Exploring Gaussian-Laplace Combinations in High-Dimensional Dynamic Spike-and-Slab Models

Guilherme C. Soares[1], Marcio P. Laurini[2]

Abstract

This study explores the role of prior specifications in dynamic variable selection models under a Spike-and-Slab framework. Gaussian and Laplace priors are employed to investigate the impact of penalty functions and their effectiveness in inducing dynamic sparsity. The analysis builds upon the framework discussed by Ročková and McAlinn (2019), addressing the dynamic selection problem through the lens of optimization using a dynamic Expectation-Maximization (EM) algorithm. The methods are analyzed in terms of their performance on simulated samples and in an application to real data forecasting.

Keywords: Optimization; Spike and Slab; Variable Selection, Sparsity.

1. Introduction

With the increasing availability of data, access to information for predictive purposes has advanced rapidly, making it significantly easier to incorporate a wide range of information into predictive models compared to the past. Combined with the growth in computational efficiency, this raises a crucial question: *"How do we handle these data?"*

From a macroeconomic perspective, this question becomes even more complex. Macroeconomic datasets often exhibit a limited number of time periods relative to the number of available variables, leading to estimation challenges inherent in scenarios where $N \ll P$, that is, when the sample size N is smaller than the number of parameters in the model.

Pioneering works such as Tibshirani (1996) and George and McCulloch (1993) introduced methods such as Lasso, Ridge, Elastic Net and spike and slab for variable selection, effectively eliminating less relevant variables from the model and estimating it without them. However, these are static approaches that remove variables without considering their potential temporal relevance.

Stock and Watson (2007) examine U.S. inflation forecasting, demonstrating how predictive dynamics evolve over time. Models that perform well in certain periods may

1. University of São Paulo - FEARP
E-mail: guilhermecsoares@usp.br, laurini@fearp.usp.br

become less effective in others due to shifts in the economic environment, such as transitions between stability and instability, the persistence of shocks, reductions in overall macroeconomic volatility, and changes in trend components. Similarly, Cogley and Sargent (2005) explore time-varying parameters, proposing models in which the coefficients of VAR equations and stochastic volatilities evolve over time. They argue that traditional fixed-parameter models fail to adequately capture economic dynamics across different periods.

However, when dealing with a large number of variables, the number of parameters to estimate grows substantially, complicating the model's implementation. To address this challenge, a growing body of literature on Dynamic Variable Selection has emerged, as explored by Ročková and George (2014), Ročková and McAlinn (2019), and Koop and Korobilis (2023). These approaches introduce specifications that allow for dynamic coefficients while inducing dynamic sparsity, both vertically, where a parameter can "enter and exit" the model over time, and horizontally, where a parameter is shrunk to zero across all periods.

This paper focuses on a model based on Spike-and-Slab priors, which operate as a mixture of two distributions: the spike, which shrinks parameters to zero to induce sparsity, and the slab, which permits parameters to deviate from zero and exhibit temporal variation. The spike-and-slab framework is typically implemented using Gaussian or Laplacian priors, each influencing the shrinkage behavior in distinct ways.

Our discussion is grounded in the work of Ročková and McAlinn (2019), who introduced the EMVS (Expectation-Maximization Variable Selection) algorithm for fitting Dynamic Spike-and-Slab (DSS) priors. The primary objective of this approach is to achieve Maximum a Posteriori (MAP) smoothing. This algorithm is particularly advantageous due to its computational efficiency in handling high-dimensional datasets, where traditional MCMC (Markov Chain Monte Carlo) methods may be computationally prohibitive, and its ease of implementation in Gaussian models, where conjugate priors facilitate tractability.

Additionally, we explore the implications of employing Laplace priors, which introduce unique challenges. These include non-differentiability at zero and a higher density near zero compared to Gaussian priors, both of which significantly influence the shrinkage properties and sparsity-inducing behavior of the model.

The results in the simulated and empirical analysis demonstrate that models incorporating shrinkage mechanisms, particularly the Gaussian-Gaussian (GG) and Laplace-Gaussian (LG) models, significantly enhance forecasting accuracy across different horizons. The GG model performs best for short-term forecasts, while the LG model is more robust over longer horizons. Fixed parameter calibration across all horizons may limit performance, suggesting the need for horizon-specific adjustments. Cumulative error analysis further supports the superiority of these models, especially in handling noise and sparsity. The study highlights the practical advantages of shrinkage in macroeconomic forecasting and suggests future research on adaptive calibrations and broader applications.

The remainder of this paper is structured as follows. Section 2 presents the formal specification of the model with Spike-and-Slab priors. Section 3 explores the influence of Normal and Laplace distributions on the loss function. Section 4 applies the proposed model to parameter recovery in a simulated dataset, while Section 5 evaluates its performance in forecasting Brazilian inflation using the discussed methods. Finally, Section 6 presents the concluding remarks.

2. Variable Selection Structure

The spike-and-slab prior (Ročková and George, 2018; Louzada et al., 2023) is a Bayesian variable selection method designed to induce sparsity in high-dimensional models. It is formulated as a mixture of two distributions: a *spike*, which shrinks coefficients toward zero, and a *slab*, which allows nonzero coefficients to vary freely.

Let β_j be a model parameter. The spike-and-slab prior is typically defined as a hierarchical mixture model:

$$\beta_j \mid \gamma_j \sim (1-\gamma_j) \cdot \text{Spike} + \gamma_j \cdot \text{Slab}$$

where $\gamma_j \in \{0,1\}$ is a latent indicator variable controlling whether β_j is included ($\gamma_j = 1$) or excluded ($\gamma_j = 0$). The spike component is often a narrow Gaussian distribution, such as $\mathcal{N}(0,\tau^2)$ with small τ^2, or a point mass at zero. The slab component is typically a broader Gaussian distribution, $\mathcal{N}(0,\sigma^2)$, or a Laplacian prior, which provides stronger shrinkage.

The primary advantage of the spike-and-slab prior lies in its ability to perform automatic variable selection by shrinking irrelevant coefficients to zero. Additionally, modern estimation techniques, such as the Expectation-Maximization Variable Selection (EMVS) algorithm, enhance computational efficiency by avoiding costly Markov Chain Monte Carlo (MCMC) sampling, making it well-suited for high-dimensional applications.

Let us consider a model that follows a Dynamic Spike and Slab structure such that:

$$y_t = \beta_{t,j} x_{t-h} + \varepsilon_t, \tag{1}$$

where

$$\begin{aligned}
\varepsilon_t &\sim N(0,\sigma), \\
\beta_{t,j} &\sim N(\gamma_{t,j}\mu_{t,j}, \gamma_t\lambda_1 + (1-\gamma_t)\lambda_0), \\
\mu_{t,j} &= \phi_0 + \phi\mu_{t-1,j} + \nu, \quad \nu \sim N(0,\sigma_\mu), \\
\gamma_{t,j} &= \begin{cases} 1 & \text{if spike,} \\ 0 & \text{if slab.} \end{cases}
\end{aligned}$$

Here, λ_1 and λ_0 represent the variances of the Dynamic Spike-and-Slab prior, respectively.

The choice of ϕ is made such that $|\phi| < 1$, ensuring stationarity of the parameters. This allows for a structure such that μ_t becomes a stationary process with mean ϕ_0 and variance $\frac{\lambda_1}{1-\phi^2}$.

The inference procedure used in this work is based on estimation using Maximum a Posteriori estimation. Maximum a Posteriori (MAP) estimation is a Bayesian approach to parameter estimation. It finds the parameter θ that maximizes the posterior probability $P(\theta \mid X)$, given the observed data X. Mathematically, MAP estimation is given by:

$$\hat{\theta}_{\text{MAP}} = \arg\max_{\theta} P(\theta \mid X)$$

Using Bayes'theorem:

$$P(\theta \mid X) = \frac{P(X \mid \theta)P(\theta)}{P(X)}$$

Since $P(X)$ is a constant with respect to θ, MAP estimation simplifies to:

$$\hat{\theta}_{\text{MAP}} = \arg\max_{\theta} P(X \mid \theta)P(\theta)$$

where $P(X \mid \theta)$ is the likelihood of the data given the parameter and $P(\theta)$ is the prior distribution of the parameter.

MAP estimation incorporates prior knowledge, which helps in regularization and prevents overfitting. Unlike Maximum Likelihood Estimation (MLE), which only considers the likelihood function, MAP includes prior information, making it more robust, particularly in small-sample settings. The prior acts as a form of regularization, ensuring that the estimates remain stable. This makes MAP especially useful when data is limited or when MLE produces ill-posed solutions.

