Whose Inflation Rates Matter Most?
A DSGE Model & ML Approach to Monetary Policy in the Euro Area

OeNB & SUERF Annual Economic Conference 2024

Daniel Stempel    Johannes Zahner

Heinrich-Heine-Universität Düsseldorf
Goethe University Frankfurt

June 2024
Table of Contents

1 Introduction

2 Combining DSGE and Machine Learning

3 Results

4 Conclusion
**Introduction**

**Inflation differentials**

- Euro area monetary policy is conducted uniformly for 20 member countries.
- How does the ECB react to deflationary and inflationary pressure in member states?
- Particularly relevant if countries deviate structurally from the euro area average inflation rate.

**Figure.** Source: ECB Website
Introduction

Inflation differentials

- Euro area monetary policy is conducted uniformly for 20 member countries.
- How does the ECB react to deflationary and inflationary pressure in member states?
- Particularly relevant if countries deviate structurally from the euro area average inflation rate.

Figure. Source: ECB Website
Introduction

Inflation differentials

- Euro area monetary policy is conducted uniformly for 20 member countries.
- How does the ECB react to deflationary and inflationary pressure in member states?
- Particularly relevant if countries deviate structurally from the euro area average inflation rate.

Figure. Source: ECB Website
**Introduction**

**Inflation Development in the Euro Area**

- EMU-members *structurally differ* in the volatility of their inflation rates.
  - Austria, Germany, the Netherlands → Low Volatility
  - Greece, Ireland, Italy, Portugal, and Spain → High Volatility

*Figure.* Average Inflation Deviations.
Introduction
Calculating the bias?

Whose inflation rates matter most for the ECB’s monetary policy?

Table. EMU Taylor Rule

<table>
<thead>
<tr>
<th>Dependent variable:</th>
<th>Interest Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(1)</td>
</tr>
<tr>
<td>HICP -2%</td>
<td>2.04***</td>
</tr>
<tr>
<td></td>
<td>(0.12)</td>
</tr>
<tr>
<td>Constant</td>
<td>−0.21</td>
</tr>
<tr>
<td></td>
<td>(0.16)</td>
</tr>
<tr>
<td>Weight on LV countries (ω) =</td>
<td>0</td>
</tr>
<tr>
<td>Observations</td>
<td>240</td>
</tr>
<tr>
<td>R²</td>
<td>0.56</td>
</tr>
</tbody>
</table>

Note: HICP is calculated as follows: \( HICP := \omega \times CPI_{LV} + (1 - \omega) \times CPI_{HV} \).

→ Which weight accurately describes historical EMU monetary policy?
**Introduction**

Calculating the bias?

Whose inflation rates matter most for the ECB’s monetary policy?

**Table. EMU Taylor Rule**

<table>
<thead>
<tr>
<th>Dependent variable:</th>
<th>Interest Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(1)</td>
</tr>
<tr>
<td>HICP -2%</td>
<td>2.04***</td>
</tr>
<tr>
<td></td>
<td>(0.12)</td>
</tr>
<tr>
<td>Constant</td>
<td>-0.21</td>
</tr>
<tr>
<td></td>
<td>(0.16)</td>
</tr>
<tr>
<td>Weight on LV countries ((\omega)) =</td>
<td>0</td>
</tr>
<tr>
<td>Observations</td>
<td>240</td>
</tr>
<tr>
<td>R(^2)</td>
<td>0.56</td>
</tr>
</tbody>
</table>

*Note: HICP is calculated as follows: \(HICP := \omega \times CPI_{LV} + (1 - \omega) \times CPI_{HV}\).*

→ Which weight accurately describes historical EMU monetary policy?
Whose inflation rates matter most for the ECB’s monetary policy?

### Table. EMU Taylor Rule

<table>
<thead>
<tr>
<th>Dependent variable:</th>
<th>Interest Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(1)</td>
</tr>
<tr>
<td>HICP -2%</td>
<td>2.04***</td>
</tr>
<tr>
<td></td>
<td>(0.12)</td>
</tr>
<tr>
<td>Constant</td>
<td>−0.21</td>
</tr>
<tr>
<td></td>
<td>(0.16)</td>
</tr>
<tr>
<td>Weight on LV countries ((\omega)) =</td>
<td>0</td>
</tr>
<tr>
<td>Observations</td>
<td>240</td>
</tr>
<tr>
<td>(R^2)</td>
<td>0.56</td>
</tr>
</tbody>
</table>

Note: HICP is calculated as follows: \(HICP := \omega \times CPI_{LV} + (1 - \omega) \times CPI_{HV}\).

