

GenGNAR: Genetic Graph Search for GNAR Models with Applications to Inflation Forecasting

Sergio Estan-Ruiz and Guy P. Nason

June 15, 2026

Abstract

Forecasting inflation accurately is central to monetary policy, yet statistical forecasting in economics often relies either on simple univariate models or on heavily parametrised multivariate models such as VARs. Recent work on Random Generalised Network Autoregressive (RaGNAR) models showed that graph-based time series models can provide accurate and timely inflation forecasts, but networks that they are based on are random. This paper introduces GenGNAR, a genetic algorithm framework for graph selection in GNAR models that replaces random exploration with a directed search based on forecasting performance validation. We apply the method to UK CPI inflation forecasting and show that GenGNAR yields systematic improvements over RaGNAR, with gains that are particularly pronounced at medium and longer forecasting horizons. We also study several variants of the framework, including changing the graph generation distribution to a Stochastic Block Model to favour important nodes, and compare the best-performing specification against standard statistical, machine learning, and official Bank of England forecasts. The results suggest that directed graph search can substantially improve the empirical performance of GNAR models in macroeconomic forecasting.

Contents

1	Introduction	3
2	Dataset, Evaluation Design and Benchmarks	4
2.1	Inflation dataset and pre-processing	4
2.2	Evaluation design	5
2.3	Statistical models and Bank of England benchmarks	6
3	Methodology	8
3.1	Generalised Network Autoregressive models	9
3.2	Network selection for GNAR	11
3.3	The genetic search GNAR: GenGNAR	12
3.4	Practicalities of network selection	14
4	Results	14
4.1	GenGNAR versus RaGNAR	15
4.2	Variants and sensitivity	17
4.3	Benchmark comparison and the Bank of England’s forecasts	19
5	Conclusions and Further Work	21
A	Macroeconomic forecasting in policy-making	24
B	CPI variable names	26
C	Limitations of McCarthy (2024)	28
D	Details of the Genetic Algorithm	29
D.1	Crossover	29
D.2	Mutation	29
E	Speed-up of validation error computation	30
F	Additional Results	30
F.1	Results for local models	30
F.2	Results with full dataset	31
G	Mann-Whitney U Test	31

1 Introduction

Effective monetary policy depends on the accurate assessment and forecasting of inflation. This, in turn, influences many aspects of the economy, including grocery prices, exchange rates, bank loan interest rates and ultimately wealth distribution (Bach, 1975; Mallick and Mohsin, 2010). Therefore, devising models which forecast inflation accurately is a primary concern for many policy-makers. In the UK, a common inflation measure is the Consumer Price Index (CPI), which tracks the average change over time in the prices of goods and services consumed by households. Institutional macroeconomic forecasting could be divided into three pillars: structural models, semi-structural models and statistical/empirical models, and a more detailed explanation of this stratification is presented in Appendix A. Despite the importance of inflation, simple univariate or vector autoregressive models dominate the purely statistical modelling conducted by many economic institutions (Ciccarelli et al., 2024).

Time series forecasting methods. In the academic sphere, early time series research was focused on linear stochastic processes such as autoregressive (AR), moving-average (MA), and their ARMA/ARIMA extensions (Mann and Wald, 1943; Quenouille, 1957; Box et al., 2016; Lütkepohl, 2005; Gooijer and Hyndman, 2006; Hyndman and Athanasopoulos, 2021). These models supplied closed-form maximum-likelihood or least-squares estimators, making them tractable even for modest sample sizes (Brockwell and Davis, 2009). Other powerful and popular models were developed based on the exponential smoothing model (Holt, 1957; Winters, 1960; Holt, 2004) and state space models and the Kalman filter from Kalman (1960). The VAR model and its structural counterpart have turned out to be extremely popular among economists and central banks; see Section 12.3 of Hyndman and Athanasopoulos (2021) who refer to their “dominance in forecasting” or the recent Bernanke Review (Independent Evaluation Office, 2024). This is understandable as they are simple to understand and fit, and allow economists to study the effect of policy and structural shocks on the variables of interest. The Bernanke Review (Independent Evaluation Office, 2024) was a comprehensive review of the Bank of England’s forecasting and related processes carried out by Ben Bernanke, commissioned by the Court of the Bank.

Despite their popularity, VAR models often suffer from over-parametrisation which can lead to poor parameter estimation and, as a result, poor forecasting, see Hamilton (1994) page 340, Lütkepohl (2005) page 80, or Nason and Palasciano (2026). Solutions to the over-parameterisation problem include imposing constraints on the coefficients, see, e.g. Lütkepohl (2005) Chapter 5, or using sparse VAR techniques, see Davis et al. (2016). Another approach to forecasting is to use graph-based models such as General Autoregressive Network (GNAR) models, introduced by Knight et al. (2016, 2020). These models use a fixed graph G to modulate lagged dependencies, yielding an extremely parsimonious model with closed-form ordinary least squares estimators and providing accurate forecasts. Recent work by Nason et al. (2023, 2026) showed that both restricted and sparse VAR methods still often result in over-parametrisation and consequently poorer forecasts compared to GNAR.

The GNAR model is central to this article and will be explored fully in Section 3.1.

Forecasting inflation with GNAR models. Recently, Nason and Palasciano (2026) introduced and explored the use of a variant of a Generalised Network Autoregressive (GNAR) model for inflation forecasting called Random GNAR (RaGNAR). Here, the graph G which modulates the lagged dependencies is selected by randomly generating graphs

and selecting the resulting model and graph which best fits the training data. They also introduced model averaging techniques at the graph and model hyperparameter level, as well as using a one-node MSE fitness function for graph selection. RaGNAR produced striking forecasting results. Not only did it generally beat all standard benchmarks, but also medium-term Bank of England forecasts, despite the latter’s significant resources expended in accurate forecasting (Independent Evaluation Office, 2024). However, RaGNAR’s graph selection process was rudimentary: sampled graphs used no information about past samples, leading to the necessity of sampling a large number of graphs before reaching a satisfactory one (or average over a set of good ones).

To integrate the strengths of the GNAR model while addressing the inefficiencies of the random sampling procedure, we propose the Genetic GNAR (GenGNAR) model. GenGNAR uses a genetic algorithm to construct new candidate graphs by exploiting information from previously evaluated samples, enabling a more directed and data-driven exploration of the graph space. Below we apply GenGNAR to the inflation forecasting task, where it delivers significant improvements in forecasting accuracy, particularly at medium- and longer-term horizons.

Our article is structured as follows. Section 2 introduces the UK inflation dataset we use, the model evaluation design and the benchmarks employed to evaluate the performance of the GenGNAR model. Section 3 details the GNAR model, the GenGNAR variant and the genetic algorithm underlying it. Section 4 compares the inflation forecasting results of Genetic GNAR to other GNAR variants, as well as with the classical statistical and modern machine learning models introduced in Section 2.3 and official Bank of England forecasts. In addition, robustness and hyperparameter sensitivity model checks are discussed in detail. Section 5 concludes and presents suggestions for further lines of research.

Contributions. The main contributions of this article are threefold. First, we introduce GenGNAR, a genetic algorithm framework for graph selection in GNAR models, replacing the purely random exploratory search used in RaGNAR with a directed search that exploits information from previously evaluated graphs. Second, on UK inflation forecasting, we show that GenGNAR yields a clear empirical improvement over the earlier RaGNAR benchmark, especially at medium and longer forecasting horizons, with supporting evidence from repeated runs and non-parametric significance tests. Third, we explain several practically relevant extensions of the framework and benchmark the resulting best specification against standard statistical and machine learning models, as well as official Bank of England forecasts.

2 Dataset, Evaluation Design and Benchmarks

2.1 Inflation dataset and pre-processing

Inflation and CPI. The UK Consumer Price Index (CPI) is a key metric employed to track inflation. It is a weighted sum of a basket of goods and services consumed by a typical household in the UK (Office for National Statistics, 2017). The weights are calculated based on the expenditure of a typical UK household. The basket of goods and services can be divided into 12 main divisions, which are food and non-alcoholic beverages; alcoholic beverages and tobacco; clothing and footwear; housing, water, electricity, gas and other fuels; furnishings, household equipment and maintenance; health; transport;

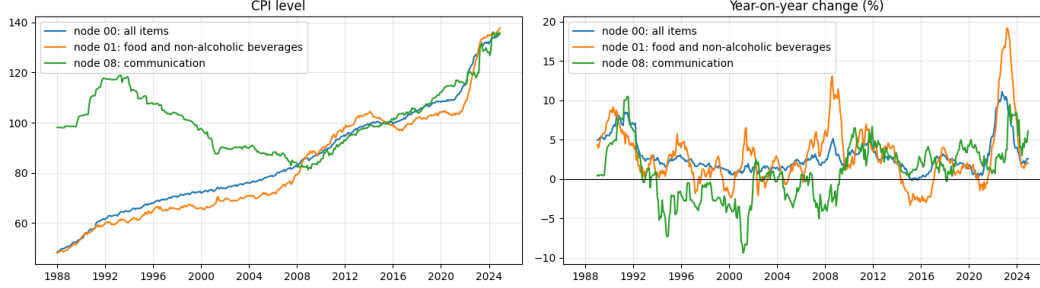


Figure 1: The figure on the left shows three time series corresponding to CPI, food and non-alcoholic beverages, and communication nodes. The figure on the right shows the year-on-year percentage change of the same three time series.

communication; recreation and culture; education; restaurants and hotels and miscellaneous goods and services. The divisions can be further divided into smaller categories, namely another 40 groups, 71 classes and 192 items, see Office for National Statistics (2017) for further details. For example, code 01.1.5.3 corresponds to the item *Olive oil*, in the class *Oil and Fats*, which belongs to the group *Food*, which itself belongs to the division *Food and non-alcoholic beverages*.

Pre-processing. After some experimental testing and pruning, the 71 classes were used to form 65 time series, plus an additional time series for the headline CPI. A complete list of the variables used in the dataset is produced in Appendix B. This seems to provide enough contextual time series for good CPI forecasting, whereas increasing the granularity of data by including more specific item-based time series makes the graph search space even larger without apparently adding relevant signals. Nason and Palasciano (2026) included divisions, groups and classes together to obtain 114 time series. However, experimental testing presented in Appendix F.2 shows that this does not produce a statistically significant improvement in forecasting performance across various experimental setups. Furthermore, using more time series increases the computational cost throughout the pipeline, so we decided to use the smaller aforementioned dataset.

We use monthly data from January 1988 to December 2024, which gives a total of 444 time points. We use year-on-year percentage change, since this reduces seasonal effects and the non-stationarity of the series, and in many instances this is the quantity of interest to economists. An example of three time series and the effect of the year-on-year percentage transformation is presented in Figure 1.