2.1 Dynamic Spike-and-Slab structure of $\beta_{t,j}$

To derive the MAP (Maximum A Posteriori) estimator for $\beta_{1:T,j}$, we maximize the posterior distribution:

$$\hat{\beta}_{1:T,j} = \arg\max_{\beta_{1:T,j}} \pi(\beta_{1:T,j}|Y_{1:T}), \tag{2}$$

where $\pi(\beta_{1:T,j}|Y_{1:T})$ is proportional to the product of the likelihood and the prior:

$$\pi(\beta_{1:T,j}|Y_{1:T}) \propto L(Y_{1:T}|\beta_{1:T,j}) \cdot \pi(\beta_{1:T,j}). \tag{3}$$

The prior for $\beta_{t,j}$ is defined as a mixture of the spike and slab distributions:

$$\pi(\beta_{t,j}|\gamma_{t,j}, \beta_{t-1,j}) = (1-\gamma_{t,j})\psi_0(\beta_{t,j}|\lambda_0) + \gamma_t\psi_1(\beta_{t,j}|\mu_t, \lambda_1), \tag{4}$$

where:

- $\psi_0(\beta_t|\lambda_0)$ is the spike distribution with variance λ_0,
- $\psi_1(\beta_t|\mu_t, \lambda_1)$ is the slab distribution with mean μ_t and variance λ_1,
- $\gamma_t \in \{0, 1\}$ is the binary indicator for spike/slab membership.

For now, we assume arbitrary distributions for ψ_0 and ψ_1. In later sections, we will discuss the implications of using Gaussian or Laplace distributions for each case. To work with the γ_{t_j} inclusion parameter, it's necessary to introduce its structure:

$$P(\gamma_t = 1 \mid \beta_{t-1}) = \theta_t. \tag{5}$$

The specification given by Ročková and McAlinn (2019) for θ_t involves introducing an interpretable parameter called the marginal importance weight Θ, where $0 \leq \Theta \leq 1$. This scalar parameter controls the overall balance between the spike and slab distributions.

Given $(\Theta, \lambda_0, \lambda_1, \phi_0, \phi_1)$, the conditional inclusion probability θ_t (or a transition function $\theta(\beta_{t-1})$) is defined as:

$$\theta_t \equiv \theta(\beta_{t-1}) = \frac{\Theta\psi_1^{ST}(\beta_{t-1}|\lambda_1, \phi_0, \phi_1)}{\Theta\psi_1^{ST}(\beta_{t-1}|\lambda_1, \phi_0, \phi_1) + (1-\Theta)\psi_0(\beta_{t-1}|\lambda_0)}. \tag{6}$$

It is interesting to note that changing the value of Θ to 0 or 1 will make the model tend toward two extremes: a dynamic linear model (DLM) without shrinkage when $\Theta = 1$, or a model composed entirely of the spike distributions when $\Theta = 0$, meaning all parameters will be shrunk to zero.

2.2 Algorithm Description

The algorithm iteratively updates the regression coefficients $\beta_{j,t}$ by calculating posterior inclusion probabilities and performing dynamic shrinkage adjustments. Below, we present the main steps of the algorithm, which incorporates spike-and-slab priors defined by the functions ψ_0 and ψ_1 for the spike and slab components, respectively.

Before starting the iterative process, the algorithm defines several auxiliary functions to compute posterior probabilities and shrinkage adjustments. These functions rely on the spike-and-slab prior distributions represented by $\psi_0(\beta_t \mid \lambda_0)$ for the spike component and $\psi_1(\beta_t \mid \lambda_1, \mu_t)$ for the slab component.

2.2.1 Posterior Inclusion Probability (p^*)

The posterior inclusion probability, denoted as $p_j^*(x)$, measures the likelihood that a variable is included in the model at time t. It is computed as follows:

$$p_j^*(x) = \frac{1}{1 + \frac{(1-\Theta)\psi_0(x|\lambda_0)}{\Theta\psi_1(x|\lambda_1,\mu_t)}} \tag{7}$$

This equation balances the relative contributions of the spike and slab components, weighted by the prior inclusion probability Θ.

2.2.2 Conditional Inclusion Probability

The conditional inclusion probability, denoted as $p_j^{\text{cond}}(x)$, incorporates information from past values of the regression coefficients and is computed as:

$$p_j^{\text{cond}}(x) = \frac{\theta_j \psi_1(x \mid \lambda_1, \mu_t)}{\theta_j \psi_1(x \mid \lambda_1, \mu_t) + (1-\theta_j)\psi_0(x \mid \lambda_0)} \tag{8}$$

This function dynamically adjusts the inclusion probability based on the past evolution of the model's coefficients.

2.2.3 Expectation Step (E-Step)

During the Expectation step, the algorithm computes the posterior inclusion probabilities for each variable j in each time period t. The main tasks in this step are as follows:

1. Compute the mixing weights θ_j and the posterior inclusion probabilities p_j^* using the auxiliary functions defined above.

2. Compute the precision terms and auxiliary quantities ν_t and d_t for the posterior distributions of the coefficients, which are used in subsequent updates.

2.2.4 Maximization Step (M-Step)

In the Maximization (M) step, the algorithm updates the regression coefficients $\beta_{j,t}$ and intercepts $\beta_{0,j}$ using the posterior probabilities computed in the E-step.

Updating Regression Coefficients ($\beta_{j,t}$) For each variable j and time period t, the regression coefficient $\beta_{j,t}$ is updated using the following expressions for the mean μ and the variance σ^2:

$$\mu = \frac{\phi_1 p_j^* \beta_{j,t-1}}{\lambda_1} + \frac{\phi_1 p_j^* \beta_{j,t+1}}{\lambda_1} + z \tag{9}$$

$$\sigma^2 = \frac{1}{\text{reg}^2 + \frac{p_j^*}{\lambda_1} + \phi_1^2 \frac{p_{j,t+1}^*}{\lambda_1} + (1-\phi_1^2)\frac{\text{shrink}}{\lambda_1}} \tag{10}$$

The term z represents the adjusted residual, and the term shrink represents the shrinkage adjustment applied to the coefficients.

The final value of $\beta_{j,t}$ is computed by applying a threshold-based shrinkage adjustment:

$$\beta_{j,t} = \max(0, \mu - \text{shrink}) - \max(0, -\mu - \text{shrink}) \tag{11}$$

2.2.5 Convergence Criterion

The algorithm iterates until the squared difference between updated and previous coefficients is below a predefined threshold ϵ, indicating convergence. The convergence criterion is given by:

$$\epsilon = \sum_{j=1}^{p} \sum_{t=1}^{T} (\beta_{j,t}^{\text{new}} - \beta_{j,t})^2 \tag{12}$$

The algorithm described in this section was implemented in Python, and the core functions were further optimized by translating them to C++ using the Cython library. This adaptation significantly accelerates the computation by leveraging compiled code, which is particularly beneficial for large datasets and complex models.

The complete implementation of the algorithm can be found in the Appendix, where the Python code and its C++-optimized version are provided. The use of Cython allows for a seamless integration of C++ code within the Python environment, maintaining the flexibility of Python while achieving the performance of compiled languages.

3. Choosing Between Gaussian and Laplace for Spike and Slab

The general case, and the most common choice, is to use a Gaussian slab and a Laplace spike. This configuration balances smooth variation for significant coefficients provided by the Gaussian slab, with strong sparsity induced by the ℓ_1-like penalty, as demonstrated by LASSO approach of Tibshirani (1996), of the Laplace spike Ročková and McAlinn (2019). However, in this work, we will explore other combinations of spike and slab distributions, including Gaussian-Gaussian and Laplace-Laplace, to evaluate their effects on model behavior and penalization structures.

The selection of spike and slab distributions significantly influences the behavior and penalization structure of the Dynamic Spike-and-Slab framework. Where the first term corresponds to the spike distribution and the second to the slab distribution. Each configuration has distinct implications for the model's penalization and sparsity-inducing properties.

3.1 DSS Penalty

As shown by Ročková (2018), the Dynamic Spike-and-Slab (DSS) penalty is defined as:

$$\text{Pen}(\beta \mid \beta_{t-1}, \beta_{t+1}) = \text{pen}(\beta \mid \beta_{t-1}) + \text{pen}(\beta_{t+1} \mid \beta) + C, \tag{13}$$

where $C \equiv -\text{Pen}(0 \mid \beta_{t-1}, \beta_{t+1})$ is a norming constant.

