VELA: NONLINEAR SOLUTIONS TO DSGE MODELS VIA MULTI-AGENT REINFORCEMENT LEARNING IN JAX

Anonymous authors
Paper under double-blind review

ABSTRACT

Dynamic stochastic general equilibrium (DSGE) models are widely used by macroeconomic institutions for policy-making and forecasting. However, traditional approaches often rely on log-linearization around a pre-specified steady state, yielding linear approximations whose validity is limited to its neighborhood. We present Vela, a JAX-based framework for defining and solving DSGE models through parallel multi-agent reinforcement learning. We implement the Kydland–Prescott RBC and Smets–Wouters DSGE models in Vela. Using OECD data for ten G20 economies from 1960 to 2026, we estimate empirical impulse responses through vector autoregression (VAR) and compare their directions with model predictions. Solutions learned via multi-agent self-play achieve higher accuracy than a conventionally solved Smets–Wouters baseline in eight of the ten G20 economies studied, with statistically significant improvements. The learned policies capture nonlinear dynamics, including interactions between simultaneous shocks, and can be meta-trained across a continuous range of economic calibration parameters. We also examine the design choices that affect learned policy quality and compare the resulting solutions with conventional DSGE solution methods and empirical data.

1 INTRODUCTION

Dynamic stochastic general equilibrium (DSGE) models are key tools used by central banks and other policymakers to assess how decisions, such as changes in interest rates or taxes, affect inflation, output, employment, and other macroeconomic indicators. By combining households' and firms' expectations, choices, and economic constraints with empirically calibrated or estimated parameters, these models provide a structured and theoretically sound framework for comparing policy options and evaluating their consequences.

A widely used approach to solving DSGE models linearizes their equilibrium conditions around a steady state (Fernández-Villaverde et al., 2016). Such approximations work best near that state and may miss important effects of large or simultaneous shocks. Nonlinear solutions can better capture binding constraints and asymmetric responses, but developing accurate and scalable global solution methods remains an active research challenge (Cao et al., 2023; Maliar et al., 2021).

In parallel, multi-agent reinforcement learning (MARL) has a long history of studying sequential interaction among agents with distinct objectives (Littman, 1994; Hu & Wellman, 2003), a setting naturally suited to households, firms, and policymakers in DSGE models. A prominent approach is self-play, including learning against mixtures of opponents' past policies through fictitious play (Brown, 1951; Heinrich et al., 2015; Heinrich & Silver, 2016). The practical methods developed by the MARL community have enabled landmark results in complex games such as Go (Silver et al., 2017; Schrittwieser et al., 2020), poker (Brown et al., 2020), Dota 2 (Berner et al., 2019), and StarCraft II (Vinyals et al., 2019).

Motivated by these advances, we formulate DSGE models as general-sum partially observable stochastic games and investigate MARL as an approach to learning nonlinear decision rules and approximating equilibrium behavior.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Our key contributions are:

1. We introduce and open-source Vela, a JAX-based framework providing modeling abstractions and computational primitives for formulating DSGE models as MARL environments, together with RBC (Kydland & Prescott, 1982) and Smets Wouters (Smets & Wouters, 2003) implementations, checkpoints, and evaluation pipelines anonymous.4open.science/r/vela-anon-DBC5.

2. To our knowledge, we provide the first MARL-based solution approach for a medium-scale DSGE model, demonstrated on the Smets Wouters model.

3. We experimentally analyze the learned nonlinear responses of the Smets Wouters model and compare them with empirical estimates and conventional solutions, finding significant gains in average empirical response-direction accuracy in eight of ten economies studied.

2 Related Work

2.1 Macroeconomic Models

Modern quantitative macroeconomics has developed as a sequence of increasingly rich, microfounded (that is, framed as a set of agents' optimization problems) frameworks. Following the Lucas critique (Lucas Jr, 1976), real business cycle (RBC) models explain aggregate fluctuations through optimizing households and firms facing productivity shocks (Kydland & Prescott, 1982; King et al., 1988). New Keynesian dynamic stochastic general equilibrium (DSGE) models added a richer set of shocks (Christiano et al., 2005; Smets & Wouters, 2003; 2007) and, notably after the 2008 crisis, financial frictions (Christiano et al., 2018). A parallel tradition, agent-based models (ABMs), is based on large-scale simulations of populations of interacting, boundedly rational agents (Tesfatsion, 2006; Dawid & Gatti, 2018). Despite the theoretical appeal of this approach and the rapid development of computational techniques, ABMs have largely evolved independently from the more classical DSGE literature (Baccini et al., 2025). Our paper belongs to the DSGE line of work: we retain a microfounded structure but let agents learn their decision rules through multi-agent reinforcement learning. This enables nonlinear solution estimation while borrowing from the ABM tradition the idea of adaptive agents that interact through the model environment.

2.2 Multi-Agent Reinforcement Learning

A wide range of methods has been proposed to address different forms of multi-agent interaction (Huh & Mohapatra, 2024). These methods differ in both their training architecture and assumed reward structure. Fully centralized approaches learn a joint policy for all agents, while independent learners apply single-agent algorithms to each agent, treating the other agents as part of its environment (Claus & Boutilier, 1998; Tan et al., 1993; De Witt et al., 2020). Centralized training with decentralized execution (CTDE) combines elements of both paradigms by providing access to additional information during training while restricting each agent's policy to local observations during execution (Lowe et al., 2020; Yu et al., 2022). A separate classification axis concerns assumptions about the game's underlying reward structure: Sunehag et al. (2017); Rashid et al. (2018); Wang et al. (2021) assume a common reward, whereas methods such as those of Lowe et al. (2020); De Witt et al. (2020) can accommodate individual rewards in general-sum settings. A related line of work addresses instability and policy cycling during multi-agent learning. Building on fictitious play (Brown, 1951), FSP (Heinrich et al., 2015) learns approximate best responses to opponents' average policies, while NFSP (Heinrich & Silver, 2016) augments this framework with neural function approximation. PSRO (Lanctot et al., 2017) maintains explicit policy populations and selects opponents through a meta-strategy solver. AlphaStar (Vinyals et al., 2019) further combines historical policies, prioritized matchmaking, and specialized exploiters within a league-based training framework. Drawing on these methods, this work formulates DSGE models as general-sum partially observable stochastic games and learns approximate solutions using IPPO (De Witt et al., 2020), with self-play implemented via a PSRO variant (Lanctot et al., 2017) that uses a uniform meta-solver and finite archives of historical policies.

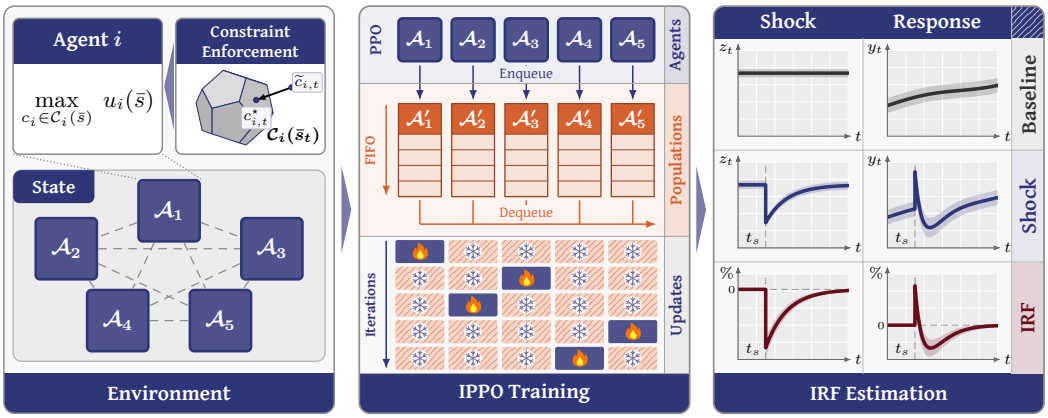


Figure 1: Vela overview. Left: multi-agent environment specification (illustrative five-agent setup) with individual rewards and state-dependent convex constraints. Center: IPPO training with self-play against FIFO archives of historical policies. Right: IRF estimation via parallel paired baseline and shocked simulations.

2.3 Reinforcement Learning for Macroeconomic Modeling

Over the last few years, we have witnessed a rise in applications of reinforcement learning to macroeconomic modeling: Atashbar & Shi (2023) describes solving RBC-type model in style of Kydland & Prescott (1982), Hinterlang & Tänzer (2021) finds optimal monetary policy rules for a RL-based central bank in simulated economy, Brusatin et al. (2024) combines RL with agent-based modeling. Recent surveys describe the field as promising but not yet mature (Wang, 2026; Rawat, 2026) – solving medium-scale DSGE such as the Smets-Wouters model with MARL is one of the gaps that still remain under explored.

2.4 Hardware-Accelerated Environments

The release of the JAX framework (Bradbury et al., 2018) facilitated the development of highly parallel hardware-accelerated environments. Engineering advantages include hosting both the agent and the environment on the same GPU or TPU, native vectorization with `vmap`, and JIT compilation of an entire training loop (Rutherford et al., 2026). From a training perspective, parallelization enables high-throughput experience collection and supports large-batch training in online reinforcement learning (Berner et al., 2019; Petrenko et al., 2020; Mayor et al., 2025). Benefits of large optimization batches have also been demonstrated in deep learning (You et al., 2020) and offline reinforcement learning (Nikulin et al., 2023; Tarasov et al., 2023). The advantages outlined above have fueled the rapid expansion of simulation frameworks and reinforcement learning environments built on JAX across multiple domains: robotics and physics-based control (Freeman et al., 2021; Matthews et al., 2025), grid-world navigation and exploration (Nikulin et al., 2024; Pignatelli et al., 2024; Matthews et al., 2024), economic and financial simulation (Ponse et al., 2025; Mohl et al., 2025), agent-based modeling (Chaturvedi et al., 2025), and other domains (Lange, 2026; Koyamada et al., 2024; Bonnet et al., 2024). We implement our macroeconomic environments using an interface adapted from the JaxMARL environment API (Rutherford et al., 2026). In our experiments, we run 16,384 environment instances in parallel on a single H100 GPU, supporting rollout collection against populations of historical policies and domain randomization over economic calibration parameters.