2.2 Evaluation design

Model Evaluation. We are interested in the forecasting accuracy for different forecasting horizons of the models being compared. In practice, this means first fixing a *forecasting horizon* h , and a *testing window* W_{test} . Then, if the time series is of length T , for each $t \in \{T - W_{\text{test}} - h, \dots, T - h\}$ the model is fitted with data up to time t , which is then used to make a h -step ahead forecast, denoted by $\hat{X}_{t,h}$. The resulting Root Mean Square Error (RMSE) measures forecasting accuracy. An important contribution of RaGNAR from Nason and Palasciano (2026) is that instead of using the total RMSE to choose appropriate

graphs, they focus only on the error of the CPI forecast. That is, if X_t^{CPI} corresponds to the univariate CPI time series, then they only used the following RMSE:

$$\text{RMSE} = \left\{ W_{\text{test}}^{-1} \sum_{t=T-h-W_{\text{test}}+1}^{T-h} (X_{t+h}^{\text{CPI}} - \hat{X}_{t,h}^{\text{CPI}})^2 \right\}^{\frac{1}{2}}. \quad (1)$$

Our article follows the same approach after experimentally validating that this procedure is computationally more efficient and leads to better CPI forecasts. It is possible to compute the RMSE of the whole model (i.e. all the nodes), but in practice we are interested in accurate CPI forecasting, and using the CPI-specific RMSE as the fitness function for graph search yields the best results in practice. Hence, CPI RMSE is the error which is reported in all later tables. We could also use alternative measures of error such as MAPE, for example. Indeed, MAPE was used by Nason and Palasciano (2026) and the conclusions were broadly similar.

If the model is of stochastic nature, meaning that each time the model is run a different result is obtained because of some random component of the model, then it is also necessary to average across different independent runs to obtain a reliable estimate of the true performance of the model. This is the case of GNAR with random or genetic graph search, because graph sampling is random and this will change for every run.

2.3 Statistical models and Bank of England benchmarks

Our proposed GenGNAR framework is compared against a set of standard statistical and machine learning benchmarks commonly used in macroeconomic and inflation forecasting. These models span univariate, multivariate, and non-linear specifications, allowing for a comprehensive assessment of forecasting performance and following the benchmarks used in the literature, see, e.g., Joseph et al. (2024); Nason and Palasciano (2026). The selection of models described below uses those used by various communities including statisticians, economists and machine learning practitioners. In all of the models presented below, hyperparameters have to be chosen. Sometimes, metrics like AIC and BIC can be used. However, in most instances, we prefer to choose the hyperparameters which produce the lowest prediction error on a validation set. This is the case for the rolling mean and VAR models. We make this methodological choice because we are comparing models based on their forecasting accuracy, so comparing this directly on a validation set seems to be an appropriate method.

Rolling mean The rolling mean model is the simplest univariate benchmark and is widely used as a baseline in macroeconomic forecasting. For each component i , the rolling mean model with lag p is given by

$$X_t^i = \frac{1}{p} \sum_{j=1}^p X_{t-j}^i + \varepsilon_t^i \quad (2)$$

$$\hat{X}_{t+h|t}^i = \frac{1}{h-1} \sum_{j=1}^{h-1} \hat{X}_{t+h-j|t}^i + \frac{1}{p-h+1} \sum_{j=h}^p X_{t+h-j}^i, \quad (3)$$

where ε_t^i is a white-noise innovation. Multi-step-ahead forecasts are made by implementing one-step ahead forecasts iteratively on predictions as per (3). Despite its simplicity, the

rolling mean model often provides a competitive benchmark for inflation series, particularly at short horizons.

ARIMA The Autoregressive Integrated Moving Average (ARIMA) (Box et al., 2016; Brockwell and Davis, 2009) model is a well-known univariate model built from an autoregressive component and a moving average one. Furthermore, the ARIMA(p, d, q) model applies a differencing operator $(1 - L)^d$ to X_t^i prior to estimation in order to induce stationarity, where L denotes the lag operator which acts as $LX_t = X_{t-1}$. To choose the model orders p, d, q , it is standard to use a criterion such as BIC, which is precisely what variants such as AutoARIMA can do given a finite set of possible values to search among. The AutoARIMA model implemented in Section 4.3 uses the pmdarima python package from Smith et al. (2017–).

VAR The vector autoregressive model of order p , VAR(p) (Quenouille, 1957; Sims, 1980; Lütkepohl, 2005), is the multivariate analogue of the univariate autoregressive model. Let $X_t \in \mathbb{R}^N$ denote the multivariate time series. The VAR(p) model is given by

$$X_t = \sum_{j=1}^p A_j X_{t-j} + \varepsilon_t, \quad (4)$$

where $A_j \in \mathbb{R}^{N \times N}$ are coefficient matrices and ε_t is a vector of temporally uncorrelated, zero-mean innovations with a temporally invariant covariance matrix $\mathbb{E}[\varepsilon_t \varepsilon_t^\top] = \Sigma$. VAR models capture linear cross-dependencies between components but quickly become parameter-intensive as the number of parameters is pN^2 . As will be shown later, GNAR models can be represented in the form presented in (4), but with restrictions on the lag matrices A_j . These restrictions yield bounds on the number of parameters of a GNAR model, which for a global specification has at most $p + pN$ parameters.

LSTM Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) are a class of recurrent neural networks designed to capture nonlinear temporal dependencies while mitigating the vanishing-gradient problem. Given the input sequence $\{X_t\}_{t=1}^T$, an LSTM updates a hidden state h_t and a cell state c_t via a system of gated nonlinear recursions,

$$\begin{aligned} f_t &= \sigma(W_f X_t + U_f h_{t-1} + b_f), \\ i_t &= \sigma(W_i X_t + U_i h_{t-1} + b_i), \\ o_t &= \sigma(W_o X_t + U_o h_{t-1} + b_o), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_c X_t + U_c h_{t-1} + b_c), \\ h_t &= o_t \odot \tanh(c_t), \end{aligned} \quad (5)$$

where $W_f, W_i, W_o, W_c, U_f, U_i, U_o, U_c$ are matrices, b_f, b_i, b_o, b_c vectors, $\sigma(\cdot)$ denotes the sigmoid function and $\odot$ element-wise multiplication. Intuitively, the hidden state h_t is a representation of the time series X_t , which can be used to make predictions by passing through an output layer, i.e. $\hat{X}_{t-1,1} = g_\theta(h_t)$, where g_θ is another activation function such as a sigmoid or tanh. On the other hand, the cell state c_t can be thought of as the long-term memory of the time series, which is constantly updated with input, output and forget gates. These control what new information is used to make a new candidate, and what information about c_{t-1} is forgotten respectively.

All the previous models used least-squares optimisation methods and were analytically tractable. Here, the model is trained using gradient descent to minimise the same RMSE prediction error. Once the trained parameters are obtained, forecasts at horizon h are obtained by iterating the network forward with these learned parameters. LSTMs provide a flexible non-linear benchmark, but require substantially more data and careful regularisation to avoid overfitting and allow for hyperparameter fine-tuning.

Chronos Recently, there has been an explosion in the popularity of models that use attention mechanisms to model sequential data, including time series. There has also been growing interest in exploring the pre-training of models in the search for the so-called foundation models. As an example of these models, Chronos from Ansari et al. (2024) has been included, which is a family of large pre-trained foundation models for time series forecasting based on transformer architectures. Unlike traditional statistical or machine learning models, Chronos is trained on a massive and heterogeneous corpus of time series and is subsequently fine-tuned or directly applied to new forecasting tasks.

Chronos produces probabilistic forecasts for future observations by modelling the joint distribution of future values using self-attention mechanisms. The transformer architecture allows the model to capture long-range temporal dependencies without relying on explicit autoregressive recursion. The model employed as a benchmark is Chronos-2 from Ansari et al. (2025), an extension of the original.

Bank of England forecasts The Monetary Policy Committee (MPC) is the official authority inside the Bank of England responsible for monetary policy. The MPC is also involved in inflation forecasting, although much of the work is performed by the Bank’s staff according to the Bernanke Review, which we draw upon here. The forecasts from specific models are modal in nature, meaning that they are maximum a posteriori predictions, but combined forecasts are described as “the best collective judgement of the Committee” (Independent Evaluation Office, 2024, p. 20). That is, the forecast is not based on a single model. Instead, several statistical and economic models are used to make forecasts, which together with contextual economic information, is used by the MPC’s members to make this best judgement. Despite this, the Bank uses as a benchmark a Dynamic Stochastic General Equilibrium (DSGE) model called COMPASS, see page 22 of Independent Evaluation Office (2024). This is an official Bank of England benchmark, but its role in constructing the Committee’s forecasts has diminished lately. This has been substituted by a selection of sectorial and purely statistical models, which are then modified by human judgement. Among the statistical models, AR(2) and VAR models still dominate the landscape (Joseph et al., 2021). More information on institutional forecasting can be found in Appendix A. This process is repeated quarterly and culminates with the Committee’s release of its quarterly *Monetary Policy Report*. The Bank’s forecasts used in this article are taken from these reports and are compared with actual inflation values to construct the RMSE of the Bank’s forecasts.

3 Methodology

This section reviews GNAR models, network selection and then explains how to use genetic algorithms with GNAR models. Subsection 3.1 explains the core Generalised Network Autoregressive model and the variant used in this paper, including model estimation and

how it produces forecasts. The GNAR model requires an *a priori* choice of graph. Previous graph selection methods are discussed in subsection 3.2, and then our proposed method based on a genetic algorithm is presented in subsection 3.3.

3.1 Generalised Network Autoregressive models

Generalised Network Autoregressive (GNAR) processes are highly parsimonious models for multivariate time series introduced by Knight et al. (2016, 2020). We first introduce some notation that underpins the GNAR models.

Definition 3.1 (Graph theory definitions). Let $G = (V, E)$ be a graph with vertex set V and edge set E . In this paper, only undirected graphs are considered; that is, elements of the edge set are unordered pairs of the form $\{i, j\}$ for $i, j \in V$. Then define:

- The **shortest path distance** $d(i, j)$ between nodes $i, j \in V$ as the number of edges in the shortest path connecting them. For undirected graphs, $d(i, j) = d(j, i)$. This is not a physical distance sometimes used when graphs are embedded in metric spaces such as in Knight et al. (2016).
- The **r -stage neighbourhood** $\mathcal{N}_i^r$ of node $i \in V$ is the set of vertices at a distance r from it, that is

$$\mathcal{N}_i^r = \{j \in V : d(i, j) = r\}.$$

Let $X_t = (X_t^1, \dots, X_t^N)^T$ be an N -dimensional multivariate time series process. This can be viewed as a process on an N -node graph $G = (V, E)$, where node $i \in V$ corresponds to the marginal time series X_t^i . Edges between nodes encode the relationships between the corresponding marginal time series. Given this notation, the model presented in Knight et al. (2020) can be synthesised as follows:

Definition 3.2 (GNAR model). The GNAR process of order $p, \mathbf{s}$, where $p \in \mathbb{N}$ and $\mathbf{s} = (s_1, \dots, s_p) \in \mathbb{N}_0^p$, for an N -dimensional time series X_t and $t \in 1, \dots, T$ is given by

$$X_t^i = \sum_{j=1}^p \left(\alpha_{i,j} X_{t-j}^i + \sum_{r=1}^{s_j} \beta_{i,j,r} \sum_{k \in \mathcal{N}_i^r} w_{i,k} X_{t-j}^k \right) + u_t^i, \quad (6)$$

where $i \in 1, \dots, K$ and

- u_t^i are independent, identically distributed white noise processes,
- $\alpha_{i,j}, \beta_{i,j,r}$ are the model parameters, and
- $w_{i,k}$ is the (usually deterministic) connection weight between node i, k . Typically, but not always, $w_{i,k} = 1/|\mathcal{N}_i^r|$ if $d(i, k) = r$.

