyp) by forecast horizon, with 90% intervals from Newey–West standard errors (Bartlett kernel, Newey–West (1994) plug-in lag). Features are standardized; numeric hyperparameters other than the Adam moments enter in $\log_{10}$. Reference categories: ReLU (ReLU6 for R-ODNN) and a single hidden layer of eight units. Horizontal axis: coefficient, on an inverse hyperbolic sine scale.

6.5 Oracle bounds and the number of candidates

To measure how close U-ODNN-RLS comes to the best feasible combination, we compare it against oracle combinations. We rank candidates by their realized out-of-sample MSE and form equal-weight combinations of the top m, for $m \in \{1, 10, 100, 1{,}000, 10{,}000\}$. These oracle combinations select candidates on their out-of-sample performance and are therefore infeasible. They show how an equal-weight average of the ex post best candidates compares with the RLS combination. Figure 7 plots them against the RLS combination at each horizon. The oracle is minimized at or near $m = 1$, $m = 10$, reaching ratios of about 0.79, 0.70, 0.67 and 0.63 across the four horizons. Thus, with out-of-sample knowledge, averaging only a few candidates is preferred. Beyond $m = 10$ the oracle deteriorates as weaker candidates enter the equal-weight average. The RLS combination performs worse than the top-10 oracle at every horizon. The gap widens with the horizon in a specific sense. At $h = 1$, 6 and 12 the RLS ratio falls between the oracles that average the top 100 and the top 1,000 candidates, whereas at $h = 24$ it falls between the top-1,000 oracle and the average of the full pool. RLS therefore behaves more like a broad average and discriminates less sharply among candidates as the horizon grows.

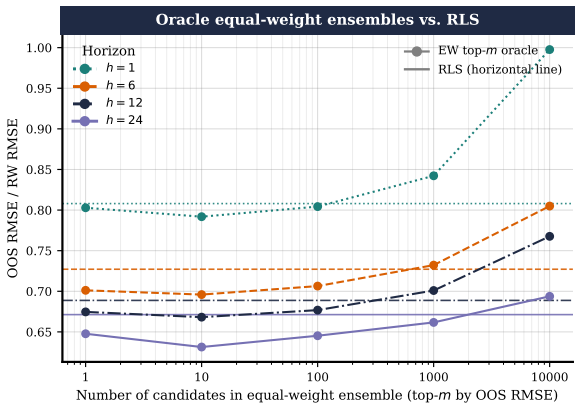

FIGURE 7: Out-of-sample performance of equally weighted oracle combinations of the top 1, 10, 100, 1,000 and 10,000 candidates, ranked by out-of-sample MSE, compared with the RLS combination across forecast horizons.

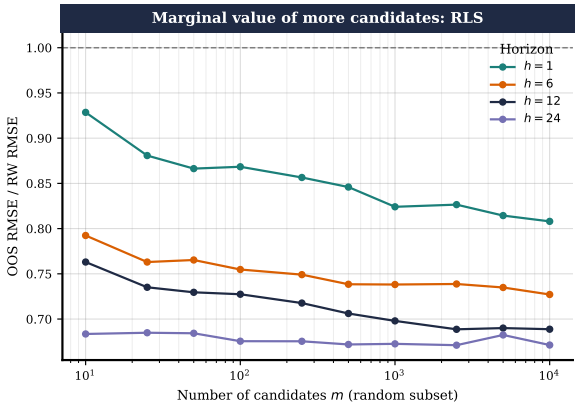

FIGURE 8: Out-of-sample RMSE of the RLS combination relative to the random walk benchmark when RLS is restricted to a random subset of m candidates, by forecast horizon. Each line is the mean across random draws of the subset.

Finally, we ask whether the full pool of 10,000 candidates is necessary to achieve good forecast performance. Figure 8 reports the RLS performance when RLS is restricted to a random subset of m candidates, averaged across draws of the subset. Performance improves with the number of candidates at every horizon. The gains over the random walk are largest at $h = 1$, where the ratio falls from 0.93 at $m = 10$ to 0.81 at $m = 10{,}000$, and smaller at the longer horizons, where the curve is flatter. Apart from small fluctuations, the forecast performance does not deteriorate when more candidates are added. The full pool of 10,000 delivers the best forecasts. This shows that RLS is effective on a large scale and overcomes the overfitting risk caused by the increasing pool of candidates. This suggests that adding more candidates can potentially further improve the results. However, the curves appear to flatten, which means that we need exponentially more computational resources to gain better forecast performance. We therefore do not increase the number of candidates further, since ten times as many candidates would make our method slower than LGBM, the fastest machine learning benchmark.

7 Feature Shapley contributions

As is standard for machine learning methods, ODNN, and therefore ODNN-RLS, allows for the computation of feature importance through Shapley values. Due to the large number of variables, we compute the Shapley values per variable group. We define eight variable groups following McCracken and Ng (2016): (i) output and income, (ii) labor market, (iii) housing, (iv) consumption, orders and inventories, (v) money and credit, (vi) interest and exchange rates, (vii) prices and (viii) stock market. Over these eight variable groups we can compute exact Shapley values.

We do this for the U-ODNN-RLS forecast at the first and twelfth forecast horizon. We denote the set of groups by $\mathcal{M}$ and a group combination by $\mathcal{C} \subseteq \mathcal{M}$. In total we evaluate $2^8 = 256$ group combinations. The neural network parameters and hyperparameter weights of each candidate are the same as in the model used in Table 2, but at every filter step in the out-of-sample period we evaluate the 10,000 candidates for each of the 256 group combinations. Each coalition value of U-ODNN-RLS is then the RLS-weighted forecast built from the outputs of the candidates evaluated at that coalition,

$$\mathcal{V}_{t,h}(\mathcal{C}) = \sum_{k=1}^{K} \hat{w}_{k,t,h}\, f(x_t^{\mathcal{C}}; \theta_{k,t,h}, \eta_k), \tag{27}$$

where $x_t^{\mathcal{C}}$ equals x_t for the variables in the groups of $\mathcal{C}$ and the trailing mean for all other variables. The Shapley value of group z is

$$\mathcal{F}_{z,t,h} = \sum_{\mathcal{C} \subseteq \mathcal{M} \setminus \{z\}} \frac{|\mathcal{C}|!\,(7 - |\mathcal{C}|)!}{8!} \left(\mathcal{V}_{t,h}(\mathcal{C} \cup \{z\}) - \mathcal{V}_{t,h}(\mathcal{C})\right). \tag{28}$$

Since we evaluate all 256 group combinations, the evaluation is exact and the eight contributions add up to the forecast minus the baseline, $\mathcal{V}_{t,h}(\mathcal{M}) - \mathcal{V}_{t,h}(\emptyset)$. The baseline is the forecast with all variables set to their trailing mean. Due to the linearity in the weights, the group contribution to U-ODNN-RLS is the RLS-weighted sum of the group contributions to the individual candidates.

As a benchmark, we compare this approach with the Shapley values of the random forest. We evaluate the random forest over the same 256 combinations and with the same baseline. We show the Shapley values at four months with large forecast errors: January 2015, June 2017, April 2020 and June 2022. Moreover, we show an aggregate measure. The share panels report the mean absolute contribution of each group over the 170 evaluation months, divided by the total over all groups, similar to Equation (26). Figure 9 shows both for the first forecast horizon and Figure 11 in the appendix for the twelfth forecast horizon. Table 5 reports the realized CPI inflation, the forecast and the baseline for each of the four months.

The U-ODNN-RLS and the random forest show similar patterns across the variable groups in the four months. Past price information is the most important group for the forecast at each of the dates. Relative to prices, U-ODNN-RLS on average puts more importance on the other seven groups than the random forest. The share of the total mean absolute contribution confirms this. Especially at the first forecast horizon, U-ODNN-RLS puts a much smaller share of its contribution on prices than the random forest. It puts a larger share on all other groups, except for housing, where the shares are equal. At the twelfth forecast horizon, both models put a smaller share of the contribution on prices. For U-ODNN-RLS, housing becomes the most important group. The random forest puts a larger share on the labor market than U-ODNN-RLS. This analysis shows that the filtered candidate combination can be interpreted through Shapley values.

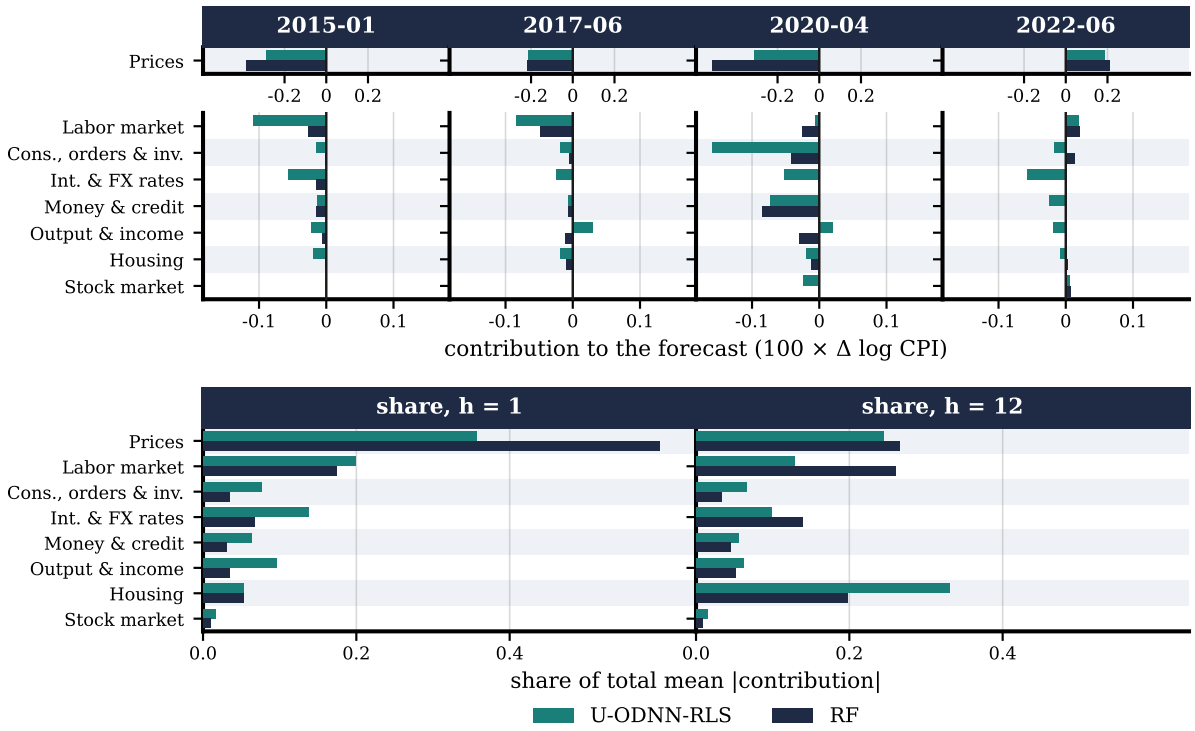

FIGURE 9: Shapley contributions of the FRED-MD variable groups to the U-ODNN-RLS and random forest forecasts. Top: four months at $h = 1$ (forecasts and baselines in Table 5). Bottom: share of each group in the total mean absolute contribution at $h = 1$ and $h = 12$.

TABLE 5: Realized values, forecasts and baselines at $h = 1$ for the months in Figure 9 ($100 \times \Delta \log \text{CPI}$).

		U-ODNN-RLS		RF	
Month	Real.	Fcst	Base	Fcst	Base
2015-01	−0.64	−0.17	0.35	−0.14	0.31
2017-06	0.07	0.02	0.35	0.03	0.33
2020-04	−0.80	−0.28	0.35	−0.38	0.33
2022-06	1.29	0.44	0.35	0.57	0.31

8 Conclusion

This paper develops an observation-driven neural network framework for macroeconomic forecasting. The weights and biases of each network evolve over time through an observation-driven update based on Adam, so the model adapts to incoming data without re-estimation over a rolling window. We treat the hyperparameters as part of the model rather than as fixed inputs chosen in advance, and run a pool of 10,000 candidate ODNNs whose hyperparameter weights are tracked online by recursive least squares. The framework links the machine learning and observation-driven parameter literatures, and it turns the usual problem of hyperparameter selection into a forecast combination problem.

We evaluate the framework on U.S. inflation using predictors from FRED-MD, over the 15 years from 2011 to 2025 and forecast horizons of 1 to 24 months. U-ODNN-RLS achieves the lowest relative RMSE at every horizon beyond the first, and it is the only method that enters every model confidence set for CPI inflation. When excluding the COVID period, it is the only model included in the average squared-error model confidence set. The hyperparameter

importance analysis shows why the approach works. RLS spreads weight over thousands of candidates with improved forecast performance compared to a single candidate choice. The method allows for rich set of configurations without overfitting and improving forecast performance as the candidate pool grows.