It is easy to see that the penalty has a dependence on the previous and next parameters. If we write the derivative of the penalty, we can see the effects of the variables in the spike-and-slab dynamics.

$$\frac{\partial \text{Pen}(\beta \mid \beta_{t-1}, \beta_{t+1})}{\partial |\beta|} = -\Lambda^\star(\beta \mid \beta_{t-1}, \beta_{t+1}), \tag{14}$$

By dividing the effects into past and future penalties, we can better understand how these dynamics affect the model. The penalty from the past can be written as:

$$\Lambda^\star(\beta \mid \beta_{t-1}, \beta_{t+1}) = \lambda^\star(\beta \mid \beta_{t-1}) + \tilde{\lambda}^\star(\beta \mid \beta_{t+1}), \tag{15}$$

and:

$$\lambda^\star(\beta \mid \beta_{t-1}) = -\frac{\partial \text{pen}(\beta \mid \beta_{t-1})}{\partial |\beta|}, \quad \tilde{\lambda}^\star(\beta \mid \beta_{t+1}) = -\frac{\partial \text{pen}(\beta_{t+1} \mid \beta)}{\partial |\beta|}.$$

Using the past penalty as an example, $\lambda^\star(\beta \mid \beta_{t-1})$ can be expanded as:

$$\lambda^\star(\beta \mid \beta_{t-1}) = -p_t^\star(\beta) \frac{\partial \log \psi_1(\beta \mid \mu_t, \lambda_1)}{\partial |\beta|} - [1 - p_t^\star(\beta)] \frac{\partial \log \psi_0(\beta \mid \lambda_0)}{\partial |\beta|},$$

where:

$$p_t^\star(\beta) \equiv \frac{\theta_t \psi_1(\beta \mid \mu_t, \lambda_1)}{\theta_t \psi_1(\beta \mid \mu_t, \lambda_1) + (1 - \theta_t) \psi_0(\beta \mid \lambda_0)}.$$

The future penalty follows the same logic, but with respect to β_{t+1}, and is defined as:

$$\tilde{\lambda}^\star(\beta_{t+1} \mid \beta) = -\frac{\partial \text{pen}(\beta \mid \beta_{t+1})}{\partial |\beta|}$$

$$= -p_t^\star(\beta_{t+1}) \frac{\partial \log \psi_1(\beta_{t+1} \mid \mu_t, \lambda_1)}{\partial |\beta_{t+1}|} - [1 - p_t^\star(\beta_{t+1})] \frac{\partial \log \psi_0(\beta_{t+1} \mid \lambda_0)}{\partial |\beta_{t+1}|}, .$$

It is important to note that there is an implicit dependence on the previous value β_{t-1} in $\text{pen}(\beta \mid \beta_{t-1})$, which is hidden in θ_t and μ_t, as discussed in Ročková and McAlinn (2019).

3.2 Gaussian-Gaussian Specification

Let us assume a Gaussian distribution for the spike, such that:

$$\psi_0(\beta | \lambda_0) = \frac{1}{\sqrt{2\pi\lambda_0}} \exp\left(-\frac{\beta^2}{2\lambda_0}\right). \tag{16}$$

where μ is the mean and σ^2 is the variance. In the case of the spike, we assume $\mu = 0$, as the spike is centered around zero, and λ_0, which defines the variance of the spike. It follows a slab gaussian distribution, given by: $\sigma^2 = \lambda_0$ and replacing x with β:

$$\psi_1(\beta|\lambda_1, \mu_t) = \frac{1}{\sqrt{2\pi\lambda_1}} \exp\left(-\frac{(x-\mu_t)^2}{2\lambda_1}\right). \tag{17}$$

This distribution represents the spike in the Dynamic Spike-and-Slab framework, with λ_0 controlling the strength of the shrinkage around zero Ročková and McAlinn (2019).Under a Gaussian slab distribution, $\psi_1(\beta_t|\mu_t, \lambda_1)$, we arrive at a model where the autoregressive process for β_t follows a stationary Gaussian AR(1) process. Specifically, the process is given by:

$$\beta_t = \phi_0 + \phi_1(\beta_{t-1} - \phi_0) + e_t,$$

where: - $|\phi_1| < 1$, ensuring stationarity, - $e_t \overset{iid}{\sim} N(0, \lambda_1)$.

This formulation highlights the smoothing property of the Gaussian slab, as it incorporates past values of β_t to ensure a structured and stationary dynamic process.

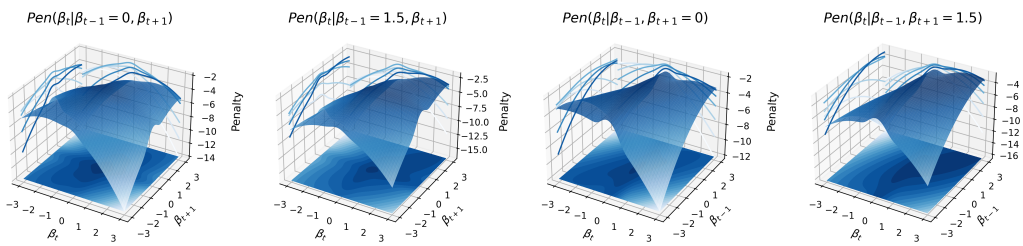

Figure 1: Gaussian-Gaussian Penalty

Figure 1 illustrates how the penalty function operates under the Gaussian spike and slab specification. This visualization highlights the interplay between the spike and slab distributions in shaping the penalty applied to the coefficients β_t.

In Figure 2, we explore the sensitivity of the transition weight parameter θ_t, which is critical for determining the balance between the spike and slab components. The parameters used for both figures are as follows: $\lambda_0 = 0.1$for the penalty and $\lambda_0 = 0.9$ for the transition weight plot, $\phi_1 = 0.96$, and $\lambda_1/(1-\phi_1^2) = 25$ for the penalty and $\lambda_1/(1-\phi_1^2) = 10$ for the transition weight. These values were chosen solely for visualization purposes to provide a practical visualization of the penalty function's behavior. Additionally, $\Theta = 0.9$ reflects the marginal importance weight.

It can be observed that a small λ_0 concentrates the distribution mass around zero, effectively shrinking the coefficients to values close to zero. This highlights the importance of finding a suitable balance between λ_0 and λ_1 to ensure adequate sparsity while simultaneously allowing the parameters to vary over time. Such flexibility is essential for capturing data patterns dynamically.

It is also noteworthy that the transition weight $\theta(\beta)$ in this specification assigns higher probabilities as $|\beta|$ increases, effectively reducing shrinkage for larger coefficients. Conversely, smaller values of $|\beta|$ are more strongly shrunk toward zero.

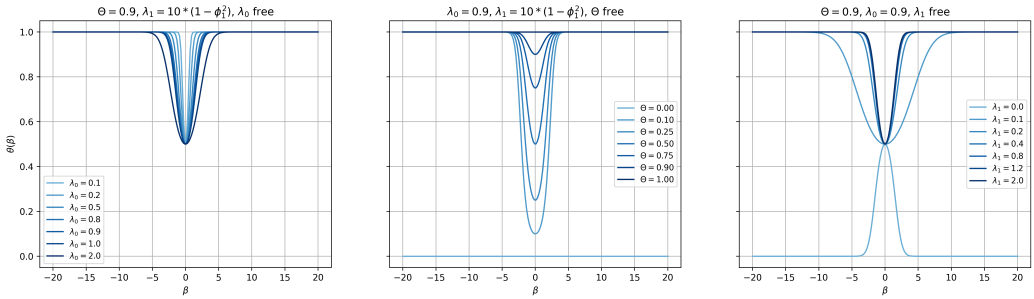

Figure 2: Gaussian-Gaussian Transition Weight

The sensitivity of the parameters is illustrated in Figure 2. The variance parameter λ_0 influences the magnitude of the coefficients that can be included in the model. Additionally, Θ acts as a global shrinkage strength parameter, while λ_1, the slab variance, determines the probability of slab inclusion.

3.3 Laplace-Gaussian Specification

The Gaussian slab is defined according to (18), while the Laplace spike is given by the following expression:

$$\psi_1(\beta_t \mid \lambda_0) = \frac{\lambda_0}{2} e^{-|\beta_t|\lambda_0}, \tag{18}$$

where $\psi_1(\beta_t \mid \lambda_0)$ represents the Laplace distribution with scale parameter λ_0, applied to the regression coefficient β_t. The behavior of β_t under a Laplace distribution changes to a Laplace Autoregressive process, which tends to concentrate the next value close to the last due to the density of the Laplace distribution. Kalli and Griffin (2014) discuss its implications.

As demonstrated by Ročková and McAlinn (2019) and Ročková (2018) (see Figure 3), the Laplace spike offers the advantage of shrinking parameters directly to zero without requiring any threshold. This property is particularly desirable for computational efficiency, as it enables focusing solely on the active parameters, which form a smaller subset for estimation. As shown by Ročková and McAlinn (2019) the prospective and retrospective

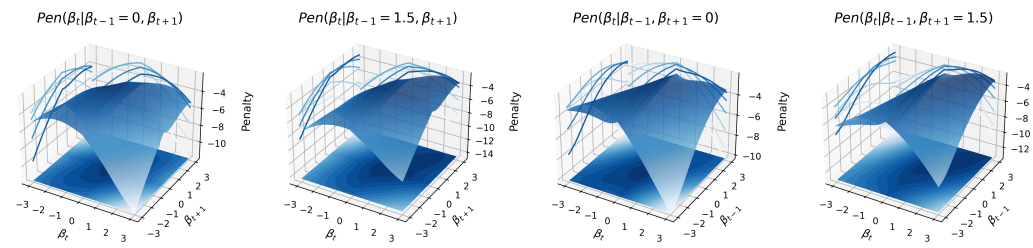

Figure 3: Laplace Gaussian Penalty

penalties depend on the mixing weight $\theta_t \equiv \theta(\beta)$ in (put equation number here). How-

ever, Laplace tails will begin to dominate for large enough $|\beta|$, where the probability $\theta(\beta)$ will start to drop (for $|\beta|$ greater than $\delta \equiv (\lambda_0 + \sqrt{2C/A})A$, where $A = \lambda_1/(1-\phi_1^2)$ and $C = \log[(1-\Theta)/\Theta\lambda_0/2\sqrt{2\pi A}]$). Nevertheless, the turning point δ can be made sufficiently large by increasing Θ and reducing λ_0. This behavior can be seen in Figure 4.