The model specified by (6) is commonly referred to as a GNAR($p, \mathbf{s}$) model. Below, we describe a variant in (7) which, rather than including every lag from 1 to p , only a specified set of lags $p_1, \dots, p_M$ are retained. This is convenient when dependence is expected to concentrate at particular horizons, for instance at seasonal frequencies. For monthly data, lags such as 12 or 24 are often useful to be incorporated directly to capture annual or biannual effects, without introducing the intermediate lags. In this way, seasonal dynamics can be accounted for, while keeping the parametrisation relatively parsimonious.

Definition 3.3 (Discrete seasonal GNAR variant). The GNAR process of order $\mathbf{p}, \mathbf{s}$, where $\mathbf{p} = (p_1, \dots, p_M) \in \mathbb{N}^M$ and $\mathbf{s} = (s_1, \dots, s_M) \in \mathbb{N}_0^M$ for X_t is given by

$$X_t^i = \sum_{j=1}^M \left(\alpha_{i,j} X_{t-p_j}^i + \sum_{r=1}^{s_j} \frac{\beta_{i,j,r}}{|\mathcal{N}_i^r|} \sum_{k \in \mathcal{N}_i^r} X_{t-p_j}^k \right) + u_t^i, \quad (7)$$

where all definitions are as before.

The hyperparameters $p_1, \dots, p_M$ need to be chosen. As discussed in Section 2, in practice this can be done by selecting a set of possible $p_1, \dots, p_M$ and then looking at criteria like AIC, BIC or the prediction error on a validation set of the resulting model. Again, we looked at a range of values for $p_1, \dots, p_M$ and chose the model with the lowest validation prediction error. This is both methodologically sound and consistent with the hyperparameter selection of the benchmark models.

Depending on the constraints imposed on the model parameters, there are three different types of GNAR models with varying numbers of parameters to estimate:

- **Global** Here $\alpha_{i,j} = \alpha_j$ and $\beta_{i,j,r} = \beta_{j,r}$ for all $i \in V$.
- **Local- α** Here constraints on the betas are still imposed, i.e. $\beta_{i,j,r} = \beta_{j,r}$ for all $i \in V$.
- **Local- α, β** No constraints are imposed on the parameters.

The choice of the model type actually fitted will depend on the assessment of model fit. With smaller data sets, the more parsimonious global model might be more suitable, for example. As with many other models, there is a trade-off between the expressiveness of the model and the accuracy of the parameter estimates.

Model estimation. As with VAR models, the nodes can be stacked up spatially and temporally to express (6) or (7) in a linear form. For instance, (7) can be expressed as a restricted-VAR model as in (4), where A_j are lag matrices formed by *repeated* $\alpha_{i,j}$ and $\beta_{i,j,r}$ parameters. This, in turn, can be stacked to include all available training time instances to get a matrix linear model $Y = XB + U$, with some linear constraints in B of the form $\text{Vec}(B) = C\gamma$, where γ represents a vector of unrestricted parameters (i.e. the $\alpha_{i,j}$ and $\beta_{i,j,r}$ parameters), following the approach of Lütkepohl (2005). Hence, under a Gaussian likelihood, the maximum-likelihood estimator is equal to an ordinary least-squares estimator which is just a solution to the following optimisation problem:

$$\hat{\gamma} \in \arg \min_{\gamma} \| \text{Vec}(Y) - (I \otimes X)C\gamma \|_2^2, \quad (8)$$

where the inside is the vectorised form of the matrix residuals $Y - XB$, given the linear constraint on B .

Forecasting. GNAR forecasts are given by iteratively obtaining one-step ahead forecasts from (7) and using historical data or past forecasts as inputs. For example, for $h < M$, forecasts are computed as follow:

$$\hat{X}_{t+h|t} = \sum_{j=1}^{h-1} \hat{A}_j \hat{X}_{t+h-j|t} + \sum_{j=h}^M \hat{A}_j X_{t+h-j}, \quad (9)$$

where $\hat{A}_j$ are the estimated lag matrices and $\hat{X}_{t+h|t}$ is the h-step ahead forecast at time t . That is, it is the forecast for time point $t + h$ using data up to and including time t .

3.2 Network selection for GNAR

Standard GNAR models require the user to select a graph to describe the pairwise relationships between the marginal time series. In the GNAR network time series literature, three main graph selection methods can be found:

Domain-informed selection In this situation, expert domain knowledge is used to propose a well-informed graph. For example: Knight et al. (2016) examined the spread of mumps across UK counties, where each node corresponded to a county and the time series was formed by looking at the number of mumps cases recorded in that county. Here, they constructed a network based on geographical proximity, that is, joining those counties which are close to each other. Similarly, Nason et al. (2026) examined the number of admissions to mechanical ventilation beds at 140 NHS Trusts in England and Wales. They also construct the network based on the geographical proximity of the Trusts.

Random graphs One way to generate graphs for GNAR processes is by randomly choosing edges. Random graph exploration for GNAR models was first proposed by Leeming (2019). Recently, this idea has been extended to form the Random GNAR (RaGNAR) method presented in Nason and Palasciano (2026). Pseudocode for a simplified version of the RaGNAR algorithm is presented in Algorithm 1.

Algorithm 1: Simplified version of RaGNAR

Input: Data $\{X_t\}_{t=1}^T \subset \mathbb{R}^N$
Input: GNAR parameters $p \in \mathbb{N}$, $\mathbf{s} \in \mathbb{N}^p$
Input: Number of generations M

- 1 **for** $i = 1$ **to** M **do**
- 2 Generate a random graph;
- 3 Fit GNAR($p, \mathbf{s}$) model with graph;
- 4 Compute validation prediction RMSE;
- 5 **return** Graph with smallest validation RMSE;

Learned graph selection Finally, some approaches try to infer a latent graph structure from the data itself, by looking directly at the relationships between the marginal time series. This is the case with many recent machine learning approaches that employ graph-based time series models. For example, Kipf et al. (2018) introduced the Neural Relational Inference, which frames graph discovery as a latent variable problem and uses variational auto-encoders to infer edge probabilities in dynamical systems. This was extended by improving message-passing and including structural priors onto the learned graphs (Chen et al., 2021). Meanwhile, spectral-temporal graph neural network (GNN) models such as StemGNN (Cao et al., 2021) learn dependencies completely in the spectral domain using attention mechanisms. Other approaches extend this idea to multivariate forecasting, coupling graph learning with temporal predictors (Satorras et al., 2022). Despite strong results, these models typically require large training datasets, which limits the opportunity to employ many of these methods to real-world economic data, which is often limited. The gradient descent based training of these large neural networks also incurs in vast

computational costs, which is not the case with GNAR models, whose parameters are rapidly found using closed-form formulae.

3.3 The genetic search GNAR: GenGNAR

We propose a novel type of exploratory graph selection by employing a *genetic algorithm* (Vose, 1999). It is an alternative to RaGNAR, where candidate graphs are not randomly sampled from an Erdős-Renyi distribution. Instead, a genetic algorithm is used to propose new candidate graphs by using knowledge of what previously proposed graphs gave rise to good models. Preliminary exploration of genetic algorithms for GNAR models was first carried out by the unpublished MSc thesis McCarthy (2024). Our work substantially develops the ideas, and extensively explores its application to forecasting particularly for CPI data. Appendix C explains in more detail the differences between McCarthy (2024) and our work.

A genetic algorithm is an optimisation algorithm inspired by natural selection, which has the aim of finding an optimal candidate in a search space Ω with respect to some fitness function $f : \Omega \rightarrow \mathbb{R}$, (Vose, 1999). That is, the objective is to solve the general optimisation problem: $\arg \max_{\omega \in \Omega} f(\omega)$.

Conventional optimisation methods such as gradient descent require Ω to have some sort of differential manifold structure as a bare minimum. In the case of graph selection, since the data consists of an N -dimensional time series, an N -node graph is required to fit a GNAR model, so Ω will be the set of all possible undirected N -node graphs with no self-loops or multiple edges between nodes. For h -step ahead forecasts, the negative h -step ahead forecasting error on a validation set is a possible objective fitness function. Nason and Palasciano (2026) used the one-step-ahead forecasting error as the objective function even though the downstream task was h -step-ahead forecasting. However, in this article the fitness function aligns with the type of out-of-sample forecasting being made. Furthermore, G is used to denote the graph and its adjacency matrix, since the labelling of the nodes does not affect the algorithm, so they can be treated interchangeably. Therefore, the optimisation problem becomes:

$$\arg \max_{G \in \Omega} \{-\text{RMSE}(h; G)\}, \text{ where } \Omega = \{G \in \{0, 1\}^{N \times N} : G = G^T, G_{ii} = 0\}, \quad (10)$$

where $\text{RMSE}(h; G)$ is the empirical forecast loss on the validation set, as presented in subsection 2.2. From now on, the h and G are dropped for notational simplicity.

The stages of the algorithm are described next. To begin, the algorithm is initialised:

- (i) **Initialisation.** Firstly, a random initial population Λ_0 is sampled from Ω . That is, a number P of N -node random graphs is generated. As in the RaGNAR case, the graphs are obtained by sampling from the Erdős-Renyi distribution.

Then, the algorithm proceeds recursively. To produce the next population Λ_1 , elements of Λ_0 *reproduce*. This involves four different steps:

- (ii) **Selection.** Two graphs G_1, G_2 from the population Λ_0 are sampled uniformly without replacement. However, there are many other variants as described in Vose (1999) which make use of the fitness score of the graphs in Λ_0 for selection.
- (iii) **Crossover.** A new graph is produced by performing a crossover between the selected graphs, denoted by $G = \chi(G_1, G_2)$. In practice, this means selecting some edges from G_1 and others from G_2 at random. The details are presented in Appendix D.1.