3 Method

Vela formulates macroeconomic modeling as a multi-agent reinforcement learning problem through three stages: translating conventional DSGE models into vectorizable JAX environments with heterogeneous agents and dynamic constraint enforcement; training agents using IPPO against populations of past agent policies; and evaluating macroeconomic responses to exogenous structural shocks

through parallel paired simulations with common random numbers. Figure 1 presents a schematic overview of Vela.

3.1 ENVIRONMENT

Vela adapts the JaxMARL environment API (Rutherford et al., 2026) to represent DSGE models as environments with a shared Markovian economic state s_t, N heterogeneous agents, and nonlinear stochastic dynamics parameterized by economic calibration parameters θ.

Heterogeneous Agents. Each agent $\mathcal{A}_i$ is defined by $(\mathcal{U}_i, h_i, g_i, \mathcal{C}_i, u_i)$: its policy-action space $\mathcal{U}_i = [-1, 1]^{d_i}$, local observation mask h_i, controller g_i, state-dependent polyhedral feasible set $\mathcal{C}_i$, and individual nonlinear economic objective u_i. At each step, agents receive local observations $o_{i,t} = h_i(s_t)$ and produce policy actions $a_{i,t}$, which the controller maps to proposed economic controls $\widetilde{c}_{i,t} = g_i(c_{i,t-1}, a_{i,t})$. We use a residual additive controller that clips actions to $\mathcal{U}_i$, scales them, and adds them to the previous controls $c_{i,t-1}$ stored in s_t.

Constraint Enforcement. Each agent's feasible control set is $\mathcal{C}_i(\widetilde{s}_t) = \{c_i : A_i(\widetilde{s}_t)c_i \leq b_i(\widetilde{s}_t)\}$, where $\widetilde{s}_t$ is the shared state updated with all agents' proposed controls. Constraints may depend on this state but must be affine in the agent's own controls c_i. For example, a household budget constraint limits spending to available income. Infeasible proposals $\widetilde{c}_{i,t}$ are numerically projected to the closest feasible point $c^\star_{i,t}$ in Euclidean distance. Projection uses a JIT-compatible OSQP solver (Stellato et al., 2020; Blondel et al., 2021), with a penalty-based L-BFGS fallback (Liu & Nocedal, 1989; DeepMind et al., 2020) when residual constraint violations exceed a tolerance ϵ.

Rewards and State Transitions. After constraint enforcement, corrected controls are incorporated into the shared state $\bar{s}_t$. Each agent's reward combines its ego-centric utility $u_i(\bar{s}_t)$ with penalties for control correction, measured by $\|c^\star_{i,t} - \widetilde{c}_{i,t}\|_1$, and action magnitude, measured by the mean absolute value of its policy-action components. We empirically find that the action cost mitigates bang-bang behavior (Seyde et al., 2021) during training; it also admits an economic interpretation as an adjustment cost, analogous to price-adjustment costs (Rotemberg, 1982). The state then evolves according to $s_{t+1} = F(\bar{s}_t, \xi_{t+1}; \theta)$, where ξ_{t+1} denotes random innovations representing economic shocks and other stochastic disturbances. Rollout pseudocode appears in section A.1, and formal model definitions are provided in section C.

Agent Definitions in Vela. States and controls use JAX-compatible `flax.struct.dataclass` structures, while utilities are JIT-compatible scalar functions of state and controls. A constraint decorator supports Python definitions of affine inequalities of the form $Ax - b \leq 0$. A simplified household definition for the RBC model (Kydland & Prescott, 1982) appears in section B.1. Successful and unsuccessful environment definition practices are discussed in section D.

3.2 TRAINING

Following IPPO (De Witt et al., 2020), we train separate actor $\pi_{\psi_i}(o_{i,t})$ and critic $V_{\phi_i}(o_{i,t})$ networks for each agent using individual rewards, consistent with agent-specific objectives in DSGE models. Our PPO update (Schulman et al., 2017) with generalized advantage estimation (GAE) (Schulman et al., 2018) adapts the PureJaxRL implementation (Lu et al., 2022). Self-play uses a PSRO variant (Lanctot et al., 2017) with a uniform meta-solver and a finite FIFO policy archive per agent (16 entries in our experiments). Training alternates between agents, with each learning agent facing uniformly weighted opponent archives across parallel environments. At the end of each agent's training block, its policy is archived. For symmetric roles, such as competing intermediate-goods producers in the Smets–Wouters model, its network parameters and optimizer state are also copied to the corresponding agents. We normalize observations using running statistics updated via an exponential moving average (EMA). Detailed hyperparameter settings are provided in section G, ablations of key design choices in sections E.1 and E.2, and training pseudocode in algorithm 2.

Throughout training process, we randomly sample model-specific economic shocks within the stochastic state transition function. For the Smets Wouters model, these include productivity, labor

preference, investment efficiency, money preference, and intertemporal preference shocks (Smets & Wouters, 2003). This enables agents to learn responses to economic disturbances while maximizing their individual expected discounted returns.

Conventional DSGE solutions are typically computed for a fixed economic calibration. Taking advantage of Vela's parallel simulation capabilities, we perform extensive domain randomization over calibration parameters during training, yielding policies that operate across economic settings. Results in section 4.4 show that this procedure improves agreement with empirical data.

3.3 Inference

DSGE models can be evaluated by introducing controlled exogenous shocks and analyzing the subsequent responses of economic variables. These responses are described by impulse response functions (IRFs) and traditionally expressed as deviations from the model's steady state. Using JAX's explicit random-key management to control randomness, Vela implements parallel paired simulations with common random numbers (Glasserman & Yao, 1992). We initialize two vectorized batches of M DSGE environments with matching random keys w_m and initial conditions and switch IPPO policies to deterministic mode. Paired trajectories therefore differ only through the imposed exogenous shock and its subsequent effects. Denoting the baseline and shocked trajectories of economic variable j by $y_{j,t}^{\text{base}}(w_m)$ and $y_{j,t}^{\text{shock}}(w_m)$, respectively, we calculate relative IRFs as

$$\widehat{\Delta}_{j,t} = \frac{1}{M} \sum_{m=1}^{M} \frac{y_{j,t}^{\text{shock}}(w_m) - y_{j,t}^{\text{base}}(w_m)}{y_{j,t}^{\text{base}}(w_m)}.$$

Cross-calibration training via domain randomization offers several advantages during evaluation. First, it allows varying the calibration vector θ at inference without retraining the model (see section 4.5). Second, parallel simulations across a range of calibrations can yield IRF uncertainty bands that incorporate calibration uncertainty.

Table 1: Difference in empirical directional accuracy between the RL model and the classically solved Smets–Wouters (2002) model under upward shocks. Values are percentage-point differences (RL minus DSGE); brackets show 95% confidence intervals. Bold cells indicate positive accuracy differences.

Country	Productivity shock (pp)	Labor preference shock (pp)	Investment shock (pp)	Money preference shock (pp)	Average (pp)
Türkiye	**+33.33 [+29.0, +36.9]**	−32.14 [−36.8, −27.0]	−9.23 [−14.4, −3.9]	**+1.79 [−3.6, +7.1]**	−1.56 [−4.2, +1.1]
Italy	**+33.33 [+29.0, +36.9]**	−6.85 [−11.5, −1.7]	−6.55 [−11.7, −1.2]	**+48.21 [+43.2, +52.7]**	**+17.04 [+14.4, +19.7]**
United States	**+41.07 [+36.4, +45.1]**	−41.96 [−46.8, −36.8]	−6.85 [−12.2, −1.6]	**+38.39 [+33.1, +43.4]**	**+7.66 [+5.0, +10.3]**
Japan	**+33.33 [+29.0, +36.9]**	**+26.79 [+21.5, +32.1]**	−27.08 [−31.3, −22.3]	−9.82 [−14.3, −4.8]	**+5.80 [+3.2, +8.5]**
South Korea	**+37.20 [+32.3, +41.6]**	−20.54 [−25.2, −15.5]	**+7.14 [+2.4, +12.3]**	**+31.85 [+26.6, +37.2]**	**+13.91 [+11.3, +16.6]**
Australia	**+41.07 [+36.4, +45.1]**	**+16.07 [+10.7, +21.3]**	**+16.37 [+11.8, +21.4]**	**+10.12 [+5.2, +15.4]**	**+20.91 [+18.2, +23.6]**
Canada	**+41.07 [+36.4, +45.1]**	**+9.82 [+4.9, +15.1]**	**+2.38 [−2.7, +7.7]**	**+22.02 [+16.8, +27.4]**	**+18.82 [+16.2, +21.5]**
France	**+33.33 [+29.0, +36.9]**	**+24.11 [+18.8, +29.4]**	−6.55 [−11.7, −1.2]	**+48.21 [+43.2, +52.7]**	**+24.78 [+22.2, +27.3]**
Germany	**+33.33 [+29.0, +36.9]**	−44.94 [−49.6, −39.8]	**+16.96 [+12.3, +22.1]**	**+10.71 [+5.4, +16.1]**	**+4.02 [+1.4, +6.7]**
United Kingdom	**+33.33 [+29.0, +36.9]**	−69.35 [−73.1, −64.9]	−19.05 [−24.2, −13.7]	−2.68 [−8.0, +2.7]	−14.43 [−17.1, −11.8]
Average	**+36.04 [+33.4, +38.7]**	−13.90 [−37.5, +9.7]	−3.24 [−13.4, +6.9]	**+19.88 [+4.8, +35.0]**	**+9.69 [+1.2, +18.2]**

4 Experiments

4.1 Empirical Response-Direction Accuracy

First, we compare the Smets Wouters model (Smets & Wouters, 2003) solved with Vela against its conventionally solved counterpart in terms of matching the directions of macroeconomic responses to imposed shocks. We chose the Smets Wouters model for its widespread use as a DSGE benchmark, including in recent studies of estimation methods (Meyer-Gohde & Saecker, 2024; Meyer-Gohde, 2025). Using a historical panel of OECD macroeconomic data (OECD, 2026) spanning

from 1960 to 2026 across ten G20 economies, we fit country-level vector autoregression (VAR) models following an evaluation procedure commonly used in the macroeconomic literature (Smets & Wouters, 2007; Francis & Ramey, 2009). The VAR-estimated response directions of key macroeconomic variables to structural shocks serve as reference labels for comparing Vela's predictions across six training seeds with those of the conventionally solved Smets Wouters model. Table 1 reports gains in response-direction accuracy ($\mathrm{Acc}_{\mathrm{Vela}} - \mathrm{Acc}_{\mathrm{classic}}$), in percentage points, with 95% confidence intervals. Results indicate statistically significant improvements in average response-direction accuracy in eight of the ten economies, with the largest mean gains for productivity and money-preference shocks. These findings suggest that Vela's learned policies may capture nonlinear response patterns that improve agreement with empirical dynamics. We further characterize these responses in section 4.3.