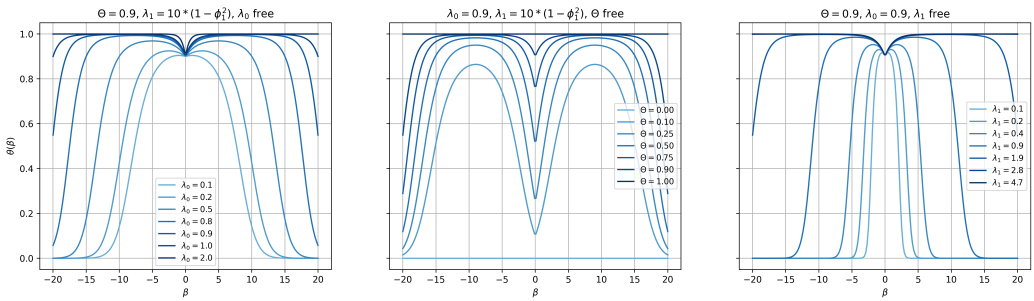

Figure 4: Laplace Gaussian Transition Weight

As shown by Ročková and McAlinn (2019) the behavior of the prospective and retrospective penalties in the Laplace spike case, is ultimately tied to the mixing weight $\theta_t \equiv \theta(\beta)$. It can be seen that the Laplace tails will begin to dominate the probabilities for large enough $|\beta|$, where the probability $\theta(\beta)$ will begin to drop for $|\beta|$ greater than

$$\delta \equiv (\lambda_0 + \sqrt{2C}/A)A$$

Where $A = \lambda_1/(1-\phi_1^2)$ and $C = \log[(1-\Theta)/\Theta\lambda_0/2\sqrt{2\pi A}])$. However, we can make the turning point δ large enough with larger values Θ and smaller values λ_0, as indicated in Figure 4.

3.4 Laplace-Laplace Specification

The Laplace slab, show in (19), and Laplace spike specification, called *Spike-and-Slab LASSO* by Ročková (2018), is a framework where the spike is assigned a smaller variance compared to the slab to induce sparsity. Additionally, the Laplace slab generates a concentration of mass around μ_t, which implies that it tends to keep the parameter more stable near its previous value than the gaussian slab. This characteristic is particularly useful in scenarios where one aims to minimize excessive changes in the parameter values, leaning towards a structural model specification.

If we want to shrink μ_t, which determines the structure of β_t, towards its past value, we can specify it as a Laplace slab. The Laplace slab distribution can be defined as:

$$\psi_1(\beta_t \mid \mu_t, \lambda_1) = \frac{\lambda_1}{2} e^{-|\beta_t - \mu_t|\lambda_1} \tag{19}$$

While both the Gaussian and Laplace slabs lead to a conditional posterior mean that shrinks towards the last value, the conditional posterior mode will shrink exactly to the past value in the case of the Laplace slab (but not for the Gaussian) Ročková and McAlinn (2019).

Ročková (2018) provide a detailed discussion on the Spike-and-Slab LASSO and demonstrate that the difference in variance between the two components of the mixture generates

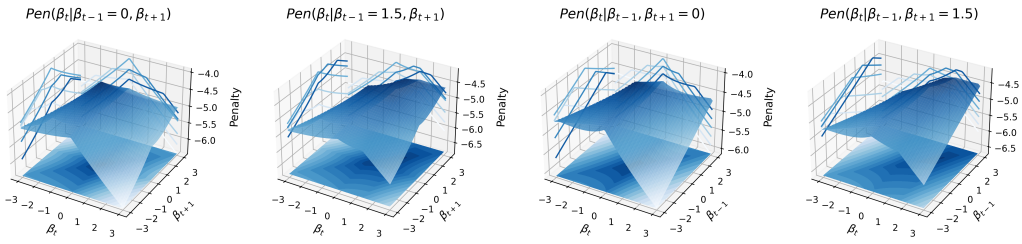

Figure 5: Laplace-Laplace Penalty

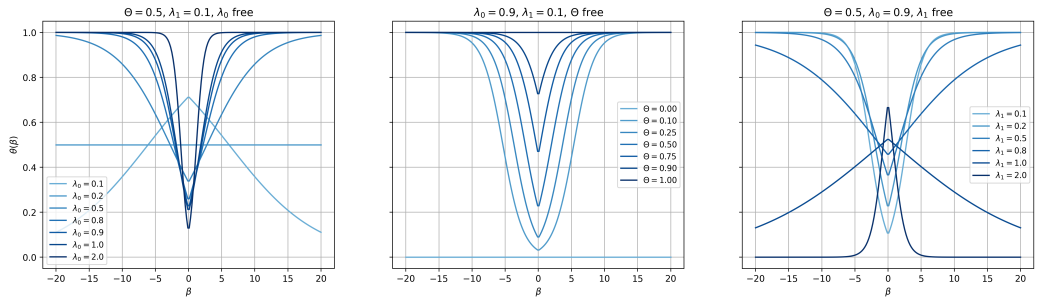

Figure 6: Laplace-Laplace Transition Weight

a threshold δ, which classifies whether a parameter belongs to the spike or the slab. The threshold, representing the point where the slab starts to dominate the spike, is given by:

$$p^{\star}_{\theta}(\beta_j) = \frac{\theta\psi_1(\beta_j)}{\theta\psi_1(\beta_j) + (1-\theta)\psi_0(\beta_j)}.$$

$$\delta = \frac{1}{\lambda_0 - \lambda_1} \log\left(\frac{1}{p^{\star}_{\theta}(0)} - 1\right).$$

This is evident in Figure 5, where the LASSO approach shrinks the parameter towards its last value. As a result, the penalty exhibits a greater point mass at μ_t. Additionally, the transition weights are significantly more sensitive to small changes in the specification of λ_0 and λ_1 compared to other specifications, as can be seen in Figure 6. This highlights the importance of carefully tuning these parameters into the Spike-and-Slab LASSO framework.

The left panel of Figure 6 illustrates that, in the case where both the slab and spike components follow Laplace distributions, if the precision parameter λ_1 is significantly larger than λ_0, the spike component dominates the inclusion probability across the entire relevant space of the function. This behavior is evident from the way the inclusion probability degenerates as λ_1 increases, effectively suppressing the contribution of the slab and leading to a near-zero probability of variable inclusion.

4. Results in Simulated Dataset

To evaluate the model, we will simulate a dataset and assess its performance using three fit metrics:

1. Sum of Squared Errors (SSE):

$$\text{SSE} = \sum_{t=1}^{T} \sum_{j=1}^{p} \left(\beta_{t,j}^{\text{true}} - \beta_{t,j}^{\text{estimated}}\right)^2$$

2. Sum of Absolute Errors (SAE):

$$\text{SAE} = \sum_{t=1}^{T} \sum_{j=1}^{p} \left|\beta_{t,j}^{\text{true}} - \beta_{t,j}^{\text{estimated}}\right|$$

3.Hamming Distance:

$$\text{Hamming Distance} = \sum_{t=1}^{T} \sum_{j=1}^{p} \left|I\left(\theta_{t,j} > \text{threshold}\right) - I\left(\beta_{t,j}^{\text{true}} \neq 0\right)\right|$$

where $I(\cdot)$ is an indicator function that equals 1 if the condition inside is true, and 0 otherwise, where the probability threshold is 0.5.

In this dataset, time series data is generated with coefficients that evolve over time according to an autoregressive process (AR(1)) while incorporating sparsity and regime switching. The goal is to simulate a dynamic environment where the relationships between predictors and the response variable are time-varying, and certain predictors may become inactive (with coefficients equal to zero) depending on the regime.

The simulation involves 200 time periods ($T = 200$) and 10 informative predictors ($p = 5$), which follow an AR(1) process. The coefficients start in an active or inactive state based on the zero probability parameter (zero_prob = 0.5). This parameter determines the likelihood that a coefficient starts in the inactive state (0) at the beginning of the series. Once initialized, the persistence parameter (persistence = 0.98) governs the probability of remaining in the current state (active or inactive) at each time step. Higher persistence values ensure smoother transitions and more prolonged periods of activity or inactivity, creating a stable temporal structure.

For coefficients in the active state, their values evolve over time based on an AR(1) process. This process is defined by the autoregressive coefficient ($\phi = 0.98$) and a noise term with a standard deviation ($\sigma = 0.25$). Together, these parameters ensure that active coefficients remain temporally dependent while allowing for some variability. Predictors from time $t-1$ are lagged and used to influence the dependent variable at time t, creating a dynamic relationship between past predictors and current outcomes.

In addition to the informative predictors, 45 uninformative noise variables are included in the dataset to mimic real-world scenarios where not all predictors are relevant. These noise variables add complexity to the simulation, making it suitable for testing models that can effectively identify and utilize only the informative predictors. The response variable is generated by summing the contributions of the active predictors (weighted by their respective coefficients) and adding random noise to introduce randomness in the outcomes.