- (iv) **Mutation.** The crossover graph then has some edges added or removed at random, as described in Appendix D.2, to produce a candidate graph G . This is intended to add some stochasticity to the algorithm and is crucial for it to function correctly. Indeed, the mutation mechanism avoids premature convergence to non-optimal local minima by encouraging continuous exploration of the graph search space.
- (v) **Replacement.** Once reproduction is complete, the fitness function is used to decide whether to *replace* an element of Λ_0 with the newly generated graph. That is, only a certain number of optimal graphs with respect to the fitness function are incorporated into Λ_1 . Genetic algorithms employ a variety of replacement techniques (Vose, 1999), but in GenGNAR we simply add G to Λ_1 if it has a higher fitness score than the worst element of Λ_0 .

The above algorithm emulates a natural selection process by producing a sequence of populations $\Lambda_0, \Lambda_1, \Lambda_2, \dots$. The idea is that, as $t \rightarrow \infty$, the population Λ_t increasingly concentrates around near-optimal solutions of the optimisation problem $\arg \min_{G \in \Omega} f(G)$. The sequence of populations $\{\Lambda_t\}_{t=0}^{\infty}$ can be viewed as a Markov chain, from which it can be deduced that the chain will converge to an optimal solution under mild regularity conditions on the search space Ω and on the reproduction, mutation and selection procedures, see Vose (1999). A simplified pseudocode is presented in Algorithm 2.

Algorithm 2: Genetic Algorithm for Network Selection

Input: $\{X_t\}_{t=1}^T \subset \mathbb{R}^N$, GNAR parameters $p \in \mathbb{N}$, $\mathbf{s} \in \mathbb{N}^p$, Number of generations M , Population size P , Selection size K

- 1 **Initialize:**
- 2 Generate an initial population $\tilde{\Lambda}_0$ of P random N -node graphs $\subset \Omega$;
- 3 **foreach** $G \in \tilde{\Lambda}_0$ **do**
- 4 Fit a GNAR($p, \mathbf{s}$) model ;
- 5 Compute prediction MSE ;
- 6 Select the top K graphs with lowest validation prediction RMSE to form Λ_0 ;
- 7 **for** $i = 1$ **to** M **do**
- 8 **Selection:** Randomly choose two parent graphs $G_1, G_2 \in \Lambda_{i-1}$;
- 9 **Crossover:** Generate offspring graph $\tilde{G} = \chi(G_1, G_2)$;
- 10 **Mutation:** Apply mutation to obtain $G = \mu(\tilde{G})$;
- 11 **Evaluation:** Compute the validation prediction RMSE of G ;
- 12 **Replacement:** If G has lower RMSE than the worst graph in Λ_{i-1} , replace it ;
- 13 **return** *Best graph in Λ_M*

The GenGNAR algorithm also includes several extensions. Much like in RaGNAR, although the graph selection is based solely on the prediction RMSE of the graph in question, an ensemble of graphs is used for out-of-sample forecasts. Specifically, at a specific time origin t , after the algorithm runs, a number of best graphs $G_{t,1}, \dots, G_{t,K}$ is selected, which in turn define GNAR models with which predictions $\hat{X}_{t+h|t}(G_{t,1}), \dots, \hat{X}_{t+h|t}(G_{t,K})$ are made. Then, the actual forecast of the whole model as a whole is an ensemble of the predictions, that is

$$\hat{X}_{t+h|t} = \frac{1}{K} \sum_{k=1}^K \hat{X}_{t+h|t}(G_{t,k}). \quad (11)$$

In addition, RaGNAR also creates an ensemble by fixing one best graph, but then changing the GNAR lag hyperparameters to define several models for that same graph. GenGNAR draws from the success of both forms of ensembling to make forecasts. This is explained more precisely in the following subsection.

3.4 Practicalities of network selection

Graph selection. How do the random and genetic algorithms evaluate a graph G which will be then used to make an out-of-sample forecast $\hat{X}_{t+h}$ for some $t \in \{T - W_{\text{test}} - h, \dots, T - h\}$? The idea is to use an estimator of the RMSE for that specific graph by making forecasts in a period before time t . To do so, a validation window W_{val} is fixed. In practice, W_{val} has been fixed to be 30. This value was chosen by testing a small range of values. Empirically, we found that small values of W_{val} gave rise to graphs which resulted in very low validation error but non-optimal out-of-sample errors. By increasing W_{val} , the chosen graphs had to be useful at a higher number of time origins, and hence were more likely to perform well out of sample.

Then, for a given graph G , forecasts are made at

$$\tau \in \{t - h - W_{\text{val}} - 1, \dots, t - h\},$$

which in turn gives rise to a validation RMSE. Concretely, for any candidate graph G , we fit the corresponding GNAR model using observations up to time τ , and evaluate its out-of-sample performance at time step $\tau + h$. Denote by $\hat{X}_{\tau+h|\tau}(G)$ the h -step-ahead forecast at time $\tau + h$ produced by the GNAR model fitted on data up to time τ under graph G . The fitness of G at forecast origin t is defined as the negative of the validation RMSE

$$\text{RMSE}_{\text{val},t}(G) := \left\{ W_{\text{val}}^{-1} \sum_{\tau=t-W_{\text{val}}-h}^{t-h} \left(X_{\tau+h} - \hat{X}_{\tau+h|\tau}(G) \right)^2 \right\}^{\frac{1}{2}}. \quad (12)$$

Hence, to make out-of-sample forecasts from the origin t , a graph is selected based on the validation RMSE, which only uses data up to that origin. In Appendix E we present an algorithm to efficiently solve the W_{val} equations needed to calculate the validation RMSE.

Single-graph vs ensemble forecasting. In the single graph specification, the algorithm selects

$$G_t^* \in \arg \min_{G \in \mathcal{G}_t} \text{RMSE}_{\text{val},t}(G), \quad (13)$$

where $\mathcal{G}_t$ is the set of graphs explored by the search algorithm at origin t . In the ensemble specification, the algorithm retains the top K graphs $\{G_{t,1}, \dots, G_{t,K}\}$ with the lowest validation RMSE. For forecasting, each selected graph is re-estimated using all observations up to time t and produces an h -step-ahead forecast $\hat{X}_{t+h|t}(G_{t,k})$. The ensemble forecast is the simple average as given by Equation 11, with $K = 1$ recovering the single-graph case. This separation between (i) graph selection, (ii) model estimation, and (iii) forecast evaluation avoids look-ahead bias by construction.

4 Results

This section presents a selection of comparisons of different forecasting methods. We predominantly compare GenGNAR with RaGNAR, as RaGNAR from Nason and Palasciano

p		Forecasting Horizon, h					
		1	2	3	4	5	6
1,2	Random	0.523±0.010	0.865±0.019	1.009±0.027	1.228±0.050	1.523±0.039	1.822±0.071
	Genetic	0.485±0.024	0.869±0.021	0.827±0.030	0.831±0.046	0.936±0.052	1.124±0.063
	Improvement	7.2%	−0.4%	18.0%	32.4%	38.5%	38.3%
1,2,12	Random	0.580±0.020	0.841±0.024	0.991±0.028	1.168±0.021	1.399±0.038	1.667±0.037
	Genetic	0.531±0.032	0.821±0.033	0.785±0.044	0.802±0.051	0.843±0.068	0.906±0.057
	Improvement	8.4%	2.4%	20.9%	31.3%	39.8%	45.7%
1,2,12,24	Random	0.512±0.017	0.830±0.021	0.975±0.034	1.162±0.037	1.398±0.058	1.661±0.056
	Genetic	0.493±0.025	0.818±0.041	0.835±0.028	0.816±0.036	0.796±0.031	0.874±0.040
	Improvement	3.7%	1.4%	14.4%	29.8%	43.1%	47.4%

Table 1: Results for different GNAR models with a global α, β specification. The p column specifies the autoregressive structure of the GNAR model, whereas the neighbourhood component is always given by $s_j = 1$. Ensembling the best 5 graphs has been used in each algorithm. Results show the mean and standard deviation of 25 independent runs using different seeds.

(2026) exhibited excellent results in forecasting CPI and better than all of the wide variety of benchmarks in that paper. We also compare GenGNAR with the benchmarks listed above as well as with the Bank of England’s own forecasts. The code used for the empirical results can be found at <https://github.com/SergioEstanRuiz/genetic-python>.

4.1 GenGNAR versus RaGNAR

The reported test RMSE is computed over monthly forecast origins from January 2021 to December 2024, and, at each forecast origin t , the validation period is taken to be 10% of the available data, which corresponds to the previous 30 months. Since both procedures are stochastic, each experimental configuration is repeated 25 times and the reported results correspond to the mean and standard deviation across runs. Averaging across 25 independent runs gives a small standard deviation to mean ratio, whilst ensuring that the computational cost of experimenting remains manageable.

Table 1 shows the results of these comparisons for the GNAR([1, 2], [1]), GNAR([1, 2, 12], [1]) and GNAR([1, 2, 12, 24], [1]) models respectively in the global α, β specification, showing that GenGNAR improves on the earlier RaGNAR benchmark in every experimental configuration but one: the GNAR([1, 2], [1]) for $h = 2$. Results for the local specifications of these three models are discussed in Appendix F.1 and further demonstrate the general superiority of GenGNAR over RaGNAR. These specific model orders were chosen for several reasons. First of all, very similar model orders were chosen in Nason and Palasciano (2026), following the tradition of selecting a mixture of small lags and annual/biannual lags to mitigate some of the seasonal components still present in the data, see for example Marcellino et al. (2006). But most importantly, different models orders were tested and these gave rise to models with the best predictive performance.

Table 1 shows that the GenGNAR generally outperforms RaGNAR, but the performance gap significantly widens as the forecasting horizon increases. A potential exception is the $h = 2$ forecasts, where they are similar.

Further, Table 1 shows that adding lags 12 and 24 tends to improve performance for both search strategies, but the benefit is larger for GenGNAR. A plausible explanation is

that these lags capture residual seasonal structure that is not fully removed by the year-on-year transformation of the data. The richer lag specification enlarges the class of candidate GNAR models, and the directed search of GenGNAR appears better able than RaGNAR to exploit this additional structure.

The same conclusion holds when averaging across GNAR specifications. Table 2 reports the results for the AvGNAR experiment from Nason and Palasciano (2026), where model averaging is performed at the level of GNAR hyperparameters. Here again, GenGNAR outperforms the RaGNAR benchmark at every forecasting horizon, with relative improvements ranging from 7.7% to 46.2%. Hence, the advantage of GenGNAR is not confined to a single model specification, but persists in the model-averaging setting as well.