4.2 Agreement with Conventional DSGE Solutions

Secondly, we assess agreement between Vela and the conventionally solved Smets Wouters model for individual shock–response pairs. Using the response-direction evaluation procedure described in section 4.1, we compute agreement rates for each combination of shock type and response variable. Figure 2 shows broad agreement, with lower rates for employment and real wages. These labor-market discrepancies warrant particular attention given the longstanding debate over labor responses to technology shocks (Gali, 1999; Christiano et al., 2004; Francis & Ramey, 2009) and evidence that such responses vary over time (Li, 2023). Overall, Vela broadly reproduces the conventional model's response directions while exhibiting larger differences in labor-market outcomes, where empirical findings remain debated.

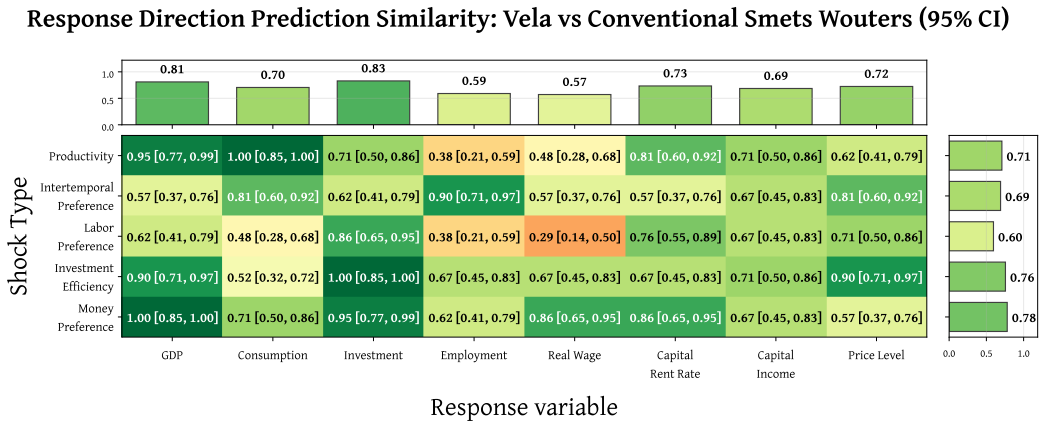


Figure 2: Response-direction agreement between Vela and the conventionally solved Smets Wouters model for upward shocks, with 95% confidence intervals.

4.2.1 IRF Case Study: United Kingdom

To examine the response trajectories more closely, we compare IRFs to a positive productivity shock from the conventional Smets Wouters solution computed using gEconpy (Grabowski, 2025), Vela's parallel paired simulations described in section 3.3, and a VAR(1) estimated on quarterly OECD data for the United Kingdom (OECD, 2026). Figure 3 shows that, as in the preceding experiment, the main differences between the model solutions concern employment and real wages. Vela's responses are generally closer to the empirical estimates; notably, Vela matches the positive initial employment response, whereas the conventional solution predicts an initial decline. More IRF comparisons could be found in section F of Appendix.

4.3 Nonlinear Responses to Simultaneous Shocks

To examine nonlinear response patterns, we evaluate the effects of simultaneous shocks. In a model solved by first-order approximation around a steady state, responses to simultaneous shocks are

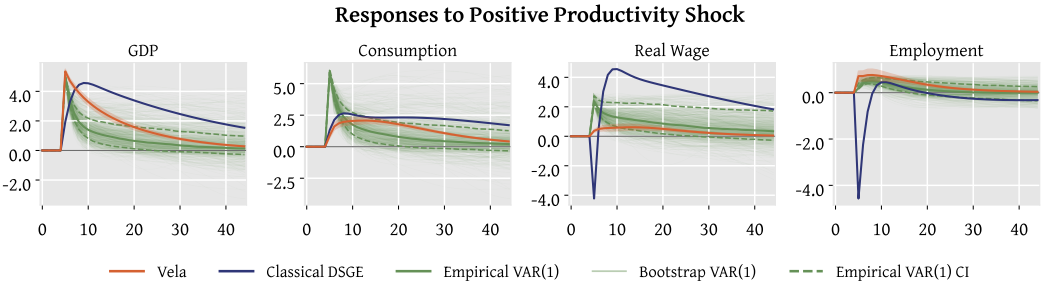


Figure 3: IRF case study for the United Kingdom: comparison of Vela, conventional Smets Wouters, and empirical VAR(1) responses to a positive productivity shock.

additive, whereas a MARL solution can capture nonlinear response patterns. To investigate this, we measured cumulative IRFs for simultaneous shocks on a grid spanning $[-0.2, 0.2]^2$ for both Vela and the conventional DSGE solution obtained using gEconpy (Grabowski, 2025). Figure 4 presents the resulting heatmaps. While the conventional solution exhibits linear color gradients, with response signs predominantly determined by the productivity shock, Vela exhibits an asymmetry: under positive productivity shocks, the signs of investment and capital responses depend strongly on the investment-efficiency shock. This pattern broadly parallels evidence of asymmetric economic responses during recessions and expansions (Auerbach & Gorodnichenko, 2012).

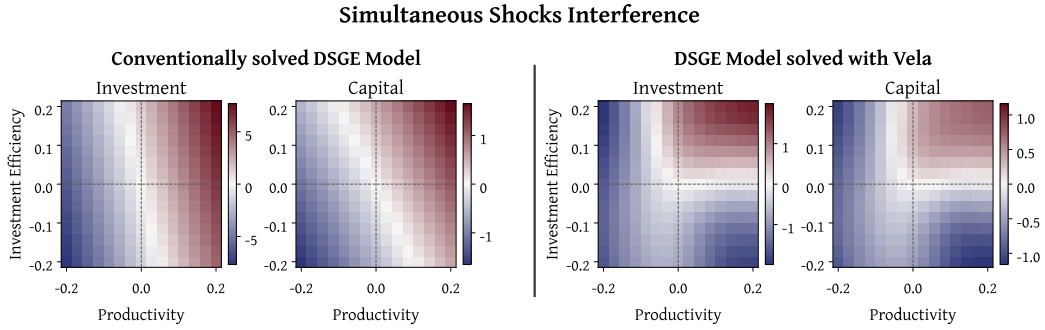


Figure 4: Investment and capital responses to simultaneous productivity and investment-efficiency shocks under conventional DSGE and Vela solutions. Colors indicate the sign and magnitude of cumulative IRFs.

4.4 Calibration-Parameter Domain Randomization

We investigate whether cross-economy training via calibration-parameter domain randomization improves empirical predictive accuracy. We repeated the response-direction accuracy comparison against the canonical Smets Wouters DSGE model described in section 4.1, using six training seeds for each setting, with and without domain randomization. As shown in figure 5, domain randomization improves accuracy across all ten economies considered, suggesting that cross-economy training improves agreement with empirically estimated response directions. This finding parallels evidence from robotics that domain randomization can improve generalization beyond the training simulator and support sim-to-real transfer (Tobin et al., 2017; Peng et al., 2018; Andrychowicz et al., 2020).

4.5 Evaluation Across Calibrations

Beyond its effect on empirical accuracy, we examine whether cross-calibration training yields economically interpretable responses across calibrations at inference. We vary the consumption-utility curvature parameter η_c, holding other calibration parameters constant, and evaluate IRFs to a negative productivity shock. Figure 7 shows that higher η_c shifts the employment response from negative

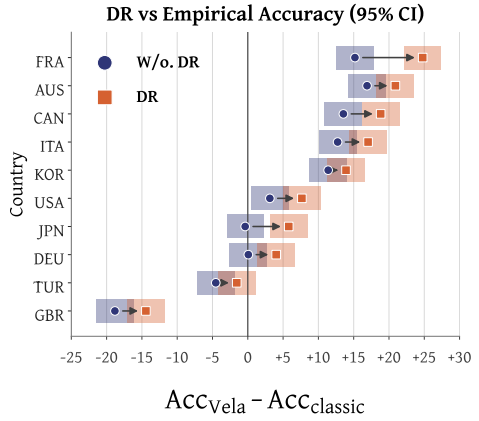


Figure 5: Difference in empirical response-direction accuracy (pp) between Vela and the conventionally solved Smets Wouters model, with and without calibration-parameter domain randomization.

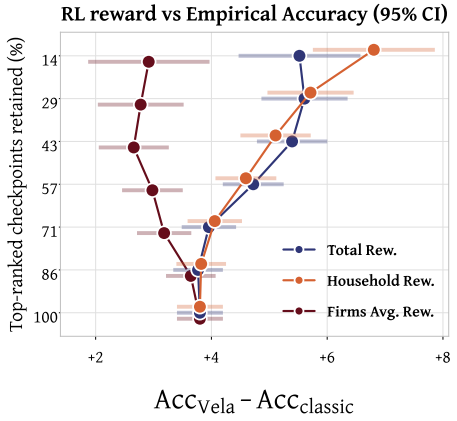


Figure 6: Empirical response-direction accuracy gains over the conventionally solved Smets Wouters model (pp), across checkpoint percentile groups ranked by total agent reward, household reward, or average firm reward.

to positive and reduces declines in consumption, investment, and GDP. This pattern is consistent with households sacrificing leisure to cushion consumption losses. The changing employment response is also relevant to mixed empirical evidence on labor-market responses to technology shocks (Li, 2023). Although these results do not establish which calibration is empirically preferred, they demonstrate that Vela supports exploration of economically distinct responses without retraining.

