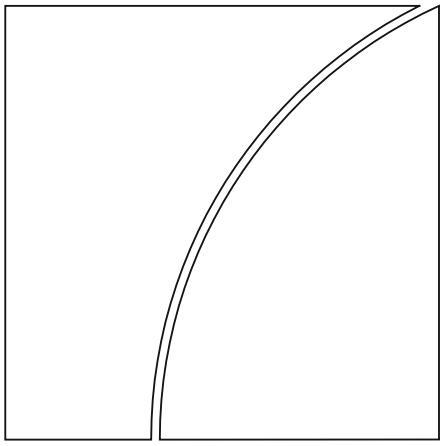


BIS Working Papers

No 1337

Introducing BISTRO: a foundational model for unconditional and conditional forecasting of macroeconomic time series

by Batuhan Koyuncu, Byeungchun Kwon, Marco Lombardi, Fernando Perez-Cruz and Hyun Song Shin

Monetary and Economic Department

March 2026

JEL classification: C32, C45, C55, C87

Keywords: forecasting, scenarios, large language models

BIS Working Papers are written by members of the Monetary and Economic Department of the Bank for International Settlements, and from time to time by other economists, and are published by the Bank. The papers are on subjects of topical interest and are technical in character. The views expressed in this publication are those of the authors and do not necessarily reflect the views of the BIS or its member central banks.

This publication is available on the BIS website (www.bis.org).

ISSN 1020-0959 (print)
ISSN 1682-7678 (online)

Introducing BISTRO: a foundational model for unconditional and conditional forecasting of macroeconomic time series

Batuhan Koyuncu Byeungchun Kwon Marco Lombardi
Fernando Perez-Cruz Hyun Song Shin

March 13, 2026

Abstract

This article introduces the BIS Time-series Regression Oracle (BISTRO), a general purpose time series model for macroeconomic forecasting. Its edge over traditional econometric approaches lies in its ability to deal with generic unconditional and conditional forecasting tasks, without requiring to adjust the model to the macroeconomic tasks being tackled. Building on the transformer architecture underlying LLMs, BISTRO is fine-tuned on the large repository of macroeconomic data maintained at the BIS. We show that BISTRO provides reliable unconditional forecasts for key macroeconomic aggregates and illustrate how using it for conditional forecasting can help unveiling patterns of nonlinearity in the data.

1 Introduction

Predictions of macroeconomic aggregates are a key ingredient to economic policy, especially for monetary policymakers. Traditionally, forecasters relied on time series models, where historical developments of key economic variables and their correlations are used to extrapolate the future path of the variable(s) of interest. According to this paradigm, models are tailored to the specific task at hand, meaning they need to be constructed, estimated and validated for a single problem. When dealing with another task involving a different set of variables in another setting, another model designed specifically for the new problem needs to be built. To use an analogy, the modeller is like a carpenter who has access to a large box of diverse tools. A new task is tackled by figuring out how to use or modify tools that were originally created for another task.

The advent of large language models (LLMs) has popularised a very different approach to problem-solving. The latest generation of LLMs are like a Swiss army knife rather than a box of tools: they can tackle a wide range of problems, even ones that were unforeseen when the model was built and trained. In the industry jargon, LLMs are dubbed "zero-shot learners" or "foundational models".

The ever-growing capabilities of LLMs have opened up the tantalising prospect that a similarly flexible approach would support time series forecasting. Rather than

having to build a bespoke model for a particular task, the same foundational model could be deployed for a wide variety of different tasks. Early proposals of automatic and flexible modelling for time series Hendry and Krolzig (2001) hinged on a necessarily task-specific model selection step. The main advantage of such a model is that forecasting can break free of the inflexibility of traditional econometric approaches, in which specific modelling choices must be imposed from the start.

This article introduces such a flexible model. We name it the BIS Time-series Regression Oracle (BISTRO). BISTRO builds on the machinery underlying LLMs and applies it to the world of economic time series.

The article starts with a general introduction to the mathematical principles underlying LLMs and explores why they have broader applicability beyond the domain of text. Just as LLMs do well in guessing the next word within the broader context of the sentences that precede it, foundational time series models do well in guessing the next realisation of a macroeconomic time series within the broader context of what else has been happening in the economy. The only difference is that they operate on macroeconomic variables rather than words. But mathematically, the task is identical.

We then showcase the forecasting performance of BISTRO. First of all, we do so in the context of an unconditional out-of-sample forecastng exercise for inflation, unemployment and GDP growth, in which we compare BISTROs' performance with that of a simple AR(1) benchmark and that of MOIRAI. We then illustrate how BISTRO can flexibly provide conditional forecasts and create scenarios: in particular we show how a researcher can produce a generic baseline forecast for inflation and then evaluate how conditioning on different explanatory variables (and different assumptions for their evolution) modifies the baseline. Hence, BISTRO constitutes a low-cost and easy-to-use forecasting tool that performs well compared with traditional econometric benchmarks.

BISTRO comes with detailed instructions and pre-compiled scripts on how to operate it, available at https://github.com/bis-med-it/bistro. To facilitate replication and practical use, these scripts can be run in Google Colab, allowing users to upload their own data set and generate baseline and conditional forecasts with BISTRO through a guided workflow.

2 From word prediction to foundational language models

Since the emergence of LLMs, economists have increasingly used them for text-based tasks such as drafting, coding and literature reviews. Korinek (2023) surveys how generative AI can support economic research, while Korinek (2025) explores specific applications, including the replication of existing results. Artificial intelligence (AI) tools are now routinely applied across many areas of economic analysis and research, see for example (Aldasoro et al., 2025), (Aquilina et al., 2025; **?**) (Cao et al., 2024), (Gambacorta et al., 2024), (Gorodnichenko et al., 2023), (Horton, 2023), (Kwon et al., 2024), (Kwon et al., 2025), (Ludwig and Mullainathan, 2024), (Siano, 2025) and (Zarifhonarvar, 2026).

LLMs are valuable because they can handle a wide range of natural language processing (NLP) tasks without any modification to their structure or parameters. Models such as the Generative Pretrained Transformer (GPT) can translate text, summarise documents and write code – even though they were trained for a single and very specific task: predicting the next word in a sequence. The transformer architecture is a deep neural network originally developed for language translation. For further details, see Vaswani et al. (2017). GPT was introduced by Radford et al. (2018) and further evaluated in Radford et al. (2019) and Brown et al. (2020). This ability to perform tasks for which the model received no explicit training is known as zero-shot learning and is what makes LLMs foundational models: they are general purpose tools applicable to a broad set of tasks, possibly unforeseen when the model was created.

The zero-shot capability of transformers is rooted in how they represent words. In any quantitative model, words must first be converted into a vector of numbers, a process known as embedding. A simple approach would assign each word a fixed vector, regardless of context. But consider the word "bond": in isolation, it could refer to a financial instrument, a family connection or a fictional spy (Bank for International Settlements, 2024). A fixed, standalone vector cannot capture this range of use. Transformers address this limitation through contextualised embeddings. Rather than assigning a static vector to each word, the transformer reads the full surrounding text and constructs a vector that reflects the word's specific meaning in that specific context. The word "bond" would thus receive a different vector representation in a sentence about sovereign debt than in one about family relationships or spy fiction. This is achieved through the attention mechanism, a core component of the transformer architecture.[1] Attention allows each word to attend to every other word in the input, weighting their contributions according to relevance. The result is a set of embeddings that encode not just the identity of each word, but its role and meaning within the input sequence.

As the model generates a different internal representation for every new input, it effectively adapts to the questions it is prompted with. This adaptation to the context is at the root of its flexibility. Within the given model, the context-dependent embeddings allow it to behave differently for each prompt. The term "foundational model" refers to such adaptability (a Swiss army knife as opposed to a box of different tools). It is this adaptability that distinguishes transformer-based LLMs from the older generation of expert systems and machine learning models.

3 From econometric models to transformer models for time series

In this section, we apply LLM logic from text to time series to illustrate how a universal time-series model would work. We begin from the econometrician's perspective and progress to an LLM-based machine-learning solution. This walk-through highlights the main difference between econometricians' and machine learners' approaches to time-series estimation, in particular, and to estimation (or learning) in general.

[1] Vaswani et al. (2017) introduced the attention mechanism for transformers, and Alammar (2018) subsequently elucidated its inner workings

Consider an econometrician forecasting inflation. She develops a hypothesis about the drivers of inflation, proposes a parsimonious model consistent with it, and collects the data required for empirical validation. She then assesses the model's errors and checks that the parameters carry significant predictive content.

To do so, she collects R time series with T observations each, arranged in a matrix $\mathbf{X}$ with R rows and T columns. This matrix is her *information set*, ie, consistently with the likelihood principle, data outside the scope of her hypothesis and model add no value. To forecast inflation at time t, she uses:

$$\widehat{x}_{1t} = f(\mathbf{X}_{t-1}, \widehat{\theta}(\mathbf{X}_{t-1})), \tag{1}$$

where $\mathbf{X}_{t-1}$ contains the first $t-1$ columns of $\mathbf{X}$ and $f(\cdot;\cdot)$ is her model. Without loss of generality, the first row of $\mathbf{X}$ represents the variable of interest (eg, inflation). The model parameters θ are estimated through likelihood-based methods (either frequentist or Bayesian) using observed data. For each value of t, a different set of parameters is estimated. If the data come from a stationary process, however, the parameters converge to stable values, and each additional data point yields diminishing returns.

If the econometrician has to tackle a different research question, she typically needs to specify a new model, possibly collect new data, and re-estimate.

A machine learner starts from the same matrix $\mathbf{X}$, splits the first 80% of columns into a training set $\mathbf{X}^{tr}$, and labels the remaining 20% as the test set $\mathbf{X}^{ts}$. She trains a flexible model (eg, a random forest, a support vector machine, or a deep neural network) on $\mathbf{X}^{tr}$ and evaluates its predictions on $\mathbf{X}^{ts}$. The model need not be tailored to inflation forecasting. The machine learner assumes that a flexible model, properly configured (hyperparameters), can solve any estimation task. Mathematically:

$$\widehat{x}_{1t} = h_{\phi}(\mathbf{X}_{t-1}, \widehat{\theta}(\mathbf{X}^{tr})), \tag{2}$$

The estimates for the first 80% of the samples will be in-sample, and those for the last 20% will be out-of-sample. The parameters are only computed once and then left unchanged.

The training-test split reflects a practical trade-off. Training machine learning models involves not only estimating parameters by maximum likelihood but also tuning hyperparameters ϕ through cross-validation, which is computationally demanding. The machine learner assumes that the training set is representative enough to generalise, and that adding columns of $\mathbf{X}$ one at a time would not meaningfully improve out-of-sample performance.

When new data arrive, or a new problem arises, the machine learner reuses the same model architecture and repeats the training and testing process. Model development focuses not on a single problem but on general architectures that work across many problems, with hyperparameter tuning yielding the best solution for each problem.

Econometricians and machine learning practitioners approach forecasting in broadly the same way. Both estimate a model that predicts future data accurately by maximising the likelihood. Machine learners split the data into training and test sets for practical convenience, rather than because they conceive of the forecasting task differently. Both camps also share a common premise: the available information set contains everything needed to solve the forecasting problem at hand.

Foundational time-series models based on the transformer architecture depart from this shared tradition in three important respects. First, they target zero-shot capability: the ability to produce forecasts for any series without retraining or tuning hyperparameters. Second, the model's parameters are estimated on a large, diverse corpus of time series that differs from the data the forecaster ultimately wants to predict. The underlying assumption is that patterns learned across many domains (eg, retail sales, energy prices, financial returns) can be transferred to new forecasting problems. Third, training does not optimise forecasts at a specific horizon. Instead, the model learns to predict randomly masked segments of each series, regardless of the end user's forecasting task. Taken together, these design choices represent a fundamentally different approach: one model, trained once, intended to solve any time-series forecasting problem out of the box. Mathematically:

$$\widehat{x}_{1t} = g_{\phi}(\mathbf{X}_{t-1}, \widehat{\mathbf{W}}), \tag{3}$$

where the function $g_{\phi}(\cdot;\cdot)$ the estimated parameters $\widehat{\mathbf{W}}$ and hyperparameters ϕ remain fixed across all estimation problems, regardless of the dimensions of $\mathbf{X}_{t-1}$, the number of variables to estimate, or whether the data come from economics, climate science, or health. This is why such models are called *foundational models* in machine learning: no changes to the strucuture of the model or to its parameters are needed to perform accurately on new tasks. This property is also known as *zero-shot learning*, because no parameters are adjusted when approaching a new problem.