We have studied a single target, CPI inflation, and it would be natural to apply the framework to other macroeconomic and financial series. The observation-driven update and the recursive estimation of the hyperparameter weights are generic, so the approach extends beyond inflation forecasting to any setting where models must adapt online to changing conditions.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework.

Asi, H. and Duchi, J. C. (2019). Stochastic (Approximate) Proximal Point Methods: Convergence, Optimality, and Adaptivity. *SIAM Journal on Optimization*, 29(3):2257–2290.

Babii, A., Ghysels, E., and Striaukas, J. (2022). Machine learning time series regressions with an application to nowcasting. *Journal of Business & Economic Statistics*, 40(3):1094–1106.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945.

Bates, J. M. and Granger, C. W. J. (1969). The combination of forecasts. *Operational Research Quarterly*, 20(4):451–468.

Belloni, A., Chen, D., Chernozhukov, V., and Hansen, C. (2012). Sparse Models and Methods for Optimal Instruments With an Application to Eminent Domain. *Econometrica*, 80(6):2369–2429. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.3982/ECTA9626.

Billio, M., Casarin, R., Ravazzolo, F., and van Dijk, H. K. (2013). Time-varying combinations of predictive densities using nonlinear filtering. *Journal of Econometrics*, 177(2):213–232.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer, New York.

Blasques, F., Gorgi, P., Koopman, S. J., and Wintenberger, O. (2018). Feasible invertibility conditions and maximum likelihood estimation for observation-driven models. *Electronic Journal of Statistics*, 12(1):1019–1052.

Blasques, F., van Brummelen, J., Koopman, S. J., and Lucas, A. (2022). Maximum likelihood estimation for score-driven models. *Journal of Econometrics*, 227(2):325–346.

Bougerol, P. (1993). Kalman filtering with random coefficients and contractions. *SIAM Journal on Control and Optimization*, 31(4):942–959.

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Cawley, G. C. and Talbot, N. L. C. (2010). On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107.

Creal, D., Koopman, S. J., and Lucas, A. (2013). Generalized autoregressive score models with applications. *Journal of Applied Econometrics*, 28(5):777–795.

Creal, D., Koopman, S. J., Lucas, A., and Zamojski, M. (2024). Observation-driven filtering of time-varying parameters using moment conditions. *Journal of Econometrics*, 238(2):105635.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314.

D'Agostino, A., Gambetti, L., and Giannone, D. (2013). Macroeconomic forecasting and structural change. *Journal of Applied Econometrics*, 28(1):82–101.

Davidson, J. (1994). *Stochastic Limit Theory: An Introduction for Econometricians*. Number 9780198774037 in OUP Catalogue. Oxford University Press.

de Castro, C. H. D. C. and Aiube, F. A. L. (2023). Forecasting inflation time series using score-driven dynamic models and combination methods: The case of Brazil. *Journal of Forecasting*, 42(2):369–401.

de Menezes, L. M., Bunn, D. W., and Taylor, J. W. (2000). Review of guidelines for the use of combined forecasts. *European Journal of Operational Research*, 120(1):190–204.

Del Negro, M., Hasegawa, R. B., and Schorfheide, F. (2016). Dynamic prediction pools: An investigation of financial frictions and forecasting performance. *Journal of Econometrics*, 192(2):391–405.

Delle Monache, D., De Polis, A., and Petrella, I. (2024). Modeling and forecasting macroeconomic downside risk. *Journal of Business & Economic Statistics*, 42(3):1010–1025.

Delle Monache, D. and Petrella, I. (2017). Adaptive models and heavy tails with an application to inflation forecasting. *International Journal of Forecasting*, 33(2):482–501.

Diebold, F. X. (1988). Serial correlation and the combination of forecasts. *Journal of Business & Economic Statistics*, 6(1):105–111.

Durbin, J. and Koopman, S. J. (2012). *Time Series Analysis by State Space Methods: Second Edition*. Oxford University Press.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1):3–42.

Giannone, D., Lenza, M., and Primiceri, G. E. (2021). Economic predictions with big data: The illusion of sparsity. *Econometrica*, 89(5):2409–2437.

Goulet Coulombe, P. (2024). The macroeconomy as a random forest. *Journal of Applied Econometrics*, 39(3):401–421.

Goulet Coulombe, P. (2025). A neural Phillips curve and a deep output gap. *Journal of Business & Economic Statistics*, 43(3):669–683.

Goulet Coulombe, P., Leroux, M., Stevanovic, D., and Surprenant, S. (2022). How is machine learning useful for macroeconomic forecasting? *Journal of Applied Econometrics*, 37(5):920–964.

Granger, C. W. J. and Ramanathan, R. (1984). Improved methods of combining forecasts. *Journal of Forecasting*, 3(2):197–204.

Gu, S., Kelly, B., and Xiu, D. (2020). 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*, 50(4):1029–1054.

Harvey, A. C. (2013). *Dynamic Models for Volatility and Heavy Tails: With Applications to Financial and Economic Time Series*. Number 52 in Econometric Society Monographs. Cambridge University Press.

He, Y. (2024). Ridge Regression Under Dense Factor Augmented Models. *Journal of the American Statistical Association*, 119(546):1566–1578.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101.

Joseph, A., Potjagailo, G., Chakraborty, C., and Kapetanios, G. (2024). Forecasting UK inflation bottom up. *International Journal of Forecasting*, 40(4):1521–1538.

Kalfa, S. Y., Timmermann, A., and van der Zwan, T. (2025). Overhyped? Can ML models reliably predict stock returns? Working Paper, University of California San Diego and Erasmus University Rotterdam.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 3146–3154. Curran Associates, Inc.

Kelly, B., Malamud, S., and Zhou, K. (2024). The Virtue of Complexity in Return Prediction. *The Journal of Finance*, 79(1):459–503.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Koop, G. and Korobilis, D. (2012). Forecasting inflation using dynamic model averaging. *International Economic Review*, 53(3):867–886.

Krengel, U. (1985). *Ergodic Theorems*. De Gruyter Studies in Mathematics. De Gruyter, Berlin.

Lai, B. and Bernstein, D. S. (2024). Generalized forgetting recursive least squares: Stability and robustness guarantees.

Lange, R.-J., van Os, B., and Van Dijk, D. (2026). Implicit score-driven filters for time-varying parameter models. *Journal of Econometrics*, 255:106251.

LeCun, Y., Bottou, L., Orr, G. B., and Müller, K.-R. (1998). Efficient backprop. In *Neural networks: Tricks of the trade*, Lecture Notes in Computer Science, pages 9–50. Springer.

Loshchilov, I. and Hutter, F. (2019). Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30, pages 4765–4774.

McCracken, M. W. and Ng, S. (2016). FRED-MD: A Monthly Database for Macroeconomic Research. *Journal of Business & Economic Statistics*, 34(4):574–589.

Medeiros, M. C., Vasconcelos, G. F. R., Veiga, , and Zilberman, E. (2021). Forecasting Inflation in a Data-Rich Environment: The Benefits of Machine Learning Methods. *Journal of Business & Economic Statistics*, 39(1):98–119.

Moroshko, E., Vaits, N., and Crammer, K. (2015). Second-order non-stationary online learning for regression. *Journal of Machine Learning Research*, 16(43):1481–1517.

Naghi, A. A., O'Neill, E., and Danielova Zaharieva, M. (2024). The benefits of forecasting inflation with machine learning: New evidence. *Journal of Applied Econometrics*, 39(7):1321–1331.

Newey, W. K. and West, K. D. (1987). A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix. *Econometrica*, 55(3):703–708.

Primiceri, G. E. (2005). Time varying structural vector autoregressions and monetary policy. *The Review of Economic Studies*, 72(3):821–852.

Quaedvlieg, R. (2021). Multi-Horizon Forecast Comparison. *Journal of Business & Economic Statistics*, 39(1):40–53.

Raftery, A. E., Kárný, M., and Ettler, P. (2010). Online prediction under model uncertainty via dynamic model averaging: Application to a cold rolling mill. *Technometrics*, 52(1):52–66.

Saunders, C., Gammerman, A., and Vovk, V. (1998). Ridge regression learning algorithm in dual variables. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998)*, pages 515–521. Morgan Kaufmann.

Sessions, D. N. and Chatterjee, S. (1989). The combining of forecasts using recursive techniques with non-stationary weights. *Journal of Forecasting*, 8(3):239–251.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Stock, J. H. and Watson, M. W. (1996). Evidence on structural instability in macroeconomic time series relations. *Journal of Business & Economic Statistics*, 14(1):11–30.

Stock, J. H. and Watson, M. W. (2004). Combination forecasts of output growth in a seven-country data set. *Journal of Forecasting*, 23(6):405–430.

Stock, J. H. and Watson, M. W. (2007). Why has U.S. inflation become harder to forecast? *Journal of Money, Credit and Banking*, 39(s1):3–33.

Straumann, D. and Mikosch, T. (2006). Quasi-maximum-likelihood estimation in conditionally heteroscedastic time series: A stochastic recurrence equations approach. *Annals of Statistics*, 34(5):2449–2495.

Štrumbelj, E. and Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, 11:1–18.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288.

Timmermann, A. (2006). Forecast combinations. In Elliott, G., Granger, C. W. J., and Timmermann, A., editors, *Handbook of Economic Forecasting*, volume 1, pages 135–196. Elsevier.

Zou, H. (2006). The Adaptive Lasso and Its Oracle Properties. *Journal of the American Statistical Association*, 101(476):1418–1429.

A Invertibility of the filter

This appendix gives the proof of Proposition 3.1 for the restricted filter specified in Section 3.1. The proof has three parts, followed by the proof of a corollary. In Appendix A.1, we give conditions on the activation functions and the predictors under which Assumption 3.1 (ii) holds. In Appendix A.2, we show that the filter shrinks the distance between any two points by at least a factor smaller than 1. In Appendix A.3, we combine this contraction with Bougerol (1993, Theorem 3.1) to prove Proposition 3.1. In Appendix A.4, we prove Corollary 3.1 on the convergence, stationarity and moments of the limit forecasts.

The main text keeps the horizon subscript h on the parameters, the hidden layers and the forecast errors of the filter. For readability, we fix the horizon h and drop this subscript throughout Appendix A. We therefore write $\tilde{\theta}_t$, $\bar{a}_t$, τ and e_t for $\tilde{\theta}_{t,h}$, $\bar{a}_{t,h}$, τ_h and $e_{t,h}$ of Section 3.1, and we do the same for all other quantities. The horizon still enters through the lag of the predictors, so that $\bar{a}_{t-h}$ denotes $\bar{a}_{t-h,h}$.

A.1 Bounds on the last hidden layer

Before addressing invertibility, we show that the bound stated in Assumption 3.1 (ii) holds for general classes of activation functions commonly used inside neural networks. We also give examples of how this bound can be derived for some popular specific activation functions. The bound holds either because the activation function itself is bounded, as in Lemma A.1 and Examples A.1 and A.2, or because the activation function is continuous and the predictors are bounded, as in Lemma A.2 and Example A.3.

LEMMA A.1: *Let the activation function satisfy $|\rho(y)| \leq \bar{\rho}$ for all y, with a constant $\bar{\rho} < \infty$. Then $\|\bar{a}_t\|_2^2 \leq \bar{\rho}^2 n_d + 1$ for all t. Therefore Assumption 3.1 (ii) holds with $V^2 = \bar{\rho}^2 n_d + 1$.*

Proof. Every element of the last hidden layer lies in $[-\bar{\rho}, \bar{\rho}]$, and the last hidden layer has n_d elements. Hence $\|\tau(\tilde{x}_t)\|_2^2 \leq \bar{\rho}^2 n_d$ and $\|\bar{a}_t\|_2^2 = 1 + \|\tau(\tilde{x}_t)\|_2^2 \leq \bar{\rho}^2 n_d + 1$ for all t. □

EXAMPLE A.1 (Tanh and Sigmoid): The Tanh function maps its input between $(-1, 1)$ and the Sigmoid function maps its input between $(0, 1)$, so both satisfy $|\rho(y)| \leq 1$ for all y. By Lemma A.1 with $\bar{\rho} = 1$, Assumption 3.1 (ii) holds with $V^2 = n_d + 1$ for any predictors.

EXAMPLE A.2 (ReLU6 and Leaky ReLU6): The ReLU6 function is the ReLU function clipped at 6, $\rho(y) = \min\{\max\{y, 0\}, 6\}$. The Leaky ReLU6 function is the Leaky ReLU function clipped at -6 and 6, $\rho(y) = \max\{-6, \min\{\max\{y, \mu y\}, 6\}\}$ with $0 < \mu < 1$. Both functions are continuous and satisfy $|\rho(y)| \leq 6$ for all y. By Lemma A.1 with $\bar{\rho} = 6$, Assumption 3.1 (ii) holds with $V^2 = 36\, n_d + 1$ for any predictors.