→ Which weight accurately describes historical EMU monetary policy?
**Introduction**

**Paper overview**

Our "data-driven" solution:

1. Build a two-country *New Keynesian model* (NKM) of a monetary union with different central bank regimes.
2. Simulate NKM to generate a data for different policy regime.
3. Train machine learning models (ML) in classifying each regimes.
4. Use the trained ML to classify historical EMU data.

Results:

1. Distribution of the ECB’s historical inflation weight is biased.
2. ECB react more strongly to countries whose inflation rates exhibit larger deviations from their long-term trend.
**Introduction**

**Paper overview**

Our "data-driven" solution:

1. Build a two-country New Keynesian model (NKM) of a monetary union with different central bank regimes.
2. Simulate NKM to generate a data for different policy regime.
3. Train machine learning models (ML) in classifying each regimes.
4. Use the trained ML to classify historical EMU data.

**Results:**

1. Distribution of the ECB’s historical inflation weight is biased.
2. ECB react more strongly to countries whose inflation rates exhibit larger deviations from their long-term trend.
**INTRODUCTION**

**Paper overview**

Our "data-driven" solution:

1. Build a two-country New Keynesian model (NKM) of a monetary union with different central bank regimes.
2. Simulate NKM to generate a data for different policy regime.
3. Train machine learning models (ML) in classifying each regimes.
4. Use the trained ML to classify historical EMU data.

**Results:**

1. Distribution of the ECB’s historical inflation weight is biased.
2. ECB react more strongly to countries whose inflation rates exhibit larger deviations from their long-term trend.
Introduction

Paper overview

Our "data-driven" solution:

1. Build a two-country New Keynesian model (NKM) of a monetary union with different central bank regimes.
2. Simulate NKM to generate a data for different policy regime.
3. Train machine learning models (ML) in classifying each regimes.
4. Use the trained ML to classify historical EMU data.

Results:

1. Distribution of the ECB’s historical inflation weight is biased.
2. ECB react more strongly to countries whose inflation rates exhibit larger deviations from their long-term trend.
**Introduction**

**Paper overview**

Our "data-driven" solution:

1. Build a two-country New Keynesian model (NKM) of a monetary union with different central bank regimes.
2. Simulate NKM to generate a data for different policy regime.
3. Train machine learning models (ML) in classifying each regimes.
4. Use the trained ML to classify historical EMU data.

**Results:**

1. Distribution of the ECB’s historical inflation weight is biased.
2. ECB react more strongly to countries whose inflation rates exhibit larger deviations from their long-term trend.
Our "data-driven" solution:

1. Build a two-country New Keynesian model (NKM) of a monetary union with different central bank regimes.
2. Simulate NKM to generate a data for different policy regime.
3. Train machine learning models (ML) in classifying each regimes.
4. Use the trained ML to classify historical EMU data.

Results:

1. Distribution of the ECB’s historical inflation weight is biased.
2. ECB react more strongly to countries whose inflation rates exhibit larger deviations from their long-term trend.
# Table of Contents

1. **Introduction**

2. **Combining DSGE and Machine Learning**

3. **Results**

4. **Conclusion**
Combining DSGE and Machine Learning

- Simple currency union with 2 countries (HV and LV) with each a household and a firm sector
- Monetary Policy Targeting Rule:
  \[ i_t = \rho + \phi_\pi \left( \omega_\pi \pi^{c,HV}_t + (1 - \omega_\pi) \pi^{c,LV}_t \right) \]
  
  1. central bank reacts to the union-wide inflation rate: \( \omega_\pi \approx 0.5 \)
  2. central bank reacts more strongly to country HV: \( \omega_\pi = 0.8 \)
  3. central bank reacts more strongly to country LV: \( \omega_\pi = 0.2 \)
- Calibrate country LV (HV) to represent the LV (HV) EMU-members\(^1\)
- Simulate 3 × 10,000 periods of macro variables (\( C, L, \pi, \ldots \)) → Split train/test set: 80/20
- Train/Evaluate ML models on the simulated data (neural network outperforms the other models).
- Use neural network to predict \( \omega \) on historical EMU data.

Combining DSGE and Machine Learning

- Simple currency union with 2 countries (HV and LV) with each a household and a firm sector
- Monetary Policy Targeting Rule:

  \[ i_t = \rho + \phi_\pi \left( \omega_\pi \pi_t^{C,HV} + (1 - \omega_\pi) \pi_t^{C,LV} \right) \]

  1. central bank reacts to the union-wide inflation rate: \( \omega_\pi \approx 0.5 \)
  2. central bank reacts more strongly to country HV: \( \omega_\pi = 0.8 \)
  3. central bank reacts more strongly to country LV: \( \omega_\pi = 0.2 \)

- Calibrate country LV (HV) to represent the LV (HV) EMU-members\(^1\)
- Simulate 3 × 10,000 periods of macro variables (C, L, π, ...) → Split train/test set: 80/20
- Train/Evaluate ML models on the simulated data (neural network outperforms the other models).
- Use neural network to predict \( \omega \) on historical EMU data.