This simulation framework produces a realistic and dynamic dataset with time-varying coefficients, sparsity, and a temporal structure. It is particularly suited for testing advanced models like dynamic spike-and-slab specifications, as it challenges the model to handle both temporal dependencies and feature selection in the presence of noise and sparsity.

We began by running a Dynamic Linear Model (DLM), that is achieved by setting $\Theta = 1$. The parameters for this model were $\phi_0 = 0$, $\phi_1 = 0.98$, $\lambda_0 = 0.9$ the spike parameter and $\lambda_{11} = 10(1 - \phi_1^2)$, the parameter for the slab. The model assumes $\Theta = 1$, indicating that it is fully dynamic without additional sparsity.

Next, we ran the Gaussian-Gaussian model. The parameters for this model were $\phi_0 = 0$, $\phi_1 = 0.95$, a slightly lower persistence of β_t parameter compared to the DLM. The variance of the observation noise was reduced to $\lambda_0 = 0.1$, introducing more sparsity, while the variance of the state noise was $\lambda_1 = 10(1 - \phi_1^2)$, dependent on the persistence parameter. The model assumed $\Theta = 0.92$, making it less dynamic and incorporating some level of sparsity.

Following this, we implemented the Laplace-Gaussian model. The parameters for this model were $\phi_0 = 0$, $\phi_1 = 0.99$, $\lambda_0 = 5$, allowing for more flexibility in the data, while the variance of the state noise was set to $\lambda_1 = 5(1 - \phi_1^2)$, scaled based on the persistence parameter. This model assumed $\Theta = 0.90$, balancing dynamics and sparsity.

Finally, we ran the Laplace-Laplace model. The parameters for this model were $\phi_0 = 0$, $\phi_1 = 0.99$, $\lambda_0 = 0.05$and $\lambda_1 = 5$, a fixed variance for the state noise. The model assumed $\Theta = 0.5$, representing a sparsity-dominant configuration that significantly reduces dynamics.

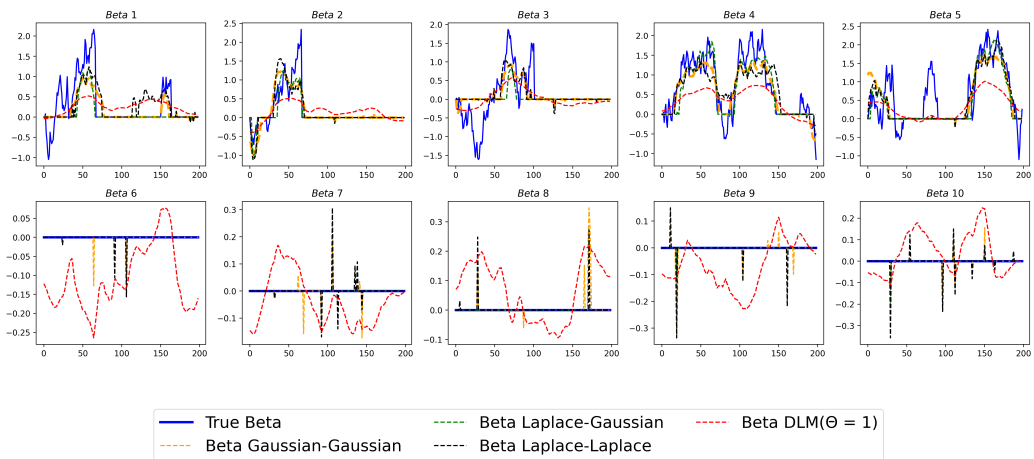

Figure 7: Coefficients Estimated

In Figure 7, we can observe the trajectories of the estimated parameters compared to the true ones for each model. The figure illustrates the fit of each model by plotting the first 10 parameters. Among these, the first five represent active parameters, while the remaining five correspond to noise parameters.

The dataset under analysis contains many noise parameters, which directly impact the performance of the DLM ($\Theta = 1$). Since the DLM captures information without incor-

porating mechanisms of sparsity, it ends up propagating noise throughout the estimates, making it difficult to distinguish between relevant and irrelevant parameters. This behavior is evident in the noise coefficients (β_6 to β_{10}), where the DLM exhibits significant fluctuations, indicating excessive sensitivity to non-informative variations.

On the other hand, models that incorporate sparsity, such as the Laplace-Gaussian and Laplace-Laplace, demonstrate a much greater ability to address this issue. In particular, the Laplace-Gaussian model appears to be less responsive to noise parameters, keeping them closer to zero, which contributes to the robustness of the estimates. At the same time, the Laplace-Gaussian achieves a good fit to the true coefficients (β_1 to β_5), balancing responsiveness and sparsity. This flexibility is a significant advantage over the Laplace-Laplace model, which, although more sparse, can be overly conservative and less adaptable to the dynamics of active coefficients.

Thus, the plot demonstrates that the Laplace-Gaussian model strikes a balance: it efficiently reduces the impact of noise while maintaining sufficient flexibility to capture the trajectories of relevant coefficients. This characteristic makes the model particularly appealing in contexts where it is necessary to handle many noise parameters without sacrificing the fit to the data.

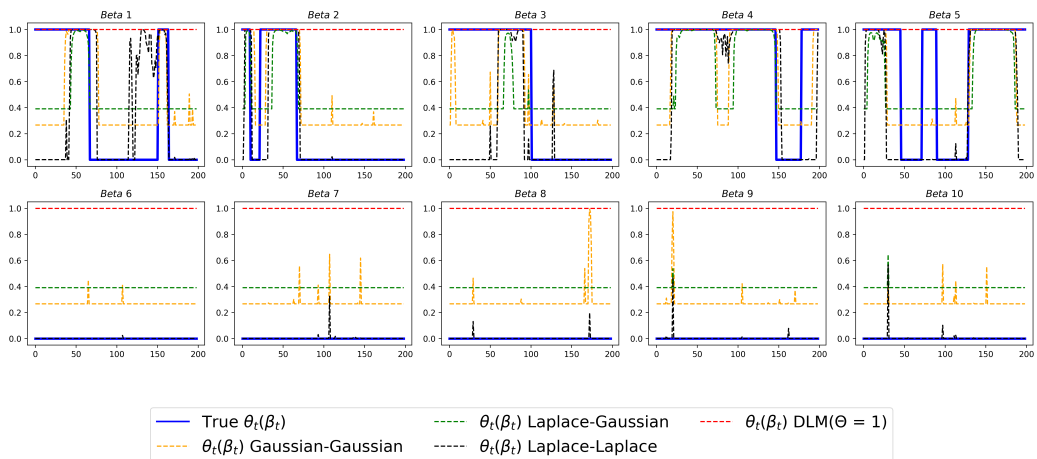

Figure 8: Inclusion Probabilities

In figure 8 it's possible to observe from the inclusion probabilities that the model demonstrates a good fit to the parameters, with some "extra" sparsity when the dynamic β approaches zero. However, as β moves away from zero, it is once again captured by the model. The noise in the Gaussian-Gaussian model becomes evident when observing the "short" jumps in inclusion probabilities, which are sporadically captured in noise variables. Despite this, these variables do not persist in the model, which helps maintain its robustness.

The double Laplace model exhibits the highest level of sparsity and appears to be highly responsive to the true variables. Among all models, the one that propagated the least noise was the Gaussian-Laplace model. In the variables shown in the figure, no "jumps" are observed, as is the case with the Gaussian-Gaussian model. Additionally, in the variables not plotted, such "jumps" are very isolated and practically do not occur.

Model	SSE	SAE	Hamming Distance
Gaussian-Gaussian	252.68	419.04	203
Laplace-Gaussian	200.77	268.23	255
Laplace-Laplace	231.87	357.83	38
DLM ($\Theta = 1$)	435.00	1298.60	9412

Table 1: Fit Metrics for Different Models

Through Table 1, it is possible to observe that the model with the best fit for the betas is the Gaussian-Gaussian model. Although it does not exhibit the best shrinkage, it replicates the models most effectively, as evidenced by its SSE and SAE metrics. However, the model with the best shrinkage is the Laplace-Laplace model, demonstrating that the structure of assigning more density to past values appears to provide a favorable characteristic for variable selection.

5. Empirical Analysis: Forecasting Brazilian Inflation

In this section, we apply the proposed method to forecast Brazilian inflation using the high-dimensional dataset from Garcia et al. (2017), which contains data on the Brazilian consumer price index (IPCA). The IPCA is the official inflation index in Brazil and is widely used as a reference for inflation-linked bonds. The dataset covers the period from January 2003 to December 2015, with a total of 156 monthly observations. The data were obtained from Bloomberg and the Central Bank of Brazil and consist of 59 macroeconomic variables and 34 variables linked to specialist forecasts (Garcia et al., 2017).