LEMMA A.2: *Let the activation function be continuous, and let the predictors lie in a compact set, $\tilde{x}_t \in \mathcal{X}$ for all t. Then $V^2 := 1 + \sup_{\tilde{x} \in \mathcal{X}} \|\tau(\tilde{x})\|_2^2$ is finite and $\|\bar{a}_t\|_2 \leq V$ for all t. Therefore Assumption 3.1 (ii) holds.*

Proof. Each hidden layer in Equation (3) is a continuous function of its input, because the activation function is continuous. Therefore the map τ is continuous. A continuous function attains its maximum over a compact set, so V is finite. Since $\tilde{x}_t \in \mathcal{X}$, we have $\|\bar{a}_t\|_2^2 = 1 + \|\tau(\tilde{x}_t)\|_2^2 \leq V^2$ for all t. □

EXAMPLE A.3 (ReLU and Leaky ReLU): ReLU and Leaky ReLU are continuous, so Lemma A.2 shows that V is finite when the predictors are bounded. For these activation functions we can also give V explicitly. Let each predictor be bounded in absolute value, $|\tilde{x}_{j,t}| \leq c$ for all j and t, so that $\mathcal{X} = [-c, c]^n$. We bound each layer sequentially. Each layer in Equation (3) is bounded

by the previous layer,

$$\|a_{i,t}\|_2 = \|\rho(W_i a_{i-1,t} + b_i)\|_2 \le \|W_i a_{i-1,t} + b_i\|_2 \le \|W_i\|_2 \, \|a_{i-1,t}\|_2 + \|b_i\|_2.$$

where we define $\|W\|_2 := \max_{y \neq 0} \|Wy\|_2 / \|y\|_2$ as the spectral norm. The first inequality holds because ReLU and Leaky ReLU do not make a vector longer. Formally, these activation functions satisfy $|\rho(y)| \le |y|$, because $\max\{y, 0\}$ equals either y or 0, and $\max\{y, \mu y\}$ equals either y or μy with $0 < \mu < 1$. Applied to each element, this gives $\|\rho(y)\|_2 \le \|y\|_2$ for any vector y. The second inequality follows from the triangle inequality and the spectral norm. The sub-multiplicativity of the spectral norm gives that $\|Wy\|_2 \le \|W\|_2 \|y\|_2$ for every y.

By the predictor being bounded in absolute value by c,

$$\|a_{0,t}\|_2 = \|\tilde{x}_t\|_2 \le \sqrt{n}\, c =: R_0.$$

We define $R_i := \|W_i\|_2 \, R_{i-1} + \|b_i\|_2$ for $i = 1, \ldots, d$. Each layer $i - 1$ $\|a_{i-1,t}\|_2 \le R_{i-1}$ gives $\|a_{i,t}\|_2 \le R_i$ for layer i. Going through each of the layers starting at $i = 1$, we get $\|\tau(\tilde{x}_t)\|_2 = \|a_{d,t}\|_2 \le R_d$. Hence $\|\bar{a}_t\|_2^2 \le R_d^2 + 1$ for all t and Assumption 3.1 (ii) holds with $V^2 = R_d^2 + 1$.

A.2 Contraction of the filter

We write the filter in Equation (14) as $\tilde{\theta}_t = \Phi_t(\tilde{\theta}_{t-1})$, with the map

$$\Phi_t(\tilde{\theta}) := \gamma \, \tilde{\theta} + \alpha \, \psi\big(\pi_t - \tilde{\theta}^\top \bar{a}_{t-h}\big) \, \bar{a}_{t-h},$$

and we define its Lipschitz constant,

$$\Lambda_t := \sup_{\tilde{\theta} \neq \tilde{\theta}^\dagger} \frac{\|\Phi_t(\tilde{\theta}) - \Phi_t(\tilde{\theta}^\dagger)\|_2}{\|\tilde{\theta} - \tilde{\theta}^\dagger\|_2}.$$

LEMMA A.3: *Let Assumption 3.1 (ii), (iii) and (iv) hold. Then there is a contraction factor $\varphi < 1$ such that $\Lambda_t \le \varphi$ for all t.*

Proof. We show that Φ_t is Lipschitz with a constant $\varphi < 1$, so that Φ_t contracts and $\Lambda_t \le \varphi$. This requires two steps. First, we use the secant slope of ψ to show that $\Phi_t(\tilde{\theta}) - \Phi_t(\tilde{\theta}^\dagger) = J_t(\tilde{\theta} - \tilde{\theta}^\dagger)$ for a symmetric matrix J_t that depends on the two points only through a scalar $\xi \in [0, 1]$. Second, we bound the eigenvalues of J_t to show that $\|J_t\|_2 \le \varphi$ for every ξ.

Take two points $\tilde{\theta}$ and $\tilde{\theta}^\dagger$, and write $e := e_t(\tilde{\theta})$ and $e^\dagger := e_t(\tilde{\theta}^\dagger)$ for their forecast errors. Subtracting the map at $\tilde{\theta}^\dagger$ from the map at $\tilde{\theta}$ gives

$$\Phi_t(\tilde{\theta}) - \Phi_t(\tilde{\theta}^\dagger) = \gamma \, (\tilde{\theta} - \tilde{\theta}^\dagger) + \alpha \, \bar{a}_{t-h} \, \big(\psi(e) - \psi(e^\dagger)\big).$$

We use the secant slope of ψ between $e^\dagger$ and e,

$$\xi := \frac{\psi(e) - \psi(e^\dagger)}{e - e^\dagger} \quad \text{if } e \neq e^\dagger, \qquad \xi := 0 \quad \text{if } e = e^\dagger.$$

This gives us

$$\psi(e) - \psi(e^\dagger) = \xi \, (e - e^\dagger),$$

which is valid because both sides are zero when $e = e^\dagger$. The choice $\xi = 0$ when $e = e^\dagger$ is arbitrary. The function ψ is nondecreasing and 1-Lipschitz by Assumption 3.1 (iii). Therefore, $\xi \in [0, 1]$. The difference of the errors is $e - e^\dagger = -\bar{a}_{t-h}^\top (\tilde{\theta} - \tilde{\theta}^\dagger)$. Substituting both into the

difference of the map gives

$$\begin{aligned}\Phi_t(\tilde{\theta}) - \Phi_t(\tilde{\theta}^\dagger) &= \gamma\,(\tilde{\theta} - \tilde{\theta}^\dagger) + \xi\,\alpha\,\bar{a}_{t-h}\,(e - e^\dagger) \\ &= \big(\gamma I - \xi\,\alpha\,\bar{a}_{t-h}\bar{a}_{t-h}^\top\big)(\tilde{\theta} - \tilde{\theta}^\dagger) =: J_t\,(\tilde{\theta} - \tilde{\theta}^\dagger).\end{aligned}$$

The gap between the filtered steps $\Phi_t(\tilde{\theta})$ and $\Phi_t(\tilde{\theta}^\dagger)$ is a matrix J_t times the gap between $\tilde{\theta}$ and $\tilde{\theta}^\dagger$. The matrix J_t is symmetric and it depends on $\tilde{\theta}$ and $\tilde{\theta}^\dagger$ only through ξ.

We now argue that the eigenvalues of J_t are $\gamma - \xi\,\alpha\,\|\bar{a}_{t-h}\|_2^2$ and γ. Multiplying J_t by $\bar{a}_{t-h}$ gives $\big(\gamma - \xi\,\alpha\,\|\bar{a}_{t-h}\|_2^2\big)\bar{a}_{t-h}$, and multiplying J_t by any vector y with $\bar{a}_{t-h}^\top y = 0$ gives $\gamma\,y$. These vectors together span $\mathbb{R}^{n_d}$, so there are no other eigenvalues.

The first eigenvalue depends on ξ, and we can bound it because $\xi \in [0, 1]$. From above, it is at most γ, because $\xi \geq 0$ and $\alpha > 0$. From below, it is at least $\gamma - \alpha V^2$, because $\xi \leq 1$ and $\|\bar{a}_{t-h}\|_2^2 \leq V^2$ by Assumption 3.1 (ii). Hence, the absolute values of all eigenvalues are at most

$$\varphi := \max\{\gamma,\ \alpha V^2 - \gamma\},$$

and $\|J_t\|_2 \leq \varphi$ for every ξ because J_t is symmetric. The contraction factor φ is smaller than one, because $\gamma < 1$ and $\alpha V^2 - \gamma < 1$ by Assumption 3.1 (iv). This gives

$$\|\Phi_t(\tilde{\theta}) - \Phi_t(\tilde{\theta}^\dagger)\|_2 = \|J_t(\tilde{\theta} - \tilde{\theta}^\dagger)\|_2 \leq \varphi\,\|\tilde{\theta} - \tilde{\theta}^\dagger\|_2.$$

As the two points were arbitrary, $\Lambda_t \leq \varphi$ for all t. □

A.3 Proof of Proposition 3.1

Proof. The map Φ_t depends on t only through the data (π_t, x_{t-h}), so by Assumption 3.1 (i) the maps Φ_t are stationary and ergodic. We will now verify the conditions of Bougerol (1993, Theorem 3.1). The theorem states that if the maps Φ_t are stationary and ergodic, and

(a) $\mathbb{E}\log^+\|\Phi_t(y_0) - y_0\|_2 < \infty$ for some fixed y_0,

(b) $\mathbb{E}\log^+\Lambda_t < \infty$,

(c) $\mathbb{E}\log\Lambda_t < 0$,

then there is a unique stationary and ergodic solution $\tilde{\theta}_t^{\mathrm{s}}$, which is a measurable function of $\{(\pi_s, x_{s-h})\}_{s\leq t}$, and every filtered path converges to it (see also Straumann and Mikosch, 2006, Theorem 2.8). We verify conditions (a) to (c) in Steps 1 and 2, and then use Krengel (1985, Proposition 4.3) to show that the forecasts are stationary and ergodic too.

Step 1: condition (a). We take $y_0 = 0$ and $\Phi_t(0) = \alpha\,\psi(\pi_t)\,\bar{a}_{t-h}$, so $\|\Phi_t(0)\|_2 = \alpha\,|\psi(\pi_t)|\,\|\bar{a}_{t-h}\|_2$. Since ψ is 1-Lipschitz by Assumption 3.1 (iii), we have $|\psi(\pi_t)| \leq |\psi(0)| + |\pi_t|$. With $\|\bar{a}_{t-h}\|_2 \leq V < \infty$ by Assumption 3.1 (ii), we get $\|\Phi_t(0)\|_2 \leq \alpha V\,(|\psi(0)| + |\pi_t|)$. The $\log^+$ of this bound has a finite expectation, because $\log^+(yz) \leq \log^+ y + \log^+ z$ and $\log^+(y+z) \leq \log 2 + \log^+ y + \log^+ z$ for $y, z \geq 0$, and $\mathbb{E}\log^+|\pi_t| < \infty$ by Assumption 3.1 (i).

Step 2: conditions (b) and (c). By Lemma A.3, $\Lambda_t \leq \varphi < 1$ for all t. In particular, $\log^+\Lambda_t = 0$ a.s., and $\log\Lambda_t \leq \log\varphi < 0$, which are conditions (b) and (c).

Step 3: invertibility of the parameters. All conditions hold, so by Bougerol (1993, Theorem 3.1) there is a unique stationary and ergodic solution $\tilde{\theta}_t^{\mathrm{s}}$. This solution satisfies the same recursion as $\tilde{\theta}_t$, $\tilde{\theta}_t^{\mathrm{s}} = \Phi_t(\tilde{\theta}_{t-1}^{\mathrm{s}})$. By Lemma A.3, the distance between both paths shrinks by at least the factor φ in every step,

$$\|\tilde{\theta}_t - \tilde{\theta}_t^{\mathrm{s}}\|_2 = \|\Phi_t(\tilde{\theta}_{t-1}) - \Phi_t(\tilde{\theta}_{t-1}^{\mathrm{s}})\|_2 \leq \varphi\,\|\tilde{\theta}_{t-1} - \tilde{\theta}_{t-1}^{\mathrm{s}}\|_2.$$

Repeating this bound t times gives

$$\|\tilde{\theta}_t - \tilde{\theta}_t^{\mathrm{s}}\|_2 \leq \varphi^t \, \|\tilde{\theta}_0 - \tilde{\theta}_0^{\mathrm{s}}\|_2.$$

So the filter converges exponentially fast a.s. to the stationary solution, for any initialization $\tilde{\theta}_0$. This shows that the filter is invertible.

Step 4: forecasts. The forecast of the stationary solution is $\tilde{\theta}_t^{\mathrm{s}\top} \bar{a}_t$. It is a measurable function of the current and past data, because $\tilde{\theta}_t^{\mathrm{s}}$ is measurable and the hidden layers in $\bar{a}_t$ are continuous, and therefore measurable. By Krengel (1985, Proposition 4.3), a measurable function of a stationary and ergodic sequence is stationary and ergodic. So the forecasts $\tilde{\theta}_t^{\mathrm{s}\top} \bar{a}_t$ are stationary and ergodic.