Combining DSGE and Machine Learning

- Simple currency union with 2 countries (HV and LV) with each a household and a firm sector
- Monetary Policy Targeting Rule:

\[ i_t = \rho + \phi_{\pi}(\omega_{\pi} \pi_{t}^{C,HV} + (1 - \omega_{\pi}) \pi_{t}^{C, LV}) \]

1. central bank reacts to the union-wide inflation rate: \( \omega_{\pi} \approx 0.5 \)
2. central bank reacts more strongly to country HV: \( \omega_{\pi} = 0.8 \)
3. central bank reacts more strongly to country LV: \( \omega_{\pi} = 0.2 \)

- Calibrate country LV (HV) to represent the LV (HV) EMU-members

- Simulate 3 \( \times \) 10,000 periods of macro variables (C, L, \( \pi \), ...) \( \rightarrow \) Split train/test set: 80/20
- Train/Evaluate ML models on the simulated data (neural network outperforms the other models).
- Use neural network to predict \( \omega \) on historical EMU data.

---


Combining DSGE and Machine Learning

- Simple currency union with 2 countries (HV and LV) with each a household and a firm sector
- Monetary Policy Targeting Rule:

\[ i_t = \rho + \phi_\pi \left( \omega_\pi \pi_t^{C,HV} + (1 - \omega_\pi) \pi_t^{C,LV} \right) \]

1. central bank reacts to the union-wide inflation rate: \( \omega_\pi \approx 0.5 \)
2. central bank reacts more strongly to country HV: \( \omega_\pi = 0.8 \)
3. central bank reacts more strongly to country LV: \( \omega_\pi = 0.2 \)
- Calibrate country LV (HV) to represent the LV (HV) EMU-members\(^1\)
- Simulate 3 x 10,000 periods of macro variables (C, L, \( \pi \), ...) → Split train/test set: 80/20
- Train/Evaluate ML models on the simulated data (neural network outperforms the other models).
- Use neural network to predict \( \omega \) on historical EMU data.

Combining DSGE and Machine Learning

- Simple currency union with 2 countries (HV and LV) with each a household and a firm sector
- Monetary Policy Targeting Rule:

\[ i_t = \rho + \phi_{\pi} (\omega_{\pi} \pi_t^{C,HV} + (1 - \omega_{\pi}) \pi_t^{C,LV}) \]

1. central bank reacts to the union-wide inflation rate: \( \omega_{\pi} \approx 0.5 \)
2. central bank reacts more strongly to country HV: \( \omega_{\pi} = 0.8 \)
3. central bank reacts more strongly to country LV: \( \omega_{\pi} = 0.2 \)
- Calibrate country LV (HV) to represent the LV (HV) EMU-members\(^1\)
- Simulate 3 \times 10,000 periods of macro variables (C, L, \pi, ...) \rightarrow Split train/test set: 80/20
- Train/Evaluate ML models on the simulated data (neural network outperforms the other models).
- Use neural network to predict \( \omega \) on historical EMU data.

----

Combining DSGE and Machine Learning

- Simple currency union with 2 countries ($HV$ and $LV$) with each a household and a firm sector
- Monetary Policy Targeting Rule:

\[ i_t = \rho + \phi_\pi \left( \omega_\pi \pi_t^{C,HV} + (1 - \omega_\pi) \pi_t^{C,LV} \right) \]

1. central bank reacts to the union-wide inflation rate: $\omega_\pi \approx 0.5$
2. central bank reacts more strongly to country $HV$: $\omega_\pi = 0.8$
3. central bank reacts more strongly to country $LV$: $\omega_\pi = 0.2$
- Calibrate country $LV$ ($HV$) to represent the $LV$ ($HV$) EMU-members\(^1\)
- Simulate $3 \times 10,000$ periods of macro variables ($C, L, \pi, ...$) → Split train/test set: 80/20
- Train/Evaluate ML models on the simulated data (neural network outperforms the other models).
- Use neural network to predict $\omega$ on historical EMU data.