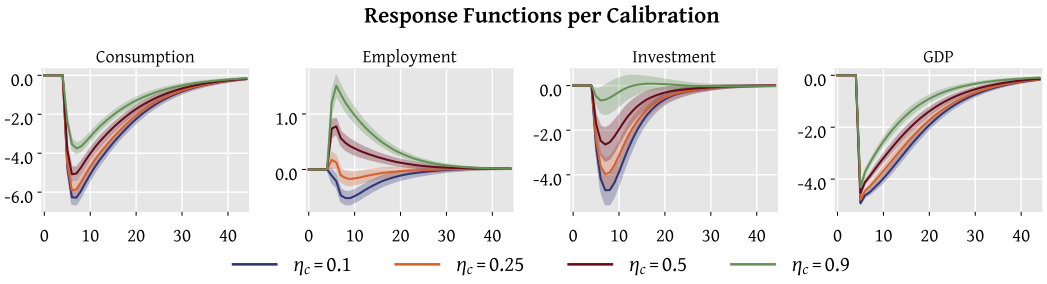


Figure 7: IRFs to a negative productivity shock across η_c calibrations in the Smets Wouters model. Higher η_c strengthens households' incentive to sacrifice leisure to cushion consumption losses.

4.6 Reward and Empirical Accuracy

Finally, we examine the relationship between agents' rewards and empirical response-direction accuracy, evaluated using the procedure described in section 4.1. Within each training seed, we rank IPPO-trained Smets Wouters checkpoints by total agent reward, household reward, and average firm reward, then evaluate progressively smaller subsets of the highest-reward checkpoints. Figure 6 shows that selection by total or household reward generally improves accuracy, with the largest gains obtained using household reward and no comparable positive association with average firm reward. These results suggest that reward-based checkpoint selection can improve empirical response-direction accuracy. The household-reward pattern may reflect households' central role in consumption, labor supply, investment, and ownership of capital and firms in the Smets Wouters model (Smets & Wouters, 2003).

5 CONCLUSION

We presented Vela, a JAX-based framework for formulating dynamic stochastic general equilibrium (DSGE) models as multi-agent reinforcement learning environments with heterogeneous agents pursuing individual objectives. Our experiments showed that Vela's learned solutions achieved higher empirical response-direction accuracy than their log-linearized counterparts across most of the economies studied. Furthermore, we showed that MARL policies captured nonlinear response patterns under simultaneous shocks and demonstrated that domain randomization can improve empirical predictive accuracy in macroeconomic settings.

While these results are promising, our observations suggest that conventional DSGE formulations, which were not originally designed for RL-based optimization, would benefit from further refinement to improve robustness to reward hacking and training instabilities. Another direction of future work is to extend Vela's constraint enforcement to sets defined by nonlinear constraints using Newton-based methods (Wächter & Biegler, 2006; De Luca et al., 1996). More ambitious directions may include bridging microfounded DSGE models with agent-based models featuring large populations of boundedly rational agents, using MARL to learn decision rules in settings beyond the reach of conventional solution methods.

AI USE STATEMENT

We used generative AI tools exclusively to polish the clarity, grammar, and style of the manuscript and to adjust the visual design of plots. We did not use these tools to generate research ideas, develop methods, conduct experiments, or analyze results. We reviewed all AI-assisted edits to ensure that they preserved the intended meaning and accurately represented the underlying data. We take full responsibility for the final content of this work, including all AI-assisted text and visual artifacts.

REFERENCES

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

Tohid Atashbar and Rui Aruhan Shi. *AI and macroeconomic modeling: Deep reinforcement learning in an RBC model*. International Monetary Fund, 2023.

Alan J Auerbach and Yuriy Gorodnichenko. Measuring the output responses to fiscal policy. *American Economic Journal: Economic Policy*, 4(2):1–27, 2012.

Alberto Baccini, Martina Cioni, Muriel Dal Pont Legrand, and Eugenio Petrovich. Does cross-fertilization occur in recent macroeconomics? a quantitative exploration of the interactions between dsge and mab models. *Journal of Economic Surveys*, 39(4):1758–1794, 2025.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal JĂłzefowicz, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. *arXiv preprint arXiv:2105.15183*, 2021.

Clément Bonnet, Daniel Luo, Donal Byrne, Shikha Surana, Sasha Abramowitz, Paul Duckworth, Vincent Coyette, Laurence I. Midgley, Elshadai Tegegn, Tristan Kalloniatis, Omayma Mahjoub, Matthew Macfarlane, Andries P. Smit, Nathan Grinsztajn, Raphael Boige, Cemlyn N. Waters, Mohamed A. Mimouni, Ulrich A. Mbou Sob, Ruan de Kock, Siddarth Singh, Daniel Furelos-Blanco, Victor Le, Arnu Pretorius, and Alexandre Laterre. Jumanji: a diverse suite of scalable reinforcement learning environments in jax, 2024. URL https://arxiv.org/abs/2306.09884.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, et al. Jax: composable transformations of python+ numpy programs. 2018.

George W Brown. Iterative solution of games by fictitious play. *Act. Anal. Prod Allocation*, 13(1): 374, 1951.

Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. Combining deep reinforcement learning and search for imperfect-information games. *Advances in neural information processing systems*, 33:17057–17069, 2020.

Simone Brusatin, Tommaso Padoan, Andrea Coletta, Domenico Delli Gatti, and Aldo Glielmo. Simulating the economic impact of rationality through reinforcement learning and agent-based modelling. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pp. 159–167, 2024.

Dan Cao, Wenlan Luo, and Guangyu Nie. Global dsge models. *Review of Economic Dynamics*, 51: 199–225, 2023.

Siddharth Chaturvedi, Ahmed El-Gazzar, and Marcel van Gerven. Abmax: A jax-based agent-based modeling framework, 2025. URL https://arxiv.org/abs/2508.16508.

Lawrence J Christiano, Martin Eichenbaum, and Robert Vigfusson. The response of hours to a technology shock: Evidence based on direct measures of technology. *Journal of the European Economic Association*, 2(2-3):381–395, 2004.

Lawrence J Christiano, Martin Eichenbaum, and Charles L Evans. Nominal rigidities and the dynamic effects of a shock to monetary policy. *Journal of political Economy*, 113(1):1–45, 2005.

Lawrence J Christiano, Martin S Eichenbaum, and Mathias Trabandt. On dsge models. *Journal of Economic Perspectives*, 32(3):113–140, 2018.

Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998(746-752):2, 1998.

Herbert Dawid and Domenico Delli Gatti. Agent-based macroeconomics. *Handbook of computational economics*, 4:63–156, 2018.

Tecla De Luca, Francisco Facchinei, and Christian Kanzow. A semismooth equation approach to the solution of nonlinear complementarity problems. *Mathematical programming*, 75(3):407–439, 1996.

Christian Schroeder De Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

DeepMind, Igor Babuschkin, Kate Baumli, Alison Bell, Surya Bhupatiraju, Jake Bruce, Peter Buchlovsky, David Budden, Trevor Cai, Aidan Clark, Ivo Danihelka, Antoine Dedieu, Claudio Fantacci, Jonathan Godwin, Chris Jones, Ross Hemsley, Tom Hennigan, Matteo Hessel, Shaobo Hou, Steven Kapturowski, Thomas Keck, Iurii Kemaev, Michael King, Markus Kunesch, Lena Martens, Hamza Merzic, Vladimir Mikulik, Tamara Norman, George Papamakarios, John Quan, Roman Ring, Francisco Ruiz, Alvaro Sanchez, Laurent Sartran, Rosalia Schneider, Eren Sezener, Stephen Spencer, Srivatsan Srinivasan, Miloš Stanojević, Wojciech Stokowiec, Luyu Wang, Guangyao Zhou, and Fabio Viola. The DeepMind JAX Ecosystem, 2020. URL http://github.com/google-deepmind.

Jesús Fernández-Villaverde, Juan Francisco Rubio-Ramírez, and Frank Schorfheide. Solution and estimation methods for dsge models. In *Handbook of macroeconomics*, volume 2, pp. 527–724. Elsevier, 2016.

Neville Francis and Valerie A Ramey. Measures of per capita hours and their implications for the technology-hours debate. *Journal of Money, credit and Banking*, 41(6):1071–1097, 2009.

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax – a differentiable physics engine for large scale rigid body simulation, 2021. URL https://arxiv.org/abs/2106.13281.

Jordi Gali. Technology, employment, and the business cycle: do technology shocks explain aggregate fluctuations? *American economic review*, 89(1):249–271, 1999.

Paul Glasserman and David D Yao. Some guidelines and guarantees for common random numbers. *Management Science*, 38(6):884–908, 1992.

Jesse Grabowski. geconpy: A collection of tools for working with dsge models in python, 2025. URL https://github.com/jessegrabowski/gEconpy.

Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games, 2016. URL https://arxiv.org/abs/1603.01121.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International conference on machine learning*, pp. 805–813. PMLR, 2015.

Natascha Hinterlang and Alina Tänzer. Optimal monetary policy using reinforcement learning. 2021.

Junling Hu and Michael P Wellman. Nash q-learning for general-sum stochastic games. *Journal of machine learning research*, 4(Nov):1039–1069, 2003.

Dom Huh and Prasant Mohapatra. Multi-agent reinforcement learning: A comprehensive survey, 2024. URL https://arxiv.org/abs/2312.10256.

Robert G King, Charles I Plosser, and Sergio T Rebelo. Production, growth and business cycles: I. the basic neoclassical model. *Journal of monetary Economics*, 21(2-3):195–232, 1988.

Sotetsu Koyamada, Shinri Okano, Soichiro Nishimori, Yu Murata, Keigo Habara, Haruka Kita, and Shin Ishii. Pgx: Hardware-accelerated parallel game simulators for reinforcement learning, 2024. URL https://arxiv.org/abs/2303.17503.

Finn E Kydland and Edward C Prescott. Time to build and aggregate fluctuations. *Econometrica: Journal of the Econometric Society*, pp. 1345–1370, 1982.

Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Perolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning, 2017. URL https://arxiv.org/abs/1711.00832.

Robert Tjarko Lange. gymnax: A JAX-based reinforcement learning environment library, 2026. URL http://github.com/RobertTLange/gymnax.

Huachen Li. The time-varying response of hours worked to a productivity shock. *Journal of Money, Credit and Banking*, 55(7):1907–1935, 2023.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pp. 157–163. Elsevier, 1994.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2020. URL https://arxiv.org/abs/1706.02275.

Chris Lu, Jakub Kuba, Alistair Letcher, Luke Metz, Christian Schroeder de Witt, and Jakob Foerster. Discovered policy optimisation. *Advances in Neural Information Processing Systems*, 35:16455–16468, 2022.

Robert E Lucas Jr. Econometric policy evaluation: A critique. In *Carnegie-Rochester conference series on public policy*, volume 1, pp. 19–46. North-Holland, 1976.

Lilia Maliar, Serguei Maliar, and Pablo Winant. Deep learning for solving dynamic economic models. *Journal of Monetary Economics*, 122:76–101, 2021.

Michael Matthews, Michael Beukman, Benjamin Ellis, Mikayel Samvelyan, Matthew Jackson, Samuel Coward, and Jakob Foerster. Craftax: A lightning-fast benchmark for open-ended reinforcement learning, 2024. URL https://arxiv.org/abs/2402.16801.

Michael Matthews, Michael Beukman, Chris Lu, and Jakob Foerster. Kinetix: Investigating the training of general agents through open-ended physics-based control tasks, 2025. URL https://arxiv.org/abs/2410.23208.

Walter Mayor, Johan Obando-Ceron, Aaron Courville, and Pablo Samuel Castro. The impact of on-policy parallelized data collection on deep reinforcement learning networks, 2025. URL https://arxiv.org/abs/2506.03404.

Alexander Meyer-Gohde. Solving linear dsge models with bernoulli iterations: A. meyer-gohde. *Computational economics*, 66(1):593–643, 2025.

Alexander Meyer-Gohde and Johanna Saecker. Solving linear dsge models with newton methods. *Economic modelling*, 133:106670, 2024.

Valentin Mohl, Sascha Frey, Reuben Leyland, Kang Li, George Nigmatulin, Mihai Cucuringu, Stefan Zohren, Jakob Foerster, and Anisoara Calinescu. Jaxmarl-hft: Gpu-accelerated large-scale multi-agent reinforcement learning for high-frequency trading. In *Proceedings of the 6th ACM International Conference on AI in Finance*, pp. 18–26, 2025.

Alexander Nikulin, Vladislav Kurenkov, Denis Tarasov, Dmitry Akimov, and Sergey Kolesnikov. Q-ensemble for offline rl: Don't scale the ensemble, scale the batch size, 2023. URL https://arxiv.org/abs/2211.11092.

Alexander Nikulin, Vladislav Kurenkov, Ilya Zisman, Artem Agarkov, Viacheslav Sinii, and Sergey Kolesnikov. Xland-minigrid: Scalable meta-reinforcement learning environments in jax, 2024. URL https://arxiv.org/abs/2312.12044.

OECD. Quarterly national accounts. OECD Data Explorer, dataflow OECD.SDD.NAD:DSD_NAMAIN1@DF_QNA, version 1.1, 2026. URL https://data-explorer.oecd.org/vis?df[ds]=dsDisseminateFinalDMZ&df[id]=DSD_NAMAIN1%40DF_QNA&df[ag]=OECD.SDD.NAD. Accessed on 14 September 2026.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3803–3810. IEEE, 2018.

Aleksei Petrenko, Zhehui Huang, Tushar Kumar, Gaurav Sukhatme, and Vladlen Koltun. Sample factory: Egocentric 3d control from pixels at 100000 fps with asynchronous reinforcement learning, 2020. URL https://arxiv.org/abs/2006.11751.

Eduardo Pignatelli, Jarek Liesen, Robert Tjarko Lange, Chris Lu, Pablo Samuel Castro, and Laura Toni. Navix: Scaling minigrid environments with jax, 2024. URL https://arxiv.org/abs/2407.19396.

Koen Ponse, Aske Plaat, Niki van Stein, and Thomas M. Moerland. Econojax: A fast scalable economic simulation in jax, 2025. URL https://arxiv.org/abs/2410.22165.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning, 2018. URL https://arxiv.org/abs/1803.11485.

Pranjal Rawat. A survey of reinforcement learning for economics. *arXiv preprint arXiv:2603.08956*, 2026.

Julio J Rotemberg. Sticky prices in the united states. *Journal of political economy*, 90(6):1187–1211, 1982.

Alexander Rutherford, Benjamin Ellis, Matteo Gallici, Jonathan Cook, Andrei Lupu, Gardar Ingvarsson, Timon Willi, Ravi Hammond, Akbir Khan, Christian Schroeder de Witt, Alexandra Souly, Saptarashmi Bandyopadhyay, Mikayel Samvelyan, Minqi Jiang, Robert Tjarko Lange, Shimon Whiteson, Bruno Lacerda, Nick Hawes, Tim Rocktaschel, Chris Lu, and Jakob Nicolaus Foerster. Jaxmarl: Multi-agent rl environments and algorithms in jax, 2026. URL https://arxiv.org/abs/2311.10090.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018. URL https://arxiv.org/abs/1506.02438.

Tim Seyde, Igor Gilitschenski, Wilko Schwarting, Bartolomeo Stellato, Martin Riedmiller, Markus Wulfmeier, and Daniela Rus. Is bang-bang control all you need? solving continuous control with bernoulli policies, 2021. URL https://arxiv.org/abs/2111.02552.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Frank Smets and Raf Wouters. An estimated dynamic stochastic general equilibrium model of the euro area. *Journal of the European economic association*, 1(5):1123–1175, 2003.

Frank Smets and Rafael Wouters. Shocks and frictions in us business cycles: A bayesian dsge approach. *American economic review*, 97(3):586–606, 2007.

B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi: 10.1007/s12532-020-00179-2. URL https://doi.org/10.1007/s12532-020-00179-2.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning, 2017. URL https://arxiv.org/abs/1706.05296.

Ming Tan et al. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pp. 330–337, 1993.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning, 2023. URL https://arxiv.org/abs/2305.09836.

Leigh Tesfatsion. Agent-based computational economics: A constructive approach to economic theory. *Handbook of computational economics*, 2:831–880, 2006.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

Andreas Wächter and Lorenz T Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1): 25–57, 2006.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning, 2021. URL https://arxiv.org/abs/2008.01062.

Yiwei Wang. From rational agents to learning agents: A structured review and conceptual framework for marl in economic modeling. *Complexity*, 2026(1):4688591, 2026.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes, 2020. URL https://arxiv.org/abs/1904.00962.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games, 2022. URL https://arxiv.org/abs/2103.01955.

A ALGORITHMS

A.1 ENVIRONMENT ROLLOUT COLLECTION

ALGORITHM 1 Environment Rollout Collection

Require: Agents $\{\mathcal{A}_i\}_{i=1}^N$, distributions $p(\theta)$ and $p(\xi)$
Ensure: Trajectory $\mathcal{D}$

Sample $\theta \sim p(\theta)$ ▷ Randomize economic calibration
$s_0 \leftarrow \mathrm{Reset}(\theta)$ ▷ Initialize the economic state
$t \leftarrow 0, \mathcal{D} \leftarrow \emptyset$
repeat
  for $i = 1, \ldots, N$ **do**
    $o_{i,t} \leftarrow h_i(s_t)$ ▷ Retrieve local observation
    $a_{i,t} \leftarrow \mathrm{GetAction}(i, o_{i,t})$ ▷ Sample policy action
    $\widetilde{c}_{i,t} \leftarrow g_i(c_{i,t-1}, a_{i,t})$ ▷ Propose economic controls
  end for
  $\widetilde{s}_t \leftarrow s_t[c \leftarrow \widetilde{c}_t]$ ▷ Insert all proposed controls
  $\bar{s}_t \leftarrow \widetilde{s}_t$ ▷ Initialize state for correction
  for $i = 1, \ldots, N$ **do**
    $c_{i,t}^\star \approx \arg\min_{c_i \in \mathcal{C}_i(\bar{s}_t)} \frac{1}{2}\|c_i - \widetilde{c}_{i,t}\|_2^2$ ▷ Enforce feasibility
    $\bar{s}_t \leftarrow \bar{s}_t[c_i \leftarrow c_{i,t}^\star]$ ▷ Insert corrected controls
  end for
  for $i = 1, \ldots, N$ **do**
    $r_{i,t} \leftarrow u_i(\bar{s}_t) - \lambda_{\mathrm{corr}}\|c_{i,t}^\star - \widetilde{c}_{i,t}\|_1 - \lambda_{\mathrm{act}}\|a_{i,t}\|_1/d_i$ ▷ Compute reward
  end for
  Sample $\xi_{t+1} \sim p(\xi)$ ▷ Sample economic shocks and noise
  $s_{t+1} \leftarrow F(\bar{s}_t, \xi_{t+1}; \theta)$ ▷ Advance global dynamics
  $o_{i,t+1} \leftarrow h_i(s_{t+1})$ for all i ▷ Construct next observations
  $(\mathrm{done}, \mathrm{truncated}) \leftarrow \mathrm{CheckTermination}(s_{t+1})$ ▷ Check episode boundaries
  Append $\big(\{o_{i,t}, a_{i,t}, r_{i,t}, o_{i,t+1}\}_{i=1}^N, \mathrm{done}, \mathrm{truncated}\big)$ to $\mathcal{D}$ ▷ Store transition
  $t \leftarrow t + 1$
until done $\vee$ truncated
return $\mathcal{D}$