	Forecasting Horizon, h					
	1	2	3	4	5	6
Random	0.567 ± 0.012	0.851 ± 0.010	0.965 ± 0.026	1.107 ± 0.027	1.285 ± 0.042	1.550 ± 0.051
Genetic	0.499 ± 0.029	0.786 ± 0.020	0.763 ± 0.039	0.783 ± 0.032	0.788 ± 0.046	0.834 ± 0.051
Improvement	12.0%	7.7%	20.9%	29.3%	38.7%	46.2%

Table 2: Results for AvGNAR with GNAR([1,2],[1]) and GNAR([1,2,12],[1]). Results shown are the mean of ten independent runs.

Statistical comparison of algorithms. To further analyse the performance of GenGNAR compared to RaGNAR we, carry out a Mann-Whitney U -test (Dalgaard, 2008) for each forecasting horizon for the models shown in Tables 1 and 2. A one-tailed test was chosen because we want to test whether the genetic algorithm is better than the random algorithm, and not simply that their distributions are different. The U -statistics and one-tailed p -values for each experimental configuration are presented in Tables 10 and 11 respectively, in Appendix G. The results are clear: GenGNAR statistically outperforms RaGNAR at any significance level in 22 out of the 24 experiments. In the two exceptions where RaGNAR outperforms GenGNAR, the results are not statistically significant at the 5% significance level.

Training dynamics comparison. We also compare how the search algorithms refine their graph selection as the number of graphs sampled increases. Figure 2 presents the RMSE of the GNAR model as the number of generations, or number of sampled graphs, increases. Since the genetic algorithm is initialised by sampling random graphs to create the initial pool Λ_0 of graphs, the RMSE is similar to that of the random algorithm at initialisation. However, as soon as the algorithms run for a few hundred generations, the gap between the algorithms widens significantly. This suggests that the genetic algorithm might reach a significantly better performance having sampled orders of magnitude fewer graphs than the random algorithm. In Nason and Palasciano (2026), the authors sampled 10,000 graphs. Here, 1000 sampled graphs for GenGNAR produces competitive results.

This behaviour might indicate that the computational efficiency of the GenGNAR model is considerably better than that of RaGNAR. A naive coding of the algorithms might support this claim. However, random graphs can be sampled in parallel, which reduces the computational time enormously if parallelised correctly. On the other hand, the genetic algorithm is sequential by nature, so such parallelisation is not possible. Only

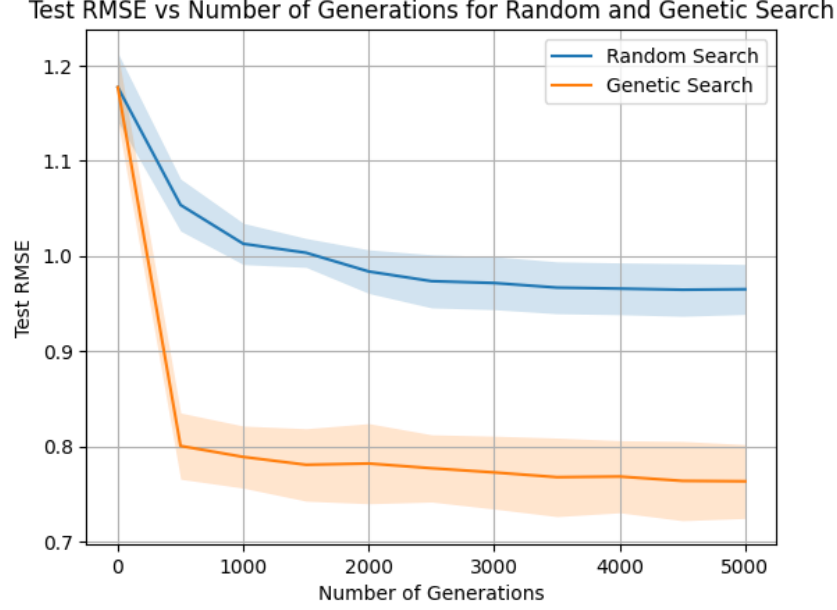


Figure 2: Test RMSE for the AvGNAR model presented in Table 2 for 3-step ahead forecasting. The shaded region between each error curve represents the standard deviation between the independent runs conducted for each search algorithm.

some parallelisation of the Island model variant described later could be carried out, which for most practical scenarios could be enough to be competitive in terms of computational efficiency.

4.2 Variants and sensitivity

This section explores two aspects of the GenGNAR model. Firstly, a robustness and sensitivity analysis is conducted, where the performance of the GenGNAR model is examined when hyperparameters are modified. Secondly, some variants of the basic genetic algorithm are explored.

Sensitivity to mutation probability. In the genetic algorithm, the mutation step randomly adds and deletes some edges from the graph in the crossover step. The number of edges changed is controlled by the mutation probability. Indeed, every 0/1 entry in the adjacency matrix is swapped for a 1/0 entry with a probability equal to the mutation probability. Table 3 reports the results for different mutation probabilities.

Two conclusions can be made about the results. First, the mutation step in the genetic algorithm is vital for its correct functioning. Indeed, if the mutation step did not exist, the algorithm would plateau much faster as the number of graphs that the algorithm could reach would be much smaller. If an edge was not present in the initial pool of graphs Λ_0 , then the algorithm could never produce a graph which included such an edge. This exploratory limitation would hinder the performance of the algorithm. Second, increasing the mutation probability improves up to a point, after which the performance starts decreasing again.

Mutation Probability	Forecasting Horizon, h		
	1	3	6
0	0.630 $\pm$ 0.028	1.125 $\pm$ 0.080	1.886 $\pm$ 0.160
0.001	0.574 $\pm$ 0.040	0.891 $\pm$ 0.059	1.022 $\pm$ 0.076
0.01	0.519 $\pm$ 0.027	0.800$\pm$0.030	0.944$\pm$0.079
0.1	0.501$\pm$0.006	0.883 $\pm$ 0.013	1.496 $\pm$ 0.029

Table 3: RMSE results for 1, 3 and 6-step ahead forecasts with different mutation probabilities. Results are the mean of ten independent runs.

Indeed, as the mutation probability approaches 1, the genetic algorithm converges towards the random algorithm.

Stochastic Block Matrix Graph Generation Variation. The CPI data we analyse is multivariate, but the key variable of interest is the CPI. Despite this, with the standard Erdős-Renyi graph-generating model underlying both RaGNAR and GenGNAR, the importance of the CPI node is not taken into account. To incorporate this inductive bias, a stochastic block model (SBM) was used to generate random graphs for both RaGNAR and GenGNAR. This contains two blocks, the first containing the CPI node and the second one containing the rest of the variables. The probability matrix implemented was

$$\text{Probability Matrix} = \begin{pmatrix} 0 & 0.03 \\ 0.03 & 0.1 \end{pmatrix}. \quad (14)$$

This permits the CPI node to be connected to many nodes, but the rest of the nodes to be sparsely connected amongst themselves. Table 4 shows that introduction of the SBM model improves forecasting some results for 1-, 3- and 6-step ahead RMSE for RaGNAR and GenGNAR. The improvement is more reliable for RaGNAR.

		Forecasting Horizon, h		
		1	3	6
RaGNAR	Block	0.551 $\pm$ 0.015	0.953 $\pm$ 0.034	1.541 $\pm$ 0.036
	Erdos	0.581 $\pm$ 0.015	0.985 $\pm$ 0.022	1.603 $\pm$ 0.058
	% improvement	5.1	3.2	3.9
GenGNAR	Block	0.520 $\pm$ 0.022	0.823 $\pm$ 0.055	0.876 $\pm$ 0.040
	Erdos	0.521 $\pm$ 0.026	0.830 $\pm$ 0.048	0.951 $\pm$ 0.053
	% improvement	0.2	0.8	7.9

Table 4: Results for RMSE of h -step ahead forecasts with different graph generating models. Results shown are the mean of ten independent runs.

Other variations explored. Here, we describe other variations which were explored but did not result in significant gains.

- **Island Model Variation.** The model was extended to include several independent populations or pools $\{\Lambda_t^0, \dots, \Lambda_t^{n_p}\}$ for some number of pools n_p . This is inspired by

the Island Model based on the seminal work of Tanese (1989). If each pool is initialised differently, then the idea is for each of the pools to converge to different local minima. With the ensembling of graphs above, this allows the model to pick up different edges and hence different forecasting improvements by having some heterogeneity across pools. Results of this experiment are presented in Table 5.

Pools	Graphs averaged per pool	Forecasting Horizon, h		
		1	3	6
1	1	0.525 $\pm$ 0.030	0.829 $\pm$ 0.044	0.949 $\pm$ 0.074
	2	0.530 $\pm$ 0.040	0.810 $\pm$ 0.036	0.954 $\pm$ 0.048
2	1	0.507 $\pm$ 0.017	0.820 $\pm$ 0.016	0.960 $\pm$ 0.053
	2	0.511 $\pm$ 0.014	0.798 $\pm$ 0.016	0.917 $\pm$ 0.048
3	1	0.504 $\pm$ 0.014	0.813 $\pm$ 0.029	0.986 $\pm$ 0.033
	2	0.508 $\pm$ 0.016	0.814 $\pm$ 0.019	0.987 $\pm$ 0.042

Table 5: Genetic algorithm with different number of pools, each containing 5 graphs. For the ensembling, 1 or 2 graphs are taken from each pool.

- **Immigration Variation.** When using the vanilla genetic algorithm, it is possible for it to become trapped in suboptimal local minima if the diversity of the population decreases too quickly. To avoid this, Yu et al. (2009) suggested an immigration variation where, every M generations, the genetic algorithm randomly samples a graph from an Erdős-Renyi distribution (the immigrant) and introduces it into the population. However, after some exploration by varying M , we discovered that the island model is a better solution to deal with the heterogeneity problem.

4.3 Benchmark comparison and the Bank of England’s forecasts

To establish a benchmark, the forecasting results of the models described in section 2.3, as well as those of the Bank of England, are presented here. Table 6 shows the forecasting RMSE over different horizons. Section 4.1 uses testing data comprising every month from January 2021 to December 2024. However, published, official forecasts made by the Bank of England are only available across all forecasting horizons quarterly from June 2022. To make fair comparisons across models, for each month t models are trained up to that month and h -step ahead forecasts are made. The root mean square error for each of the models across the available months is reported in Table 6. The table also reports the results for the best GenGNAR variant. In this case, it is a GenGNAR model with the following variations:

- Averaging of global α, β GNAR($[1,2],[1]$) and local α GNAR($[1,2,12],[1]$).
- A block model generating the random graphs with the same probability matrix as the one presented in Section 4.2.
- Island variation with two pools, with five graphs in each pool, but only the best two graphs per pool are selected for model averaging.

To account for the stochasticity of the genetic algorithm, ten independent runs are carried out and the reported RMSE is the mean across runs.