Table of Contents

1 Introduction

2 Combining DSGE and Machine Learning

3 Results

4 Conclusion
**Results**

**Monetary Policy Regime Classifications**

1. Biased weight: disproportional emphasis (80%) on HV inflation rates
2. ECB is reacting more strongly to greater deviations of inflation rates from their long-term trend → potential explanation for 1.
**Results**

**Monetary Policy Regime Classifications**

1. **Biased weight**: disproportional emphasis (80%) on *HV* inflation rates

2. ECB is reacting more strongly to greater deviations of inflation rates from their long-term trend → potential explanation for 1.
**Results**

**Monetary Policy Regime Classifications**

1. **Biased weight**: disproportional emphasis (80%) on HV inflation rates
2. ECB is reacting more strongly to greater deviations of inflation rates from their long-term trend → potential explanation for 1.
RESULTS

ON THE ECB’S TAYLOR RULE AND LOSS FUNCTION

Standard central bank loss function:

\[ L_t = -\frac{1}{2} \left( \pi_t^{EMU} \right)^2 \]

where \( \pi_t^{EMU} \) is the EMU-wide inflation rate. The corresponding Taylor rule is given by:

\[ i_t = \rho + \phi_\pi \pi_t^{EMU} \]
RESULTS

ON THE ECB’S TAYLOR RULE AND LOSS FUNCTION

If ECB’s losses arise from individual deviations rather than from aggregated ones:

\[ L_t = -\frac{1}{2} \sum_{k=1}^{K} \omega^k \left( \pi^k_t \right)^2 \]

The interest rate rule becomes:

\[ i_t = \rho + \phi_\pi \left( \sum_{k=1}^{K} \Omega^k_t \pi^k_t \right) \]

\[ \Omega^k_t = \omega^k - \nu \left( |\pi^EMU_t| - |\pi^k_t| \right) \]

Example:
- \( HV \) inflation deviation is greater than \( LV \)’s (\( |\pi^HV_t| > |\pi^EMU_t| \))
- \( HV \) weight in the Taylor Rule exceeds the "true" \( HV \) weight: \( \Omega^HV_t > \omega^HV \).
**Results**

**Regression Model**

- Problem: We require weights in continuous space
- Adjustments:
  1. NKM: redefine the inflation weight: \( \Omega_\pi \in [0.1, 0.9] \)
  2. Simulate the NKM in 0.1 \( \Omega_\pi \) increments
  3. Regression NN
- Repeat Training and evaluation of NN
- (As expected:) **biased weight** (0.67) favors the high-volatility countries.

![Figure. Density Inflation Weight Prediction.](image-url)
**RESULTS**

**Regression Model**

- Problem: We require weights in continuous space
- Adjustments:
  1. NKM: redefine the inflation weight: \( \Omega_\pi \in [0.1, 0.9] \)
  2. Simulate the NKM in 0.1 \( \Omega_\pi \) increments
  3. Regression NN
- Repeat Training and evaluation of NN
- (As expected:) biased weight (0.67) favors the high-volatility countries.

![Figure. Density Inflation Weight Prediction.](image)
Test our hypothesis (greater weight on greater deviation) empirically.

- OLS regression:
  \[ \Omega_t^H = \beta_0 + \beta_1 (|\pi_t^{EMU}| - |\pi_t^L|) + \epsilon_t \]

- \( \beta_0 \) can be interpreted as the true weight on HV countries \( \omega^H \)
- \( \beta_1 \) can be interpreted as \( \nu \) (reaction parameter on deviations from EMU inflation)
- Expectation: \( \beta_0 \approx 0.5 \) and \( \beta_1 > 0 \) \( (|\pi^L| \uparrow \rightarrow \downarrow \rightarrow \Omega \downarrow) \)
Test our hypothesis (greater weight on greater deviation) empirically.

- **OLS regression:**

\[ \Omega_t^H = \beta_0 + \beta_1 (|\pi_t^{EMU}| - |\pi_t^L|) + \epsilon_t \]

- \( \beta_0 \) can be interpreted as the true weight on HV countries \( \omega_t^H \)
- \( \beta_1 \) can be interpreted as \( \nu \) (reaction parameter on deviations from EMU inflation)
- Expectation: \( \beta_0 \approx 0.5 \) and \( \beta_1 > 0 \) (\(|\pi_t^L| \uparrow \rightarrow \ldots \downarrow \rightarrow \Omega \downarrow\))
Results
Regression Model

▶ Test our hypothesis (greater weight on greater deviation) empirically.
▶ OLS regression:
\[ \Omega_t^H = \beta_0 + \beta_1 (|\pi_t^{EMU}| - |\pi_t^L|) + \epsilon_t \]
▶ \(\beta_0\) can be interpreted as the true weight on HV countries \(\omega^H\)
▶ \(\beta_1\) can be interpreted as \(\nu\) (reaction parameter on deviations from EMU inflation)
▶ Expectation: \(\beta_0 \approx 0.5\) and \(\beta_1 > 0\) \(|\pi^L| \uparrow \rightarrow (.) \downarrow \rightarrow \Omega \downarrow\)
Test our hypothesis (greater weight on greater deviation) empirically.