Table 2: Macroeconomic variables

Column 1	Column 2	Column 3
Brazil CPI IPCA MoM	Brazil Unemployment Statistic Male	Brazil Central Government Net Revenue
FGV Brazil General Prices IGP-	Brazil Unemployment Statistic Total	Brazil Central Government Revenue from the Central Bank
FGV Brazil General Prices IGP-DI	IMF Brazil Unemployment Rate in Percentage	Brazil Central Government Total Expenditures
FGV Brazil General Prices IGP-M	CNI Brazil Manufacturing Industry Employment	Brazil National Treasury Gross Revenue
FGV Brazil General Prices IGP-10	Brazil Industry Working Hours	Brazil Importing Tax Income
Brazil CPI IPCA Median Market Expectations	Brazil Average Real Income	BNDES Brazil Income Taxes
Brazil Total Electricity Consumption	Brazil Minimum Wage	Brazil Treasury Revenue from Import Tax
Brazil Industrial Electricity Consumption	USD-BRL X-RATE	Brazil Treasury Revenue from Social Security
BofA Merrill Lynch Economic Conditions Index	USD-BRL X-RATE Tourism	Brazil Current Account Balance
CNI Brazil Manufacture Industrial Confidence Index	EUR-BRL X-RATE	Brazil Trade Balance FOB
CNI Brazil Manufacture Industrial Installed Capacity	BRAZIL IBOVESPA INDEX	Brazil Public Net Fiscal Debt as Percentage of GDP
Brazil Industrial Production Seasonally Adjusted	Brazil Savings Accounts Deposits	Brazil Public Net Fiscal Debt
Brazil Industrial Production Annual	Brazil Total Savings Deposits	Brazilian Federal Government Domestic Debt
IBGE Brazil Unemployment Rate	Brazil BNDES Long Term Interest Rate	Brazilian Public Net Government and Central Bank Domestic Debt
Brazil Unemployment Statistic Female	Brazil Selic Target Rate	Brazilian States Debt to Consolidated Net Debt
Brazil Cetip DI Interbank Deposits	Brazilian States Debt to Foreigners	Brazilian Cities Debt to Foreigners
Brazil Monetary Base	Brazilian Cities Debt Total	Brazilian Cities Debt to Foreigners Total
Brazil Money Supply M1 Brazil	t+1 median	Top5 t+1 median
Brazil Money Supply M2 Brazil	t+2 median	Top5 t+2 median
Brazil Money Supply M3 Brazil	t+3 median	Top5 t+3 median
Brazil Money Supply M4 Brazil	t+4 median	Top5 t+4 median
t+5 median	Top5 t+5 median	t+6 median
Top5 t+6 median	t+7 median	Top5 t+7 median
t+8 median	Top5 t+8 median	t+9 median
Top5 t+9 median	t+10 median	Top5 t+10 median
t+11 median	Top5 t+11 median	t+12 median
Top5 t+12 median	t+13 median	Top5 t+13 median
1 median^2	1 mean	1 mean^2
1 Std	2 median	2 mean
2 mean^2	2 Std	fmed2
fmean	fmean2	fdp
lfmed2	lfmean	lfmean2
lfdp		

The macroeconomic variables cover a wide range of indicators, including several inflation and industry indexes, unemployment rates and other labor-related variables, energy consumption, exchange rates, stock markets, government accounts, expenditure and debt, taxes, monetary variables, and the exchange of goods and services (Garcia et al., 2017). The dataset also includes expert forecasts from the FOCUS survey, produced by the Central Bank of Brazil. The expectation variables include the median of the h-period-ahead specialist forecasts, the median of the top five (Top5) experts—i.e., the five experts who produced the best forecasts in the previous period—and, finally, the mean and the standard deviation of the Top5 experts. These expectation variables aim to provide a more accurate assessment of future inflation expectations. The variables can be seen in table 2.

To ensure stationarity, we conduct the augmented Dickey-Fuller test for each time series in the dataset. If a unit root is detected, we apply the logarithmic difference to achieve stationarity. Once all variables are stationary, we normalize them to have zero mean and unit variance to facilitate model convergence and interpretability. The transformed variables are then used as inputs for the forecasting model discussed in previous sections.

The primary objective of this empirical analysis is to assess the forecasting performance of the dynamic variable selection model with spike-and-slab priors when applied to real-world macroeconomic data, particularly in a high-dimensional setting. By leveraging both macroeconomic indicators and expectation variables, the model aims to improve predictive accuracy by dynamically selecting the most relevant variables at each time step. The inclusion of the Top5 expert forecasts is particularly relevant, as previous studies have shown that these forecasts tend to perform better in short-term horizons but lose accuracy as the forecast horizon increases.

To implement the forecasting models, we test several parameterizations to calibrate the models described in Table 3. The models are constructed following the methodology discussed in the previous sections, leveraging the theoretical framework and estimation procedures detailed earlier in this paper. The implementation of the models is carried out using the Python and Cython code provided in the appendix, ensuring computational efficiency and reproducibility.

Table 3: Specification of Spike-and-Slab Models

Model	**Spike** (ψ_0)	**Slab** (ψ_1)
Gaussian-Gaussian	$\psi_0(\beta_t \mid \lambda_0) = \frac{1}{\sqrt{2\pi\lambda_0}} \exp\left(-\frac{\beta_t^2}{2\lambda_0}\right)$	$\psi_1(\beta_t \mid \mu_t, \lambda_1) = \frac{1}{\sqrt{2\pi\lambda_1}} \exp\left(-\frac{(\beta_t-\mu_t)^2}{2\lambda_1}\right)$
Laplace-Gaussian	$\psi_0(\beta_t \mid \lambda_0) = \frac{1}{2\lambda_0} \exp(\frac{-\lvert\beta_t\rvert}{\lambda_0})$	$\psi_1(\beta_t \mid \mu_t, \lambda_1) = \frac{1}{\sqrt{2\pi\lambda_1}} \exp\left(-\frac{(\beta_t-\mu_t)^2}{2\lambda_1}\right)$
Laplace-Laplace	$\psi_0(\beta_t \mid \lambda_0) = \frac{1}{2\lambda_0} \exp(\frac{-\lvert\beta_t\rvert}{\lambda_0})$	$\psi_1(\beta_t \mid \mu_t, \lambda_1) = \frac{1}{2\lambda_1} \exp(\frac{-\lvert\beta_t-\mu_t\rvert}{\lambda_1})$

These parameterizations aim to explore the sensitivity of the models to different configurations and identify the optimal settings for forecasting Brazilian inflation.

In this analysis, we will first run a model with $\Theta = 1$, which is equivalent to a Dynamic Linear Model (DLM). Subsequently, we will compare its performance with five well-calibrated parameterizations for each of the specifications described in Table 3. This approach allows us to benchmark the proposed dynamic variable selection models with spike-and-slab priors against a simpler baseline model, evaluating their relative performance in forecasting Brazilian inflation.

Before proceeding, it is essential to discuss the recursive forecasting algorithm employed in this study. The data are divided into training and testing sets, where the training set consists of 100 samples and the testing set includes 56 samples. Initially, the model estimates parameters with $\Theta = 1$, corresponding to a Dynamic Linear Model (DLM). These estimated parameters are then used as priors to estimate the model with active shrinkage ($\Theta < 1$). This approach is particularly useful for avoiding local minima, a common issue in Expectation-Maximization (EM) algorithms, as discussed by Ročková and McAlinn (2019).

For each forecast, the parameters are first estimated using the DLM, and a prediction is made with active shrinkage ($\Theta < 1$) by using the DLM estimates for β as priors for the model. This iterative procedure ensures that the parameters estimated at the last time step t, denoted as β_t, are projected forward using the dynamic equation for μ_t, which defines $\beta_{t+1} = \phi_0 + \phi_1 \beta_t$. The estimated value of β_{t+1} is then multiplied by the corresponding x_{t+1} (which is possible since we use a lagged data in the input x_t) values to forecast y_{t+1}, without the model having observed the actual value of y_{t+1}.

Finally, we compute the predicted mean squared error (MSE) and the predicted mean absolute error (MAE) for each model. This methodology allows for a robust comparison of forecast performance across different specifications, while ensuring the models are evaluated in a forward-looking, out-of-sample framework.

Table 4 presents forecast results across different horizons: short-term ($h = 1$), medium-term ($h = 6$), and long-term ($h = 12$). The error metrics used are the Mean Forecast Squared Error (MFSE) and the Mean Absolute Forecast Error (MAFE). The AR(1) model and the DLM ($\Theta = 1$) model are considered benchmarks to compare the performance of more complex models such as Gaussian-Gaussian, Laplace-Gaussian, and Laplace-Laplace.