Models	Forecasting Horizon					
	1	2	3	4	5	6
Bank of England	0.177	0.336	0.408	0.854	0.929	0.920
Best GenGNAR	0.499	0.786	0.763	0.783	0.788	0.834
Rolling Mean $p = 1$	0.998	1.175	1.314	2.055	2.305	2.453
Rolling Mean, $p = 4$	1.364	1.367	1.503	2.290	2.594	2.703
AR, $p = 2$	1.221	1.319	1.456	2.255	2.475	2.580
Ensemble AR*	1.117	1.044	1.091	1.794	1.841	1.852
AutoARIMA	1.145	1.192	1.285	2.098	2.320	2.470
VAR(1)	1.216	1.277	1.396	2.047	2.371	2.495
VAR(2)	1.189	1.310	1.366	1.898	2.211	2.340
LSTM**	3.866	4.925	4.871	5.032	5.398	5.542
Chronos-2	0.311	1.167	1.158	1.245	1.934	2.072

Table 6: Benchmark results. *The ensemble of autoregressive models are of orders 2, 12, 24. **LSTM hyperparameters like the hidden dimension of the linear layers has been optimised by carrying out a sweep.

The benchmark results provide some interesting insights into inflation forecasting. Firstly, the ensemble AR model and Chronos-2 show the strongest results across all time horizons. It is particularly notable how a pre-trained model like Chronos-2 provides state-of-the-art forecasting, including some remarkable results like those seen in the one-step ahead predictions: whereas most models have an RMSE of around 1.1, Chronos’ RMSE goes as low as 0.311. Similarly, the RMSE for Chronos’ 4-step ahead forecast is only of 1.245, compared to the mean performance of 2.062 from the classical models. This suggests that foundation time series models are certainly a promising direction of research. However, since the pre-training dataset is not disclosed, the model might have been previously exposed to the UK CPI dataset during pre-training. If this is the case, a truly out-of-sample performance of Chronos might be significantly worse.

Despite this, the ensemble AR model provides a stronger and more consistent benchmark across most forecasting horizons, outperforming Chronos at the 2-, 3-, 5- and 6-step-ahead horizons. It is also notable that the LSTM performs substantially worse than the classical benchmarks. During the test period, which includes the COVID-19 and Russia–Ukraine crisis years of 2021–2022, the inflation dynamics appear to change sharply relative to the earlier training sample. However, all methods need to adapt to this and the AR benchmarks come out best.

How do the benchmarks compare to the Bank of England forecasts and those of the GenGNAR model? Firstly, the forecasts made by the Bank are significantly better than that of the benchmarks by a considerable margin. This is not surprising as the Monetary Policy Committee has a group of Bank of England expert statisticians and economists that provide both time series forecasts with inflation-tailored models, as well as domain-specific knowledge about the economic conditions to be able to employ contextual economic information to adjust forecasts. This can be especially useful when there are obvious shocks (e.g. wars) which cannot be easily predicted by analysing historical data, but which can be used to modify predictions made by models.

On the other hand, GenGNAR also outperforms all benchmarks except for Chronos

at time horizon 1. Of special interest is the comparison with its traditional multivariate counterpart: the VAR model. Both have a similar linear lag structure, but as was discussed in Section 2.3, the number of parameters in VAR models grows quadratically. Although this gives the VAR model flexibility, in most applications relative data scarcity leads to poor parameter estimation and hence poor forecasting. GNAR models utilise a network structure to dramatically decrease the number of parameters, which result in more accurate parameter estimation. This is then reflected in the impressive improvement in forecasting performance. GenGNAR is able to outperform the Bank’s forecasts at forecasting horizons 4,5 and 6. This is remarkable given the aforementioned amount of resources available to the Monetary Policy Committee, and the relatively lightweight structure of GenGNAR.

5 Conclusions and Further Work

This article introduces GenGNAR, a genetic algorithm powered graph selection method for GNAR models. This is motivated by the inefficiency of sampling large numbers of unrelated graphs in RaGNAR when selecting the network structure. The key methodological idea is to turn graph selection into an explicit optimisation problem, where candidate graphs are iteratively improved through a genetic algorithm with a validation RMSE-based fitness function. Empirically, the genetic search strategy yields systematic forecasting improvements over the purely random search conducted by RaGNAR, with gains that are particularly evident in high forecasting horizons.

Extensions. Several natural extensions follow. First, economic priors could be built directly into the graph search space, for example by biasing edge proposals toward relationships suggested by CPI basket structure or economic theory. Second, the framework is well-suited to inter-country or inter-sector datasets where inflation can propagate through trade and supply chains exposure. For example, inflationary pressures derived from supply shocks during the Ukraine crisis were first felt in countries with stronger trading ties to Ukraine or those with a stronger energy-dependence on Russian oil. This then propagated to countries which were better isolated from these shocks. Hence, trying to use inter-country data to exploit the economic dependencies between them could serve to better refine forecasting during high-volatility, shock-induced periods.

Limitations. The evidence presented in this article is based on a single application, namely UK CPI inflation forecasting, so the extent to which the gains of GenGNAR generalise to other macroeconomic datasets remains to be established. Moreover, although GenGNAR appears substantially more search-efficient than the earlier random-search approach, this does not automatically imply lower wall-clock cost, since random graph search is easier to parallelise whereas the genetic algorithm is inherently more sequential.

References

Ansari, A. F., Stella, L., Turkmen, C., Zhang, X., Mercado, P., Shen, H., Shchur, O., Rangapuram, S. S., Pineda Arango, S., Kapoor, S., Zschiegner, J., Maddix, D. C., Mahoney, M. W., Torkkola, K., Gordon Wilson, A., Bohlke-Schneider, M., and Wang, Y. (2024) Chronos: Learning the language of time series, *Transactions on Machine Learning Research*, p. 10/2024.

- Ansari, A. F., Shchur, O., Küken, J., Auer, A., Han, B., Mercado, P., Rangapuram, S. S., Shen, H., Stella, L., Zhang, X., Goswami, M., Kapoor, S., Maddix, D. C., Guerron, P., Hu, T., Yin, J., Erickson, N., Desai, P. M., Wang, H., Rangwala, H., Karypis, G., Wang, Y., and Bohlke-Schneider, M. (2025) Chronos-2: From univariate to universal forecasting, *arXiv preprint arXiv:2510.15821*.
- Bach, G. L. (1975) The economic effects of inflation, *Proceedings of the Academy of Political Science*, **31**, 20–33.
- Bank of England (2025) Monetary Policy Report, November 2025, Technical report, Bank of England, London, United Kingdom, <https://www.bankofengland.co.uk/monetary-policy-report/2025/november-2025>.
- Box, G. E. P., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2016) *Time Series Analysis: Forecasting and Control*, Wiley, Hoboken, New Jersey, USA, 5th edition.
- Brockwell, P. J. and Davis, R. A. (2009) *Time Series: Theory and Methods*, Springer, 2nd edition.
- Cao, D., Wang, Y., Duan, J., Zhang, C., Zhu, X., Huang, C., Tong, Y., Xu, B., Bai, J., Tong, J., and Zhang, Q. (2021) Spectral temporal graph neural network for multivariate time-series forecasting, *arXiv preprint: 2103.07719*.
- Chen, S., Wang, J., and Li, G. (2021) Neural relational inference with efficient message passing mechanisms, in *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Ciccarelli, M., Darracq Paries, M., Priftis, R., Angelini, E., Banbura, M., Bokan, N., Rigato, R. D., Fagan, G., Gumiel, J. E., Kase, H., Kornprobst, A., Koutsoulis, I., Lalik, M., Montes-Galdón, C., Müller, G., Paredes, J., Santoro, S., Warne, A., Zimic, S., Brunotte, S., Cocchi, S., Giammaria, A., Invernizzi, M., and Von-Pine, E. (2024) ECB macroeconomic models for forecasting and policy analysis, Technical Report 344, European Central Bank.
- Coenen, G., Karadi, P., Schmidt, S., and Warne, A. (2018) The New Area-Wide Model II: an extended version of the ECB’s micro-founded model for forecasting and policy analysis with a financial sector, Working Paper 2200, European Central Bank, Frankfurt am Main, Germany.
- Dalgaard, P. (2008) *Introductory Statistics with R*, Springer, New York.
- Davis, R., Zang, P., and Zheng, T. (2016) Sparse vector autoregressive modeling, *Journal of Computational and Graphical Statistics*, **25**, 1077–1096.
- Gooijer, J. G. D. and Hyndman, R. J. (2006) 25 years of time series forecasting, *International Journal of Forecasting*, **22**, 443–473.
- Hamilton, J. (1994) *Time Series Analysis*, Princeton University Press, Princeton, NJ, USA.
- Hochreiter, S. and Schmidhuber, J. (1997) Long short-term memory, *Neural Computation*, **9**, 1735–1780.
- Holt, C. C. (1957) Forecasting seasonals and trends by exponentially weighted averages, Technical Report 52, Carnegie Institute of Technology, ONR Research Memorandum.

- Holt, C. C. (2004) Forecasting seasonals and trends by exponentially weighted averages, *International Journal of Forecasting*, **20**, 5–10.
- Hyndman, R. and Athanasopoulos, G. (2021) *Forecasting: principles and practice*, OTexts, Melbourne, Australia, 3rd edition, [OTexts.com/fpp3](https://otexts.com/fpp3).
- Independent Evaluation Office (2024) Forecasting for monetary policy making and communication at the Bank of England: A review, Technical report, Bank of England, <https://www.bankofengland.co.uk/-/media/boe/files/independent-evaluation-office/2024/forecasting-for-monetary-policy-making-and-communication-at-the-bank-of-england-a-review.pdf>.
- Joseph, A., Potjagailo, G., Kalamara, E., Chakraborty, C., and Kapetanios, G. (2021) Forecasting UK inflation bottom up, Staff Working Paper 915, Bank of England.
- Joseph, A., Potjagailo, G., Chakraborty, C., and Kapetanios, G. (2024) Forecasting UK inflation bottom up, *International Journal of Forecasting*, **40**, 1521–1538.
- Kalman, R. E. (1960) A new approach to linear filtering and prediction problems, *Journal of Basic Engineering*, **82**, 35–45.
- Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., and Zemel, R. (2018) Neural relational inference for interacting systems, *arXiv preprint: 1802.04687*.
- Knight, M., Nason, G. P., and Nunes, M. (2016) Modelling, detrending and decorrelation of network time series, *arXiv preprint: 1603.03221*.
- Knight, M., Leeming, K., Nason, G. P., and Nunes, M. (2020) Generalized Network Autoregressive Processes and the GNAR package, *Journal of Statistical Software*, **96**.
- Leeming, K. (2019) *New Methods in Time Series Analysis: Univariate Testing and Network Autoregression Modelling*, Ph.D. thesis, University of Bristol.
- Lütkepohl, H. (2005) *New Introduction to Multiple Time Series Analysis*, Springer.
- Mallick, S. K. and Mohsin, M. (2010) On the real effects of inflation in open economies: theory and empirics, *Empirical Economics*, **39**, 643–673.
- Mann, H. B. and Wald, A. (1943) On the statistical treatment of linear stochastic difference equations, *Econometrica*, **11**, 173–220.
- Marcellino, M., H. Stock, J., and Watson, M. W. (2006) A comparison of direct and iterated multistep ar methods for forecasting macroeconomic time series, *Journal of Econometrics*, **135**, 499–526.
- McCarthy, H. (2024) *Smarter Networks, Sharper Forecasts; Improving GNAR Prediction Accuracy by Network Search*, Master’s thesis, Imperial College London.
- Nason, G. P. and Palasciano, H. A. (2026) Forecasting UK consumer price inflation with RaGNAR: Random generalised network autoregressive processes, *International Journal of Forecasting*, **42**, 181–202.