- **OLS regression:**
  \[
  \Omega_t^H = \beta_0 + \beta_1 (|\pi_t^{EMU}| - |\pi_t^L|) + \epsilon_t
  \]

- **\(\beta_0\)** can be interpreted as the **true weight on HV countries** \(\omega^H\)

- **\(\beta_1\)** can be interpreted as \(\nu\) (**reaction parameter** on deviations from EMU inflation)

- Expectation: \(\beta_0 \approx 0.5\) and \(\beta_1 > 0\) \((|\pi_t^L| \uparrow \rightarrow (. \downarrow) \rightarrow \Omega \downarrow)\)
Test our hypothesis (greater weight on greater deviation) empirically.

OLS regression:

\[
\Omega_t^H = \beta_0 + \beta_1 (|\pi_{t}^{EMU}| - |\pi_{t}^{L}|) + \epsilon_t
\]

- \(\beta_0\) can be interpreted as the true weight on HV countries \(\omega^H\)
- \(\beta_1\) can be interpreted as \(\nu\) (reaction parameter on deviations from EMU inflation)
- Expectation: \(\beta_0 \approx 0.5\) and \(\beta_1 > 0\) (\(|\pi^L| \uparrow \rightarrow \cdot \downarrow \rightarrow \Omega \downarrow\))
## Results

### Regression Model

**Table.** Main Regression Results.

<table>
<thead>
<tr>
<th></th>
<th>Column (1)</th>
<th>Column (2)</th>
<th>Column (3)</th>
<th>Column (4)</th>
<th>Column (5)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dependent variable:</strong> Inflation weight $: \Omega_t$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HICP ($= \nu$)</td>
<td>25.09***</td>
<td>24.06**</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(9.41)</td>
<td>(9.56)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>3.23**</td>
<td>3.59**</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1.36)</td>
<td>(1.44)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>−1.83</td>
<td>−2.90</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2.60)</td>
<td>(2.73)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>8.95</td>
<td>7.44</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(6.51)</td>
<td>(6.29)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Constant ($= \omega_k$)</td>
<td>0.62***</td>
<td>0.62***</td>
<td>0.64***</td>
<td>0.63***</td>
<td>0.62***</td>
</tr>
<tr>
<td></td>
<td>(0.02)</td>
<td>(0.02)</td>
<td>(0.02)</td>
<td>(0.02)</td>
<td>(0.02)</td>
</tr>
<tr>
<td>Observations</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.09</td>
<td>0.08</td>
<td>0.01</td>
<td>0.03</td>
<td>0.21</td>
</tr>
<tr>
<td>Adjusted $R^2$</td>
<td>0.08</td>
<td>0.06</td>
<td>−0.01</td>
<td>0.01</td>
<td>0.16</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS

1 Introduction

2 Combining DSGE and Machine Learning

3 Results

4 Conclusion
CONCLUSION

- We investigate whose inflation rates matter most for ECB’s monetary policy.
- Theoretical model with different monetary policy rules as data-generating process
- Train machine learning model to separate rules
- Use machine learning model to classify historical EMU data between 2004 and 2021.
- Findings:
  1. Disproportional emphasis on high volatility countries
  2. Stronger reaction to countries whose inflation rates exhibit larger deviations from their long-term trend
ROBUSTNESS TESTS

1. Model Extension: Investment and Capital
2. Adjustment of Taylor Parameter
3. Inclusion of ECB Board Composition
4. Use of Inflation Expectations

→ No change in findings.

Current work:

▶ NKM estimation
▶ ...
Literature

▶ **ML in monetary policy** (Tiffin, 2019; Hinterlang, 2020; Hinterlang and Hollmayr, 2021; Paranhos, 2021; Doerr et al., 2021; Fouliard et al., 2021)

▶ Assessment of inflation differentials within New Keynesian models (Canzoneri et al., 2006; Angeloni and Ehrmann, 2007; Andres et al., 2008; Duarte and Wolman, 2008; Rabanal, 2009; Neyer and Stempel, 2022)
we propose a modification to the Pagan frontier by combining DSGE and machine learning models to study inflation dynamics in the EMU.
## Calibration