Most econometricians would expect such a model to perform poorly. Economic data are heterogeneous, and a model with no problem-specific calibration would seem unlikely to deliver reliable results. Yet the same econometrician knows that ChatGPT or Claude handles virtually any NLP task competently, which is a domain arguably as broad as the universe of time-series problems. How does the LLM achieve this? And can the same approach extend to time series?

The neural network implementing $g(\cdot;\cdot)$ is the transformer model. When presented with new data $\mathbf{X}_{t-1}$, the transformer reconfigures how it computes its output, effectively adapting to each estimation problem without changing the weights $\widehat{\mathbf{W}}$. The name "transformer" reflects this capacity to dynamically transform its internal representations.[2] This is achieved through the attention mechanism, which reweights the model's internal representations for each input, yielding accurate solutions across a wide range of tasks. We detail the transformer architecture and the attention mechanism in the next section.

The transformer exhibits foundational capacity when two conditions hold: the neural network is very large (at least hundreds of millions of parameters), and the training dataset is sufficiently large and diverse. To build a foundational time-series model, the LLM machine learner requires access to a new matrix $\mathbf{Z}$ that ideally contains all recorded time series. She trains the model on a subset of series in $\mathbf{Z}$, masks some observations at the end of selected series, and asks the model to predict the masked values. She repeats this process with different subsets until the model generalises well

[2]The term was introduced in Vaswani et al. (2017) and refers to the model's ability to transform sequence representations via the attention mechanism. The name of this neural network is inspired by the TV series and movie franchise of the same name.

across all of them.[3] This process yields the parameter matrix $\widehat{\mathbf{W}}$, which remains fixed for all future tasks.

Crucially, the goal of training is not to memorise every problem that might arise. The set of possible problems is too large, and new tasks will emerge that the model never saw during training. Instead, the goal is to learn the underlying regularities of time series, the equivalent of grammar and orthography in language, and to absorb enough general patterns to combine them when facing novel tasks. We expect a well-trained transformer to learn a *common language* of time series, thereby yielding a genuinely universal model.

The transformer's universality holds only if the training data cover the domain of interest. Current text-based LLMs contain hundreds of billions of parameters and have been trained on tens of trillions of words drawn from the internet across many languages. A model trained only in English cannot translate into Spanish or respond in Korean. Likewise, text-based LLMs cannot handle time-series tasks because they have not been trained on time-series data.[4]

To build a foundational time-series model, one must collect broad and representative time-series data and adapt the transformer to accept time-series inputs rather than words. The training data must span all domains and tasks the model is expected to address, with proportions that support accurate predictions across these domains and tasks. Woo et al. (2024) adapted the transformer architecture for time-series analysis and assembled 27 billion observations from diverse time series to train a universal time-series transformer with hundreds of millions of parameters. This model, called MOIRAI, performs well on standard machine learning benchmarks. When we applied it to macroeconomic time series, however, it underperformed and largely ignored the provided covariates. This is likely because the training data in Woo et al. (2024) primarily came from the energy, transportation, and climate domains, with economic and financial data accounting for roughly 0.1% of the total. This data imbalance need not be a permanent limitation, but it matters: if the non-economic series do not share sufficient structure with macroeconomic data, the model fails to transfer what it has learned.

We would likely need a substantially larger neural network and training dataset to develop a foundational model that can be applied to all time-series problems and domains. Instead, we adapted the MOIRAI architecture and fine-tuned it on macroeconomic data collected at the BIS to produce the BIS time-series regression oracle (BISTRO), a foundational model for macroeconomic time series described in Section 5. The need to fine-tune MOIRAI illustrates a broader point: achieving a truly universal time-series model may require far more data (trillions of observations) and a much larger model (tens of billions of parameters), just as text-based LLMs did. Until that bar is reached, sectoral foundational models, purpose-built for specific domains, are a practical and principled alternative. BISTRO is BIS's solution for macroeconomic time series.

We also trained MOIRAI from scratch on BIS data alone, erasing the original Woo

[3] Making this training efficient requires many engineering choices, including decisions about which series to feed jointly into the model. We abstract from these details here.

[4] Some work (Jin et al., 2024; Zhou et al., 2023; Faria-e Castro and Leibovici, 2024) has attempted to adapt text-based LLMs for time-series prediction, but such models are generally considered inadequate, because text and time series share little structure beyond their sequential nature.

et al. (2024) pre-training. This yielded weaker results (not reported in this document), confirming that our macroeconomic dataset alone is insufficient for stable pre-training. The combination of MOIRAI pre-training and BIS fine-tuning achieves the best performance for BISTRO.

One final note on expectations. Just as one should not accept ChatGPT output uncritically, the output of a foundational time-series model is best treated as a strong first estimate. When the problem falls within the model's training domain, a tailored econometric model may offer little additional gain. When the foundational model's output is lacking, the practitioner should consider either fine-tuning the foundational model on domain-specific data or developing a bespoke model from scratch. This applies to BISTRO and any other foundational model (for time series).

4 Foundational models for time series analysis

Foundation models represent a paradigm shift in machine learning (Bommasani et al., 2021): large-scale transformer architectures Vaswani et al. (2017) pre-trained on vast datasets and adapted for a wide range of downstream tasks Devlin et al. (2019) have shown the capacity for zero-shot learning Radford et al. (2018). These models learn general-purpose representations of temporal patterns and dependencies across domains and frequencies Liang et al. (2024). Their scalability enables high predictive accuracy across tasks such as nowcasting, forecasting and scenario construction. Scenario analysis is a particularly important use case, as traditional methods struggle to condition forecasts on variables not present in the estimation dataset.

From text models to time-series models. Early attempts to apply LLMs to time series, such as Time-LLM (Jin et al., 2024) and One Fits All (Zhou et al., 2023), repurposed pre-trained text models. The transformers trained on text are unable to capture the nuances of time series, underperforming because text-trained transformers lack the inductive biases needed to capture the distinct dynamics of numerical sequences (Tan et al., 2024).

A second wave of research developed purpose-built *univariate* time-series foundation models, including LagLLaMA (Rasul et al., 2024), TimeGPT (Garza et al., 2024), Time-MoE (Shi et al., 2025), TimeFound (Xiao et al., 2025a), TimeFM (Das et al., 2024), Chronos (Ansari et al., 2024), PatchTST (Nie et al., 2023), Moment (Goswami et al., 2024) and Pathformer (Chen et al., 2024). These models achieve strong forecasting performance for individual series but cannot accommodate covariates, which is a fundamental requirement in macroeconomic analysis, where forecasts typically depend on several related variables.

Applications to economics and finance. Several studies have applied foundation models to economic and financial data (Yu et al., 2023; Xiao et al., 2025b; Allard et al., 2024). Most rely on a text-based LLM to predict asset prices or inflation from textual sources such as news articles (Yu et al., 2023; Allard et al., 2024). Xiao et al. (2025b) takes a different approach, using retrieval-augmented generation to match time

series, but the method does not scale to multivariate settings or support probabilistic modelling.

Multivariate foundation models. For macroeconomic applications, models must handle multiple series jointly. Two recent architectures meet this requirement: MOIRAI (Woo et al., 2024) and Chronos 2.0 (Ansari et al., 2024).

MOIRAI uses a masked encoder to model multivariate time-series data. Trained on the Large-Scale Open Time Series Archive (LOTSA), which is a corpus of more than 27 billion observations spanning diverse domains. MOIRAI supports cross-frequency learning, an arbitrary number of input series and flexible probabilistic outputs. These properties make it well-suited to macroeconomic data, which exhibit complex temporal dependencies and require distributional forecasts.

Chronos 2.0 takes a different design approach. Rather than allowing full attention across all series, it alternates self-attention along the time axis with group attention across related series at each patch index. This design scales linearly rather than quadratically in the number of series. The model is first trained on univariate data and then fine-tuned on synthetic multivariate series. A drawback relative to MOIRAI is that Chronos 2.0 cannot attend jointly across different series and time patches. This limits its capacity to extract information from the history of one variable to predict another, which is a capability central to many econometric tasks.

Finally, Tiny Time Mixers (TTMs) Ekambaram et al. (2024) achieve strong zero-shot and few-shot performance with a parameter-efficient architecture. TTMs, however, provide only point estimates and require task-specific fine-tuning to incorporate covariates, severely limiting their capacity as a foundational model for macroeconomic time-series tasks.

Foundation models in macroeconomics. Foundation models have begun to address complex forecasting challenges in macroeconomics Carriero et al. (2020). Their ability to handle nonstationary dynamics and capture nonlinear relationships among economic variables makes them attractive for policy applications in uncertain environments. In particular, the capacity for zero-shot and few-shot learning Liang et al. (2024) across different regions and economic conditions reduces the need for separate models for each economy—from advanced to emerging markets.

Remaining gaps. Despite this progress, standardised datasets and evaluation frameworks for economic time series remain scarce. Most studies rely on small datasets: for example, Carriero et al. (2020) uses 120 US time series. Research has focused on unconditional forecasting and nowcasting, without addressing the construction and evaluation of conditional scenarios. Performance has generally been assessed without distinguishing between normal periods and episodes of high volatility, structural breaks or regime changes.

Our dataset and benchmark address these limitations. We provide a comprehensive macroeconomic time-series dataset spanning multiple countries, designed for benchmarking nowcasting, forecasting and scenario analysis. We compare traditional statistical methods, pre-trained foundation models and foundation models trained on macroe-

conomic data. This makes our contribution valuable for central banks and policymakers with limited machine-learning expertise or computational resources, who can use our baseline models without additional training to obtain reliable forecasts and scenarios for their own economies and the external environment.

4.1 Attention mechanism for text or univariate time series

The attention mechanism is the core building block of the transformer architecture behind LLMs Vaswani et al. (2017). In standard time series models, eg, a vector autoregression (VAR), the coefficients linking past observations to forecasts are fixed once estimated. The attention mechanism replaces these fixed coefficients with context-dependent weights, assigning greater importance to the past observations most relevant to the prediction at hand.

The mechanism was originally designed for machine translation, where a single set of weights determined which source words mattered for each translated word. Vaswani et al. (2017) found that a single attention head was insufficient for longer, more complex texts.[5] They proposed running several attention mechanisms in parallel (ie, multi-head attention) so that each head can capture a different type of dependency (syntactic, semantic or positional, for instance). The transformer architecture stacks multiple layers of multi-head attention, with each layer refining the embeddings from the previous one.[6]

Intuition. An attention head operates through three learned linear projections, called the query, key and value. The query asks, "What information do I need?"; the key signals, "What information do I carry?"; and the value delivers the actual content. For a given element in the sequence (eg the i-th word or observation), the model computes the dot product between that element's query vector and the key vectors of all preceding elements. A larger dot product indicates stronger dependence. A normalising function converts these raw scores into non-negative weights that sum to 1. The contextualised representation of the i-th element is then the weighted sum of the value vectors, with weights determined by these data-dependent weights.

For an econometrician, this mechanism resembles kernel regression or a time-varying parameter model, with a key difference. In conventional time-varying parameter models, the weights are estimated from historical data and then fixed. The attention mechanism, by contrast, recomputes the weights for every new input. The model therefore adapts to structural breaks and shifting relationships without re-estimation, which is a valuable property for macroeconomic forecasting, where regimes can change abruptly.

Formal description. Each input element w_i is mapped to a numerical vector through a linear projection, $\mathbf{x}_i = \mathbf{E}w_i$. From this initial embedding, three vectors are derived

[5] An attention head computes a set of attention weights and produces contextualised embeddings. Multi-head attention runs several such computations in parallel, each with its own set of weight matrices for the key, query and value projections.