Since the network is linear in $\tilde{\theta}$, the difference in the forecasts is $(\tilde{\theta}_t - \tilde{\theta}_t^{\mathrm{s}})^\top \bar{a}_t$. By the Cauchy–Schwarz inequality and $\|\bar{a}_t\|_2 \leq V$ by Assumption 3.1 (ii),

$$\left|\tilde{\theta}_t^\top \bar{a}_t - \tilde{\theta}_t^{\mathrm{s}\top} \bar{a}_t\right| \leq V \, \|\tilde{\theta}_t - \tilde{\theta}_t^{\mathrm{s}}\|_2 \leq V \, \varphi^t \, \|\tilde{\theta}_0 - \tilde{\theta}_0^{\mathrm{s}}\|_2.$$

The forecasts therefore converge exponentially fast to the stationary and ergodic forecasts, for any initialization $\tilde{\theta}_0$. □

A.4 Proof of Corollary 3.1

Proof. We show the three statements of the corollary one by one. In part 1 we show that the forecasts converge e.a.s. In part 2 we show that the limit forecasts and the target are jointly stationary and ergodic. In part 3 we show that the limit forecasts have finite second moments.

Part 1: e.a.s. convergence. By Proposition 3.1, the forecast of every candidate satisfies $|\tilde{\theta}_t^\top \bar{a}_t - \tilde{\theta}_t^{\mathrm{s}\top} \bar{a}_t| \leq V \, \varphi^t \, \|\tilde{\theta}_0 - \tilde{\theta}_0^{\mathrm{s}}\|_2$. The norm $\|\tilde{\theta}_0 - \tilde{\theta}_0^{\mathrm{s}}\|_2$ is finite almost surely. Consequently, the bound multiplied by any number between 1 and $1/\varphi$ to the power t converges to zero almost surely, and the gap of every candidate converges to zero e.a.s. There are finitely many candidates. We can take the largest contraction factor for all candidates such that the gap $\|\mathbf{f}_t - \mathbf{f}_t^{\mathrm{s}}\|_2$ of the stacked forecasts also converges to zero e.a.s.

Part 2: stationarity and ergodicity. By Proposition 3.1, the limit forecast of every candidate is a measurable function of the current and past data. Together with the target π_{t+h}, the vector $(\mathbf{f}_t^{\mathrm{s}\top}, \pi_{t+h})^\top$ is then a measurable function of $\{(\pi_s, \tilde{x}_s)\}_{s \leq t+h}$. The data are stationary and ergodic by Assumption 3.1 (i). By Krengel (1985, Proposition 4.3), the limit forecasts and the target are jointly stationary and ergodic.

Part 3: moments. We follow the moment bound in the proof of Blasques et al. (2022, Technical Appendix, Proposition TA.3). We consider a single candidate. The stationary solution satisfies $\tilde{\theta}_t^{\mathrm{s}} = \Phi_t(\tilde{\theta}_{t-1}^{\mathrm{s}})$. By the triangle inequality and Lemma A.3,

$$\|\tilde{\theta}_t^{\mathrm{s}}\|_2 \leq \|\Phi_t(\tilde{\theta}_{t-1}^{\mathrm{s}}) - \Phi_t(0)\|_2 + \|\Phi_t(0)\|_2 \leq \varphi \, \|\tilde{\theta}_{t-1}^{\mathrm{s}}\|_2 + \|\Phi_t(0)\|_2.$$

Unfolding this bound N steps backwards gives

$$\|\tilde{\theta}_t^{\mathrm{s}}\|_2 \leq \varphi^N \, \|\tilde{\theta}_{t-N}^{\mathrm{s}}\|_2 + \sum_{s=t-N+1}^{t} \varphi^{t-s} \, \|\Phi_s(0)\|_2.$$

The factor φ^N goes to zero at an exponential rate as $N \to \infty$, and $\{\|\tilde{\theta}_{t-N}^{\mathrm{s}}\|_2\}$ is stationary by Proposition 3.1. For N large enough, the first term is therefore smaller than 1, which gives

$$\|\tilde{\theta}_t^{\mathrm{s}}\|_2 \leq 1 + \sum_{s=t-N+1}^{t} \varphi^{t-s} \, \|\Phi_s(0)\|_2.$$

We take the root of the expected square on both sides. By the subadditivity of the norm $\big(\mathbb{E}\|\cdot\|_2^2\big)^{1/2}$, and since the maps Φ_s are stationary,

$$\big(\mathbb{E}\|\tilde{\theta}_t^{\mathrm{s}}\|_2^2\big)^{1/2} \le 1 + \sum_{s=t-N+1}^{t} \varphi^{t-s}\big(\mathbb{E}\|\Phi_s(0)\|_2^2\big)^{1/2} \le 1 + \frac{\big(\mathbb{E}\|\Phi_t(0)\|_2^2\big)^{1/2}}{1-\varphi}.$$

In Step 1 of the proof of Proposition 3.1 we showed that $\|\Phi_t(0)\|_2 \le \alpha V\,(|\psi(0)| + |\pi_t|)$. By the same subadditivity, $\big(\mathbb{E}\|\Phi_t(0)\|_2^2\big)^{1/2} \le \alpha V\,\big(|\psi(0)| + (\mathbb{E}[\pi_t^2])^{1/2}\big) < \infty$, so $\mathbb{E}\|\tilde{\theta}_t^{\mathrm{s}}\|_2^2 < \infty$. For the limit forecast, the Cauchy–Schwarz inequality and $\|\bar{a}_t\|_2 \le V$ give $(\tilde{\theta}_t^{\mathrm{s}\top}\bar{a}_t)^2 \le V^2\,\|\tilde{\theta}_t^{\mathrm{s}}\|_2^2$, which has a finite expectation. Since there are finitely many candidates, the stacked limit forecasts also satisfy $\mathbb{E}\|\mathbf{f}_t^{\mathrm{s}}\|_2^2 < \infty$.

Our setting differs from their Proposition TA.3 in two respects. Their state is scalar, while our state $\tilde{\theta}_t^{\mathrm{s}}$ is a vector. The argument carries over, because it only uses the triangle inequality and the subadditivity of the norm. Their result also holds uniformly over a compact parameter space, which we do not need because the hyperparameters of each candidate are fixed. □

B Consistency and optimality of the hyperparameter weights

This appendix gives the proofs of the results in Section 3.2. In Appendix B.1, we prove the ridge representation of Lemma 3.1. In Appendix B.2, we use this representation to prove the consistency of the hyperparameter weights in Proposition 3.2. In Appendices B.3 and B.4, we prove Corollaries 3.2 and 3.3. Throughout, we fix the horizon h and normalize $\sigma_h^2 = 1$.

B.1 Proof of Lemma 3.1

Proof. The recursion in Equations (9) to (11) is the Kalman filter for the state-space model in Equations (7) and (8), with $\mathbf{Q} = \mathbf{0}$ and $\sigma_h^2 = 1$. The update at time $t > h$ uses the pair $(\mathbf{f}_{t-h,h}, \pi_t)$ and becomes

$$P_{t,h}^{-1} = \Big(\big(I - \boldsymbol{\kappa}_{t,h} \mathbf{f}_{t-h,h}^\top\big) P_{t-1,h} \Big)^{-1} = P_{t-1,h}^{-1} + \mathbf{f}_{t-h,h} \mathbf{f}_{t-h,h}^\top,$$
$$P_{t,h}^{-1} \hat{\mathbf{w}}_{t,h} = P_{t,h}^{-1} \Big(\hat{\mathbf{w}}_{t-1,h} + \boldsymbol{\kappa}_{t,h} \big(\pi_t - \mathbf{f}_{t-h,h}^\top \hat{\mathbf{w}}_{t-1,h}\big) \Big) = P_{t-1,h}^{-1} \hat{\mathbf{w}}_{t-1,h} + \mathbf{f}_{t-h,h}\, \pi_t.$$

Every update adds the new pair to the values of the previous period. The filter stays at $\hat{\mathbf{w}}_{0,h}$ and $P_{0,h}$ for $t \le h$. Repeating the update from time $h+1$ to time t gives

$$P_{t,h}^{-1} = P_{0,h}^{-1} + \sum_{s=1}^{t-h} \mathbf{f}_{s,h} \mathbf{f}_{s,h}^\top = P_{0,h}^{-1} + \mathbf{F}_{1:t-h,h}^\top \mathbf{F}_{1:t-h,h},$$
$$P_{t,h}^{-1} \hat{\mathbf{w}}_{t,h} = P_{0,h}^{-1} \hat{\mathbf{w}}_{0,h} + \sum_{s=1}^{t-h} \mathbf{f}_{s,h} \pi_{s+h} = P_{0,h}^{-1} \hat{\mathbf{w}}_{0,h} + \mathbf{F}_{1:t-h,h}^\top \boldsymbol{\pi}_{1+h:t}.$$

The matrix $P_{0,h}^{-1} + \mathbf{F}_{1:t-h,h}^\top \mathbf{F}_{1:t-h,h}$ is invertible, because $P_{0,h}^{-1}$ is positive definite and $\mathbf{F}_{1:t-h,h}^\top \mathbf{F}_{1:t-h,h}$ is positive semidefinite. Together, after rewriting, the two equations give Equation (15).

We now show that $\hat{\mathbf{w}}_{t,h}$ minimizes the criterion in Equation (16). The derivative of the criterion with respect to $\mathbf{w}$ is

$$2\big(P_{0,h}^{-1} + \mathbf{F}_{1:t-h,h}^\top \mathbf{F}_{1:t-h,h}\big)\mathbf{w} - 2\big(P_{0,h}^{-1} \hat{\mathbf{w}}_{0,h} + \mathbf{F}_{1:t-h,h}^\top \boldsymbol{\pi}_{1+h:t}\big),$$

which is zero at $\mathbf{w} = \hat{\mathbf{w}}_{t,h}$ by Equation (15). The criterion is strictly convex because $P_{0,h}^{-1}$ is positive definite, so $\hat{\mathbf{w}}_{t,h}$ is its unique minimizer. □

B.2 Proof of Proposition 3.2

Proof. We hold K and h fixed and let T denote the number of completed forecast and realization pairs. These pairs are available at information time $T + h$. In Step 1 we show that the sample moments of the limit forecasts converge. In Step 2 we show that the sample moments of the filtered forecasts have the same limits. In Step 3 we use these limits in the ridge representation of Lemma 3.1.

Step 1: the sample moments of the limit forecasts converge. By Assumption 3.2 (i), the limit forecasts and the target are jointly stationary and ergodic and have finite second moments. By the Cauchy–Schwarz inequality, the products $\mathbf{f}_{t,h}^{\mathrm{s}} \mathbf{f}_{t,h}^{\mathrm{s}\top}$ and $\mathbf{f}_{t,h}^{\mathrm{s}} \pi_{t+h}$ then have finite expectations. By the ergodic theorem, their sample averages converge almost surely to these expectations,

$$\tfrac{1}{T} \sum_{s=1}^{T} \mathbf{f}_{s,h}^{\mathrm{s}} \mathbf{f}_{s,h}^{\mathrm{s}\top} \to M_h, \qquad \tfrac{1}{T} \sum_{s=1}^{T} \mathbf{f}_{s,h}^{\mathrm{s}} \pi_{s+h} \to \mathbb{E}\big[\mathbf{f}_{t,h}^{\mathrm{s}} \pi_{t+h}\big] = M_h \mathbf{w}_h^*.$$

The last equality is Equation (17).