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>H</td>
</tr>
<tr>
<td><strong>Households</strong></td>
<td></td>
</tr>
<tr>
<td>Habit parameter Ψ_k</td>
<td>0.77</td>
</tr>
<tr>
<td>Inverse Frisch elasticity φ_k</td>
<td>2.01</td>
</tr>
<tr>
<td>Preference shock strength η^Z_k</td>
<td>1</td>
</tr>
<tr>
<td>Weight of domestic goods γ_k</td>
<td>0.75</td>
</tr>
<tr>
<td>Elasticity of substitution θ^C_k</td>
<td>1.42</td>
</tr>
<tr>
<td>Cost-push shock strength η^A_k</td>
<td>1</td>
</tr>
<tr>
<td>Calvo parameter λ_k</td>
<td>0.737</td>
</tr>
<tr>
<td>Price elasticity of demand ε</td>
<td>6</td>
</tr>
<tr>
<td>Discount rate β</td>
<td>0.995</td>
</tr>
<tr>
<td><strong>Firms</strong></td>
<td></td>
</tr>
<tr>
<td>Output elasticity labor α_k</td>
<td>0.33</td>
</tr>
<tr>
<td>Cost-push shock strength η^A_k</td>
<td>1</td>
</tr>
<tr>
<td>Calvo parameter λ_k</td>
<td>0.737</td>
</tr>
<tr>
<td><strong>Central Bank</strong></td>
<td></td>
</tr>
<tr>
<td>Taylor rule coefficient φ_π</td>
<td>1.5; 2.5</td>
</tr>
<tr>
<td>HICP inflation weight ω_π</td>
<td>( \frac{c^H_{SS}}{c^H_{SS}+c^L_{SS}}; [0.1, 0.9] )</td>
</tr>
</tbody>
</table>
Historical EMU Data

- Data: Quarterly consumption, employment, output and price level → consumption weighted
- EMU wide interest rate → MRO + Wu and Xia (2020) shadow rate
- NKM reports percentage deviations from steady state → Hamilton (2018) filter to extract the cyclical component

→ Classification of historical inflation weight on a quarterly basis between 2004Q4 and 2022Q1.
We compare the performance of several algorithms in a horserace-style assessment.

All models have the following structure where \( y \in (\omega_H, \omega_L, \omega_C) \) and \( X \in (Y, C, \pi, ...) \):

\[
y_t = h_\beta(X_t) + \epsilon_t
\]

Accuracy of models is assessed out-of-sample.

The NN outperforms the other models by quite a margin.

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed guess</td>
<td>0.33</td>
</tr>
<tr>
<td>MLR</td>
<td>0.34</td>
</tr>
<tr>
<td>Ridge regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Lasso regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.33</td>
</tr>
<tr>
<td>K-nearest-neighbor</td>
<td>0.38</td>
</tr>
<tr>
<td>Decision tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Random forest</td>
<td>0.67</td>
</tr>
<tr>
<td>Neural network</td>
<td>0.97</td>
</tr>
</tbody>
</table>

Next: Use NN on historical EMU data.
We compare the performance of several algorithms in a horserace-style assessment.

All models have the following structure where \( y \in (\omega_H, \omega_L, \omega_C) \) and \( X \in (Y, C, \pi, ...) \):

\[
y_t = h_\beta(X_t) + \epsilon_t
\]

Accuracy of models is assessed out-of-sample.

The NN outperforms the other models by quite a margin.

---

Table. Out-of-sample evaluation.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed guess</td>
<td>0.33</td>
</tr>
<tr>
<td>MLR</td>
<td>0.34</td>
</tr>
<tr>
<td>Ridge regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Lasso regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.33</td>
</tr>
<tr>
<td>K-nearest-neighbor</td>
<td>0.38</td>
</tr>
<tr>
<td>Decision tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Random forest</td>
<td>0.67</td>
</tr>
<tr>
<td>Neural network</td>
<td>0.97</td>
</tr>
</tbody>
</table>

---

Next: Use NN on historical EMU data.
We compare the performance of several algorithms in a horserace-style assessment.

All models have the following structure where \( y \in (\omega_H, \omega_L, \omega_C) \) and \( X \in (Y, C, \pi, \ldots) \):

\[
y_t = h\beta(X_t) + \epsilon_t
\]

Accuracy of models is assessed out-of-sample.