[6] Alammar (2018) provides a step-by-step visual guide to the transformer, with clear examples that serve as a useful starting point.

through separate linear transformations: the query, $\mathbf{q}_i = \mathbf{W}^Q \mathbf{x}_i$, the key, $\mathbf{k}_i = \mathbf{W}^K \mathbf{x}_i$, and the value, $\mathbf{v}_i = \mathbf{W}^V \mathbf{x}_i$. The attention weight that element j exerts on element i is then:

$$\alpha_{ij}(\mathbf{q}_i, \mathbf{k}_1, \cdots, \mathbf{k}_i) = \frac{\exp\left(\mathbf{q}_i^\top \mathbf{k}_j\right)}{\sum_{l=1}^{i} \exp\left(\mathbf{q}_i^\top \mathbf{k}_l\right)}. \tag{4}$$

When α_{ij} is large, element j carries significant information for element i; when it is close to zero, element j contributes little. The normalising function, known as the softmax function, ensures that the weights are non-negative and sum to 1. The contextualised embedding is the weighted sum of the value vectors:

$$\mathbf{z}_i = \sum_{j=1}^{i} \alpha_{ij}(\mathbf{q}_i, \mathbf{k}_1, \cdots, \mathbf{k}_i)\, \mathbf{v}_j \tag{5}$$

The contextualised embedding depends on all preceding elements and defines a single attention head. The three projections partition the workload: the value vector carries the information content, while the key and query vectors determine which of that content is relevant. Each new input produces different weights and, consequently, a different embedding for every element. This context sensitivity is what gives transformer models their adaptability.

4.2 A universal time-series forecasting transformer

The *Masked encOder-based unIveRsAl* (MOIRAI) time series forecasting transformer is designed as a *universal forecaster*: a single pre-trained model that handles heterogeneous datasets and downstream tasks without per-dataset customisation Woo et al. (2024). Time-series foundation models face three practical hurdles that distinguish them from their counterparts in vision or text: (i) cross-frequency learning (from minutely to yearly sampling), (ii) any-variate inputs (arbitrary numbers of target and covariate series) and (iii) flexible probabilistic outputs (because distributional supports and shapes vary across domains). MOIRAI addresses all three within a single architecture, trained once on a large multi-domain corpus and then applied broadly.

The model retains a masked-encoder transformer backbone but introduces three innovations. First, multi-patch-size input/output projections assign larger patches to high-frequency series and smaller patches to low-frequency ones, specialising the projections by frequency. Second, any-variate attention flattens a multivariate sequence across time *and* variates, encodes time with rotary positional embeddings (RoPE) and learns a binary bias that distinguishes within-series from cross-series attention. This preserves permutation invariance and accommodates arbitrary numbers of variates.

From words to patches. The attention mechanism in MOIRAI, which underpins our model BISTRO, follows the procedure described above with adaptations for time-series data. Consider R time series, each with n observations: y_{rj}, i runs from 1 to R and j from 1 to n. One could draw a direct analogy between individual observations y_{rj}

(or equivalently w_i) and individual words. But just as language models process words rather than individual characters, the MOIRAI model groups p consecutive observations into a fixed-length patch: $\mathbf{p}_{ri} = \big(y_{r,\,p\cdot(i-1)+1},\; y_{r,\,p\cdot(i-1)+2},\; \ldots,\; y_{r,\,p\cdot i}\big)$. Each patch serves as the minimal unit processed by the transformer, which is analogous to a single word in language modelling. MOIRAI and BISTRO process p samples of the time series as a unit, and their predictions are grouped into multiple consecutive samples of size p.

Attention with two indices. Because the model operates over multiple time series, the attention mechanism must accommodate two indices. Each input element $\mathbf{p}_{ri}$ is mapped to a numerical vector through a linear projection, $\mathbf{x}_{ri} = \mathbf{E}\,\mathbf{p}_{ri}$. The model then computes the query $\mathbf{q}_{ri} = \mathbf{W}^Q\mathbf{x}_{ri}$, the key $\mathbf{k}_{ri} = \mathbf{W}^K\mathbf{x}_{ri}$ and the value $\mathbf{v}_{ri} = \mathbf{W}^V\mathbf{x}_{ri}$ vectors. Here, r indexes the time series and i the time stamp. The attention weights therefore have four indices:

$$\alpha_{rs,ij} = \frac{\exp\Big(\mathbf{q}_{ri}^\top\,\mathbf{R}_{i-j}\,\mathbf{k}_{sj} \;+\; u^{(1)}_{r=s} \;+\; u^{(2)}_{r\neq s}\Big)}{\displaystyle\sum_{t,\,l}\exp\Big(\mathbf{q}_{ri}^\top\,\mathbf{R}_{i-l}\,\mathbf{k}_{tl} \;+\; u^{(1)}_{r=t} \;+\; u^{(2)}_{r\neq t}\Big)} \tag{6}$$

This attention mechanism differs from the original text-oriented version in three ways:

1. Two scalars were added to distinguish between same-series and cross-series observations: $u^{(1)}_{r-s}$ and $u^{(2)}_{r\neq s}$, where r is the index of the query vector and s the index for the key vector. These scalars allow the model to weight its own series history differently from cross-series information. Using only two constants, regardless of the number of series, means the model is not constrained in how many series it can process. It also ensures that reordering the series does not change the forecast.

2. $\mathbf{R}_{i-j}$ is a rotation matrix that encodes the temporal distance between two patches. The index i represents the time index for the query vector, and j represents the time index for the key vector. $\mathbf{R}_{i-j}$ indicates to the model the temporal separation between the two observations, which is essential for capturing lead-lag relationships.

3. The normalisation in the denominator runs over both time and series dimensions for the key vector, so that the attention weights sum to one across all patches and all series. This enables the model to draw on both the history of the target variable and the evolution of covariates when forming its prediction.

The updated embedding for each element is computed as:

$$\mathbf{z}_{ri} = \sum_{s,\,j}\alpha_{rs,ij}\,\mathbf{v}_{sj} \tag{7}$$

The rest of the transformer architecture follows the standard design, as explained by Alammar (2018).

Training and performance. To enable universal training, the authors curate LOTSA, spanning nine domains with more than 27 billion observations, and train three model sizes (approximately 14M, 91M and 311M parameters). Training alternates between random context/horizon sampling and sequence packing to improve efficiency. On the Monash Time Series Forecasting Benchmark, MOIRAI outperforms strong baselines that are typically trained per-dataset. Ablations show that each proposed ingredient (eg, multi-patch-size projections, any-variate attention, and the mixture distribution) contributes materially. The mixture model yields narrower prediction intervals than those of a symmetric Student's-t distribution when forecasting sharp peaks.

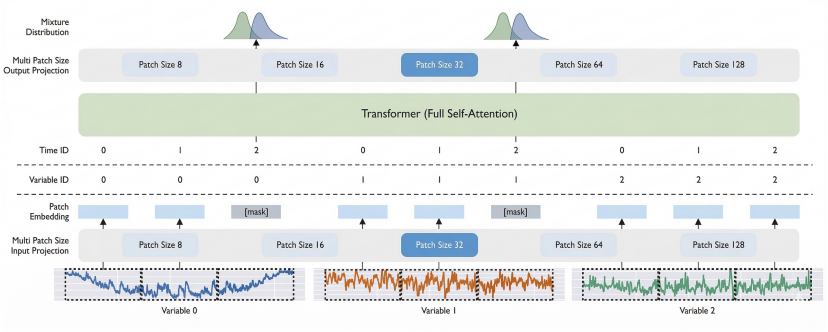


Figure 1: MOIRAI architecture reproduced from the second figure in Woo et al. (2024).

5 Training BISTRO

5.1 Dataset

The full internal data set is made up of 4,925 time series across 63 economies, observed from 1970 through 2024. The training and evaluation window is set at 1984–2024, which is intended to improve coverage consistency. The data set mirrors the real-world information set used by central bank economists and forecasters: indicators arrive at different frequencies and with publication lags. So the data set includes series with varying periodicity and release lags.

The series are arranged into macro blocks to ensure broad coverage of macro-financial conditions. The biggest categories are exchange rates (1,699 series), prices and earnings (1,130), domestic interest rates and bond yields (731) and national accounts (554). Smaller macro blocks broaden coverage, reaching real economy indicators as well as monetary and balance sheet indicators. The topics they cover include labour market data (325), monetary aggregates (235), output indicators (153), demand indicators (83) and items from central bank balance sheets and income statements (15).

The data set has a mixed-frequency structure, and it supports analysis across different time horizons. Of the total 4,925 series, 2,151 are monthly, 1,314 are quarterly, 861 are annual, 536 are daily, 61 are weekly, and two are semi-annual. Country coverage is broad, with dense coverage in several major economies and regions, thereby support-

ing cross-country comparisons. The series are spread across other advanced economies (1,020), Asia (368), Latin America (318), the G3 major advanced economies (309), emerging Europe (291) and Africa and the Middle East (206).

5.2 Data preprocessing

To ensure consistency and facilitate model training, we have used the appropriate transformations for each variable depending on its properties. These transformations are critical for achieving stationarity, managing scale differences, and capturing the relevant dynamics of each economic indicator.

Data transformation: We use the series in four different configurations, which are the most relevant in most econometric analysis. We use the time series as is: Level (no transformation) applied to variables that are already stationary or represent rates (e.g., interest rates, unemployment rates); First difference (Δx_t) applied to nonstationary series where the change between consecutive observations is relevant (e.g., bond yields); Log of the time series ($\log(x_t)$) to consider series expanding over several orders of magnitudes; and Log difference ($\Delta \log(x_t) = \log(x_t) - \log(x_{t-1})$) applied to variables with exponential growth patterns or where percentage changes are more interpretable than absolute changes (e.g., GDP, price indices). The difference transformations are applied at the frequency level of the time series, rather than at the daily level.

Temporal alignment: After applying the transformations, we face another key challenge in economic forecasting, which is handling time series with different frequencies and publication lags. For each univariate variable X, we apply a daily forward-fill operation $\mathcal{F}$ that propagates the last observed value forward in time: $X_t = \mathcal{F}(X_\tau) = X_\tau$ where $\tau = \max\{s \leq t : X_s \text{ is observed}\}$. This enables consistent temporal alignment across variables with different frequencies, allowing our model to be used with a daily frequency that is consequential for the scenario planning application. For example, quarterly GDP values are forward-filled as $X_t^{GDP} = \mathcal{F}(X^{GDP}_{\lfloor t/D_Q \rfloor \cdot D_Q})$, while monthly inflation is treated as $X_t^{INF} = \mathcal{F}(X^{INF}_{\lfloor t/D_M \rfloor \cdot D_M})$, where D_Q and D_M represent, respectively, the number of days in each quarter and month. We account for reporting lags by shifting time indices: $\tilde{X}_t = \mathcal{L}(X_t, \lambda) = X_{t+\lambda}$ where λ represents the reporting delay in days. This ensures that our models only use information that would have been available at each point in time. After applying these steps, on any day t, our feature vector consists of target variables and covariates, e.g. $\mathbf{X}_t = \{X_t^i\}_{i=1}^d$. This unified representation allows models to effectively integrate information across different economic variables regardless of their original frequency or publication schedule.

5.3 Training/validation/test configurations

Train, validation, and test splits: Using our dataset, we use time series from 1984 to 2024, encompassing multiple business cycles, structural economic changes, and periods of both stability and crisis. To establish a reproducible training and evaluation

protocol, we implement a rolling window approach for splitting the train test. We designate specific years (1995, 2005, 2015, and 2023+) as test periods, each representing different economic regimes. For each test period, the training data comprises all available observations from 1984 to 2024, excluding the designated test years, to ensure that there is no information leakage between train and test splits. We also use training data for validation purposes. This approach offers several advantages: it allows evaluation of model performance across different macroeconomic settings (pre- and post-Great Recession, low and high inflation periods, etc.); moreover, it mitigates the risk of models overfitting to specific economic conditions; and it provides insight into how various forecasting methods perform when challenged with structural breaks or regime changes in economic relationships. By evaluating models in the diverse test periods, we can assess both their predictive accuracy and their robustness to different economic circumstances, providing a more comprehensive benchmark for macroeconomic forecasting capabilities.