Table 4: Forecasting Results

Model	ϕ_1	λ_{11}	λ_0	Θ	**h = 1**		**h = 6**		**h = 12**	
					MFSE	**MAFE**	**MFSE**	**MAFE**	**MFSE**	**MAFE**
AR(h)	-	-	-	-	0.6651	0.6611	1.4661	0.9280	1.3049	0.8760
DLM ($\Theta = 1$)	0.98	1.3	0.9	1.0	0.5181	0.4876	0.7937	0.5480	0.7047	0.5593
	0.98	1.3	0.024	0.9	0.3052	0.4129	0.8792	0.4736	0.5845	0.4729
	0.98	1.3	0.023	0.9	**0.2827**	**0.3803**	0.6711	0.4645	0.5635	0.4754
Gaussian-Gaussian	0.98	1.3	0.025	0.9	0.3001	0.4125	0.7702	0.4604	**0.5215**	**0.4597**
	0.98	1.3	0.024	0.92	0.3179	0.4126	0.7146	0.4689	0.5340	0.4588
	0.98	1.3	0.023	0.92	0.2997	0.3926	**0.6594**	**0.4642**	0.5478	0.4695
	0.98	0.65	0.02	0.5	0.3253	0.3905	0.8748	0.5053	0.4807	0.4376
	0.98	0.65	0.02	0.4	0.3168	0.3903	0.8523	0.5007	0.4634	0.4334
Laplace-Gaussian	0.98	0.65	0.02	0.3	0.3112	0.3849	0.8620	0.4965	0.5111	0.4452
	0.98	0.65	0.02	0.2	0.2959	0.3821	0.7543	0.4747	0.4830	0.4397
	0.98	0.65	0.02	0.1	**0.2843**	**0.3769**	**0.7009**	**0.4653**	**0.4432**	**0.4177**
	0.98	0.65	0.04	0.45	**0.2922**	0.3883	0.9118	0.4959	0.5807	0.4662
	0.985	0.67	0.037	0.42	0.2967	**0.3863**	**0.6345**	**0.4472**	**0.4886**	0.4407
Laplace-Laplace	0.98	0.65	0.038	0.42	0.2976	0.3867	1.2607	0.5279	0.5412	0.4605
	0.975	0.65	0.035	0.4	0.3020	0.3898	0.7265	0.4647	0.5549	0.4677
	0.98	0.65	0.032	0.38	0.3064	0.3998	0.7145	0.4664	0.5125	**0.4404**

For the short-term horizon ($h = 1$), both the Gaussian-Gaussian and Laplace-Gaussian models showed superior performance compared to the benchmarks. The Gaussian-Gaussian model with $\lambda_0 = 0.023$ and $\Theta = 0.9$ achieved the lowest MFSE (0.2827) and one of the lowest MAFE values (0.3803). Similarly, the Laplace-Gaussian model with $\lambda_0 = 0.02$ and $\Theta = 0.1$ also performed very well, presenting the lowest MAFE (0.3769). The AR(1) benchmark model, by contrast, exhibited significantly higher error values ($MFSE = 0.6651$, $MASE = 0.6611$), indicating that the use of shrinkage in these models provided better predictive accuracy at the short-term horizon.

For the medium-term horizon ($h = 6$), the Laplace-Laplace model with $\lambda_0 = 0.037$ and $\Theta = 0.42$ delivered the best results in both MFSE and MAFE metrics. The model showed a clear improvement over the DLM benchmark, which presented an MFSE of 0.7937 and a MAFE of 0.5480. The AR(1) model for this horizon performed even worse, with an MFSE of 1.4661 and a MAFE of 0.9280. These results confirm that incorporating more flexible shrinkage structures improves predictive performance, even for medium-term horizons.

For the long-term horizon ($h = 12$), the best-performing model was again from the Laplace-Gaussian family, with $\lambda_0 = 0.02$ and $\Theta = 0.1$, showing an MFSE of 0.4432 and a MAFE of 0.4177. The performance gap between the benchmarks (AR(1) and DLM) and the more sophisticated models becomes even more evident in this horizon. The AR(1) model presented an MFSE of 1.3049 and a MAFE of 0.8759, while the DLM model, despite being better than AR(1), still showed worse performance compared to the Gaussian-Gaussian and Laplace-based models.

Overall, the results indicate that shrinkage plays a crucial role in improving forecasting accuracy. The more advanced models outperform the simpler benchmarks across all horizons, particularly in longer horizons. However, it is essential to note that a simplification was made in this study by applying the same parameter calibration across all forecasting horizons. Ideally, different parameter settings would be recalibrated for each horizon to better capture the dynamics of each timeframe. In this case, we selected five combinations for each family that performed relatively well and used them across all horizons, acknowledging that this approach may not fully optimize the models for each specific forecast length.

Figure 9 presents the cumulative forecasting errors for the horizons $h = 1$, $h = 6$, and $h = 12$. These errors were calculated based on the predictions of different models, namely DLM (Dynamic Linear Model), Gaussian-Gaussian (GG), Laplace-Gaussian (LG), and Laplace-Laplace (LL), compared to the actual data. The cumulative errors provide an intuitive understanding of how the models' forecasting accuracy evolves over time.

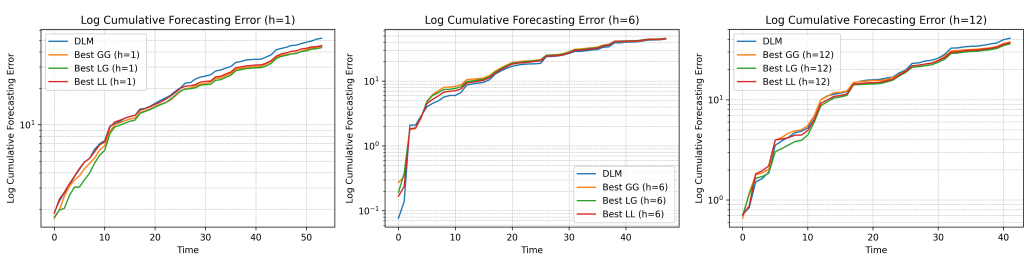

Figure 9: Cumulative Forecasting Errors

Starting with the horizon $h = 1$, we observe that the DLM model serves as a benchmark and exhibits a relatively higher cumulative error compared to the best-performing Gaussian-Gaussian (GG) model. This finding aligns with the results in Table 4, where the best GG combination for $h = 1$ (with $\lambda_0 = 0.023$ and $\Theta = 0.9$) achieved the lowest MFSE and MASE. The figure clearly shows that the best GG model accumulates errors more slowly over time, demonstrating superior predictive performance for short-term horizons.

For horizon $h = 6$, the cumulative forecasting errors are slightly higher across all models, as expected due to the increased forecasting uncertainty over longer periods. Nevertheless, the best-performing models (specifically LG and LL) still outperform the DLM benchmark. The shrinkage effect introduced by these models appears to contribute positively to reducing forecast errors, which is evident in both the cumulative error plots and the numerical results from Table 4. The LG model, in particular, shows robust performance in reducing errors over the medium-term horizon.

In the $h = 12$ horizon, the cumulative errors further increase, as longer-term forecasts naturally involve greater uncertainty. However, even in this challenging forecasting scenario, the best LL model continues to outperform the DLM benchmark, as seen in both the cumulative error plot and the table. The LL model with parameters $\lambda_0 = 0.037$ and $\Theta = 0.42$ shows the lowest cumulative error growth among the models, indicating that the Laplace prior effectively handles long-term shrinkage requirements.

Overall, the more sophisticated spike-and-slab models (GG, LG, and LL) generally outperform the simpler benchmark models such as AR(1) and DLM, particularly over longer horizons. The results also reinforce the importance of properly calibrated shrinkage parameters, as the best-performing models demonstrate that shrinkage plays a key role in reducing cumulative forecasting errors. However, it is important to note that the models were calibrated using the same parameter values across different horizons, which may have influenced the results. Ideally, parameter calibration should be horizon-specific to account for the different dynamics at various forecast lengths.

6. Conclusions

In this study, we explored the application of dynamic variable selection models in forecasting, focusing on Gaussian and Laplace priors within the Spike-and-Slab framework. The primary objective was to compare these models against simpler benchmarks, such as the AR(1) and Dynamic Linear Model (DLM), across different forecasting horizons.

The results indicate that more sophisticated models incorporating shrinkage mechanisms significantly improve forecasting performance compared to traditional benchmarks. Specifically, the Gaussian-Gaussian (GG) and Laplace-Gaussian (LG) models consistently outperformed the AR(1) and DLM across short, medium, and long-term horizons. For instance, the GG model showed superior performance for short-term forecasts (h = 1), while the LG model demonstrated robustness over longer horizons (h = 6 and h = 12).

The analysis also revealed that applying the same parameter calibration across all horizons may limit the models' potential. Ideally, each horizon would require a recalibration of parameters to better capture the distinct dynamics of each timeframe. Nevertheless, the models selected for this study, using a fixed calibration, still provided significant improvements over the benchmark models. This finding shows the importance of shrinkage in dynamic forecasting models, as it helps control overfitting and enhances predictive

accuracy.

The cumulative error plots further support the numerical results, highlighting the effectiveness of the GG and LG models in reducing forecasting errors over time. These models accumulate errors more slowly compared to the benchmarks, especially in longer forecasting horizons. The Laplace-based models, in particular, showcased their ability to handle noise and sparsity, making them suitable for high-dimensional data forecasting.

Overall, this study contributes to the literature by demonstrating the practical advantages of dynamic variable selection models with Spike-and-Slab priors in macroeconomic forecasting. The results suggest that incorporating more flexible shrinkage structures is essential for improving forecasting accuracy, particularly in high-dimensional settings. Future research could further enhance these findings by exploring horizon-specific calibrations and testing the models in other real-world forecasting scenarios.