- Nason, G. P., Salnikov, D., and Cortina-Borja, M. (2023) New tools for network time series with an application to COVID-19 hospitalisations, *arXiv:2312.00530 [stat]*.
- Nason, G. P., Salnikov, D., and Cortina-Borja, M. (2026) New tools for network time series with an application to COVID-19 hospitalisations, *Journal of the Royal Statistical Society, Series A*, **189**, 139–163.
- Office for National Statistics (2017) Consumer price indices: a brief guide, <https://www.ons.gov.uk/economy/inflationandpriceindices/articles/consumerpriceindicesabriefguide/2017>.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019) Pytorch: An imperative style, high-performance deep learning library, in H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché Buc, E. Fox, and R. Garnett, eds., *Proceedings of the 33rd International Conference on Neural Information Proceedings*, pp. 8026–8037, Curran Associates, Inc.
- Quenouille, M. H. (1957) *The Analysis of Multiple Time Series*, Griffin.
- Satorras, V. G., Rangapuram, S. S., and Januschowski, T. (2022) Multivariate time series forecasting with latent graph inference, *arXiv preprint: 2203.03423*.
- Sims, C. A. (1980) Macroeconomics and reality, *Econometrica*, **48**, 1–48.
- Smith, T. G. et al. (2017–) pmdarima: Arima estimators for Python.
- Tanese, R. (1989) Distributed genetic algorithms, in J. D. Schaffer, ed., *Proceedings of the 3rd International Conference on Genetic Algorithms*, pp. 434–439, Morgan Kaufman Publishers, San Francisco, USA.
- Vose, M. D. (1999) *The Simple Genetic Algorithm: Foundations and Theory*, The MIT Press, Cambridge, MA, USA.
- Winters, P. R. (1960) Forecasting sales by exponentially weighted moving averages, *Management Science*, **6**, 324–342.
- Yu, X., Tang, K., Chen, T., and Yao, X. (2009) Empirical analysis of evolutionary algorithms with immigrants schemes for dynamic optimization, *Memetic Computing*, **1**, 3–24.

A Macroeconomic forecasting in policy-making

Institutional macroeconomic forecasting. Macroeconomic forecasting lies at the core of many economic public policies. All relevant economic institutions, from the Treasury to the Bank of England, passing through international institutions like the International Monetary Fund, have groups of experts whose job is to try to accurately predict the evolution of key macroeconomic aggregates. In a recent report by the European Central Bank (ECB) (Ciccarelli et al., 2024), the modelling used by institutions is classified into three main types of modelling:

- (i) **Dynamic Stochastic General Equilibrium (DSGE) models.** These correspond to the classical structural models implemented by economists. There is a great variety of DSGE models, but in essence they stipulate a set of equations which relate macroeconomic aggregates and explain how agents (i.e. households, firms and the government) interact with each other. The equations are derived from micro-fundamentals (e.g. maximisation of utility by households and firms) as well as macroeconomic theory (e.g. variants of Phillips curve). Hence, the model is generally given by a system of non-linear stochastic equations of the form

$$\mathbb{E}[f(x_{t+1}, x_t, x_{t-1}, \varepsilon_t; \theta)] = 0,$$

where x_t are endogenous variables like inflation, ε_t are structural shocks and θ are structural parameters to be estimated. In many instances, a state-space form is required for estimation. That is, the model equations are solved to form a state-space system of the form

$$\begin{aligned} z_t &= A(\theta)z_{t-1} + B(\theta)\varepsilon_t \\ y_t &= C(\theta)z_t + D(\theta)\eta_t, \end{aligned}$$

where z_t are latent variables like technology, and y_t are observed variables like inflation. Then, a Kalman filter is used to infer the unobserved data z_t and construct the likelihood. Finally, Bayesian methods are typically employed to produce forecasts and evaluate the effect of possible monetary or fiscal rules. A typical example of such a model is the ECB's New Area-Wide Model and its variants (Coenen et al., 2018).

- (ii) **Statistical/empirical models.** These models lie at the opposite end of the spectrum of models used by institutions. They are data-driven and very minimal economic theories goes into their specification. Here, VAR models and their variants are predominant.
- (iii) **Semi-structural models.** These lie halfway through the DSGE models and the time-series or empirical models. They are still constructed like DSGE models and estimated using the state-space formulation and Bayesian methods. However, in DSGE models, the parameters to be estimated inside the state-space matrices are heavily constrained based on economic principles. This is not the case in semi-structural models, where parameters are freely estimated.

Inflation policy in the UK. Inflation describes the process by which the prices of goods and services change over a period of time. Inflation is of central importance to policymakers, as it has an effect on savings, consumption, wages and investment. In fact, since as early as 1992 the Bank of England (BoE) adopted explicit inflation tracking, CPI being the primary index which has shaped the monetary policy of the UK since then. In the UK, the Bank of England (BoE) is in charge of monetary policy, which is the main policy tool to control a country or region's inflation. The control of official interest rates is in turn the main tool employed by the BoE to influence the UK's inflation. To control inflation well, a primary job of the BoE is to accurately forecast inflation.

Inside the BoE, the Monetary Policy Committee (MPC) is a group of experts who, with the advice of the BoE's staff, determines the monetary policy, which should always support the mission of maintaining the Consumer Price Index (CPI) around 2% (Bank of

England, 2025). Every quarter, the MPC releases a report where they set out the economic analysis and inflation projections produced by the BoE, which then informs their decision on the official interest rates. The MPC uses all of the modelling devices described above to elaborate the aforementioned quarterly report.

B CPI variable names

Here we briefly enumerate all of the variables used in the dataset. They are organised by the 12 divisions mentioned in Section 2.

CPI node.

Code	CPI category
00	All items

Food and non-alcoholic beverages.

Code	CPI category
01.1.1	Bread & cereals
01.1.2	Meat
01.1.3	Fish
01.1.4	Milk, cheese & eggs
01.1.5	Oils & fats
01.1.6	Fruit
01.1.7	Vegetables including potatoes and other tubers
01.1.8	Sugar, jam, honey, syrups, chocolate & confectionery
01.1.9	Food products
01.2.1	Coffee, tea, cocoa
01.2.2	Mineral waters, soft drinks and juices

Alcoholic beverages, tobacco & narcotics.

Code	CPI category
02.1.1	Spirits
02.1.2	Wine
02.1.3	Beer

Clothing and footwear.

Code	CPI category
03.1.2	Garments
03.1.3	Other articles of clothing & accessories
03.1.4	Cleaning, repair and hire of clothing

Housing, water and fuels.

Code	CPI category
04.3.1	Materials for maintenance and repair
04.3.2	Services for maintenance and repair
04.5.3	Liquid fuels
04.5.4	Solid fuels

Furniture, household equipment & routine repair of house.

Code	CPI category
05.1.1	Furniture and furnishings
05.1.2	Carpets & other floor coverings
05.3.1/2	Major appliances and small electric goods
05.3.3	Repair of household appliances
05.6.1	Non-durable household goods
05.6.2	Domestic services and household services

Health.

Code	CPI category
06.1.1	Pharmaceutical products
06.1.2/3	Other medical and therapeutic equipment
06.2.1/3	Medical services and paramedical services
06.2.2	Dental services

Transport.

Code	CPI category
07.1.1A	New cars
07.1.1B	Second-hand cars
07.1.2/3	Motor cycles and bicycles
07.2.1	Spare parts & accessories
07.2.2	Fuels & lubricants
07.2.3	Maintenance & repairs
07.2.4	Other services
07.3.1	Passenger transport by railway
07.3.2/6	Passenger transport by road and other transport services
07.3.3	Passenger transport by air
07.3.4	Passenger transport by sea and inland waterway

Recreation & culture.

Code	CPI category
09.1.1	Reception and reproduction of sound and pictures
09.1.2	Photographic, cinematographic & optical equipment
09.1.3	Data processing equipment
09.1.4	Recording media
09.1.5	Repair of audio-visual equipment & related products
09.3.1	Games toys and hobbies
09.3.2	Equipment for sport camping and open-air recreation
09.3.3	Garden plants and flowers
09.3.4/5	Pets, related products and services
09.4.1	Recreational and sporting services
09.4.2	Cultural services
09.5.1	Books
09.5.2	Newspapers and periodicals
09.5.3/4	Misc. printed matter stationery & drawing materials

Hotels, cafes and restaurants.

Code	CPI category
11.1.1	Restaurants & cafes
11.1.2	Canteens

Miscellaneous goods and services.

Code	CPI category
12.1.1	Hairdressing and personal grooming establishments
12.1.2/3	Appliances, articles & products for personal care
12.3.1	Jewellery clocks and watches
12.3.2	Other personal effects
12.5.2	House contents insurance
12.5.3/5	Health insurance and other insurance
12.5.4	Transport insurance

C Limitations of McCarthy (2024)

McCarthy (2024) conducted preliminary investigations into the use of a genetic algorithm for GNAR graph selection. McCarthy used a genetic algorithm based on modifying minimum spanning trees for GNAR graphs and applied these models to OECD unemployment data. However, although pioneering, McCarthy (2024) was subject to certain limitations, some of which are listed below:

- (i) The work did not make use of separate validation and testing datasets. Hence, it used a single “testing” window to select the best graph and report out-of-sample performance, which means that, in practice, it could not rigorously conclude that the genetic algorithm provided graphs which led to improved out-of-sample prediction accuracy.
- (ii) The approach was limited to non-ensemble, single graph GNAR models. The graph and GNAR hyperparameter level ensembling in Nason and Palasciano (2026) was a significant contribution and we leverage it here.

- (iii) The work’s graph exploration was limited to minimum spanning trees with 26 edges, which was hard-wired to a specific narrow problem of study. A more comprehensive study, as here, would enable complete free choice of edges in the network.
- (iv) The work used the overall RMSE fitness function. We followed the approach of Nason and Palasciano (2026) and use CPI-only RMSE which yields better results and is more computationally efficient. However, Nason and Palasciano (2026) used 1-step ahead validation forecasting RMSE as a fitness function, whereas we aligned the type of h -step ahead validation forecasting RMSE with the downstream forecasting task.