Model configuration: For our implementation of BISTRO, we made a strategic adaptation to optimize MOIRAI for macroeconomic forecasting.

Although the original architecture employs multiple patch-size projection layers tailored to different frequencies, this approach becomes problematic when handling multivariate economic data where channels exhibit multiple frequencies simultaneously. To address this limitation, we standardized our configuration by selecting a uniform patch size of 32 time steps (days), a choice that effectively captures a complete monthly cycle in our daily-frequency preprocessed data (see Section 5.1). This standardization provides an optimal balance: it preserves sufficient granularity for high-frequency indicators while providing adequate context for identifying patterns in slower-moving economic variables. Below we provide details for training, validation, and testing of BISTRO.

Training: We train our foundation model $\mathcal{F}_\theta$ using a masked feature prediction approach as detailed in Appendix A. For each training iteration, we process batches of multivariate time series data. Each batch element $\boldsymbol{Z}^{(i)} = (\boldsymbol{X}^{(i)}, \boldsymbol{C}^{(i)}) \in \mathbb{R}^{4 \cdot d_{z_i}}$ consists of target variables and their associated covariates, with d_{z_i} distinct time series, that is, features. The dimensionality is scaled by a factor of 4 due to the transformations applied during preprocessing (see subsection 5.1).

1. **Feature sampling:** We sample up to $\mathcal{L} = 14$ distinct time series from available features (or use all available if $d_{z_i} < \mathcal{L}$). This process yields $4 \cdot \mathcal{L}$ time series in total due to the transformations defined in Section 5.1.
2. **Transformation selection:** For each sampled feature, we randomly select a single transformation from set $\mathcal{T}$ rather than applying all transformations. This approach encourages the model to learn robust representations across different statistical properties while maintaining computational efficiency. We end up with $\mathcal{L}' < 4 \cdot \mathcal{L}$ time series after this step.
3. **Patching:** We segment each time series into patches of size 32, ensuring a minimum of 48 patches per feature to provide sufficient context (i.e., over 4 years worth of

context).

4. **Random masking:** Out of $\mathcal{L}'$ series, we randomly select $\mathcal{L}_m$ to be masked and used in the loss computation. For these selected series, we mask between 1.5% and 7% of patches from the end, e.g. the most recent time steps, simulating real-world forecasting conditions.
5. **Leakage prevention:** We create a temporal consistency mask ($\mathcal{M}_j^{\text{tc}}$) that identifies the values appearing in both the history and the prediction patches due to the forward-filling process. This mask will exclude repeated observations from the loss computation, ensuring that the model cannot access target values through temporal misalignment.
6. **Loss computation:** When computing the loss, we employ a combined masking strategy that incorporates both a transformation-selection mask ($\mathcal{M}_j^{\text{tr}}$) and our temporal-consistency mask ($\mathcal{M}_j^{\text{tc}}$). The transformation-selection mask ensures that only one transformation per available feature contributes to the loss and excludes certain high-frequency series such as commodities and daily exchange rates. The temporal-consistency mask eliminates duplicated values that appear in both historical context and prediction windows due to forward-filling operations, ensuring the model learns genuine forecasting patterns rather than exploiting data contamination across temporal boundaries.

Validation: We use early-stopping by using the validation loss. In Step 4, instead of randomly selecting the features to mask, we use a predefined subset of the features to perform sampling $\mathcal{S}_{\text{val}}$ to ensure consistent evaluation across iterations. This enables reliable early stopping when optimization plateaus for defined tasks, e.g. forecasting GDP, unemployment, or inflation. We provide more details in Appendix A.

Testing: When testing the model, we use pre-defined tasks, e.g. different sets of target and covariate time-series, including history and prediction window lengths. We provide these details in Appendix A.

In all cases, for final predictions, we generate 25 samples from the probabilistic output layer and aggregate them to produce robust forecasts. In case of predictions of a lower frequency variable, we use daily predictions until the next availability date and aggregate them to get a single prediction corresponding to the original frequency.

The combination of uniform patch size, strategic masking, and careful attention to frequency alignment enables our model to effectively learn from heterogeneous economic time series without the complexity of multiple projection layers for different frequencies, while still capturing the essential temporal dynamics across variables with diverse reporting cadences. The attention mechanism operates efficiently across these 32-day patches, allowing the model to capture both short-term fluctuations and medium-term trends relevant to macroeconomic analysis while maintaining computational efficiency.

6 How to operate BISTRO

BISTRO was designed as a ready-to-use and off-the-shelf tool for macroeconomic forecasting. This section describes the workflow, which comprises three stages: data preparation, parameter modelling, and forecasting and evaluation (Graph 2). A Google Colab notebook supports replication, with sample scripts and step-by-step code for the entire workflow, including rolling-window forecast generation.

In the data preparation step, the user aligns the target variable (eg inflation or unemployment) with selected covariates (eg oil prices or exchange rates) along the time dimension. Indicators released at different frequencies are converted to daily by carrying forward the last known value. This ensures that the data set contains only information available as of each date. For example, second quarter GDP appears in the data set only on its release date in the third quarter.[7] The result is a (pseudo) real-time information set that mirrors the operational environment faced by central bank modellers and prevents forecasts from using data that were not yet available.

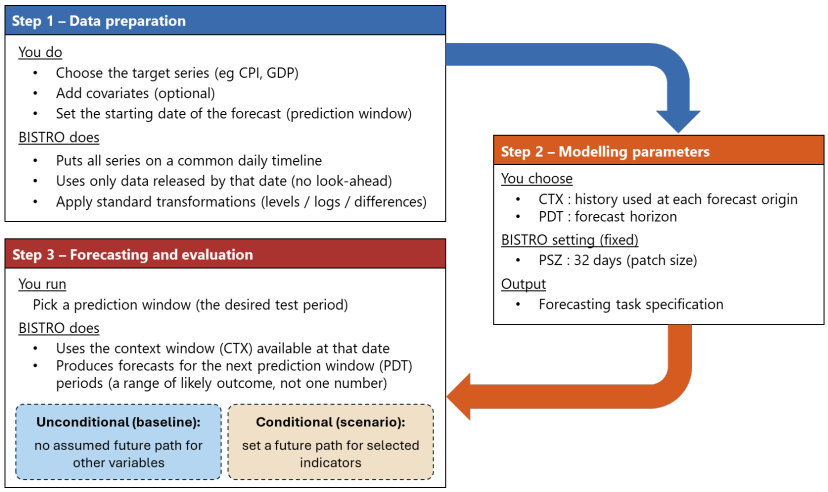


Figure 2: Workflow for BISTRO. CTX = context window length; PDT = prediction window length; PSZ = patch size (fixed to 32). The workflow has three steps. Step 1 builds the real-time data set for each forecast origin (forecast date): all series are put on a common daily timeline, and BISTRO uses only data released by that date (no look-ahead), carrying forward the last available value until the next release. Step 2 defines the task by choosing context length (CTX) and prediction length (PDT). Step 3 runs a rolling-origin back test over the evaluation window, produces probabilistic forecasts and checks accuracy using out-of-sample errors (forecast errors) and standard measures (eg root mean square error, or RMSE). Forecasts can be unconditional (baseline) or conditional (scenario) based on an assumed path for selected indicators.

The aligned data set is then passed to the model interface, which automatically

[7]For simplicity, we abstract from data revisions. While revisions are an important issue for GDP, most other macroeconomic time series are revised only sporadically and by small amounts.

handles the required preprocessing. It applies standard data transformations (levels, logs, differences), imputes missing values due to publication lags and determines the number of past observations used for each prediction, based on the user-defined context window length (CTX). For instance, a context window of 100 patches corresponds to roughly 8.75 years of historical data. Users do not need to manually transform or reformat the data.

A simple example illustrates how running the model mirrors operational forecasting. Suppose the goal is to forecast inflation from 2023 through end-2025, estimating the next 12 months of inflation (12 patches of 32 days) at each forecast date, using 20 years of history (228 patches). BISTRO is first prompted with observations from 1 January 2003 through 31 December 2022 and then predicts inflation for the next 12 months. It then shifts the data window forward by one patch, using data from 1 February 2003 through 31 January 2023 to forecast the next 12 months. This process repeats until the end of the sample.

BISTRO automates the entire estimation process and calculates prediction errors at each step. It benchmarks the results against an AR(1) model, enabling a quick assessment of forecast quality relative to a common benchmark. In the example, the user supplies consumer price index (CPI) inflation data for the country or area of interest over the period 2003–25, sets the CTX to 228 patches and the prediction window length (PDT) to 12 patches. The AR model is trained on data through December 2022 and is updated monthly, while BISTRO uses all 228 context patches at each forecast origin without retraining.

To produce forecasts that also depend on covariates such as oil prices or exchange rates, the user adds these series and aligns them with the target variable. BISTRO then conditions its prediction on both the past values of the target and the explanatory variables (Graph 3A). BISTRO can also produce conditional forecasts. To this end, the user fixes the future path of one or more covariates and feeds them to the model (Graph 3B). For example, BISTRO can forecast 2026 inflation under alternative oil price trajectories. This allows researchers to explore different macroeconomic scenarios in a straightforward way.

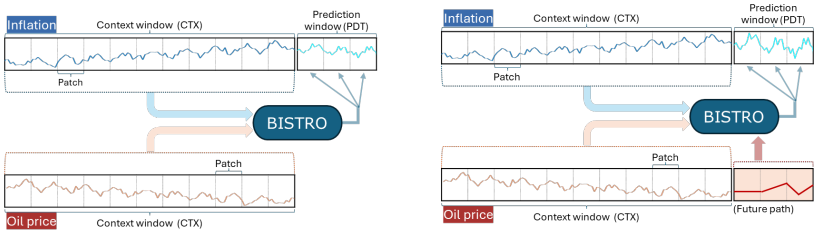


Figure 3: Forecasting modes in BISTRO: baseline and scenario. In panel A, BISTRO produces a baseline inflation forecast using only information available at the forecast date. In panel B, the inflation forecast is updated by imposing an assumed path for oil prices over the forecast horizon. The difference between the two forecasts reflects historical co-movements in the data, not causal effects. Source: Authors' elaboration.

7 Forecasting performance and scenario analysis

7.1 Unconditional forecasts

In this section, we validate the reliability of the forecasts provided by BISTRO through a simple out-of-sample univariate forecasting exercise. The objective is to establish a clean and transparent comparison against standard time-series models—namely a simple AR(1)[8] as well as the MOIRAI (base) foundation model.

While BISTRO can flexibly predict any macroeconomic time series, we illustrate its performance using inflation, GDP growth and unemployment. More specifically, the basic forecasting task consists of predicting monthly inflation (measured as year-on-year (YoY) growth in the consumer price index (CPI)), monthly unemployment and quarterly YoY GDP growth, at short- and medium-term horizons. We consider one-step-ahead ($h = 1$), three-steps-ahead ($h = 3$), six-steps-ahead ($h = 6$) and two-steps-ahead ($h = 12$) forecasts; these are horizons of typical relevance for monetary policy makers.

The forecasting performance is evaluated on four distinct out-of-sample evaluation windows: 1995, 2005, 2015, and 2023+. These windows correspond to markedly different macroeconomic regimes, ranging from the pre- and post-Great Financial Crisis period to the recent episode of elevated inflation. This design allows us to assess model performance across periods characterised by different macroeconomic regimes and levels of volatility.

The evaluation windows are out-of-sample in the sense that models are trained using an expanding sample that starts in 1984 and excludes observations belonging to the corresponding test period. Forecasts are then generated sequentially over the test window as it progresses forward in time. Forecast accuracy is measured using relative root mean squared forecast errors (R-RMSFE) with respect to the AR(1) benchmark and averaged across evaluation windows within each group.