The NN outperforms the other models by quite a margin.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed guess</td>
<td>0.33</td>
</tr>
<tr>
<td>MLR</td>
<td>0.34</td>
</tr>
<tr>
<td>Ridge regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Lasso regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.33</td>
</tr>
<tr>
<td>K-nearest-neighbor</td>
<td>0.38</td>
</tr>
<tr>
<td>Decision tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Random forest</td>
<td>0.67</td>
</tr>
<tr>
<td>Neural network</td>
<td>0.97</td>
</tr>
</tbody>
</table>

Next: Use NN on historical EMU data.
We compare the performance of several algorithms in a horserace-style assessment.

All models have the following structure where $y \in (\omega_H, \omega_L, \omega_C)$ and $X \in (Y, C, \pi, ...)$:

$$y_t = h_\beta(X_t) + \epsilon_t$$

Accuracy of models is assessed out-of-sample.

The NN outperforms the other models by quite a margin.

### Table. Out-of-sample evaluation.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed guess</td>
<td>0.33</td>
</tr>
<tr>
<td>MLR</td>
<td>0.34</td>
</tr>
<tr>
<td>Ridge regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Lasso regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.33</td>
</tr>
<tr>
<td>K-nearest-neighbor</td>
<td>0.38</td>
</tr>
<tr>
<td>Decision tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Random forest</td>
<td>0.67</td>
</tr>
<tr>
<td>Neural network</td>
<td>0.97</td>
</tr>
</tbody>
</table>
We compare the performance of several algorithms in a horserace-style assessment.

All models have the following structure where \( y \in (\omega_H, \omega_L, \omega_C) \) and \( X \in (Y, C, \pi, ...) \):

\[
y_t = h_\beta(X_t) + \epsilon_t
\]

Accuracy of models is assessed out-of-sample.

The NN outperforms the other models by quite a margin.

Next: Use NN on historical EMU data

---

**Table.** Out-of-sample evaluation.

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uninformed guess</td>
<td>0.33</td>
</tr>
<tr>
<td>MLR</td>
<td>0.34</td>
</tr>
<tr>
<td>Ridge regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Lasso regression</td>
<td>0.33</td>
</tr>
<tr>
<td>Elastic net</td>
<td>0.33</td>
</tr>
<tr>
<td>K-nearest-neighbor</td>
<td>0.38</td>
</tr>
<tr>
<td>Decision tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Prune complex tree</td>
<td>0.48</td>
</tr>
<tr>
<td>Random forest</td>
<td>0.67</td>
</tr>
<tr>
<td>Neural network</td>
<td>0.97</td>
</tr>
</tbody>
</table>
EMU time series