References

Cogley, T. and Sargent, T. J. (2005). Drift and volatilities: Monetary policies and outcomes in the post WWII U.S. *Review of Economic Dynamics*, 8(2):262–302.

Garcia, M., Medeiros, M., and Vasconcelos, G. F. (2017). Real-time inflation forecasting with high-dimensional models: The case of Brazil. *International Journal of Forecasting*, 33(3):679–693.

George, E. and McCulloch, R. (1993). Variable selection via Gibbs sampling. *Journal of The American Statistical Association*, 88:881–889.

Kalli, M. and Griffin, J. E. (2014). Time-varying sparsity in dynamic regression models. *Journal of Econometrics*, 178(2):779–793.

Koop, G. and Korobilis, D. (2023). Bayesian dynamic variable selection in high dimensions. *International Economic Review,*, 64(1):1047–1074.

Louzada, F., Ferreira, P. H., and Nascimento, D. C. (2023). *Spike-and-Slab Priors and Their Applications*, pages 1–8. John Wiley & Sons, Ltd.

Ročková, V. (2018). Bayesian estimation of sparse signals with a continuous spike-and-slab prior. *The Annals of Statistics*, 46(1):401 – 437.

Ročková, V. and George, E. I. (2014). EMVS: The EM approach to Bayesian variable selection. *Journal of the American Statistical Association*, 109(506):828–846.

Ročková, V. and George, E. I. (2018). The Spike-and-Slab LASSO. *Journal of the American Statistical Association*, 113(521):431–444.

Ročková, V. and McAlinn, K. (2019). Dynamic variable selection with Spike-and-Slab process priors. *Bayesian Analysis*, 0:01.

Stock, J. H. and Watson, M. W. (2007). Why Has U.S. Inflation Become Harder to Forecast? *Journal of Money, Credit and Banking*, 39(s1):3–33.

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

7. Appendix

A. Code Implementation

The full implementation of the EM-DSS algorithm used in this study. The code was written in Python and optimized using Cython for computational efficiency.

Listing 1: Cython implementation of the EM-DSS algorithm

```
%%cython
#cython: profile=True, boundscheck=False, wraparound=False, cdivision=True, language_level=3

import numpy as np
cimport numpy as cnp
from libc.math cimport sqrt, exp, pi, fabs

# Auxiliary functions in Cython
cdef double gaussian_spike(double x, double lambda0):
    return (1 / (sqrt(2 * pi) * lambda0)) * exp(-0.5 * (x / lambda0) ** 2)

cdef double gaussian_slab(double x, double mu, double sigma):
    return (1 / (sqrt(2 * pi) * sigma)) * exp(-0.5 * ((x - mu) / sigma) ** 2)

cdef double laplace_spike(double x, double lambda0):
    return (1 / (2 * lambda0)) * exp(-fabs(x) / lambda0)

cdef double laplace_slab(double x, double mu, double scale):
    return (1 / (2 * scale)) * exp(-fabs(x - mu) / scale)

# Function to select the spike-and-slab model
cdef void select_spike_and_slab(char* model_type,
                                double (**spike_func)(double, double),
                                double (**slab_func)(double, double, double)):
    if model_type == b"gg":
        spike_func[0] = gaussian_spike
        slab_func[0] = gaussian_slab
    elif model_type == b"lg":
        spike_func[0] = laplace_spike
        slab_func[0] = gaussian_slab
    elif model_type == b"ll":
        spike_func[0] = laplace_spike
        slab_func[0] = laplace_slab
    else:
        raise ValueError("Invalid model type. Use 'gg', 'lg', or 'll'.")

# Main calculation functions
cdef double p_star_stat(double x, double phi0, double phi1, double Theta
```

```
    , double lambda1, double lambda0,
                          double (*spike_func)(double, double), double (*
                              slab_func)(double, double, double)):
    cdef double mu = phi0 / (1 - phi1)
    cdef double sigma = sqrt(lambda1 / (1 - phi1**2))
    cdef double numerator = (1 - Theta) * spike_func(x, lambda0)
    cdef double denominator = Theta * slab_func(x, mu, sigma)
    return 1 / (1 + numerator / denominator)

cdef double pstar_cond(double x, double previous, double phi0, double
    phi1, double theta, double lambda1, double lambda0,
                         double (*spike_func)(double, double), double (*
                             slab_func)(double, double, double)):
    cdef double mu = phi0 + phi1 * previous
    cdef double numerator = theta * slab_func(x, mu, sqrt(lambda1))
    cdef double denominator = (1 - theta) * spike_func(x, lambda0)
    return numerator / (numerator + denominator)

cdef double shrink_extra(double x, double future, double phi0, double
    phi1, double Theta, double lambda1, double lambda0, int plus,
                           double (*spike_func)(double, double), double (*
                               slab_func)(double, double, double)):
    cdef double theta_k = p_star_stat(x, phi0, phi1, Theta, lambda1,
        lambda0, spike_func, slab_func)
    cdef double pstar = pstar_cond(future, x, phi0, phi1, theta_k,
        lambda1, lambda0, spike_func, slab_func)
    cdef double der1 = lambda0 if plus == 1 else -lambda0
    return pstar * (1 - theta_k) - (1 - pstar) * theta_k

# Main EM-DSS model function
def EM_DSS_cy(cnp.ndarray[double, ndim=1] y, cnp.ndarray[double, ndim=2]
     X,
            double phi0, double phi1, double lambda11, double lambda0,
            cnp.ndarray[double, ndim=2] start, double epsilon, double
                Theta, int max_iter, str model_type):
    # Select spike-and-slab functions
    cdef double (*spike_func)(double, double)
    cdef double (*slab_func)(double, double, double)
    select_spike_and_slab(model_type.encode('utf-8'), &spike_func, &
        slab_func)

    cdef int T = X.shape[0]
    cdef int p = X.shape[1]
    cdef cnp.ndarray[double, ndim=2] beta = np.copy(start)
    cdef cnp.ndarray[double, ndim=2] beta_new = np.copy(beta)
    cdef double eps = epsilon + 1
    cdef cnp.ndarray[double, ndim=1] beta0 = np.zeros(p)
    cdef cnp.ndarray[double, ndim=1] beta0_new = np.copy(beta0)
    cdef cnp.ndarray[double, ndim=2] pstar = np.ones((p, T + 1)) * 0.5
    cdef cnp.ndarray[double, ndim=2] theta = np.ones((p, T)) * 0.5
    cdef double lambda1 = lambda11
```

```
    cdef int niter = 0
    cdef double sigmasq, mu, z, shrink, shrink1, shrink2
    cdef int i, j, k

    print(f"Now you are running EM_DSS with model type: {model_type.upper()}")

    # EM algorithm iteration
    while eps > epsilon and niter < max_iter:
        if niter % 100 == 0 and niter > 0:
            print(f"Iteration {niter}: eps = {eps:.6f}")

        for i in range(T):
            for j in range(p):
                # Calculate z by removing element j
                z = y[i]
                for k in range(p):
                    if k != j:
                        z -= X[i, k] * beta[k, i]

                previous = beta[j, i - 1] if i > 0 else beta0[j]
                future = beta[j, i + 1] if i < T - 1 else phi0 + phi1 * beta[j, i]
                reg = X[i, j]

                # Calculate pstar and theta using auxiliary functions
                pstar[j, i] = pstar_cond(beta[j, i], previous, phi0, phi1, theta[j, i], lambda1, lambda0, spike_func, slab_func)

                # Update beta
                if i < T - 1:
                    sh = shrink_extra(beta[j, i], future, phi0, phi1, Theta, lambda1, lambda0, 1, spike_func, slab_func)
                    sigmasq = 1 / (reg**2 + pstar[j, i] / lambda1 + phi1**2 * pstar[j, i + 1] / lambda1 + (1 - phi1**2) / lambda1 * sh)
                    mu = sigmasq * (phi1 * pstar[j, i] * previous / lambda1 + phi1 * pstar[j, i + 1] * future / lambda1 + reg * z)
                    shrink1 = sigmasq * sh * lambda0
                    shrink2 = -sigmasq * sh * lambda0
                else:
                    sigmasq = 1 / (reg**2 + pstar[j, i] / lambda1)
                    mu = sigmasq * (phi1 * pstar[j, i] * previous / lambda1 + reg * z)
                    shrink1 = 0
                    shrink2 = 0

                shrink = sigmasq * (1 - pstar[j, i]) * lambda0
```

```
                beta[j, i] = 0

                if mu > (shrink - shrink1):
                    beta[j, i] = mu - shrink + shrink1
                elif mu < (-shrink - shrink2):
                    beta[j, i] = mu + shrink + shrink2
                theta[j, i] = p_star_stat(previous, phi0, phi1, Theta,
                    lambda1, lambda0, spike_func, slab_func)
        eps = 0
        for j in range(p):
            for i in range(T):
                eps += (beta_new[j, i] - beta[j, i])**2

        beta_new[:, :] = beta[:, :]
        niter += 1

    print(f"Total iterations: {niter}, eps = {eps:.6f}")
    return beta.T, theta.T, pstar.T
```