To ensure comparability across models, parameters are estimated once at the beginning of each evaluation window and kept fixed throughout the forecasting exercise. This applies also to the AR(1) benchmark, whose coefficients are not updated throughout the evaluation windows. The same training and testing splits are used for all models. The analysis is conducted over all countries for which sufficient data are available in each testing window.

To give a first sense of the performance of BISTRO, we start by showing aggregate results. Table 1 report the median relative RMSFE (R-RMSFE) across countries, thereby summarising the typical forecasting performance of each model and specification. Values below one indicate an improvement over the AR(1) benchmark.

For **inflation**, the results reveal a clear horizon-dependent pattern. In the unconditional setting, MOIRAI performs marginally better at the shortest horizon ($h = 1$, 1.002 vs. 1.092), but BISTRO becomes superior at medium and long horizons, particularly at $h = 12$ (0.851 vs. 0.988).

For **unemployment**, BISTRO consistently outperforms MOIRAI across all horizons in both unconditional and conditional settings. In the unconditional case, me-

[8] Despite their simplicity, autoregressive models are a tough benchmark to beat when forecasting macroeconomic aggregates. See for example Faust and Wright (2013) and Hall et al (2023).

Table 1: Median relative RMSFE across countries. Values below one indicate an improvement over the AR(1) benchmark.

Variable	Model	h=1	h=3	h=6	h=12
Inflation (M)	BISTRO	1.17	0.97	0.95	1.03
	MOIRAI	1.01	0.98	1.04	1.69
Unemployment (M)	BISTRO	0.85	0.72	0.78	1.23
	MOIRAI	0.97	0.98	1.08	1.57
GDP (Q)	BISTRO	0.79	0.74	0.81	1.44
	MOIRAI	1.33	1.18	1.26	2.12

dian R-RMSFE values for BISTRO range between 0.720 and 0.904, compared to 0.944–1.163 for MOIRAI. The gains are especially pronounced at short and medium horizons.

For **quarterly GDP**, the differences are even more marked. BISTRO delivers large improvements relative to the benchmark at all horizons (except 2 year out), with median R-RMSFE values around 0.62–0.70, both with and without covariates. In contrast, MOIRAI systematically underperforms the AR(1) benchmark, with values above one at all horizons.

Taken together, these aggregated median results indicate that BISTRO tends to deliver meaningful and reliable forecasts. The improvements over the AR(1) benchmark and MOIRAI appear particularly strong for unemployment and GDP, and at longer horizons for inflation.

The median performance, however, conceals considerable heterogeneity. Focusing on inflation, Table 2 reports the forecasting accuracy of BISTRO and MOIRAI relative to an AR(1) benchmark across forecast horizons, where forecasts are aggregated across four country groups: the United States (US), the euro area (EA), other advanced economies (Other AE), and emerging market economies (EMEs).

On average, BISTRO improves upon the AR(1) benchmark in most cases, with particularly strong gains at the longer horizons. For the United States, BISTRO outperforms both the AR(1) and MOIRAI for all horizons except the one-month-ahead, with its relative performance improving as the forecasting horizon lengthens. Results are similar for the euro area, where both BISTRO and MOIRAI models perform similarly on average and improve with the length of the forecasting horizon. Results for other advanced economies are more mixed, with average performance closer to the AR(1) benchmark. Similarly, for EMEs, BISTRO delivers systematic improvements at the medium-term horizon.

To formally test the superior predictive ability of BISTRO over the AR(1) benchmark, we resort to Diebold-Mariano (DM) tests. These, however, need to be conducted at the country-specific level: detailed country-by country results are showcased in Tables 3, 4 and 5 for, respectively, inflation, unemployment and GDP growth; statistically significant differences at the 5% level are highlighted in bold blue (statistically significant improvements) and bold red (statistically significant deteriorations).

Note first how the blue colour tends to prevail in the BISTRO columns, in contrast

Table 2: **Univariate inflation forecasting performance.** Relative RMSFE (R-RMSFE) with respect to an AR(1) benchmark. Entries report the mean R-RMSFE across countries, averaged over the test windows. For each evaluation, a 20-year historical estimation window is used when available. Values below one indicate an improvement over the AR(1) benchmark.

(a) Short horizon forecasts (h=1,3).

	h=1		h=3	
Region	BISTRO	MOIRAI	BISTRO	MOIRAI
United States	**1.024**	1.136	**0.946**	1.241
Euro Area	1.065	**0.987**	**0.927**	0.932
Other Advanced Econ.	1.157	**1.053**	**1.052**	1.108
Emerging Economies	1.009	**0.874**	0.876	**0.844**

(b) Long horizon forecasts (h=6,12).

	h=6		h=12	
Region	BISTRO	MOIRAI	BISTRO	MOIRAI
United States	**0.804**	1.335	**0.481**	1.349
Euro Area	**0.890**	0.956	**0.898**	1.689
Other Advanced Econ.	**0.972**	1.220	**0.813**	1.097
Emerging Economies	0.895	**0.854**	0.969	**0.850**

to MOIRAI. This is especially so at longer horizons, and for unemployment and GDP growth. Statistically-significant improvements, however, tend to be more rare, and occur at short-to-medium horizons. Note also that there are a few instances in which BISTRO (and even more so MOIRAI) yield very bad forecasts, with RMSE ratios as high as two.

Figures 4, 5 and 6 summarise the results. In the case of inflation, both BISTRO performs relatively worse than the AR(1) benchmark as well as MOIRAI at short horizons, but its performance improves over time. By contrast, the performance of MOIRAI over the AR(1) benchmark tends to deteriorate at longer horizons. For unemployment and GDP growth, the comparative advantage of BISTRO is much clearer, especially at shorter horizons: not only the vast majority of the RMSFE ratios are below one, but a large proportion is also statistically significant.

7.2 Forecasting the 2021 inflation surge

In this subsection we illustrate BISTRO's forecasting capabilities on a specific episode: the 2021–22 US inflation surge. Figure 7 compares YoY inflation forecasts from BISTRO, MOIRAI and a benchmark AR(1) model, each initiated at a different stage of the surge. Every panel displays multi-step-ahead forecasts generated dynamically over the forecast horizon, without using inflation readings that were not yet available at the

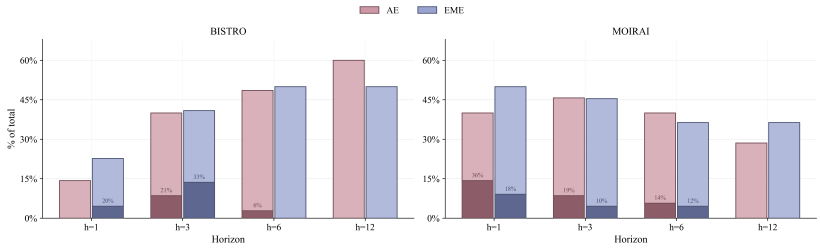


Figure 4: Percentages of R-RMSFE below 1 and their statistical significance for inflation

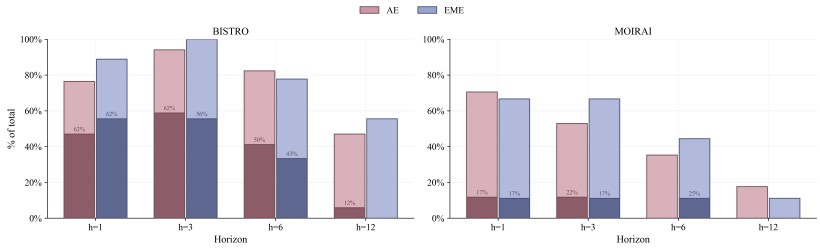


Figure 5: Percentages of R-RMSFE below 1 and their statistical significance for unemployment

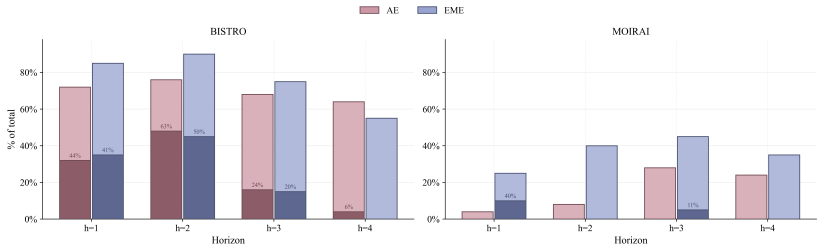


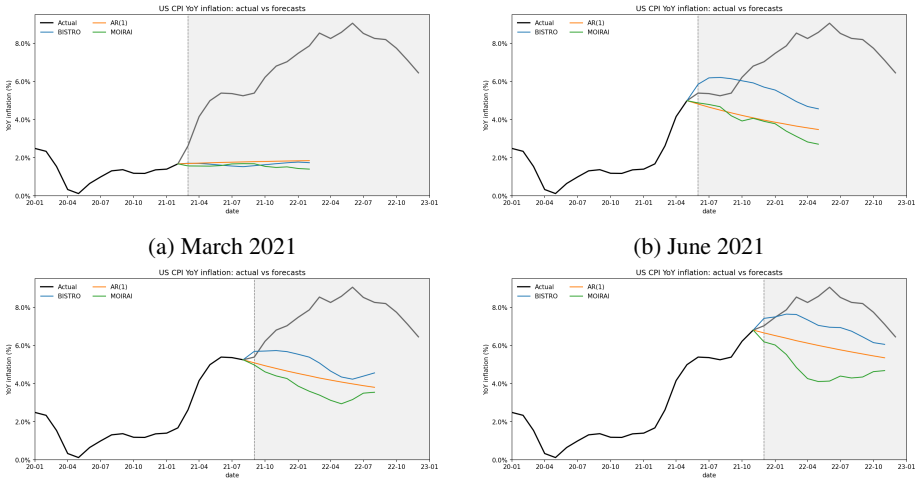
Figure 6: Percentages of R-RMSFE below 1 and their statistical significance for GDP growth

time of generation.

In March 2021, before the surge had materialised, all three models projected inflation converging towards the 2% target (Figure 8a). By June 2021, however, the first signs of rising inflation caused the projections to diverge (Figure **??**). The AR(1) benchmark mechanically decayed towards the historical average, while MOIRAI indicated an even faster decline. BISTRO, by contrast, correctly anticipated a more persistent wave of inflation.

By September 2021, inflation appeared to have plateaued. The AR(1) and MOIRAI models continued to project a declining trajectory, but BISTRO signalled a shallower decline (Figure 8c). In December 2021, a second wave had already taken hold. BISTRO projected rising and persistently high inflation, close to the actual outcome (Figure **??**). In **?** employ a text-based LLM to predict the 2021 US inflation surge, achieving results comparable to AR(1) predictions.

The 2021–22 inflation episode is one that simple linear time-series models struggle to capture. The exercise illustrates how BISTRO can detect potential structural changes and regime shifts. That said, the model also performs competitively and consistently across other time periods that do not feature such extreme circumstances.



(a) March 2021

(b) June 2021

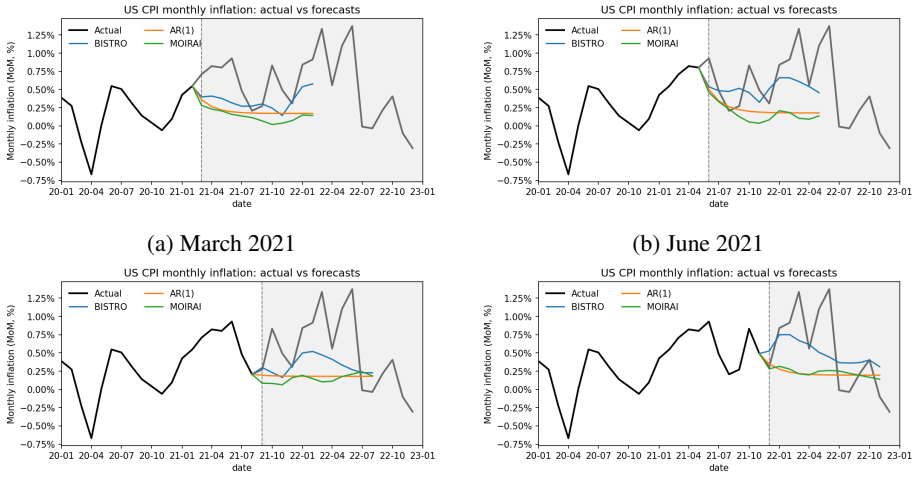
(c) September 2021

(d) December 2021

Figure 7: **Univariate YoY inflation forecasts for the United States for 2021 during the inflation surge.**

Admittedly, this period was included in the training sample. However, during training, the inflation series was never presented in isolation or explicitly labelled as "inflation", and it was expressed in month-on-month (MoM) rather than YoY terms. The model received only the numerical values within the time series, with no additional metadata. Had the model memorised the series exactly, it would have perfectly replicated the forthcoming surge. All other comparisons in the paper use only test periods that were excluded from training.