Step 2: the filtered forecasts have the same limits. We want to show that

$$\tfrac{1}{T}\sum_{s=1}^{T}\mathbf{f}_{s,h}\mathbf{f}_{s,h}^{\top}\to M_h,\qquad \tfrac{1}{T}\sum_{s=1}^{T}\mathbf{f}_{s,h}\pi_{s+h}\to M_h\mathbf{w}_h^{*}.$$

By Step 1, the sample moments of the limit forecasts have these limits. So we have to show that the sample moments of the filtered forecasts have the same limits as those of the limit forecasts. We show this by evaluating the difference at each time step. It suffices to show that

$$\left\|\mathbf{f}_{s,h}\mathbf{f}_{s,h}^{\top}-\mathbf{f}_{s,h}^{\mathrm{s}}\mathbf{f}_{s,h}^{\mathrm{s}\top}\right\|_2\to 0,\qquad \left\|\mathbf{f}_{s,h}\pi_{s+h}-\mathbf{f}_{s,h}^{\mathrm{s}}\pi_{s+h}\right\|_2\to 0$$

almost surely as $s\to\infty$, because a sequence that converges to zero has an average that converges to zero. Both differences are bounded by the gap $\|\mathbf{f}_{s,h}-\mathbf{f}_{s,h}^{\mathrm{s}}\|_2$ between the filtered and the limit forecasts,

$$\begin{aligned}\left\|\mathbf{f}_{s,h}\mathbf{f}_{s,h}^{\top}-\mathbf{f}_{s,h}^{\mathrm{s}}\mathbf{f}_{s,h}^{\mathrm{s}\top}\right\|_2 &\le \|\mathbf{f}_{s,h}-\mathbf{f}_{s,h}^{\mathrm{s}}\|_2\left(\|\mathbf{f}_{s,h}\|_2+\|\mathbf{f}_{s,h}^{\mathrm{s}}\|_2\right)\\ &\le 2\,\|\mathbf{f}_{s,h}-\mathbf{f}_{s,h}^{\mathrm{s}}\|_2\,\|\mathbf{f}_{s,h}^{\mathrm{s}}\|_2+\|\mathbf{f}_{s,h}-\mathbf{f}_{s,h}^{\mathrm{s}}\|_2^2,\\ \left\|\mathbf{f}_{s,h}\pi_{s+h}-\mathbf{f}_{s,h}^{\mathrm{s}}\pi_{s+h}\right\|_2 &= \|\mathbf{f}_{s,h}-\mathbf{f}_{s,h}^{\mathrm{s}}\|_2\,|\pi_{s+h}|.\end{aligned}$$

To show that these bounds converge to zero, we use Straumann and Mikosch (2006, Lemma 2.1). By this lemma, a sequence that converges to zero exponentially fast almost surely (e.a.s.), multiplied by a stationary sequence with a $\log^+$ moment, also converges to zero e.a.s. We first note that the gap $\|\mathbf{f}_{s,h}-\mathbf{f}_{s,h}^{\mathrm{s}}\|_2$ of the stacked forecasts converges to zero e.a.s. by Assumption 3.2 (ii), and therefore so does its square. The sequences $\|\mathbf{f}_{s,h}^{\mathrm{s}}\|_2$ and $|\pi_{s+h}|$ are stationary by Step 1, and they have a $\log^+$ moment because they have a finite second moment by Step 1. Thus, the lemma implies that the terms on both right-hand sides converge to zero e.a.s, which concludes this step.

Step 3: the weights converge. We evaluate Equation (15) at $t=T+h$ and write the matrix products as sums. We then divide both the matrix in the inverse and the vector behind it by T, which leaves the product unchanged,

$$\hat{\mathbf{w}}_{T+h,h}=\left(\tfrac{1}{T}P_{0,h}^{-1}+\tfrac{1}{T}\sum_{s=1}^{T}\mathbf{f}_{s,h}\mathbf{f}_{s,h}^{\top}\right)^{-1}\left(\tfrac{1}{T}P_{0,h}^{-1}\hat{\mathbf{w}}_{0,h}+\tfrac{1}{T}\sum_{s=1}^{T}\mathbf{f}_{s,h}\pi_{s+h}\right).$$

We let $T\to\infty$. The initialization terms $\frac{1}{T}P_{0,h}^{-1}$ and $\frac{1}{T}P_{0,h}^{-1}\hat{\mathbf{w}}_{0,h}$ converge to zero, because the initial values do not depend on T. By Step 2, the two sample moments converge almost surely to M_h and $M_h\mathbf{w}_h^{*}$. Together, the matrix in the inverse converges to M_h, and the vector behind it converges to $M_h\mathbf{w}_h^{*}$. The matrix M_h is positive definite by Assumption 3.2 (iii), so it is invertible and matrix inversion is continuous at M_h. By the continuous mapping theorem,

$$\hat{\mathbf{w}}_{T+h,h}\xrightarrow{\text{a.s.}}M_h^{-1}M_h\,\mathbf{w}_h^{*}=\mathbf{w}_h^{*}.$$

The initialization $(\hat{\mathbf{w}}_{0,h},P_{0,h})$ only enters through terms of order $1/T$, so the filter forgets it. The weights therefore converge to the best linear combination of the limit forecasts. □

B.3 Proof of Corollary 3.2

Proof. We hold K and h fixed. We split the forecast error of the combined forecast into three terms,

$$\pi_{s+h}-\mathbf{f}_{s,h}^{\top}\hat{\mathbf{w}}_{s,h}=\mathcal{E}_s^{(1)}-\mathcal{E}_s^{(2)}-\mathcal{E}_s^{(3)},$$

where

$$\mathcal{E}_s^{(1)} = \pi_{s+h} - \mathbf{f}_{s,h}^{\mathrm{s}\top}\mathbf{w}_h^*, \qquad \mathcal{E}_s^{(2)} = \big(\mathbf{f}_{s,h} - \mathbf{f}_{s,h}^{\mathrm{s}}\big)^\top \hat{\mathbf{w}}_{s,h}, \qquad \mathcal{E}_s^{(3)} = \mathbf{f}_{s,h}^{\mathrm{s}\top}\big(\hat{\mathbf{w}}_{s,h} - \mathbf{w}_h^*\big).$$

The first term is the forecast error of the best linear combination of the limit forecasts. The second term is due to the gap between the filtered and the limit forecasts, and the third term is due to the estimation error of the weights. In Step 1 we show that the average of $(\mathcal{E}_s^{(1)})^2$ converges to $\mathcal{L}_h(\mathbf{w}_h^*)$. In Steps 2 and 3 we show that the averages of $(\mathcal{E}_s^{(2)})^2$ and $(\mathcal{E}_s^{(3)})^2$ converge to zero. In Step 4 we combine these results.

Step 1: the first term. By Step 1 of the proof of Proposition 3.2, the limit forecasts and the target are jointly stationary and ergodic and have finite second moments. So $(\mathcal{E}_s^{(1)})^2$ is stationary and ergodic and has a finite expectation. By the ergodic theorem, as $T \to \infty$,

$$\tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(1)}\big)^2 \to \mathcal{L}_h(\mathbf{w}_h^*) \quad \text{almost surely.}$$

Step 2: the second term. By the Cauchy–Schwarz inequality,

$$\big(\mathcal{E}_s^{(2)}\big)^2 \le \|\mathbf{f}_{s,h} - \mathbf{f}_{s,h}^{\mathrm{s}}\|_2^2 \, \|\hat{\mathbf{w}}_{s,h}\|_2^2.$$

The weights converge almost surely by Proposition 3.2, so $\|\hat{\mathbf{w}}_{s,h}\|_2$ is bounded over s almost surely. The gap $\|\mathbf{f}_{s,h} - \mathbf{f}_{s,h}^{\mathrm{s}}\|_2$ converges to zero e.a.s. by Assumption 3.2 (ii). The bound converges to zero almost surely, and its average converges to zero, because a sequence that converges to zero has an average that converges to zero.

Step 3: the third term. By the Cauchy–Schwarz inequality,

$$\big(\mathcal{E}_s^{(3)}\big)^2 \le \|\mathbf{f}_{s,h}^{\mathrm{s}}\|_2^2 \, \|\hat{\mathbf{w}}_{s,h} - \mathbf{w}_h^*\|_2^2.$$

The second factor converges to zero almost surely by Proposition 3.2, and the average of the first factor converges to $\mathbb{E}\|\mathbf{f}_{t,h}^{\mathrm{s}}\|_2^2 < \infty$ by the ergodic theorem. The average of the bound therefore converges to zero almost surely.

Step 4: combining the terms. Expanding the square of the forecast error gives

$$\tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(1)} - \mathcal{E}_s^{(2)} - \mathcal{E}_s^{(3)}\big)^2 = \tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(1)}\big)^2 + \tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(2)} + \mathcal{E}_s^{(3)}\big)^2 - \tfrac{2}{T}\sum_{s=1}^{T}\mathcal{E}_s^{(1)}\big(\mathcal{E}_s^{(2)} + \mathcal{E}_s^{(3)}\big).$$

The first average converges to $\mathcal{L}_h(\mathbf{w}_h^*)$ by Step 1. The second average converges to zero by Steps 2 and 3, because

$$\tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(2)} + \mathcal{E}_s^{(3)}\big)^2 \le \tfrac{2}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(2)}\big)^2 + \tfrac{2}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(3)}\big)^2.$$

The third average converges to zero, because by the Cauchy–Schwarz inequality

$$\Big|\tfrac{1}{T}\sum_{s=1}^{T}\mathcal{E}_s^{(1)}\big(\mathcal{E}_s^{(2)} + \mathcal{E}_s^{(3)}\big)\Big| \le \Big(\tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(1)}\big)^2\Big)^{1/2}\Big(\tfrac{1}{T}\sum_{s=1}^{T}\big(\mathcal{E}_s^{(2)} + \mathcal{E}_s^{(3)}\big)^2\Big)^{1/2},$$

where the first factor converges to $\mathcal{L}_h(\mathbf{w}_h^*)^{1/2}$ and the second factor converges to zero. Therefore, the average squared forecast error of the combined forecast converges to $\mathcal{L}_h(\mathbf{w}_h^*)$ almost surely.

The weights $\mathbf{w}_h^*$ minimize $\mathcal{L}_h$ over $\mathbb{R}^K$. The weight vector with weight one on candidate k and zero on the others gives the expected squared error $\mathbb{E}\big[(\pi_{t+h} - f_{k,t,h}^{\mathrm{s}})^2\big]$. Since $\mathbf{w}_h^*$ minimizes $\mathcal{L}_h$, the value $\mathcal{L}_h(\mathbf{w}_h^*)$ is at most this expected squared error, for every candidate k. □

B.4 Proof of Corollary 3.3

Proof. We hold K, q and h fixed. RLS-hyp is the filter in Equations (9) to (11) with the feature forecasts $\boldsymbol{\phi}_{t,h} = \mathbf{Z}^\top \mathbf{f}_{t,h}$ in place of the candidate forecasts $\mathbf{f}_{t,h}$, and with the initial values $\hat{\boldsymbol{\psi}}_{0,h}$ and $\tilde{P}_{0,h}$. The proofs of Lemma 3.1, Proposition 3.2 and Corollary 3.2 use the candidate forecasts only through Assumption 3.2. In Step 1 we show that the feature forecasts satisfy this assumption. In Step 2 we apply the proposition and the corollary to the feature forecasts. In Step 3 we derive Equation (19). In Step 4 we derive the limit distribution in part (iv).

Step 1: the feature forecasts satisfy Assumption 3.2. We define the limit feature forecasts $\boldsymbol{\phi}^{\mathrm{s}}_{t,h} = \mathbf{Z}^\top \mathbf{f}^{\mathrm{s}}_{t,h}$. The matrix $\mathbf{Z}$ is fixed, so the limit feature forecasts are a fixed linear function of the limit forecasts. By Assumption 3.2 (i), the limit feature forecasts and the target are then jointly stationary and ergodic and have finite second moments. The gap between the feature forecasts and the limit feature forecasts satisfies

$$\|\boldsymbol{\phi}_{t,h} - \boldsymbol{\phi}^{\mathrm{s}}_{t,h}\|_2 = \big\|\mathbf{Z}^\top\big(\mathbf{f}_{t,h} - \mathbf{f}^{\mathrm{s}}_{t,h}\big)\big\|_2 \leq \|\mathbf{Z}\|_2 \, \|\mathbf{f}_{t,h} - \mathbf{f}^{\mathrm{s}}_{t,h}\|_2,$$

which converges to zero e.a.s. by Assumption 3.2 (ii). The second moment matrix of the limit feature forecasts is

$$\mathbb{E}\big[\boldsymbol{\phi}^{\mathrm{s}}_{t,h}\boldsymbol{\phi}^{\mathrm{s}\top}_{t,h}\big] = \mathbf{Z}^\top \mathbb{E}\big[\mathbf{f}^{\mathrm{s}}_{t,h}\mathbf{f}^{\mathrm{s}\top}_{t,h}\big]\mathbf{Z} = \mathbf{Z}^\top M_h \mathbf{Z},$$

which is positive definite by the condition of the corollary. The feature forecasts therefore satisfy Assumption 3.2 when we consider q instead of K.