A.2 IPPO WITH HISTORICAL-POLICY SELF-PLAY

ALGORITHM 2 IPPO with Historical-Policy Self-Play

Require: Environment $\mathcal{E}$, N agents, C alternation cycles, updates per block $\{K_i\}_{i=1}^{N}$, archive capacity P, symmetry mapping $\mathcal{S}$
Ensure: Trained policies $\{\pi_{\psi_i}\}_{i=1}^{N}$

for $i = 1, \ldots, N$ **do**
  Initialize actor π_{ψ_i} and critic V_{ϕ_i}
  Initialize archive $\mathcal{P}_i$ with P random policies
end for
Initialize parallel instances of $\mathcal{E}$
for $k = 1, \ldots, C$ **do**
  $\sigma \leftarrow \mathrm{RandomPermutation}(1, \ldots, N)$ ▷ Randomize agent order
  for i in σ **do**
    for $b = 1, \ldots, K_i$ **do**
      $\mathcal{D} \leftarrow \mathrm{CollectRollouts}(\mathcal{E}, i, \pi_{\psi_i}, \{\mathcal{P}_j\}_{j \neq i})$ ▷ Current learner; archived opponents
      $\widehat{A}_i, \widehat{R}_i \leftarrow \mathrm{GAE}(\mathcal{D}_i, V_{\phi_i})$ ▷ Use individual rewards
      $(\psi_i, \phi_i) \leftarrow \mathrm{PPOUpdate}(\mathcal{D}_i, \widehat{A}_i, \widehat{R}_i, \psi_i, \phi_i)$
    end for
    $q_i \leftarrow \mathrm{Snapshot}(\pi_{\psi_i})$ ▷ Freeze updated policy
    for $j \in \{i\} \cup \mathcal{S}(i)$ **do**
      $\mathcal{P}_j \leftarrow \mathrm{FIFOInsert}(\mathcal{P}_j, q_i, P)$ ▷ Replace oldest entry
    end for
    for $j \in \mathcal{S}(i)$ **do**
      $(\psi_j, \phi_j) \leftarrow (\psi_i, \phi_i)$ ▷ Synchronize symmetric agents
    end for
  end for
end for
return $\{\pi_{\psi_i}\}_{i=1}^{N}$

B CODE SNIPPETS

B.1 HOUSEHOLD: RBC MODEL

```python
import flax
import jax
import jax.numpy as jnp
from vela.types import ControlParameters, State, constraint

@flax.struct.dataclass
class HouseholdAction(ControlParameters):
    labor: float = 0.8
    consumption: float = 0.2
    investment: float = 0.2

# Household utility
@jax.jit
def household_objective(a: ControlParameters, s: State) ->
   chex.Scalar:
    c = jnp.maximum(a.consumption, 0.0)
    leisure = jnp.maximum(1.0 - a.labor, 0.0)
    return (
        s.gamma * (c ** (1 - s.eta_c) - 1) / (1 - s.eta_c)
        + s.psi * (leisure ** (1 - s.eta_l) - 1) / (1 - s.eta_l)
    )

# Budget constraint: spending <= income
@constraint(parameter=HouseholdAction)
def household_budget_constraint(a: ControlParameters, s: State)
   -> chex.Scalar:
    income = (a.labor * s.firm.wage
              + s.capital * s.firm.interest_rate)
    return a.consumption + a.investment - income
```

C Models Formal Definition

C.1 Kydland–Prescott RBC

The economy consists of a representative household and a firm. Let A_t denote productivity, K_t capital, and $\theta = (\eta_C, \eta_L, \gamma, \psi, \delta, \rho_A, \alpha_K, \alpha_L)$ the calibration vector.

Household. The household controls consumption C_t, labor supply L_t, and investment I_t:

$$c_{H,t} = (C_t, L_t, I_t).$$

Its instantaneous utility is

$$u_H = \frac{1}{10}\left[\gamma \frac{C_t^{1-\eta_C} - 1}{1-\eta_C} + \psi \frac{(1-L_t)^{1-\eta_L} - 1}{1-\eta_L}\right],$$

where γ and ψ weight consumption and leisure, while η_C and η_L determine their utility curvature. The feasible controls satisfy

$$C_t + I_t \le W_t L_t + R_t K_t, \qquad C_t \ge 0, \quad I_t \ge 0, \quad 0 \le L_t \le 1.$$

Here, W_t and R_t denote the wage and capital rental rate. Household income consists of labor and capital payments.

Firm. The firm controls its wage and capital rental rate,

$$c_{F,t} = (W_t, R_t),$$

and produces output according to

$$Y_t = A_t K_t^{\alpha_K} L_t^{\alpha_L}.$$

Its instantaneous objective is scaled profit:

$$u_F = \frac{1}{10}\left(Y_t - W_t L_t - R_t K_t\right),$$

subject to

$$0.01 \le W_t \le 5, \qquad 0.01 \le R_t \le 5.$$

The factor $1/10$ in both objectives follows the implementation's utility scaling.

State Dynamics. Capital evolves through depreciation and investment, subject to numerical bounds:

$$K_{t+1} = \mathrm{clip}_{[0.01,\,1.1]}\left((1-\delta)K_t + I_t\right).$$

Productivity follows

$$A_{t+1} = \mathrm{clip}_{[0.01,\,1.2]}\left[\exp\left(\rho_A \log A_t + B_{t+1}\varepsilon_{t+1}\right) + \nu_{t+1}\right],$$

where independent innovations satisfy

$$B_t \sim \mathrm{Bernoulli}(0.01), \qquad \varepsilon_t \sim \mathcal{N}(0, 0.15^2), \qquad \nu_t \sim \mathcal{N}(0, 0.0025^2).$$

Thus, productivity combines persistent dynamics with infrequent shocks and small per-period disturbances.

Calibration and Domain Randomization. Following the RBC training configuration, calibration parameters are sampled independently at each environment reset and held fixed throughout the episode:

$$\begin{aligned}
\eta_C, \eta_L &\sim \mathcal{U}(0.1, 0.9),\\
\gamma, \psi &\sim \mathcal{U}(0.1, 3.0),\\
\alpha_K, \alpha_L &\sim \mathcal{U}(0.1, 2.5),\\
\delta &\sim \mathcal{U}(0.05, 0.3),\\
\rho_A &\sim \mathcal{U}(0.8, 0.99).
\end{aligned}$$

The production exponents are sampled separately, without imposing $\alpha_K + \alpha_L = 1$. The training discount factor is $\beta = 0.99$.

Initial conditions are randomized as

$$\begin{aligned}
A_0 &= 1, & K_0 &\sim \mathcal{U}(0.8, 1.1),\\
W_0 &\sim \mathcal{U}(0.5, 1.2), & R_0 &\sim \mathcal{U}(0.1, 0.5),\\
L_0 &\sim \mathcal{U}(0.7, 1.0), & C_0, I_0 &\sim \mathcal{U}(0, 0.3).
\end{aligned}$$

C.2 SMETS–WOUTERS

We describe the implemented Smets Wouters adaptation with a representative household, N intermediate-goods firms, an implicit final-goods aggregator, and an optional central bank. The following economic specification omits numerical feasibility corrections and utility gating.

Household. The household controls consumption, firm-specific labor and investment, and, when the central bank is included, bond holdings:

$$c_{H,t} = \left(C_t, (L_{i,t}, I_{i,t})_{i=1}^{N}, B_t\right).$$

Let

$$L_t = \sum_{i=1}^{N} L_{i,t}, \qquad K_t = \sum_{i=1}^{N} K_{i,t}.$$

Consumption and investment are parameterized relative to aggregate capital:

$$C_t = q_{C,t} \max(K_t, 1), \qquad I_{i,t} = q_{I_i,t} \max(K_t, 1).$$

The household's instantaneous utility is

$$u_{H,t} = \frac{\varepsilon_t^B}{10}\left[\gamma \frac{C_t^{1-\eta_C} - 1}{1-\eta_C} + \varepsilon_t^L \psi \frac{(1-L_t)^{1-\eta_L} - 1}{1-\eta_L} + \varepsilon_t^M \beta_M \frac{(M_t/P_t)^{1-\eta_M} - 1}{1-\eta_M}\right].$$

Here, γ, ψ, and β_M weight consumption, leisure, and real money balances, respectively; η_C, η_L, η_M determine utility curvature. The processes ε_t^B, ε_t^L, and ε_t^M represent intertemporal, labor-preference, and money-preference shocks.

Nominal factor income and dividends are

$$F_t = \sum_{i=1}^{N} \left(W_{i,t} L_{i,t} + R_{i,t} K_{i,t}\right), \qquad D_t = (1-\tau) \sum_{i=1}^{N} \Pi_{i,t-1},$$

where τ is the profit-tax rate and $\Pi_{i,t}$ denotes nominal firm profit. The household's real disposable income is

$$Y_t^{\text{bal}} = \frac{F_t + D_t}{P_t}.$$

Money balances evolve according to

$$M_t = M_{t-1} + B_{t-1} - b_t B_t + P_t \left(Y_t^{\text{bal}} - C_t - \sum_{i=1}^{N} I_{i,t}\right),$$

where b_t is the price of a one-period bond paying one unit of money at maturity.

Household controls satisfy

$$\begin{aligned} C_t + \sum_{i=1}^{N} I_{i,t} &\le Y_t^{\text{bal}}, \\ C_t,\ I_{i,t},\ L_{i,t},\ B_t &\ge 0, \\ \sum_{i=1}^{N} L_{i,t} &\le 1, \\ b_t B_t &\le M_{t-1} + B_{t-1} + P_t \left(Y_t^{\text{bal}} - C_t - \sum_{i=1}^{N} I_{i,t}\right). \end{aligned}$$

Thus, consumption and investment are financed from current disposable income, while bond purchases additionally draw on existing money balances and maturing bonds.