D Details of the Genetic Algorithm

This appendix provides pseudocode for the crossover and the mutation steps of the genetic algorithm presented in Section 3. Here $\text{ErdosRenyi}(p)$ denotes an Erdős-Renyi probability distribution with edge probability p over the space of $N \times N$ matrices. Furthermore, the operator *triu* turns a matrix into an upper triangular matrix by zeroing all elements below the main diagonal. This is in line with the notation from the PyTorch package Paszke et al. (2019) used in this article’s repository.

D.1 Crossover

Algorithm 3: Crossover

Input: $G_1, G_2 \in \mathbb{R}^{N \times N}$
Output: G_{new}

- 1 $M \sim \text{ErdosRenyi}(0.5)$ (upper triangular)
- 2 $G_{\text{new}} \leftarrow G_1$
- 3 $G_{\text{new}}[M] \leftarrow G_2[M]$ (on masked entries change G_1 for G_2)
- 4 $G_{\text{new}} \leftarrow \text{triu}(G_{\text{new}})$ (make upper triangular)
- 5 $G_{\text{new}} \leftarrow G_{\text{new}} + G_{\text{new}}^\top$
- 6 **return** G_{new}

D.2 Mutation

Algorithm 4: Mutation

Input: $G \in \mathbb{R}^{N \times N}$
Output: G_{new}

- 1 $M \sim \text{ErdosRenyi}(p)$ (upper triangular)
- 2 $G_{\text{new}} \leftarrow G$
- 3 $G_{\text{new}}[M] \leftarrow 1 - G_{\text{new}}[M]$ (on masked entries change sign $0 \leftrightarrow 1$)
- 4 $G_{\text{new}} \leftarrow \text{triu}(G_{\text{new}})$ (make upper triangular)
- 5 $G_{\text{new}} \leftarrow G_{\text{new}} + G_{\text{new}}^\top$
- 6 **return** G_{new}

E Speed-up of validation error computation

Recall that for a specific graph G , at a time origin t , we need to compute the following validation RMSE as a measure of fitness for G :

$$\text{RMSE}_{\text{val},t}(G) := \left\{ W_{\text{val}}^{-1} \sum_{\tau=t-W_{\text{val}}-h}^{t-h} \left(X_{\tau+h} - \hat{X}_{\tau+h|\tau}(G) \right)^2 \right\}^{\frac{1}{2}}. \quad (15)$$

This means that W_{val} different GNAR model have to be fitted to make the predictions $\hat{X}_{\tau+h|\tau}(G)$. Let

$$Z_{\tau} \in \mathbb{R}^{K \times M}, \quad y_{\tau} \in \mathbb{R}^K,$$

denote the GNAR design block and target block a time step τ . If we want the OLS coefficients using all data up to prefix τ , we solve the optimisation problem presented in Equation 8. The naive approach forms

$$\tilde{Z}_{\tau} = \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_{\tau} \end{bmatrix} \in \mathbb{R}^{(sN) \times M}, \quad \tilde{y}_{\tau} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{\tau} \end{bmatrix} \in \mathbb{R}^{sN},$$

and solves

$$\hat{\gamma}_{\tau} = \arg \min_{\gamma \in \mathbb{R}^M} \left\| \tilde{y}_{\tau} - \tilde{Z}_{\tau} \gamma \right\|_2^2 \quad (16)$$

Then uses this to make h -step ahead predictions. However, this recomputes many of the previously calculated matrix multiplications. Indeed, solving Equation 16 is equivalent to solving the normal equations:

$$\begin{aligned} \tilde{Z}_{\tau}^{\top} \tilde{Z}_{\tau} \hat{\gamma}_{\tau} - \tilde{Z}_{\tau}^{\top} \tilde{y}_{\tau} &= \left(\sum_{s=1}^{\tau} Z_s^{\top} Z_s \right) \hat{\gamma}_{\tau} - \sum_{s=1}^{\tau} Z_s^{\top} y_s \\ &:= G_{\tau} \hat{\gamma}_{\tau} - b_{\tau}. \end{aligned}$$

This means that to solve the $\tau + 1$ equation, we only need to add an extra term to G_{τ} and b_{τ} , i.e.,

$$G_{\tau+1} = G_{\tau} + Z_{\tau+1}^{\top} Z_{\tau+1}, \quad b_{\tau+1} = b_{\tau} + Z_{\tau+1}^{\top} y_{\tau+1}.$$

This leads to a faster algorithm to compute the validation RMSE because at each step you efficiently compute the updates $G_{\tau+1}$ and $b_{\tau+1}$. A further direction to explore is whether using an iterative solver which avoids having to solve the normal equations at each time step (e.g. updating the Cholesky factorisation) improves the speed of the overall algorithm.

F Additional Results

F.1 Results for local models

Local α . Replication of Table 1, but with a local α specification.

Local α, β . Replication of Table 1, but with a local α, β specification.

p		Forecasting Horizon, h					
		1	2	3	4	5	6
1,2	Random	0.519±0.009	0.867±0.018	0.970±0.031	1.084±0.093	1.256±0.120	1.521±0.171
	Genetic	0.503±0.019	0.853±0.036	0.779±0.028	0.745±0.030	0.832±0.057	0.886±0.062
	Improvement	3.1%	1.6%	19.7%	31.2%	33.8%	41.7%
1,2,12	Random	0.601±0.013	0.853±0.021	0.966±0.017	1.005±0.077	1.098±0.116	1.275±0.163
	Genetic	0.550±0.042	0.806±0.042	0.736±0.034	0.740±0.036	0.782±0.035	0.772±0.031
	Improvement	8.4%	5.5%	23.8%	26.3%	28.8%	39.5%
1,2,12,24	Random	0.532±0.018	0.850±0.020	0.956±0.030	1.009±0.069	1.141±0.084	1.334±0.136
	Genetic	0.528±0.044	0.808±0.043	0.764±0.041	0.758±0.050	0.804±0.032	0.820±0.034
	Improvement	0.7%	4.9%	20.1%	24.8%	29.5%	38.5%

Table 7: Results for different GNAR models with a local α specification. Ensembling of best 5 graphs has been used in each algorithm.

p		Forecasting Horizon, h					
		1	2	3	4	5	6
1,2	Random	0.604±0.005	0.949±0.014	1.196±0.015	1.465±0.017	1.796±0.030	2.171±0.046
	Genetic	0.564±0.009	0.774±0.019	0.817±0.011	0.919±0.017	1.084±0.023	1.378±0.028
	Improvement	6.7%	18.5%	31.7%	37.2%	39.6%	36.5%
1,2,12	Random	0.570±0.006	0.863±0.016	1.043±0.021	1.254±0.010	1.478±0.031	1.733±0.028
	Genetic	0.556±0.014	0.765±0.022	0.763±0.038	0.847±0.036	0.883±0.018	0.983±0.038
	Improvement	2.5%	11.3%	26.9%	32.4%	40.2%	43.3%
1,2,12,24	Random	0.556±0.006	0.841±0.013	1.015±0.016	1.210±0.016	1.425±0.020	1.690±0.042
	Genetic	0.539±0.012	0.755±0.029	0.772±0.031	0.877±0.037	0.927±0.037	1.044±0.061
	Improvement	2.9%	10.2%	23.9%	27.5%	34.9%	38.2%

Table 8: Results for different GNAR models with a local α, β specification. Ensembling of best 5 graphs has been used in each algorithm.

F.2 Results with full dataset

This subsection looks at experiment runs using the whole CPI dataset. The results presented in Table 9 suggested that using the full dataset does not improve the forecasting accuracy of the model, and it comes at a computational costs since it increases the size of all of the design matrices in the GNAR model. This led us to use the reduced CPI dataset described in Section 2 to build our models. In the full dataset experiments, we continue to see the two conclusions which we draw from the paper: GenGNAR consistently outperforms RaGNAR, and in turn both outperform the Bank of England forecasts at medium to large forecasting horizons.

G Mann-Whitney U Test

The U -statistics and one-tailed p -values for each experimental configuration used for Table 1 is calculated and presented below. From Table 10, it is clear that the genetic algorithm outperforms the random algorithm across configurations and time horizons, with the exception of $h = 2$. Here, for $p = [1, 2]$ and $p = [1, 2, 12, 24]$, the difference in the performances of the random and genetic search algorithms are not statistically significant at any reasonable significance level. Similarly, the U -statistics and the respective p -values for

		Forecasting Horizon					
		1	2	3	4	5	6
Reduced Dataset	Random Genetic	0.601 $\pm$ 0.013	0.853 $\pm$ 0.021	0.966 $\pm$ 0.017	1.005 $\pm$ 0.077	1.098 $\pm$ 0.116	1.275 $\pm$ 0.163
		0.550 $\pm$ 0.042	0.806 $\pm$ 0.042	0.736 $\pm$ 0.034	0.740 $\pm$ 0.036	0.782 $\pm$ 0.035	0.772 $\pm$ 0.031
Full Dataset	Random Genetic	0.599 $\pm$ 0.016	0.849 $\pm$ 0.011	0.949 $\pm$ 0.015	1.047 $\pm$ 0.043	1.227 $\pm$ 0.056	1.450 $\pm$ 0.066
		0.517 $\pm$ 0.030	0.812 $\pm$ 0.015	0.819 $\pm$ 0.032	0.764 $\pm$ 0.037	0.820 $\pm$ 0.059	0.884 $\pm$ 0.047

Table 9: Results comparing local α GNAR(1,2,12,[1]) models with the reduced and full datasets as inputs.

		Forecasting Horizon, h					
p	Statistic	1	2	3	4	5	6
[1, 2]	U -statistic	0.070	0.598	0.000	0.000	0.000	0.000
	p -value	0.000	0.856	0.000	0.000	0.000	0.000
[1, 2, 12]	U -statistic	0.107	0.318	0.000	0.000	0.000	0.000
	p -value	0.000	0.025	0.000	0.000	0.000	0.000
[1, 2, 12, 24]	U -statistic	0.233	0.412	0.000	0.000	0.000	0.000
	p -value	0.002	0.177	0.000	0.000	0.000	0.000

Table 10: Normalised U -statistics and corresponding p -values for the Mann–Whitney U -test to quantify the statistical significance of the improvement of GenGNAR versus RaGNAR for the experiments presented in Table 1.

the AvGNAR experiment in Table 2 are presented below. Here, GenGNAR beats RaGNAR at any significance level.

		Forecasting Horizon, h					
		1	2	3	4	5	6
U -statistic		0.00	0.00	0.00	0.00	0.00	0.00
p -value		0.00	0.00	0.00	0.00	0.00	0.00

Table 11: Normalised U -statistics and corresponding p -values for the Mann–Whitney U -test to quantify the statistical significance of the improvement of GenGNAR versus RaGNAR for the experiments presented in Table 2.