Step 2: the coefficients and the average squared forecast error converge. By Proposition 3.2 applied to the feature forecasts, the coefficients $\hat{\boldsymbol{\psi}}_{t,h}$ converge almost surely to

$$\big(\mathbf{Z}^\top M_h \mathbf{Z}\big)^{-1}\mathbb{E}\big[\boldsymbol{\phi}^{\mathrm{s}}_{t,h}\pi_{t+h}\big] = \big(\mathbf{Z}^\top M_h \mathbf{Z}\big)^{-1}\mathbf{Z}^\top \mathbb{E}\big[\mathbf{f}^{\mathrm{s}}_{t,h}\pi_{t+h}\big] = \boldsymbol{\psi}^*_h,$$

where the last equality is Equation (18). By Corollary 3.2 applied to the feature forecasts, the average squared forecast error converges almost surely to

$$\mathbb{E}\Big[\big(\pi_{t+h} - \boldsymbol{\phi}^{\mathrm{s}\top}_{t,h}\boldsymbol{\psi}^*_h\big)^2\Big] = \mathbb{E}\Big[\big(\pi_{t+h} - \mathbf{f}^{\mathrm{s}\top}_{t,h}\mathbf{Z}\boldsymbol{\psi}^*_h\big)^2\Big] = \mathcal{L}_h(\mathbf{Z}\boldsymbol{\psi}^*_h).$$

Step 3: the loss of the restricted weights. The function $\mathcal{L}_h$ is a convex quadratic function of $\mathbf{w}$ that is bounded from below by zero, so it attains its minimum over $\mathbf{w}$. Let $\mathbf{w}^*_h$ be any minimizer of $\mathcal{L}_h$ over $\mathbb{R}^K$. The derivative of $\mathcal{L}_h$ with respect to $\mathbf{w}$ is $2\big(M_h\mathbf{w} - \mathbb{E}\big[\mathbf{f}^{\mathrm{s}}_{t,h}\pi_{t+h}\big]\big)$. It is zero at the minimizer, which gives $\mathbb{E}\big[\mathbf{f}^{\mathrm{s}}_{t,h}\pi_{t+h}\big] = M_h\mathbf{w}^*_h$. This does not require M_h to be invertible. For any $\mathbf{w} \in \mathbb{R}^K$, we write the forecast error as $\pi_{t+h} - \mathbf{f}^{\mathrm{s}\top}_{t,h}\mathbf{w} = \big(\pi_{t+h} - \mathbf{f}^{\mathrm{s}\top}_{t,h}\mathbf{w}^*_h\big) - \mathbf{f}^{\mathrm{s}\top}_{t,h}\big(\mathbf{w} - \mathbf{w}^*_h\big)$. Expanding the square and taking expectations gives

$$\mathcal{L}_h(\mathbf{w}) = \mathcal{L}_h(\mathbf{w}^*_h) + \big(\mathbf{w} - \mathbf{w}^*_h\big)^\top M_h\big(\mathbf{w} - \mathbf{w}^*_h\big) - 2\big(\mathbf{w} - \mathbf{w}^*_h\big)^\top \mathbb{E}\Big[\mathbf{f}^{\mathrm{s}}_{t,h}\big(\pi_{t+h} - \mathbf{f}^{\mathrm{s}\top}_{t,h}\mathbf{w}^*_h\big)\Big].$$

The expectation in the last term equals $\mathbb{E}\big[\mathbf{f}^{\mathrm{s}}_{t,h}\pi_{t+h}\big] - M_h\mathbf{w}^*_h = \mathbf{0}$. Setting $\mathbf{w} = \mathbf{Z}\boldsymbol{\psi}^*_h$ gives Equation (19).

By Equation (19), the second term equals $\mathcal{L}_h(\mathbf{Z}\boldsymbol{\psi}^*_h) - \mathcal{L}_h(\mathbf{w}^*_h)$, which is zero if and only if $\mathbf{Z}\boldsymbol{\psi}^*_h$ minimizes $\mathcal{L}_h$ over $\mathbb{R}^K$. In that case $\mathcal{L}_h$ has a minimizer of the form $\mathbf{Z}\boldsymbol{\psi}$. Conversely, let $\mathbf{Z}\boldsymbol{\psi}$ minimize $\mathcal{L}_h$ over $\mathbb{R}^K$ for some $\boldsymbol{\psi} \in \mathbb{R}^q$. Then $\boldsymbol{\psi}$ also minimizes $\mathcal{L}_h(\mathbf{Z}\boldsymbol{\psi})$ over $\mathbb{R}^q$, and $\boldsymbol{\psi} = \boldsymbol{\psi}^*_h$ because the minimizer in Equation (18) is unique. The vector $\mathbf{Z}\boldsymbol{\psi}^*_h$ therefore minimizes $\mathcal{L}_h$ over $\mathbb{R}^K$.

Step 4: the coefficients are asymptotically normal. In this step we show that, under the condition in part (iv) of Corollary 3.3, the coefficients are asymptotically normal. We again use

the representation of the ridge regression of the target on the feature forecasts as stated in Lemma 3.1.

Subtracting $\boldsymbol{\psi}_h^*$ and multiplying by $\sqrt{T}$ gives

$$\sqrt{T}\big(\hat{\boldsymbol{\psi}}_{T+h,h} - \boldsymbol{\psi}_h^*\big) = \Big(\tfrac{1}{T}\tilde{P}_{0,h}^{-1} + \tfrac{1}{T}\sum_{s=1}^{T}\boldsymbol{\phi}_{s,h}\boldsymbol{\phi}_{s,h}^\top\Big)^{-1} \times \Big(\tfrac{1}{\sqrt{T}}\tilde{P}_{0,h}^{-1}\big(\hat{\boldsymbol{\psi}}_{0,h} - \boldsymbol{\psi}_h^*\big) + \tfrac{1}{\sqrt{T}}\sum_{s=1}^{T}\boldsymbol{\phi}_{s,h}\big(\pi_{s+h} - \boldsymbol{\phi}_{s,h}^\top\boldsymbol{\psi}_h^*\big)\Big).$$

We write the forecast error of the filtered feature forecasts in terms of the projection error,

$$\pi_{s+h} - \boldsymbol{\phi}_{s,h}^\top\boldsymbol{\psi}_h^* = r_{s,h} - \big(\boldsymbol{\phi}_{s,h} - \boldsymbol{\phi}_{s,h}^{\mathrm{s}}\big)^\top\boldsymbol{\psi}_h^*.$$

Substituting this, the vector behind the inverse becomes the sum of three terms,

$$\tfrac{1}{\sqrt{T}}\sum_{s=1}^{T}\boldsymbol{\phi}_{s,h}^{\mathrm{s}}r_{s,h} + \tfrac{1}{\sqrt{T}}\tilde{P}_{0,h}^{-1}\big(\hat{\boldsymbol{\psi}}_{0,h} - \boldsymbol{\psi}_h^*\big) + \tfrac{1}{\sqrt{T}}\sum_{s=1}^{T}\Big(\big(\boldsymbol{\phi}_{s,h} - \boldsymbol{\phi}_{s,h}^{\mathrm{s}}\big)r_{s,h} - \boldsymbol{\phi}_{s,h}\big(\boldsymbol{\phi}_{s,h} - \boldsymbol{\phi}_{s,h}^{\mathrm{s}}\big)^\top\boldsymbol{\psi}_h^*\Big).$$

The first term converges in distribution to $\mathcal{N}(\mathbf{0}, \Omega_h)$ by the condition in part (iv) of Corollary 3.3. The second term converges to zero, because the initial values do not depend on T. The third term is bounded in norm by

$$\tfrac{1}{\sqrt{T}}\sum_{s=1}^{T}\|\boldsymbol{\phi}_{s,h} - \boldsymbol{\phi}_{s,h}^{\mathrm{s}}\|_2\Big(|r_{s,h}| + \big(\|\boldsymbol{\phi}_{s,h}^{\mathrm{s}}\|_2 + \|\boldsymbol{\phi}_{s,h} - \boldsymbol{\phi}_{s,h}^{\mathrm{s}}\|_2\big)\|\boldsymbol{\psi}_h^*\|_2\Big).$$

By Step 1, the gap $\|\boldsymbol{\phi}_{s,h} - \boldsymbol{\phi}_{s,h}^{\mathrm{s}}\|_2$ converges to zero e.a.s., and therefore so does the square of the gap. The projection error and the limit feature forecasts are fixed linear functions of the target and the limit forecasts, so by Step 1 they are stationary and have finite second moments, and therefore a $\log^+$ moment. By Straumann and Mikosch (2006, Lemma 2.1), as in Step 2 of the proof of Proposition 3.2, every term in the sum converges to zero e.a.s. A sequence that converges to zero e.a.s. has a finite sum almost surely, so the sum stays bounded as T grows. Due to the factor $1/\sqrt{T}$, the third term converges to zero almost surely. By the proof of Proposition 3.2 applied to the feature forecasts, the matrix in the inverse converges almost surely to $\mathbf{Z}^\top M_h\mathbf{Z}$, which is positive definite and therefore invertible. By Slutsky's theorem,

$$\sqrt{T}\big(\hat{\boldsymbol{\psi}}_{T+h,h} - \boldsymbol{\psi}_h^*\big) \to \mathcal{N}\Big(\mathbf{0}, \big(\mathbf{Z}^\top M_h\mathbf{Z}\big)^{-1}\Omega_h\big(\mathbf{Z}^\top M_h\mathbf{Z}\big)^{-1}\Big) = \mathcal{N}(\mathbf{0}, \Sigma_h)$$

in distribution as $T \to \infty$, which is part (iv) of Corollary 3.3. □

C Additional results

This appendix collects additional results for the empirical analysis in Sections 5 to 7. In Appendix C.1, we report the forecast accuracy that complements Table 2. Table 6 gives the relative MAE for CPI inflation, and Tables 7 and 8 give the relative RMSE and MAE for PCE inflation. In Appendix C.2, we show the relative importance of the hyperparameter values of U-ODNN-RLS at $h = 6$ and $h = 12$, which completes Figure 5 for $h = 1$ and $h = 24$. In Appendix C.3, we report the Shapley contributions of the FRED-MD variable groups at $h = 12$ for the same months as in Section 7. In Appendix C.4, we repeat the candidate analysis of Section 6 for the restricted candidates. We show the distribution of the R-ODNN candidate RMSE, the forecast paths of a good and a bad candidate, and the oracle bounds for R-ODNN-RLS.

C.1 Forecast accuracy

Table 6: Relative MAE by forecast horizon for CPI inflation.

	Direct horizon					Cumulative horizon				Multi-horizon MCS			
Model	$h=1$	$h=3$	$h=6$	$h=12$	$h=24$	Σ_{1-3}	Σ_{1-6}	Σ_{1-12}	Σ_{1-24}	unif.$_{12}$	avg.$_{12}$	unif.$_{24}$	avg.$_{24}$
Panel A: Full sample													
RW	0.0019	0.0026	0.0025	0.0027	0.0029	0.0060	0.0112	0.0221	0.0469	—	—	—	—
AR	0.89*	0.77*	0.78*	0.76*	0.73*	0.72*	0.66*	0.65*	0.64*	0.05	0.00	0.05	0.00
U-ODNN-RLS	**0.81***	0.68*	0.70*	**0.67***	**0.64***	0.64*	0.53*	0.44*	**0.40***	**1.00†**	**1.00†**	**1.00†**	**1.00†**
R-ODNN-RLS	0.83*	0.68*	**0.69***	0.69*	0.70*	0.61*	**0.50***	**0.44***	0.42*	**1.00†**	0.35†	**1.00†**	0.00
UCSV	0.94	0.83*	0.85*	0.89*	0.89*	0.77*	0.75*	0.75*	0.79*	0.00	0.00	0.00	0.00
ERT	0.88	**0.68***	0.71*	0.70*	0.65*	0.64*	0.54*	0.52*	0.47*	0.71†	0.01	**1.00†**	0.45†
LGBM	0.81	0.68*	0.71*	0.68*	0.65*	**0.60***	0.52*	0.48*	0.44*	0.88†	0.13†	**1.00†**	0.47†
CatBoost	0.86*	0.71*	0.71*	0.77*	0.69*	0.63*	0.54*	0.47*	0.45*	0.07	0.00	0.12†	0.00
NN	0.85	0.74*	0.71*	0.68*	0.71*	0.64*	0.58*	0.55*	0.50*	0.07	0.00	0.12†	0.00
RF	0.83*	0.76*	0.76*	0.74*	0.68*	0.68*	0.63*	0.55*	0.53*	0.07	0.00	0.05	0.00
RLasso	1.44*	0.90	0.95	0.88	1.01	1.04	0.98	0.90	0.90	0.00	0.00	0.00	0.00
AdaLasso	0.84*	0.71*	0.76*	0.76*	0.81	0.64*	0.56*	0.49*	0.50*	0.07	0.00	0.05	0.00
Panel B: COVID excluded (2020-03–2022-06)													
RW	0.0017	0.0024	0.0023	0.0023	0.0025	0.0057	0.0104	0.0212	0.0456	—	—	—	—
AR	0.85*	0.74*	0.75*	0.71	0.71*	0.67*	0.60*	0.56*	0.64*	0.00	0.00	0.00	0.00
U-ODNN-RLS	**0.76***	**0.62***	0.64*	**0.61***	**0.57***	0.55*	0.42*	**0.33***	**0.37***	1.00†	**1.00†**	**1.00†**	**1.00†**
R-ODNN-RLS	0.78*	0.63*	**0.63***	0.61*	0.62*	**0.54***	**0.40***	0.34*	0.38*	**1.00†**	0.24†	**1.00†**	0.10†
UCSV	0.92	0.83*	0.84*	0.85*	0.85*	0.76*	0.72*	0.68*	0.78*	0.00	0.00	0.00	0.00
ERT	0.79*	0.63*	0.66*	0.64*	0.59*	0.54*	0.43*	0.43*	0.47*	0.34†	0.00	**1.00†**	0.53†
LGBM	0.76*	0.65*	0.69*	0.61*	0.57*	0.54*	0.45*	0.41*	0.42*	0.18†	0.00	**1.00†**	0.01
CatBoost	0.82*	0.67*	0.70*	0.71*	0.61*	0.56*	0.46*	0.39*	0.43*	0.00	0.00	0.07	0.00
NN	0.78*	0.64*	0.67*	0.66*	0.68*	0.54*	0.46*	0.48*	0.50*	0.06	0.00	0.08	0.00
RF	0.78*	0.65*	0.70*	0.65*	0.60*	0.56*	0.47*	0.42*	0.51*	0.06	0.00	0.10†	0.00
RLasso	1.28	0.81	0.85	0.80	0.97	0.88	0.86	0.84	0.95	0.00	0.00	0.00	0.00
AdaLasso	0.78*	0.65*	0.75*	0.68*	0.80	0.56*	0.49*	0.41*	0.44*	0.05	0.00	0.04	0.00

Notes: The first data row of each panel gives the absolute MAE of the random walk (RW); all other entries are relative to RW. Multi-horizon MCS p-values are computed over per-horizon absolute losses (subscripts denote the maximum horizon $h_{\max}$ included). Bold = best and underlined = second-best competitor per column (RW excluded). Stars denote Diebold–Mariano equal-predictive-accuracy tests against the random-walk benchmark (two-sided, squared-loss): * $p < 0.05$.