Intermediate-Goods Firms. Each firm $i \in \{1, \ldots, N\}$ controls a wage coefficient, a capital rental-rate coefficient, and a proportional price change:

$$c_{F_i,t} = (w_{i,t}, r_{i,t}, d_{i,t}).$$

Nominal factor payments and prices are determined by

$$W_{i,t} = w_{i,t} P_{t-1}, \qquad R_{i,t} = r_{i,t} P_{t-1}, \qquad p_{i,t} = \max\{1, p_{i,t-1}(1 + d_{i,t})\}.$$

The control bounds are

$$w_{i,t} \geq 0, \qquad r_{i,t} \geq 0, \qquad -0.05 \leq d_{i,t} \leq 0.05.$$

Firm output and nominal profit are

$$Y_{i,t} = A_t K_{i,t}^{\alpha_K} L_{i,t}^{\alpha_L}, \qquad \Pi_{i,t} = p_{i,t} Y_{i,t} - W_{i,t} L_{i,t} - R_{i,t} K_{i,t}.$$

Each firm maximizes scaled real profit:

$$u_{F_i,t} = \frac{\Pi_{i,t}}{10 P_t}.$$

Final-Goods Aggregation. The final firm is not explicitly controlled; its role is embedded in the output and price aggregation rules:

$$Y_t = \sum_{i=1}^{N} Y_{i,t}, \qquad P_t = \frac{\sum_{i=1}^{N} p_{i,t} \bar{Y}_{i,t-1}}{\sum_{i=1}^{N} \bar{Y}_{i,t-1}},$$

where output weights are updated as

$$\bar{Y}_{i,t} = 0.85 \bar{Y}_{i,t-1} + 0.15 Y_{i,t}.$$

This formulation connects production and income accounting through the nominal identity

$$\sum_{i=1}^{N} p_{i,t} Y_{i,t} = F_t + \sum_{i=1}^{N} \Pi_{i,t}.$$

Smoothed price weights, lagged dividends, and profit taxation mean that household disposable income Y_t^{bal} need not equal aggregate output Y_t in each period.

Central Bank. In the extended specification, the central bank controls the bond price

$$c_{\text{CB},t} = b_t, \qquad 0.5 \leq b_t \leq 1.$$

Define inflation and output growth as

$$\pi_t = \frac{P_t}{P_{t-1}} - 1, \qquad g_t = \frac{Y_t}{Y_{t-1}} - 1.$$

The bank minimizes absolute deviations from its targets, with a capped loss:

$$u_{\text{CB},t} = -\min\left\{10,\, c_\pi |\pi_t - \pi^\star| + c_g |g_t - g^\star|\right\}, \qquad c_\pi = c_g = 1.$$

This agent did not learn reliable inflation control through the indirect transmission mechanism in our experiments. We therefore omit it from the final configuration, set $b_t = 1$, and remove bond trading by setting $B_t = 0$.

Capital Accumulation. Investment efficiency depends on the investment shock ε_t^I and changes in investment:

$$\chi_{i,t} = \max\left\{0,\, 1 - \left(\varepsilon_t^I \frac{I_{i,t}}{I_{i,t-1}} - 1\right)^2\right\}.$$

Capital evolves according to

$$K_{i,t+1} = \text{clip}_{[0.001,\, 10]}\left[(1 - \delta) K_{i,t} + \chi_{i,t} I_{i,t}\right].$$

Numerical safeguards for zero denominators are omitted from this notation.

Exogenous Processes. Productivity and preference and investment shocks follow a common transition rule. For $x \in \{A, \varepsilon^B, \varepsilon^L, \varepsilon^I, \varepsilon^M\}$,

$$x_{t+1} = \text{clip}_{[0.01,\,1.2]}\left[\exp\left(\rho_x \log x_t + J_{x,t+1}\zeta_{x,t+1}\right) + \nu_{x,t+1}\right],$$

with independent innovations

$$J_{x,t} \sim \text{Bernoulli}(0.003), \qquad \zeta_{x,t} \sim \mathcal{N}(0, 0.15^2), \qquad \nu_{x,t} \sim \mathcal{N}(0, 0.0025^2).$$

Calibration and Domain Randomization. Calibration parameters are sampled independently at each environment reset and remain fixed throughout the episode:

$$\begin{aligned}
\eta_C, \eta_L &\sim \mathcal{U}(0.1, 0.9),\\
\eta_M &\sim \mathcal{U}(0.5, 0.8),\\
\gamma, \psi, \beta_M &\sim \mathcal{U}(0.25, 2.5),\\
\alpha_K, \alpha_L &\sim \mathcal{U}(0.1, 0.7),\\
\delta &\sim \mathcal{U}(0.05, 0.3),\\
\rho_A, \rho_B, \rho_L, \rho_I, \rho_M &\sim \mathcal{U}(0.8, 0.99).
\end{aligned}$$

The training configuration fixes $\tau = 0.5$. Production exponents are sampled separately, without imposing $\alpha_K + \alpha_L = 1$. For the central-bank extension, the earlier implementation randomizes the targets as

$$\pi^\star \sim \mathcal{U}(0.02, 0.06), \qquad g^\star \sim \mathcal{U}(0.01, 0.05).$$

D Design Choices and Implementation Findings

D.1 Effective Design Choices

More parallel environments. Increasing the number of parallel environments improved training stability by providing more diverse experience for each update. The comparison in section E.1 illustrates this benefit.

Training against historical policies. Training agents against populations of past opponent policies improved convergence compared with using only the latest policies (section E.2). Historical opponents provide a broader range of behaviors and reduce dependence on the current opponent configuration.

Intermediate-firm profit taxation. Without profit taxation, wages and capital rental rates approached zero: the household could receive firms' profits through dividends instead of factor payments. Taxing profits reduced the attractiveness of this outcome and improved theoretical response-direction accuracy (section E.3).

Logarithmic prices and scaled real variables. We represented the price level in logarithms, expressed wages and capital rentals relative to prices, and scaled consumption and investment by capital. This reformulation limited the numerical effects of nominal growth and helped stabilize training.

Money utility conditional on consumption and leisure. Unconditional utility from money holdings encouraged accumulation at the expense of consumption and produced drifting trajectories. Making money utility conditional on consumption and leisure helped avoid this behavior and yielded more balanced household decisions.

Control-magnitude penalties. Penalizing large control actions suppressed pronounced oscillations in the RBC model and produced smoother trajectories. This adjustment cost discouraged abrupt decisions while preserving agents' ability to respond to shocks.

Preserving normalization statistics. We stored observation-normalization statistics alongside each historical policy and restored them when sampling that policy. Using current statistics with older policies changed their effective behavior; preserving the original statistics substantially reduced training noise.

Sharing parameters between symmetric firms. Independent policies for otherwise identical firms produced asymmetric behavior. Sharing policy parameters between symmetric firms improved the symmetry of their economic responses.

D.2 Unsuccessful Configurations and Limitations

Removing capital bounds under broad calibrations. Removing the upper bound on capital initially produced divergent trajectories when the capital and labor production exponents, α_K and α_L, varied over $[0.1, 2.5]$. Narrowing both ranges to $[0.1, 0.7]$ improved stability in these experiments.

Learning central-bank policy through indirect transmission. In our tested configurations, the central-bank agent did not learn to reliably control inflation through the effects of its policy on household decisions and firm pricing. We therefore omitted the central-bank agent from the final experiments.

Habit formation. Including consumption habit formation yielded IRFs with poorer agreement with reference responses in our experiments. We therefore excluded this mechanism from the final specification.

E Additional Experiments

E.1 Number of Parallel Environments

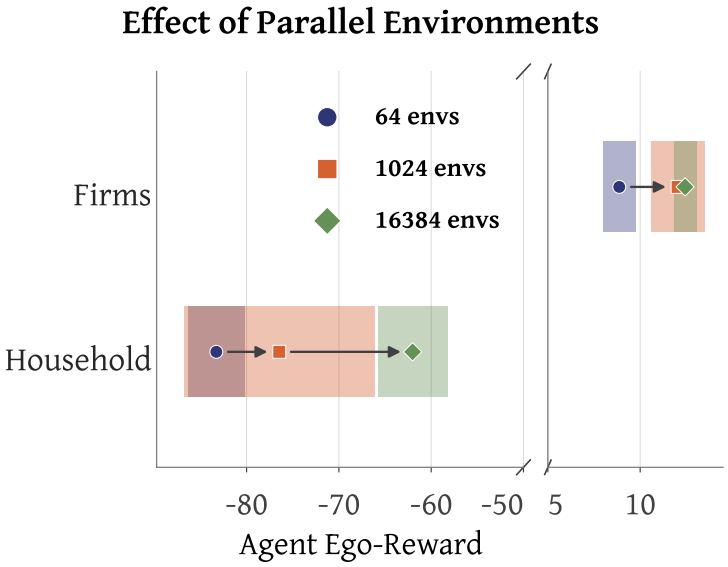


Figure 8: Effect of the number of parallel training environments on agents' individual rewards. Markers show mean rewards for evaluated checkpoints with shaded bands indicating one standard deviation across training seeds. Increasing parallelism improves mean rewards for both agent types, with the largest additional gain from 1,024 to 16,384 environments occurring for the household, while firm rewards largely plateau. These results support extensive parallelization as a practical means of improving learned policy performance, although they do not establish improved sample efficiency.

E.2 FIFO Historical Populations

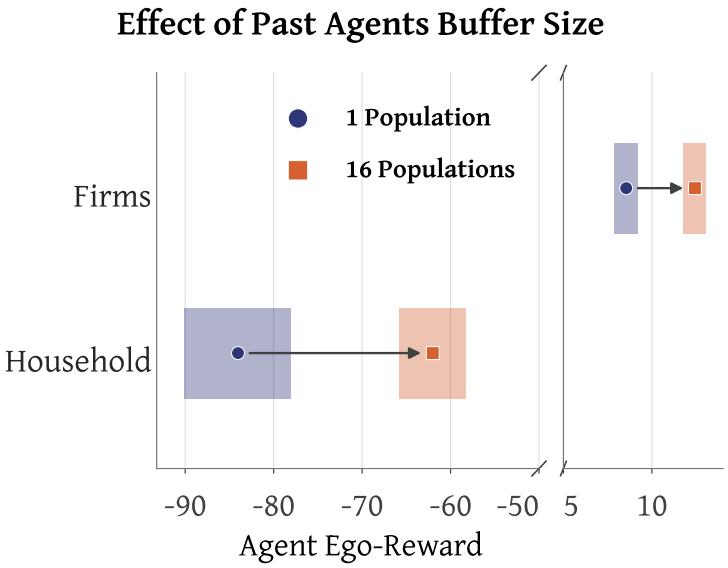


Figure 9: Effect of the historical policy archive size on agents' individual rewards. Markers show mean rewards for evaluated checkpoints, with shaded bands indicating one standard deviation across training seeds. Expanding the archive from one to sixteen policies per agent improves mean rewards for both households and firms, while reducing variability across seeds for the household. These results support training against a broader range of historical opponent policies as a means of improving learned policy performance.