Furthermore, Figure 8 repeats the four-panel comparison, this time prompting the model with MoM inflation rather than YoY. The MoM series differs markedly from the YoY series and is considerably noisier. If BISTRO had memorised any training series, it would have been this one. Yet BISTRO's estimates clearly diverge from the realised MoM path, confirming that memorisation has not occurred. BISTRO nonetheless infers that MoM inflation would remain higher than the MOIRAI and AR(1) forecasts, which predict a rapid return to between 0.1% and 0.25% within a few months—unable to capture that inflation remains elevated.



(a) March 2021

(b) June 2021

(c) September 2021

(d) December 2021

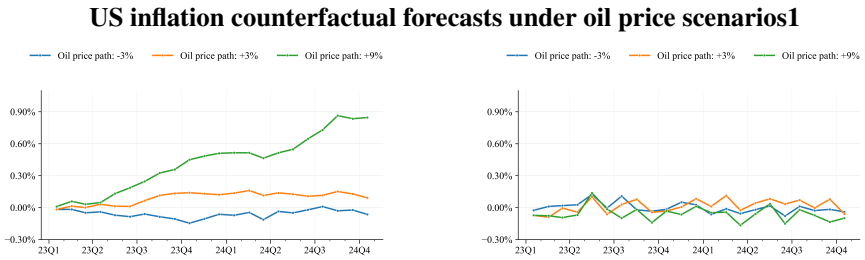
Figure 8: **Univariate MoM inflation forecasts for the United States for 2021 during the inflation surge.**

7.3 Running conditional forecasts

Having established the solid performance of our model in unconditional forecasting, we turn to scenarios, that is, conditional modelling exercises. One of the main advantages of the flexibility of our model lies in its ability to naturally handle relevant covariates that one may want to condition the forecast on. In standard time series models, conditioning is possible only on the set of variables that were considered and explicitly included in the model's specification and estimation. We can instead prompt BISTRO with trajectories of additional macroeconomic variables (not necessarily featured in the training sample) and then generate projections that are conditional on their realisations. The way the model handles this is by relying on the historical patterns that emerge from the training data set and hence shape the embeddings.

To illustrate this, we consider conditional forecasts of inflation based on different possible future trajectories of the oil price. Starting from its realised path, which serves as a baseline, we introduce counterfactual oil price scenarios in which the oil price would be 3% lower and 3% and 9% higher than it actually was. In general, conditioning

on oil prices produces systematically higher or lower oil price paths relative to the baseline (Figure 9(a)). But even more importantly, the model unveils a substantial degree of non-linearity in the reaction of inflation to oil prices. First, a decline in the price of oil has smaller medium-run effects on inflation than an increase by the same amount. Second, larger increases in the price of oil have a more than proportional effect on inflation. A clear bearing on the relationship between oil prices and inflation is one of the advantages of our model over MOIRAI. The latter, trained on a wider set of non-macroeconomic time series, fails to highlight any meaningful connection between oil prices and inflation (Figure 9(b)).



(a) BISTRO (b) MOIRAI

Figure 9: United States inflation counterfactual forecasts under oil price shocks. The figure compares projections generated by BISTRO and MOIRAI when conditioning on alternative oil price paths (baseline, ± 0.03, and $+0.09$). Inflation is expressed in percentage terms. Each panel reports the realized path and the corresponding counterfactual projections under the different oil price scenarios.

Table 3: Mean RRMSFE by country for unconditional monthly inflation forecasts. Values < 1 indicate that the model outperforms the baseline (blue), while values > 1 indicate that the baseline performs better (red). Bold values denote statistical significance at the 5% level according to the Diebold–Mariano test.

	BIS				**MOIRAI**			
Country	$h = 1$	$h = 3$	$h = 6$	$h = 12$	$h = 1$	$h = 3$	$h = 6$	$h = 12$
AE	**1.21**	1.12	1.08	1.05	1.09	1.13	1.02	2.00
AR	**1.31**	0.74	0.65	1.16	0.86	0.62	0.75	1.83
AT	1.16	1.01	0.93	0.81	1.04	1.00	1.00	0.95
AU	**1.48**	1.23	1.04	1.71	1.05	1.18	1.47	6.44
BE	1.26	1.14	1.07	1.10	1.04	0.97	0.88	0.81
BG	**1.46**	**0.69**	**0.51**	0.36	**0.52**	0.59	0.74	0.64
BR	**3.58**	1.17	0.71	0.22	**2.90**	0.84	0.63	0.23
CA	1.08	1.07	1.19	0.94	1.06	1.14	1.17	1.06
CH	1.10	1.07	1.12	1.16	1.04	1.03	1.06	0.99
CL	1.23	1.01	0.94	0.86	1.01	1.02	0.98	0.73
CN	1.13	1.15	1.32	1.13	0.93	0.98	1.30	3.98
CO	**1.42**	0.91	0.78	0.83	1.04	1.04	1.03	1.17
CY	0.91	0.89	0.90	0.95	0.93	0.91	0.92	1.09
CZ	0.97	0.84	0.72	0.45	0.93	0.91	**0.94**	1.05
DE	1.09	0.98	0.96	0.71	1.03	1.09	1.22	1.50
DK	1.13	1.07	1.02	1.06	0.96	0.92	0.93	0.83
DZ	1.00	0.97	0.91	0.76	1.00	1.06	1.25	1.20
EE	1.13	1.07	1.00	0.54	0.89	0.83	0.83	1.04
ES	1.11	1.05	1.02	0.82	1.03	0.99	1.03	0.80
FI	1.07	0.94	0.93	0.92	1.04	1.08	1.15	1.03
FR	1.09	1.04	1.08	0.90	1.03	**1.00**	1.42	2.90
GB	1.37	1.19	1.00	1.35	1.16	1.22	1.32	2.20
GR	1.05	1.00	0.90	0.77	**0.93**	0.84	0.86	1.05
HK	1.11	0.98	1.02	1.00	0.96	1.02	1.00	0.93
HU	1.20	1.03	1.06	1.03	1.04	1.04	1.07	0.93
ID	1.18	1.04	1.10	0.94	0.97	0.94	**0.89**	0.71
IE	1.26	1.24	1.52	2.00	1.07	1.10	1.19	2.47
IL	**1.83**	1.43	1.34	0.89	1.07	1.05	1.63	2.29
IN	1.10	1.15	1.10	2.20	0.96	1.16	1.29	1.40
IT	1.18	1.04	1.05	2.64	1.04	0.98	1.00	9.17
JP	1.11	1.10	1.04	1.12	1.00	1.02	1.10	1.24
KR	1.00	0.94	0.87	1.03	1.03	1.06	1.00	1.13
KW	1.02	1.12	1.32	1.48	0.84	0.97	1.07	0.90
LT	**1.07**	**0.45**	0.33	0.21	**0.62**	0.94	1.06	1.22
LU	1.09	1.07	1.20	1.14	1.04	**1.09**	**1.13**	1.16
LV	0.96	**0.48**	0.33	0.19	**0.36**	**0.39**	0.68	0.86
MA	1.06	1.11	1.23	1.34	1.06	1.10	1.21	2.10
MK	**1.02**	**0.46**	0.31	0.19	1.32	0.62	0.63	1.05
MT	**1.21**	1.17	1.15	0.93	1.09	1.21	1.26	1.47
MX	0.98	0.84	0.81	0.89	**0.96**	0.76	0.68	0.64
MY	1.18	1.10	1.12	1.59	1.03	1.03	1.05	1.39
NL	1.16	1.09	1.24	1.07	1.03	1.05	1.12	1.40
NZ	1.29	1.00	0.86	0.83	0.99	1.03	1.29	2.92
PE	0.97	**0.38**	0.20	0.09	0.52	**0.30**	0.18	0.17
PH	1.04	1.04	1.03	0.75	1.07	1.13	1.39	2.04
PT	**1.21**	1.10	0.99	0.66	0.94	0.94	0.90	1.78
RO	**0.90**	**0.66**	0.50	0.44	**0.64**	0.52	0.48	0.66
RS	0.93	0.85	1.32	1.54	0.70	0.69	0.83	1.26
RU	0.95	1.01	1.09	0.77	1.04	1.04	1.28	2.05
SA	**1.10**	1.00	0.97	1.11	1.07	1.13	1.20	1.27
SE	**1.16**	1.00	1.02	3.87	0.92	0.89	1.00	2.07
SG	1.03	0.95	1.02	2.35	1.06	**1.23**	**1.41**	2.95
SI	1.02	0.94	0.79	0.73	**0.93**	1.10	1.22	0.96
SK	0.91	0.71	0.60	0.51	0.84	**0.78**	**0.76**	0.89
TH	0.85	**0.72**	0.76	0.81	0.99	0.97	1.01	1.07
TR	1.24	0.88	0.93	1.09	0.92	0.84	0.91	1.62
US	1.24	0.94	0.81	0.48	1.14	1.24	1.36	1.35
XM	1.15	0.93	0.85	1.18	1.08	1.00	1.13	7.27
ZA	1.07	1.16	1.25	1.18	0.97	1.12	1.26	1.66
All Countries	1.17	0.97	0.95	1.03	1.00	0.96	1.04	1.69

Table 4: Mean RRMSFE by country for unconditional monthly unemployment forecasts. Values < 1 indicate that the model outperforms the baseline (blue), while values > 1 indicate that the baseline performs better (red). Bold values denote statistical significance at the 5% level according to the Diebold–Mariano test.

	BIS				**MOIRAI**			
Country	$h = 1$	$h = 3$	$h = 6$	$h = 12$	$h = 1$	$h = 3$	$h = 6$	$h = 12$
AT	**0.40**	**0.30**	**0.29**	0.36	0.96	**0.63**	0.62	0.90
BE	0.89	0.81	0.92	1.06	1.03	1.02	1.00	3.33
BG	**0.64**	**0.62**	0.64	1.27	0.95	1.01	1.08	1.51
BR	**0.64**	**0.70**	**0.86**	1.18	0.97	1.06	1.19	1.51
CA	**0.57**	**0.54**	**0.55**	0.88	0.95	1.05	1.29	2.26
CH	**0.72**	**0.64**	**0.63**	0.95	0.95	0.94	0.97	1.62
CL	1.07	0.96	1.08	1.63	0.99	0.99	1.19	2.67
CO	**0.60**	**0.42**	**0.37**	0.41	0.80	0.75	0.79	1.46
CY	1.07	0.95	0.87	0.65	**0.86**	0.92	0.94	0.70
CZ	**0.73**	**0.62**	0.73	1.24	0.94	0.93	0.87	1.00
DE	0.80	0.79	1.04	1.62	0.91	0.91	1.05	1.16
DK	0.69	**0.63**	0.68	0.88	**0.95**	0.96	0.99	1.08
FI	**0.60**	**0.51**	**0.45**	1.20	0.97	**0.90**	1.10	2.62
GB	1.36	0.99	0.94	0.77	1.10	1.14	1.20	1.35
LU	0.96	**0.65**	**0.70**	1.24	1.05	0.99	1.08	0.84
LV	**0.76**	**0.69**	**0.58**	0.65	0.97	1.20	**1.48**	2.35
MT	2.08	1.12	1.05	1.96	1.09	0.99	0.98	1.16
MX	**0.63**	**0.50**	**0.37**	0.52	0.87	**0.86**	**0.76**	1.45
NL	0.82	0.92	1.69	3.29	1.07	1.11	1.68	2.36
PE	0.91	**0.81**	0.76	0.43	1.11	1.14	**1.40**	1.38
PL	**0.82**	0.84	1.10	2.73	0.98	0.98	0.94	1.64
PT	1.15	1.00	0.98	1.67	0.96	1.11	1.30	2.02
RO	0.89	0.79	0.72	0.79	**0.86**	0.85	0.93	1.15
RU	0.97	0.81	0.82	3.12	1.04	0.98	1.01	0.85
TR	**0.74**	**0.63**	0.84	0.85	1.02	1.01	1.03	1.02
US	**0.62**	**0.53**	**0.50**	**0.60**	0.99	1.02	**1.10**	1.37
All Countries	0.85	0.72	0.78	1.23	0.97	0.98	1.08	1.57

Table 5: Mean RRMSFE by country for unconditional quarterly GDP forecasts. Values < 1 indicate that the model outperforms the baseline (blue), while values > 1 indicate that the baseline performs better (red). Bold values denote statistical significance at the 5% level according to the Diebold–Mariano test.