C.2 Hyperparameter importance at intermediate horizons

C.3 Feature Shapley contributions at the twelfth horizon

C.4 Analysis of the restricted candidates

TABLE 7: Relative RMSE by forecast horizon for PCE inflation.

	Direct horizon					Cumulative horizon				Multi-horizon MCS			
Model	$h=1$	$h=3$	$h=6$	$h=12$	$h=24$	$\Sigma_{1\text{–}3}$	$\Sigma_{1\text{–}6}$	$\Sigma_{1\text{–}12}$	$\Sigma_{1\text{–}24}$	unif.$_{12}$	avg.$_{12}$	unif.$_{24}$	avg.$_{24}$
Panel A: Full sample													
RW	0.0019	0.0022	0.0023	0.0025	0.0029	0.0054	0.0103	0.0208	0.0468	—	—	—	—
AR	0.87*	0.81*	0.81*	0.82*	0.78*	0.73*	0.66*	0.66*	0.66*	0.13†	0.04	0.14†	0.06
U-ODNN-RLS	**0.80***	0.78*	0.76*	**0.69***	0.73*	0.68*	0.59*	0.52*	**0.45***	1.00†	**1.00†**	**1.00†**	**1.00†**
R-ODNN-RLS	0.80*	0.76*	**0.74***	0.74*	0.74*	0.66*	0.57*	0.54*	0.51*	0.98†	0.05	0.99†	0.11†
UCSV	0.91*	0.88*	0.87*	0.94*	0.94*	0.83*	0.80*	0.82*	0.87*	0.10†	0.04	0.09	0.05
ERT	0.83*	0.76*	0.75*	0.74*	**0.70***	0.68*	0.58*	0.55*	0.51*	0.98†	0.10†	**1.00†**	0.11†
LGBM	0.81*	**0.76***	0.77*	0.73*	0.71*	**0.63***	**0.54***	**0.52***	0.49*	**1.00†**	0.20†	**1.00†**	0.11†
CatBoost	0.81*	0.77*	0.80*	0.76*	0.74*	0.65*	0.56*	0.52*	0.51*	0.98†	0.05	0.66†	0.11†
NN	0.92	0.99	0.79*	0.76*	0.73*	0.83	0.74*	0.58*	0.52*	0.08	0.04	0.09	0.11†
RF	0.80*	0.81*	0.79*	0.80*	0.73*	0.72*	0.67*	0.64*	0.62*	0.13†	0.04	0.14†	0.11†
RLasso	1.57*	1.19	1.13	1.04	1.16	1.36	1.29	1.19	1.05	0.10†	0.03	0.09	0.08
AdaLasso	0.86*	0.78*	0.80*	0.91	0.87	0.72*	0.59*	0.58*	0.65*	0.10†	0.04	0.07	0.11†
Panel B: COVID excluded (2020-03–2022-06)													
RW	0.0018	0.0021	0.0022	0.0021	0.0024	0.0051	0.0099	0.0195	0.0436	—	—	—	—
AR	0.84*	0.79*	0.76*	0.79*	0.74	0.70*	0.62*	0.59*	0.65*	0.05	0.00	0.07	0.00
U-ODNN-RLS	0.73*	**0.67***	0.63*	**0.62***	0.62*	0.56*	0.43*	**0.38***	**0.43***	**1.00†**	**1.00†**	**1.00†**	**1.00†**
R-ODNN-RLS	**0.72***	0.68*	**0.62***	0.64*	0.60*	**0.55***	**0.41***	0.39*	0.46*	0.99†	0.01	**1.00†**	0.03
UCSV	0.88*	0.86*	0.84*	0.90*	0.89*	0.79*	0.76*	0.75*	0.83*	0.05	0.00	0.07	0.00
ERT	0.73*	0.68*	0.66*	0.67*	0.59*	0.56*	0.45*	0.43*	0.50*	0.15†	0.00	0.79†	0.03
LGBM	0.76*	0.70*	0.71*	0.68*	**0.58***	0.57*	0.46*	0.43*	0.48*	0.13†	0.00	0.93†	0.01
CatBoost	0.76*	0.70*	0.74*	0.71*	0.62*	0.56*	0.45*	0.41*	0.49*	0.13†	0.00	0.79†	0.01
NN	0.75*	0.71*	0.70*	0.72*	0.73*	0.59*	0.49*	0.49*	0.54*	0.12†	0.00	0.07	0.01
RF	0.75*	0.69*	0.70*	0.68*	0.61*	0.57*	0.47*	0.46*	0.59*	0.15†	0.00	0.62†	0.01
RLasso	1.34*	1.04	0.95	0.97	1.18	1.15	1.11	1.12	1.14	0.05	0.00	0.03	0.01
AdaLasso	0.79*	0.71*	0.73*	0.73*	0.88	0.61*	0.51*	0.45*	0.48*	0.12†	0.00	0.07	0.01

Notes: The first data row of each panel gives the absolute RMSE of the random walk (RW); all other entries are relative to RW. Multi-horizon MCS p-values are computed over per-horizon squared losses (subscripts denote the maximum horizon $h_{\max}$ included). Bold = best and underlined = second-best competitor per column (RW excluded). Stars denote Diebold–Mariano equal-predictive-accuracy tests against the random-walk benchmark (two-sided, squared-loss): * $p < 0.05$.

TABLE 8: Relative MAE by forecast horizon for PCE inflation.

	Direct horizon					Cumulative horizon				Multi-horizon MCS			
Model	$h=1$	$h=3$	$h=6$	$h=12$	$h=24$	$\Sigma_{1\text{–}3}$	$\Sigma_{1\text{–}6}$	$\Sigma_{1\text{–}12}$	$\Sigma_{1\text{–}24}$	unif.$_{12}$	avg.$_{12}$	unif.$_{24}$	avg.$_{24}$
Panel A: Full sample													
RW	0.0014	0.0018	0.0018	0.0018	0.0021	0.0041	0.0077	0.0154	0.0337	—	—	—	—
AR	0.89*	0.81*	0.81*	0.83*	0.78*	0.74*	0.68*	0.69*	0.72*	0.06	0.00	0.05	0.00
U-ODNN-RLS	0.82*	0.75*	0.76*	**0.70***	0.73*	0.69*	0.58*	**0.49***	**0.45***	**1.00**†	**1.00**†	**1.00**†	0.01
R-ODNN-RLS	0.83*	0.73*	**0.72***	0.73*	0.73*	0.67*	0.54*	0.49*	0.49*	**1.00**†	0.05	**1.00**†	0.00
UCSV	0.93*	0.88*	0.88*	0.93*	0.93*	0.81*	0.79*	0.81*	0.87*	0.06	0.00	0.05	0.00
ERT	0.85*	**0.73***	0.73*	0.74*	**0.67***	0.66*	0.55*	0.53*	0.51*	**1.00**†	0.05	**1.00**†	**1.00**†
LGBM	0.81*	0.74*	0.76*	0.73*	0.67*	**0.63***	**0.54***	0.52*	0.50*	**1.00**†	0.00	**1.00**†	0.00
CatBoost	0.82*	0.77*	0.80*	0.77*	0.73*	0.65*	0.57*	0.51*	0.51*	0.37†	0.00	0.57†	0.00
NN	0.87	0.82	0.76*	0.73*	0.74*	0.74	0.66*	0.60*	0.53*	0.06	0.00	0.05	0.00
RF	**0.80***	0.79*	0.80*	0.78*	0.70*	0.69*	0.63*	0.58*	0.60*	0.26†	0.00	0.30†	0.00
RLasso	1.57*	1.12	1.11	1.08	1.21	1.34	1.27	1.21	1.19	0.01	0.00	0.01	0.00
AdaLasso	0.85*	0.76*	0.81*	0.83	0.88	0.69*	0.59*	0.57*	0.54*	0.08	0.00	0.05	0.00
Panel B: COVID excluded (2020-03–2022-06)													
RW	0.0014	0.0016	0.0017	0.0016	0.0018	0.0039	0.0075	0.0151	0.0327	—	—	—	—
AR	0.84*	0.78*	0.76*	0.79*	0.76	0.69*	0.61*	0.60*	0.71*	0.00	0.00	0.00	0.00
U-ODNN-RLS	**0.72***	**0.65***	0.65*	**0.64***	0.66*	0.58*	0.44*	**0.36***	**0.41***	**1.00**†	**1.00**†	**1.00**†	0.01
R-ODNN-RLS	0.73*	0.65*	**0.62***	0.65*	0.64*	0.56*	**0.41***	0.37*	0.43*	**1.00**†	0.16†	**1.00**†	0.00
UCSV	0.89*	0.86*	0.84*	0.89*	0.90*	0.78*	0.75*	0.74*	0.84*	0.00	0.00	0.00	0.00
ERT	0.74*	0.66*	0.66*	0.68*	0.60*	**0.54***	0.42*	0.41*	0.49*	0.17†	0.01	**1.00**†	**1.00**†
LGBM	0.76*	0.68*	0.70*	0.68*	**0.59***	0.56*	0.45*	0.42*	0.48*	0.02	0.00	**1.00**†	0.00
CatBoost	0.76*	0.70*	0.75*	0.71*	0.65*	0.56*	0.46*	0.41*	0.48*	0.04	0.00	0.16†	0.00
NN	0.75*	0.67*	0.67*	0.72*	0.74*	0.58*	0.48*	0.50*	0.53*	0.02	0.00	0.02	0.00
RF	0.73*	0.67*	0.70*	0.68*	0.62*	0.56*	0.45*	0.44*	0.58*	0.04	0.00	0.16†	0.00
RLasso	1.32*	0.98	0.94	1.00	1.22	1.12	1.08	1.12	1.25	0.00	0.00	0.00	0.00
AdaLasso	0.78*	0.69*	0.74*	0.73*	0.88	0.61*	0.51*	0.47*	0.46*	0.00	0.00	0.01	0.00

Notes: The first data row of each panel gives the absolute MAE of the random walk (RW); all other entries are relative to RW. Multi-horizon MCS p-values are computed over per-horizon absolute losses (subscripts denote the maximum horizon $h_{\max}$ included). Bold = best and underlined = second-best competitor per column (RW excluded). Stars denote Diebold–Mariano equal-predictive-accuracy tests against the random-walk benchmark (two-sided, squared-loss): * $p < 0.05$.