Figure. Hamilton-Filtered Data.
### Model fit

**Table.** Comparison of Simulated Moments with Data.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>$\omega_\pi = \frac{C_{SS}^H}{C_{SS}^H + C_{SS}^L}$</th>
<th>$\omega_\pi = 0.8$</th>
<th>$\omega_\pi = 0.2$</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{SS}^H / C_{SS}^L$</td>
<td>Relative consumption per capita $H$, $L$</td>
<td>$0.962$</td>
<td>$0.962$</td>
<td>$0.962$</td>
<td>$0.805$</td>
</tr>
<tr>
<td>$Y_{H,SS} / Y_{L,SS}$</td>
<td>Relative GDP per capita $H$, $L$</td>
<td>$0.980$</td>
<td>$0.980$</td>
<td>$0.980$</td>
<td>$0.773$</td>
</tr>
<tr>
<td>$\sigma (\hat{y}<em>{L,i}) / \sigma (\hat{y}</em>{H,i})$</td>
<td>Relative volatility GDP $L$, $H$</td>
<td>$0.779$</td>
<td>$0.773$</td>
<td>$0.783$</td>
<td>$0.587$</td>
</tr>
<tr>
<td>$\sigma (\hat{y}<em>{L,i}) / \sigma (\hat{y}</em>{H,i})$</td>
<td>Relative volatility union-wide GDP, $H$</td>
<td>$0.857$</td>
<td>$0.888$</td>
<td>$0.862$</td>
<td>$0.671$</td>
</tr>
<tr>
<td>$\sigma (\hat{y}<em>{L,i}) / \sigma (\hat{y}</em>{H,i})$</td>
<td>Relative volatility union-wide GDP, $L$</td>
<td>$1.010$</td>
<td>$1.149$</td>
<td>$1.010$</td>
<td>$1.144$</td>
</tr>
<tr>
<td>$\sigma (\hat{c}<em>{L,i}) / \sigma (\hat{c}</em>{H,i})$</td>
<td>Relative volatility consumption $L$, $H$</td>
<td>$0.152$</td>
<td>$0.149$</td>
<td>$0.158$</td>
<td>$0.559$</td>
</tr>
<tr>
<td>$\sigma (\hat{h}<em>{L,i}) / \sigma (\hat{h}</em>{H,i})$</td>
<td>Relative volatility labor $L$, $H$</td>
<td>$0.779$</td>
<td>$0.773$</td>
<td>$0.783$</td>
<td>$0.718$</td>
</tr>
<tr>
<td>$\sigma (\hat{a}<em>{L,i}) / \sigma (\hat{a}</em>{H,i})$</td>
<td>Relative volatility inflation $L$, $H$</td>
<td>$0.913$</td>
<td>$0.921$</td>
<td>$0.904$</td>
<td>$0.842$</td>
</tr>
<tr>
<td>$\rho (\hat{y}<em>{L,i}, \hat{y}</em>{H,i})$</td>
<td>Correlation GDP $L$, $H$</td>
<td>$0.859$</td>
<td>$0.844$</td>
<td>$0.871$</td>
<td>$0.591$</td>
</tr>
<tr>
<td>$\rho (\hat{y}<em>{C,i}, \hat{y}</em>{C,i})$</td>
<td>Correlation inflation $L$, $H$</td>
<td>$0.931$</td>
<td>$0.990$</td>
<td>$0.991$</td>
<td>$0.989$</td>
</tr>
<tr>
<td>$\rho (\hat{c}<em>{L,i}, \hat{c}</em>{H,i})$</td>
<td>Correlation consumption $L$, $H$</td>
<td>$0.603$</td>
<td>$0.536$</td>
<td>$0.640$</td>
<td>$0.636$</td>
</tr>
<tr>
<td>$\rho (\hat{a}<em>{L,i}, \hat{a}</em>{H,i})$</td>
<td>Correlation labor $L$, $H$</td>
<td>$0.859$</td>
<td>$0.844$</td>
<td>$0.871$</td>
<td>$0.132$</td>
</tr>
<tr>
<td>$\rho (\hat{a}<em>{H,i}, \hat{c}</em>{H,i})$</td>
<td>Correlation labor, consumption $H$</td>
<td>$0.943$</td>
<td>$0.942$</td>
<td>$0.944$</td>
<td>$0.627$</td>
</tr>
<tr>
<td>$\rho (\hat{a}<em>{L,i}, \hat{c}</em>{L,i})$</td>
<td>Correlation labor, consumption $L$</td>
<td>$0.482$</td>
<td>$0.437$</td>
<td>$0.513$</td>
<td>$0.466$</td>
</tr>
</tbody>
</table>

*Note:* $\hat{x}_i$ denotes the deviation of a variable $X$ from its zero inflation steady state.
Neural networks in a nutshell II

- A neural network consists of $i \in I$ layers, with each $k$ perceptrons.
- The input for to layer:

$$X_i = f(W_i \times X_{i-1} + b_i)$$

- Two activation functions $f(\cdot)$ in this paper:

$$f(x) = \max(0, x) \quad \text{ReLu}$$

$$f(x) = \frac{e^{x_k}}{\sum_{k=1}^{K} e^{x_k}} \quad \text{Softmax}$$

- Training process: optimize $W_i$ and $b_i$
Regression Model III

Figure. Inflation Weight from Regression NN 2004Q4 - 2022Q1.

→ Stronger deviations $L$ coincide with periods of higher ($L$) weight, e.g. 2011, 2017/18 and vice versa.
**Neural networks in a nutshell**

Figure. Illustration of a Neural Network.
Neural networks in a nutshell

Figure. Illustration of a Logistic Regression.
Neural networks in a nutshell

Input layer

\( \pi_L \)

\( \pi_H \)

\( C_L \)

\( i_{EMU} \)

Output layer

\( \omega_H \)

\( \omega_L \)

\( \omega_C \)

Figure. Illustration of a Multinomial Logistic Regression.
Neural networks in a nutshell

Figure. Illustration of a Neural Network.

Notes: This figure illustrates the model architecture of a feed-forward NN with four layers: One input layer, two hidden layers, and an output layer. The connections between the layers represent the weighting matrix $W_i$ and are adjusted during the training process.
**Evaluation II**

**Table.** Confusion matrix of out-of-sample prediction by NN

<table>
<thead>
<tr>
<th>Prediction</th>
<th>True label</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Neutral</td>
<td>LV</td>
<td>HV</td>
<td></td>
</tr>
<tr>
<td>Neutral</td>
<td>2405</td>
<td>50</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>LV</td>
<td>48</td>
<td>2442</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>HV</td>
<td>47</td>
<td>7</td>
<td>2452</td>
<td></td>
</tr>
</tbody>
</table>

→ neural network does not suffer from biased predictions