E.3 Intermediate Firms Profit Taxation

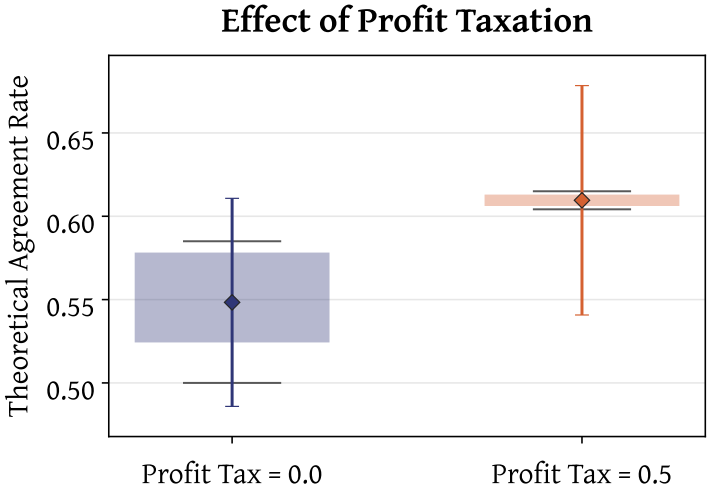


Figure 10: Effect of intermediate firms' profit taxation on response-direction agreement between Vela and conventional Smets Wouters solutions. Higher profit taxes increase the relative importance of factor income to households compared with dividend payments, reducing their incentive to accept zero wages and capital rental payments.

F IRFs Examples

F.1 vs Empirically estimated VAR and classic DSGE

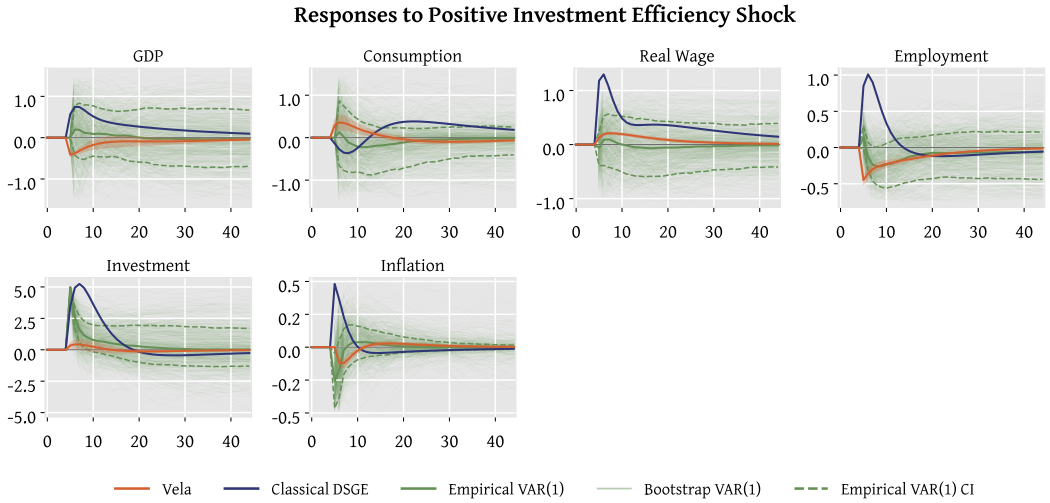


Figure 11: IRFs to a positive investment-efficiency shock: comparison of Vela, conventional Smets Wouters, and empirical VAR(1) estimates for Germany.

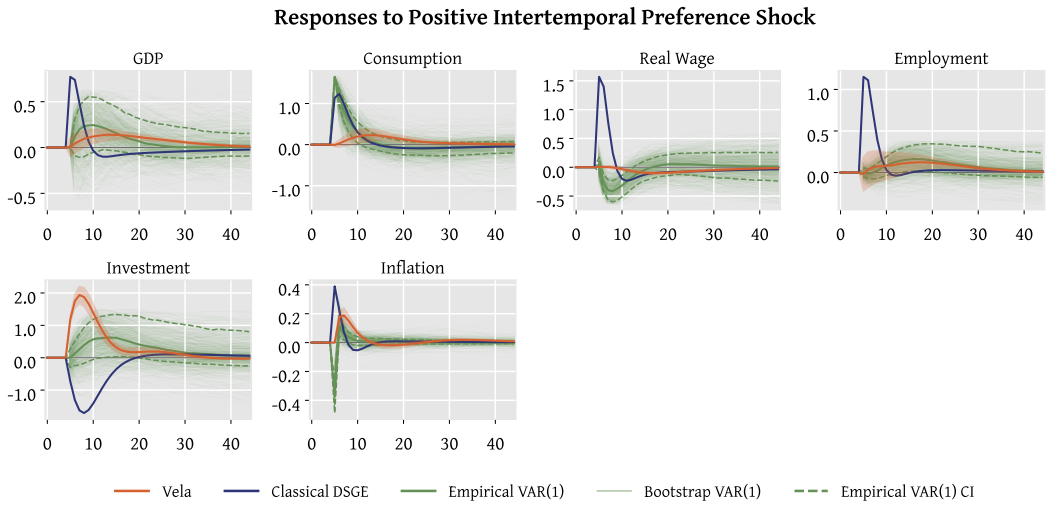


Figure 12: IRFs to a positive intertemporal-preference shock: comparison of Vela, conventional Smets Wouters, and empirical VAR(1) estimates for Japan.

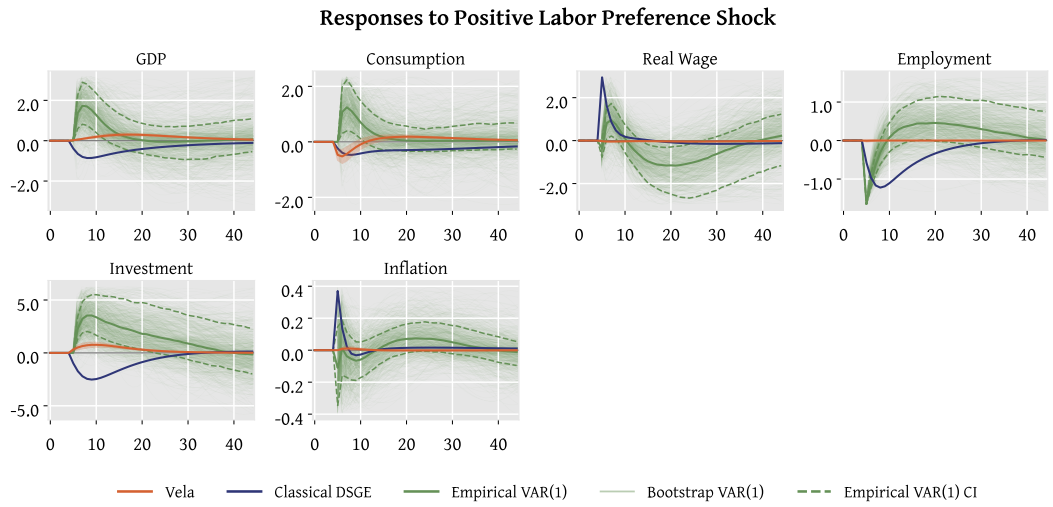


Figure 13: IRFs to a positive labor-preference shock: comparison of Vela, conventional Smets Wouters, and empirical VAR(1) estimates for France.

G HYPER-PARAMETER SETUP

Table 2: Hyperparameters for IPPO training on Smets Wouters environment. Intervals denote uniform domain-randomization ranges.

Description	**Parameter Group**	**Value**
Number of parallel environments	Environment	16384
Action-space scaling factor	Environment	1.0
Constraint-violation tolerance for the action projection	Environment	10^{-3}
OSQP solver tolerance	Environment	10^{-3}
L-BFGS solver tolerance	Environment	10^{-4}
Reward penalty for constraint violation	Environment	0.5
Reward penalty for action projection distance	Environment	0.2
Consumption utility curvature	Env. / Domain randomization	$[0.1,\ 0.9]$
Leisure utility curvature	Env. / Domain randomization	$[0.1,\ 0.9]$
Real money balances utility curvature	Env. / Domain randomization	$[0.5,\ 0.8]$
Consumption utility weight	Env. / Domain randomization	$[0.25,\ 2.5]$
Leisure utility weight	Env. / Domain randomization	$[0.25,\ 2.5]$
Real money balances utility weight	Env. / Domain randomization	$[0.25,\ 2.5]$
TFP shock persistence	Env. / Domain randomization	$[0.8,\ 0.99]$
Preference shock persistence	Env. / Domain randomization	$[0.8,\ 0.99]$
Labor supply shock persistence	Env. / Domain randomization	$[0.8,\ 0.99]$
Investment-efficiency shock persistence	Env. / Domain randomization	$[0.8,\ 0.99]$
Money demand shock persistence	Env. / Domain randomization	$[0.8,\ 0.99]$
Capital depreciation rate	Env. / Domain randomization	$[0.05,\ 0.3]$
Output elasticity of capital	Env. / Domain randomization	$[0.1,\ 0.7]$
Output elasticity of labor	Env. / Domain randomization	$[0.1,\ 0.7]$
Activation function	Agent (policy network)	SiLU
Number of hidden layers	Agent (policy network)	3
Hidden layer width	Agent (policy network)	128
Policy output scaling factor	Agent (policy network)	1.0
Environment steps per rollout	Training / Sampling	20
Minibatches per update step	Training / Sampling	10
Learning rate	Training / PPO	2×10^{-4}
Discount factor	Training / PPO	0.95
GAE lambda	Training / PPO	0.95
PPO clipping range	Training / PPO	0.2
Entropy bonus coefficient	Training / PPO	10^{-6}
Value loss coefficient	Training / PPO	0.5
Maximum gradient norm	Training / PPO	0.5