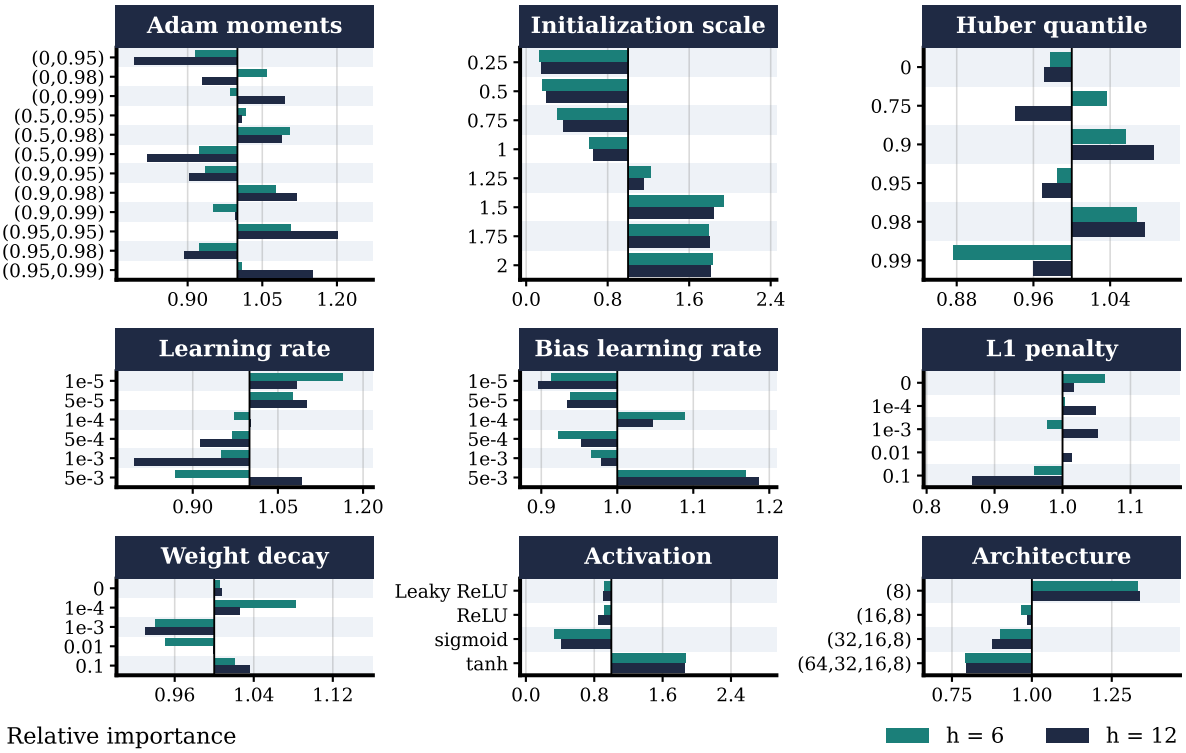

FIGURE 10: Relative importance of the hyperparameter values for U-ODNN-RLS at $h = 6$ and $h = 12$. See Figure 5 for details.

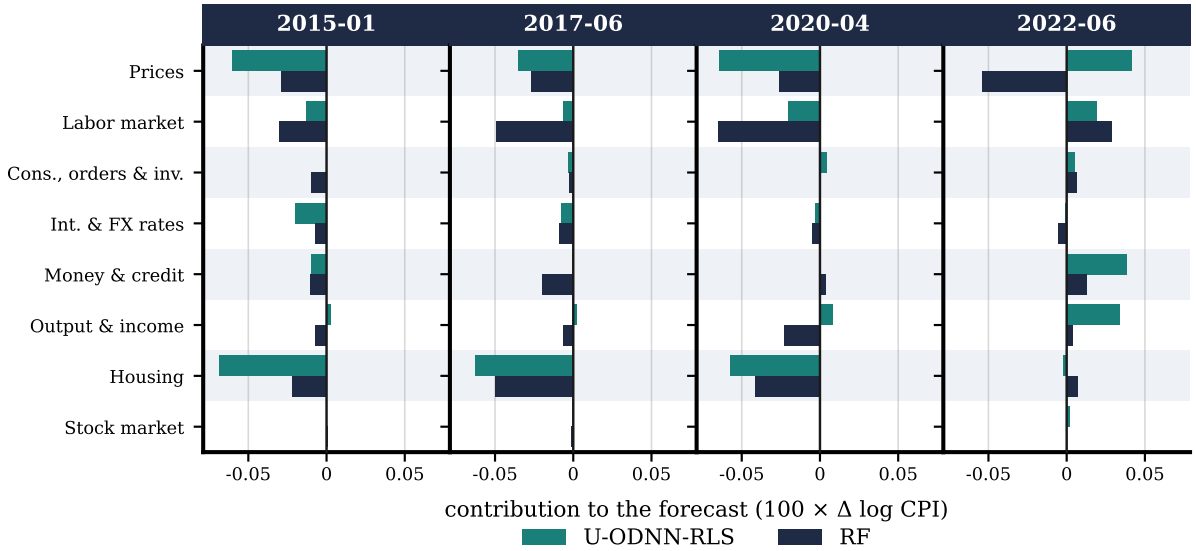

FIGURE 11: Shapley contributions of the eight FRED-MD variable groups to the U-ODNN-RLS (teal) and random forest (navy) forecasts at $h = 12$, for the same months as Figure 9, in units of $100 \times \Delta \log \text{CPI}$. For each model, the contributions sum to its forecast minus its baseline in Table 9. See Figure 9 for details.

TABLE 9: Realized values, forecasts and baseline forecasts at $h = 12$ for the months in Figure 11, in units of $100 \times \Delta \log \text{CPI}$.

		U-ODNN-RLS		RF	
Month	Real.	Fcst	Base	Fcst	Base
2015-01	−0.64	0.09	0.26	0.16	0.28
2017-06	0.07	0.13	0.24	0.13	0.29
2020-04	−0.80	0.12	0.25	0.13	0.28
2022-06	1.29	0.39	0.25	0.28	0.28

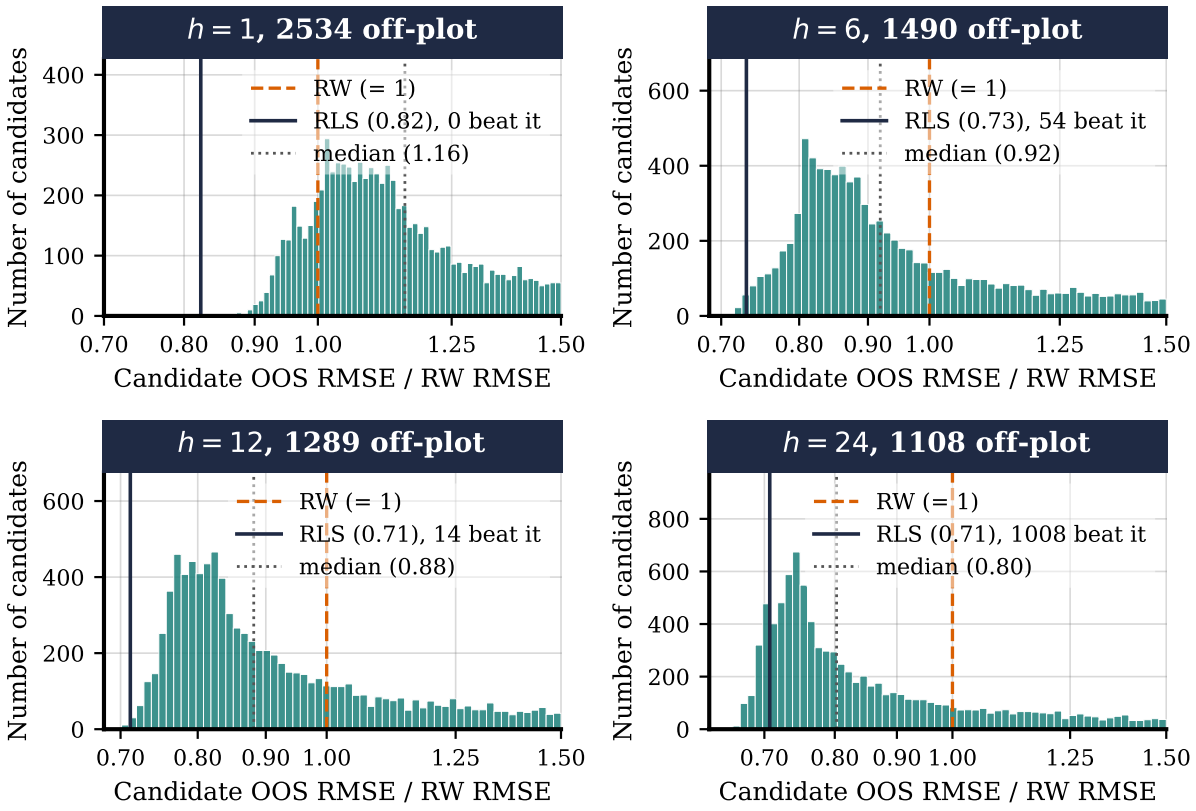

FIGURE 12: Distribution of individual R-ODNN candidate out-of-sample RMSE relative to the random walk benchmark. As Figure 3, for the R-ODNN candidates; the RLS line is R-ODNN-RLS.

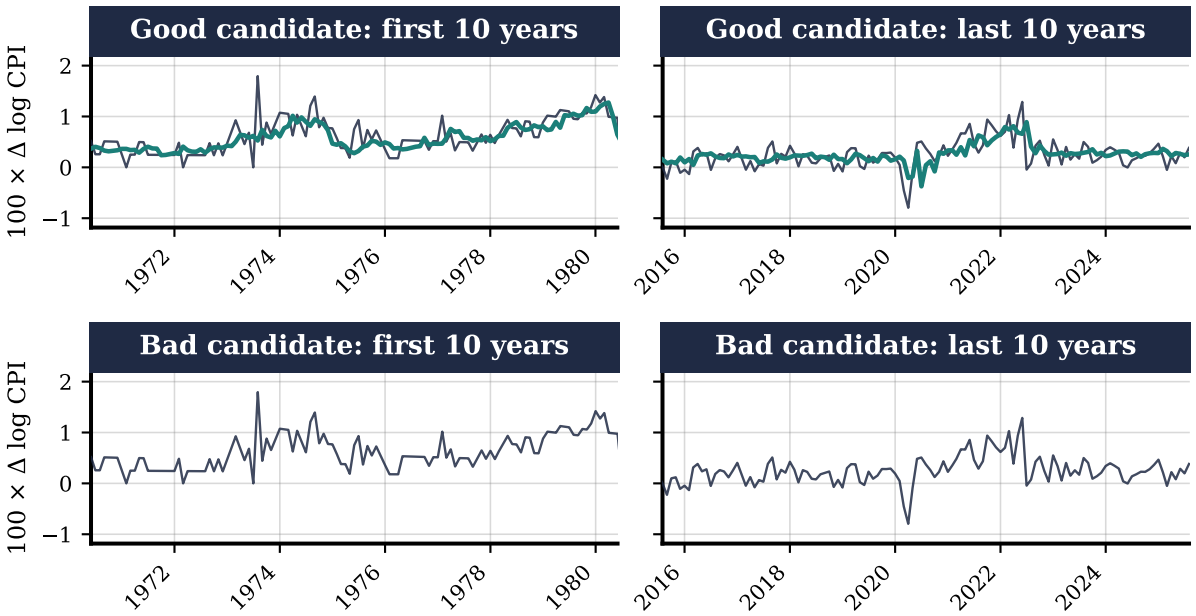

FIGURE 13: Forecast paths of a good (top) and a bad (bottom) R-ODNN candidate at the first horizon. As Figure 4; the first panel starts after the 10-year pretraining window, before which the R-ODNN candidates do not forecast.

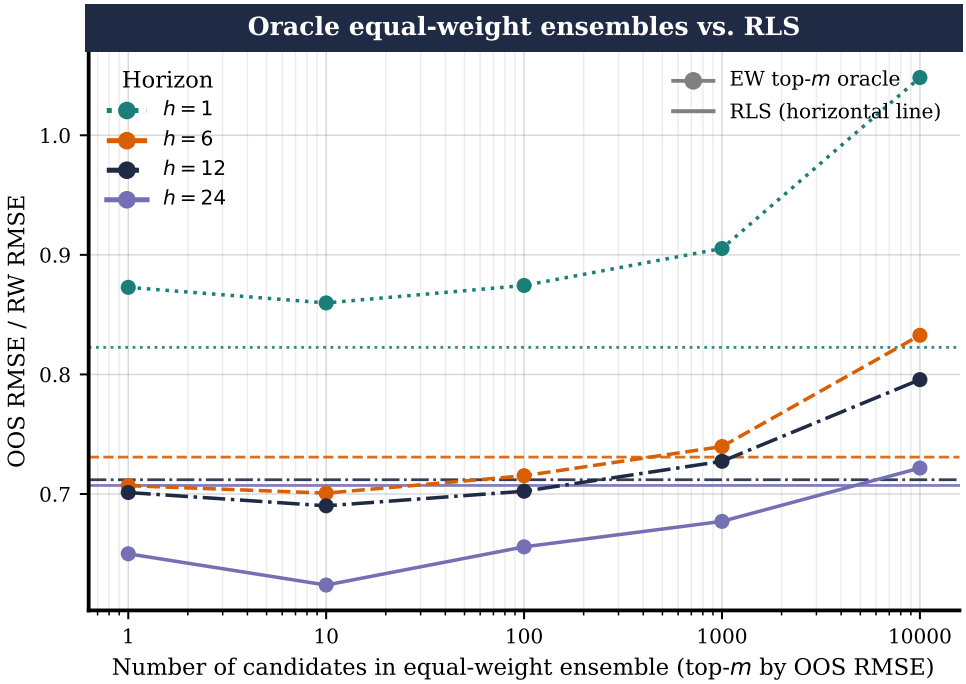

FIGURE 14: Out-of-sample performance of equally weighted oracle combinations of the top R-ODNN candidates against R-ODNN-RLS. As Figure 7, for the R-ODNN candidates.