	BIS				**MOIRAI**			
Country	$h = 1$	$h = 2$	$h = 3$	$h = 4$	$h = 1$	$h = 2$	$h = 3$	$h = 4$
AE	1.25	0.98	0.89	0.95	1.02	0.89	0.78	0.89
AR	**0.53**	0.64	0.44	1.20	**0.71**	0.82	0.64	2.11
AT	0.92	0.75	0.65	5.85	**1.73**	1.34	1.14	9.69
BA	0.77	0.97	0.86	1.07	0.95	0.92	0.83	0.61
BE	1.06	1.02	1.77	0.63	**2.44**	**1.71**	**3.41**	1.26
BG	**0.49**	**0.43**	0.59	2.07	1.14	1.41	1.59	3.71
BR	**0.43**	**0.49**	0.32	0.31	**0.75**	0.75	**0.59**	0.61
CA	**0.61**	**0.66**	**0.47**	**0.23**	0.94	1.01	0.79	0.57
CH	0.87	0.65	0.63	0.77	**1.60**	**1.15**	**1.38**	0.72
CL	0.73	0.86	1.52	0.61	1.26	0.96	1.48	0.95
CO	0.80	1.14	1.12	5.15	1.10	1.07	0.72	3.72
CZ	0.79	**0.57**	0.44	2.62	**1.42**	1.05	0.82	2.91
DE	**0.67**	0.61	0.79	0.71	1.05	1.06	1.15	0.76
DK	1.23	1.14	1.08	0.74	**2.16**	**1.56**	**1.75**	1.47
EE	1.11	**0.65**	0.71	0.66	**1.40**	1.10	1.01	1.78
ES	1.84	1.86	2.23	0.92	**4.40**	**3.69**	**5.04**	1.36
FI	**0.80**	**0.56**	0.53	1.84	**1.65**	1.01	0.89	2.06
GR	**0.53**	**0.40**	0.90	5.32	1.03	0.90	1.02	1.53
HK	**0.57**	**0.49**	**0.63**	0.58	1.10	**1.21**	0.93	0.51
HR	**0.41**	**0.52**	**0.22**	**0.13**	**0.66**	0.90	**0.82**	**0.51**
HU	**0.71**	**0.57**	**0.31**	1.67	**1.31**	1.03	0.73	3.01
ID	0.67	0.88	0.79	1.04	1.14	1.33	0.64	0.89
IE	1.05	1.29	1.28	2.59	1.28	1.14	1.50	1.43
IN	0.71	**0.34**	**0.34**	0.87	1.18	0.96	1.97	17.59
IS	0.98	0.94	0.69	1.29	1.17	1.10	0.57	0.66
IT	**0.66**	**0.51**	0.54	0.32	**1.54**	1.13	1.22	1.13
LU	1.10	0.74	**0.49**	0.75	**1.98**	1.15	**1.42**	3.20
LV	**0.57**	**0.49**	0.28	0.91	1.04	0.99	0.77	0.52
MA	1.23	0.98	0.65	1.60	1.32	1.32	0.82	1.89
MX	0.84	**0.77**	0.61	0.74	1.06	1.03	1.28	1.45
MY	0.59	**0.65**	1.33	0.50	1.15	0.96	1.38	1.91
NL	1.00	1.06	0.62	0.77	**2.23**	**1.91**	**1.40**	1.49
NO	0.94	0.85	1.43	3.25	1.18	1.22	1.36	2.54
NZ	0.61	**0.73**	1.37	0.39	**1.36**	1.13	**2.11**	1.13
PE	**0.36**	0.33	0.23	0.24	0.93	1.05	1.38	2.51
PH	0.67	**0.64**	0.97	1.10	**1.62**	1.14	**1.61**	1.52
PL	**0.57**	**0.53**	1.28	1.59	1.28	0.96	1.57	2.04
RO	**0.57**	**0.51**	0.65	0.23	1.02	0.87	0.69	0.52
RS	0.57	0.52	0.69	1.04	**1.15**	0.89	1.11	0.85
RU	0.60	0.88	0.48	8.15	0.89	1.60	1.05	7.30
SA	1.21	1.27	1.80	1.85	1.04	1.10	1.44	1.37
SG	0.84	0.95	1.25	2.27	**1.21**	1.27	1.18	1.45
SI	0.62	**0.48**	0.45	1.75	1.37	1.13	0.89	3.46
SK	0.78	**0.65**	**0.28**	0.57	1.14	1.12	0.56	1.03
TH	**0.46**	**0.44**	**0.56**	0.24	**1.20**	**1.31**	1.27	0.82
TR	0.56	0.54	0.56	0.49	0.92	1.04	0.82	0.54
US	1.31	1.32	1.39	0.98	1.03	1.18	1.31	1.26
XM	0.71	0.48	0.63	0.27	**1.83**	1.35	2.03	0.79
ZA	0.83	0.73	0.78	0.76	**1.25**	1.10	1.08	1.92
All Countries	0.79	0.74	0.81	1.44	1.33	1.18	1.26	2.12

8 Conclusions

We introduced BISTRO, a foundational time series model explicitly fine-tuned for the unconditional and conditional forecasting of macroeconomic time series. The key advantage of BISTRO compared with alternatives lies in its flexibility. It can accommodate prediction of a multitude of time series in unconditional as well as conditional terms. Importantly, it is also able to unveil and deal with non-linearities. While we provided a few examples, BISTRO lends itself to plenty of different macroeconomic exercises, ranging from pure forecasting to scenario analysis.

References

Alammar, J. (2018). The illustrated transformer.

Aldasoro, I., Hoerdahl, P., Schrimpf, A., and Zhu, S. (2025). Predicting financial market stress with machine learning. BIS Working Papers 1250, Bank for International Settlements.

Allard, M.-A., Teiletche, P., and Zinebi, A. (2024). Enhancing inflation nowcasting with llm: Sentiment analysis on news. *arXiv preprint arXiv:2410.20198*.

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Arango, S. P., Kapoor, S., et al. (2024). Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*.

Aquilina, M., Araujo, D., Gelos, G., Park, T., and Perez-Cruz, F. (2025). Harnessing artificial intelligence for monitoring financial markets. BIS Working Papers 1291, Bank for International Settlements.

Bank for International Settlements (2024). Artificial intelligence and the economy: Implications for central banks. In *Annual Economic Report*, chapter III.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Cao, S., Jiang, W., Wang, J. L., and Yang, B. (2024). From man vs. machine to man + machine: The art and AI of stock analyses. *Journal of Financial Economics*, 160(103910).

Carriero, A., Clark, T. E., and Massimiliano, M. (2020). Nowcasting tail risks to economic activity with many indicators. Technical report, Federal Reserve Bank of Cleveland.

Chen, P., Zhang, Y., Cheng, Y., Shu, Y., Wang, Y., Wen, Q., Yang, B., and Guo, C. (2024). Pathformer: Multi-scale transformers with adaptive pathways for time series forecasting. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*.

Das, A., Kong, W., Sen, R., and Zhou, Y. (2024). A decoder-only foundation model for time-series forecasting. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F., editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10148–10167. PMLR.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186. Association for Computational Linguistics.

Ekambaram, V., Jati, A., Dayama, P., Mukherjee, S., Nguyen, N. H., Gifford, W. M., Reddy, C., and Kalagnanam, J. (2024). Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series.

Faria-e Castro, M. and Leibovici, F. (2024). Artificial intelligence and inflation forecasts. *Federal Reserve Bank of St. Louis Review*, 106(12).

Gambacorta, L., Kwon, B., Park, T., Patelli, P., and Zhu, S. (2024). CB-LMs: Language models for central banking. BIS Working Papers 1215, Bank for International Settlements.

Garza, A., Challu, C., and Mergenthaler-Canseco, M. (2024). Timegpt-1.

Gorodnichenko, Y., Pham, T., and Talavera, O. (2023). The voice of monetary policy. *American Economic Review*, 113(2).

Goswami, M., Szafer, K., Choudhry, A., Cai, Y., Li, S., and Dubrawski, A. (2024). MOMENT: A family of open time-series foundation models. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F., editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16115–16152. PMLR.

Hendry, D. and Krolzig, H.-M. (2001). *Automatic Econometric Model Selection Using PcGets 1.0*. Timberlake Consultants Press.

Horton, J. (2023). Large language models as simulated economic agents: What can we learn from Homo Silicus? NBER Working Papers 31122, National Bureau of Economic Research.

Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S., and Wen, Q. (2024). Time-llm: Time series forecasting by reprogramming large language models. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*.

Korinek, A. (2023). Generative AI for economic research: Use cases and implications for economists. *Journal of Economic Literature*, 61(4):1281–1317.

Korinek, A. (2025). AI agents for economic research. NBER Working Papers 34202, National Bureau of Economic Research.

Kwon, B., Park, T., Perez-Cruz, F., and Rungcharoenkitkul, P. (2024). Large language models: a primer for economists. *BIS Quarterly Review*, pages 37–52.

Kwon, B., Park, T., Rungcharoenkitkul, P., and Smets, F. (2025). Parsing the pulse: Decomposing macroeconomic sentiment with LLMs. BIS Working Papers 1294, Bank for International Settlements.

Liang, Y., Wen, H., Nie, Y., Jiang, Y., Jin, M., Song, D., Pan, S., and Wen, Q. (2024). Foundation Models for Time Series Analysis: A Tutorial and Survey. *KDD*, 24.

Ludwig, J. and Mullainathan, S. (2024). Machine learning as a tool for hypothesis generation. *Quarterly Journal of Economics*, 139(2).

Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. (2023). A time series is worth 64 words: Long-term forecasting with transformers. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training. Preprint, OpenAI.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. Technical report, OpenAI.

Rasul, K., Ashok, A., Williams, A. R., Ghonia, H., Bhagwatkar, R., Khorasani, A., Bayazi, M. J. D., Adamopoulos, G., Riachi, R., Hassen, N., Biloš, M., Garg, S., Schneider, A., Chapados, N., Drouin, A., Zantedeschi, V., Nevmyvaka, Y., and Rish, I. (2024). Lag-llama: Towards foundation models for probabilistic time series forecasting.

Shi, X., Wang, S., Nie, Y., Li, D., Ye, Z., Wen, Q., and Jin, M. (2025). Time-moe: Billion-scale time series foundation models with mixture of experts. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*. Spotlight Presentation.

Siano, F. (2025). The news in earnings announcement disclosures: Capturing word context using LLM methods. *Management Science*, 71(11).

Tan, M., Merrill, M. A., Gupta, V., Althoff, T., and Hartvigsen, T. (2024). Are language models actually useful for time series forecasting? In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gómez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Woo, G., Liu, C., Kumar, A., Xiong, C., Savarese, S., and Sahoo, D. (2024). Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592*.

Xiao, C., Zhou, J., Xiao, Y., Lu, X., Zhang, L., and Xiong, H. (2025a). Timefound: A foundation model for time series forecasting. *arXiv preprint arXiv:2503.04118*.

Xiao, M., Jiang, Z., Qian, L., Chen, Z., He, Y., Xu, Y., Jiang, Y., Li, D., Weng, R.-L., Peng, M., Huang, J., Ananiadou, S., and Xie, Q. (2025b). Enhancing financial time-series forecasting with retrieval-augmented large language models. *arXiv preprint arXiv:2502.05878*.

Yu, X., Chen, Z., Ling, Y., Dong, S., Liu, Z., and Lu, Y. (2023). Temporal data meets llm – explainable financial time series forecasting. *arXiv preprint arXiv:2306.11025*.

Zarifhonarvar, A. (2026). Generating inflation expectations with large language models. *Journal of Monetary Economics*, 157(C).

Zhou, T., Niu, P., Wang, X., Sun, L., and Jin, R. (2023). One fits all: Power general time series analysis by pretrained lm. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

A Training details

In this section, we provide further details of the training and configurations mentioned in Section 5.3. We train our foundation model $\mathcal{F}_\theta$ using a masked feature prediction approach. For each training iteration, we process batches of multivariate time series data. Each batch element $\boldsymbol{Z}^{(i)} = (\boldsymbol{X}^{(i)}, \boldsymbol{C}^{(i)}) \in \mathbb{R}^{4 \cdot dz_i}$ consists of target variables and their associated covariates, with d_{z_i} distinct time series. The dimensionality is scaled by a factor of 4 due to the transformations applied during preprocessing. The training process follows these steps:

1. **Feature sampling:** We sample up to $\mathcal{L} = 14$ distinct time series from available features (or use all available if $d_{z_i} < \mathcal{L}$). This process yields $4 \cdot \mathcal{L}$ time series in total due to the transformations from set $\mathcal{T}$.
2. **Transformation selection:** For each sampled feature, we randomly select a single transformation from set $\mathcal{T}$ rather than applying all transformations. This approach encourages the model to learn robust representations across different statistical properties while maintaining computational efficiency. We end up with $\mathcal{L}' < 4 \cdot \mathcal{L}$ time series after this step.
3. **Patching:** We segment each time series into patches of size 32, ensuring a minimum of 48 patches per feature to provide sufficient context (i.e., over 4 years worth of context).
4. **Random masking:** Out of $\mathcal{L}'$ series, we randomly select $\mathcal{L}_m$ to be masked and used in the loss computation. For these selected series, we mask between 1.5% and 7% of patches from the end, corresponding to the most recent time steps, simulating real-world forecasting conditions.
5. **Leakage prevention:** We create a temporal consistency mask ($\mathcal{M}_j^{\text{tc}}$) that identifies values appearing in both the history and prediction patches due to the forward-filling process. This mask excludes repeated observations from the loss computation, ensuring that the model cannot access target values through temporal misalignment.
6. **Loss computation:** When computing the loss, we employ a combined masking strategy that incorporates both a transformation-selection mask ($\mathcal{M}_j^{\text{tr}}$) and our temporal-consistency mask ($\mathcal{M}_j^{\text{tc}}$). The transformation-selection mask ensures that only one transformation per available feature contributes to the loss and excludes certain high-frequency series such as commodities and daily exchange rates. The temporal-consistency mask eliminates duplicated values that appear in both historical context and prediction windows due to forward-filling operations.

During validation, we use a predefined subset of features $\mathcal{S}_{\text{val}}$ rather than randomly selecting features to mask in Step 4, allowing for consistent evaluation across iterations and enabling reliable early stopping when optimization plateaus for defined tasks. For final predictions, we generate 25 samples from the probabilistic output layer and aggregate them to produce robust forecasts. In case of predictions of a lower frequency variable, we use daily predictions until the next availability date and aggregate them to get a single prediction corresponding to the original frequency.

Algorithm 1 Pseudo-Code Algorithm for Foundation Model Training and Validation

Require: $D_{\text{train,val}} = \{(\boldsymbol{X}^{(i)}, \boldsymbol{C}^{(i)})\}_{i=1}^{N}, \mathcal{T}, \mathcal{F}_\theta, \mathcal{B}$ ▷ Dataset, transformations, foundation model, batch size
Require: $\mathcal{L}, \mathcal{L}_m, \mathcal{S}_{\text{val}}$ ▷ Max # features to sample, # features to mask, validation feature subset

total_loss $\leftarrow 0$
best_val_loss $\leftarrow \infty$
patience_counter $\leftarrow 0$ ▷ Initialize the training loop
repeat ▷ Training Phase
  batch_samples $\leftarrow$ sample_batch($D_{\text{train}}, \mathcal{B}$) ▷ Sample batch of $\mathcal{B}$
  for each $\boldsymbol{Z}^{(i)} = (\boldsymbol{X}^{(i)}, \boldsymbol{C}^{(i)}) \in \mathbb{R}^{4 \cdot d_{z_i}}$ in batch_samples **do**
    $\tilde{\boldsymbol{Z}} \leftarrow$ sample_features($\boldsymbol{Z}^{(i)}, \mathcal{L} = 14$) ▷ Sample up to $\mathcal{L}$ features
    transform $\leftarrow$ random_select_single_transformation($\mathcal{T}$) ▷ Select single transformation
    $\tilde{\boldsymbol{Z}} \leftarrow$ apply(transform, $\tilde{\boldsymbol{Z}}$) ▷ Apply transformation
    $\tilde{\boldsymbol{Z}} \leftarrow$ apply(transform, $\tilde{\boldsymbol{Z}}$) ▷ Apply transformation
    $\tilde{\boldsymbol{Z}} \leftarrow$ create_patches($\tilde{\boldsymbol{Z}}'$, patch_size=32, min_patches=48) ▷ Ensure min 48 patches
    mask_ratio $\leftarrow$ uniform_sample(0.015, 0.07) ▷ Mask ratio between 1.5%–7%
    $\mathcal{M} \leftarrow$ select_end_patches($\boldsymbol{P}, \mathcal{L}_m$, mask_ratio) ▷ Select $\mathcal{L}_m$ series
    $\boldsymbol{P}' \leftarrow$ mask($\boldsymbol{P}, \mathcal{M}$) ▷ Apply masking
    $\mathcal{M}_j^{\text{tc}} \leftarrow$ create_temporal_consistency_mask($\boldsymbol{P}, \mathcal{M}$) ▷ Prevent duplicates
    $\mathcal{M}_j^{\text{tr}} \leftarrow$ create_transformation_selection_mask($\boldsymbol{P}$) ▷ One transform per feature
    $\mathcal{M}_j \leftarrow$ combine_masks($\mathcal{M}_j^{\text{tc}}, \mathcal{M}_j^{\text{tr}}$) ▷ Combine masks
    $\Phi \leftarrow \mathcal{F}_\theta(\boldsymbol{P}', \mathcal{M}_j)$ ▷ Feed to model
    batch_loss $\leftarrow$ compute_LL($\boldsymbol{P}', \mathcal{M}_j, \Phi$) ▷ Compute log-likelihood
  end for
  update_parameters(θ, total_loss) ▷ Update model parameters
  ▷ Validation Phase
  if val_step **then**
    val_loss $\leftarrow 0$
    batch_samples $\leftarrow$ sample_batch($D_{\text{val}}, \mathcal{B}$) ▷ Sample validation batch
    for each $\boldsymbol{Z}^{(i)} = (\boldsymbol{X}^{(i)}, \boldsymbol{C}^{(i)}) \in \mathbb{R}^{4 \cdot d_{z_i}}$ in batch_samples **do** ▷ Feature sampling
      $\tilde{\boldsymbol{Z}} \leftarrow$ sample_features($\boldsymbol{Z}^{(i)}, \mathcal{L} = 14$) ▷ Sample up to $\mathcal{L}$ features
      ▷ Transformation selection
      transform $\leftarrow$ predefined_transformation($\mathcal{T}$) ▷ Use consistent transformation for validation
      $\tilde{\boldsymbol{Z}}' \leftarrow$ apply(transform, $\tilde{\boldsymbol{Z}}$) ▷ Apply transformation
      ▷ Patching
      $\boldsymbol{P} \leftarrow$ create_patches($\tilde{\boldsymbol{Z}}'$, patch_size=32, min_patches=48) ▷ Create patches
      ▷ Predefined masking
      $\mathcal{M} \leftarrow$ select_predefined_features($\boldsymbol{P}, \mathcal{S}_{\text{val}}$) ▷ Use predefined subset for consistent evaluation
      $\boldsymbol{P}' \leftarrow$ mask($\boldsymbol{P}, \mathcal{M}$) ▷ Apply masking
      ▷ Leakage prevention
      $\mathcal{M}_j^{\text{tc}} \leftarrow$ create_temporal_consistency_mask($\boldsymbol{P}, \mathcal{M}$) ▷ Identify forward-fill duplicates
      $\mathcal{M}_j^{\text{tr}} \leftarrow$ create_transformation_selection_mask($\boldsymbol{P}$) ▷ One transformation per feature mask
      $\mathcal{M}_j \leftarrow$ combine_masks($\mathcal{M}_j^{\text{tc}}, \mathcal{M}_j^{\text{tr}}$) ▷ Combined masking strategy
      ▷ Loss computation
      $\Phi \leftarrow \mathcal{F}_\theta(\boldsymbol{P}', \mathcal{M}_j)$ ▷ Feed to foundation model
      val_batch_loss $\leftarrow$ compute_LL($\boldsymbol{P}', \mathcal{M}_j, \Phi$) ▷ Compute log-likelihood
      val_loss $\leftarrow$ val_loss + val_batch_loss ▷ Accumulate validation loss
    end for
    ▷ Early stopping check
    if val_loss ¡ best_val_loss **then**
      best_val_loss $\leftarrow$ val_loss
      patience_counter $\leftarrow 0$
      save_checkpoint($\mathcal{F}_\theta$) ▷ Save best model
    else
      patience_counter $\leftarrow$ patience_counter + 1
    end if
  end if
  total_loss $\leftarrow 0$ ▷ Reset loss for next iteration
until patience_counter ¿ max_patience
load_best_checkpoint($\mathcal{F}_\theta$) ▷ Restore best model
return trained model $\mathcal{F}_\theta$ ▷ The trained foundation model

Previous volumes in this series

1336 March 2026	A public-private partnership? Central bank funding and credit supply	Matthieu Chavaz, David Elliott and Win Monroe
1335 March 2026	Tokenomics and blockchain fragmentation	Hyun Song Shin
1334 March 2026	Robots, ICT and employment: evidence from advanced and emerging EU countries	Costanza Bosone, Leonardo Gambacorta, Paolo Giudici, Enisse Kharroubi and Ulf Lewrick
1333 March 2026	Generative AI for surveys on payment apps: AI views on privacy and technology	Koji Takahashi and Joon Suk Park
1332 February 2026	Dollar funding and housing markets: the role of non-US global banks	Torsten Ehlers, Mathias Hoffmann and Alexander Raabe
1331 February 2026	Lending to vulnerable households and consumption: evidence from Korea	Jieun Lee and Ilhyock Shim
1330 February 2026	Passive investors and loan spreads	Konrad Adler, Sebastian Doerr and Sonya Zhu
1329 February 2026	Adoption and welfare effects of payment innovations: the case of digital wallets in Peru	Arturo Andia, Jose Aurazo and Marcelo Paliza
1328 February 2026	The perils of narrowing fiscal spaces	Hanno Kase, Leonardo Melosi, Sebastian Rast and Matthias Rottner
1327 January 2026	A tractable menu cost model with an aggregate markup drift	Ko Munakata
1326 January 2026	Monetary policy and private equity acquisitions: tracing the links	Fernando Avalos, Boris Hofmann and Jose Maria Serena
1325 January 2026	AI adoption, productivity and employment: evidence from European firms	Iñaki Aldasoro, Leonardo Gambacorta, Rozalia Pal, Debora Revoltella, Christoph Weiss and Marcin Wolfski

All volumes are available on our website www.bis.org.