Autonomous Agents and Multi-Agent Systems (2021) 35:46
https://doi.org/10.1007/s10458-021-09528-4

# Automatic calibration of dynamic and heterogeneous parameters in agent-based models

**Dongjun Kim[1] · Tae-Sub Yun[1] · Il-Chul Moon[1] · Jang Won Bae[2]**

Accepted: 6 August 2021 / Published online: 24 August 2021


**Abstract**
Simulation has been applied to diverse domains such as urban growth modeling and market dynamics modeling. Some of these applications may require validations, based on some real-world observations modeled in the simulation. This validation can be conducted as either qualitative face-validation or quantitative empirical validation; however, as the importance and accumulation of data grows, the importance of quantitative validation has been highlighted in recent studies. The key component of quantitative validation is finding a calibrated set of parameters to regenerate the real-world observations in the simulation models. While the parameter of interest to be calibrated has hitherto been fixed throughout simulation executions, we expand the static parameter calibration in two dimensions in this study, dynamically and heterogeneously. The dynamic calibration changes the parameter values over the simulation period by reflecting the simulation output trend, and the heterogeneous calibration changes the parameter values per simulated entity clusters by considering the similarities of the entity states. We experimented with the proposed calibrations on a hypothetical case and a real-world case. For the hypothetical scenario, we used the wealth distribution model to illustrate how our calibration works. For the real-world scenario, we selected the real estate market model. The models were selected, because of two reasons. First, they have heterogeneous entities, being agent-based models. Second, they are agent-based models exhibiting real-world trends over time.

**Keywords** Simulation validation · Parameter calibration · Machine learning · Agent-based model

✉ Jang Won Bae
jangwon_bae@koreatech.ac.kr

Dongjun Kim
dongjoun57@kaist.ac.kr

Tae-Sub Yun
andy2402@kaist.ac.kr

Il-Chul Moon
icmoon@kaist.ac.kr

[1] Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea

[2] KOREATECH: Korea University of Technology and Education, Cheonan, Republic of Korea

## 1 Introduction

Simulation has been useful in market modeling [9], traffic management [56], and urban planning [35]. These simulations are real-world-based validated simulations of well-defined scenarios. Because of the detail and fitness to the real world, simulation becomes a meaningful tool for designing, managing, and analyzing modeled real world. While modelers accept the fundamental requirement of the validation [4], many simulation models are validated through model input parameter calibrations that can be easily invalidated, as time progresses or heterogeneity arises in the simulations. Some modelers manually calibrate the parameters to improve the fitness, but such an effort often doesn't work well, in particular, in the case of agent-based models which describe an emergence through the interactions among individuals.

A simulation world often diverges from the real world, as the simulation progresses, and the parameter calibration should accommodate this deviation over the simulation period. If we utilize handpicked or heuristically chosen parameters, we cannot mitigate the natural divergence as often as necessary. Hence, a common practice in the field is to calibrate the simulation parameters up-front, or over the predetermined period [83], and use them, regardless of the divergence of the simulation from the real world in the simulation runtime. Therefore, if we intend to accommodate the parameter calibration as often as necessary, we need to design an automatic calibration method that can resolve the natural divergence of simulation from the real world.

This automatic calibration is based on two functionalities: *when to calibrate* and *how to calibrate*. As the simulation diverges from the real world, the automatic calibration needs to determine the best timestep in the simulation to calibrate, for instance, calibrations in every simulation timestep in extreme cases. Moreover, if we consider the heterogeneity of simulated entities in the model as the source of divergence, the automatic calibration needs to determine the groups of entities to be calibrated. After deciding *when to calibrate*, the automatic calibration requires a component of *how to calibrate*.

In this study, the parameters to be calibrated are divided into two groups: dynamic parameters and heterogeneous parameters. The dynamic calibration is based on the assumption that there is a set of dynamically varying parameters that is shared across simulated entities at a certain timestep. The economic trend is an example of dynamic parameters. To calibrate the dynamically varying parameters, we extract unobservable regimes [30] that explain the temporal dynamics of interest. The unobservable regime is a hidden structure of time-dependent simulation that is extracted by applying a regime detection method [7, 24] to the deviation of observations and simulation results. The algorithm iteratively minimizes the fitness by exploring the dynamic parameters on under-fitted regimes, and exploiting the dynamic parameters on well-fitted regimes.

On the other hand, the heterogeneous parameters are a set of parameters that (1) vary from agent sub-population and (2) is static on the timesteps. The algorithm estimates the optimal heterogeneous parameter values by assigning the most suitable values to each cluster. The validity of heterogeneous calibration lies on the assumption that the agents with similar state variables will have similar criteria on action. Therefore, the agent clusters, a hidden structure for simulation, are extracted from the agent-level state variables. More specifically, agents with similar state variables are grouped together. Because the clustering methods suffer from the curse of dimension, to enhance the clustering performance, the algorithm learns the dimension-reduced latent vector for each high-dimensional agent state variables [37], and applies a clustering method [7, 79] to the obtained latent vectors.

The suggested calibration methodologies, dynamic calibration and heterogeneous calibration, can be deployed in a single framework wherein the methods utilize hidden structures extracted from simulation state variables for improving the calibration. Specifically, our framework captures the important observation that the simulation results of interest are driven by the hidden structures embedded in the simulation. For instance, dynamic calibration can capture the deviation patterns by extracting the hidden dynamic regimes; heterogeneous calibration can find a clustered group of agents sharing the same parameter, and the heterogeneity of individuals will be abstracted as their group heterogeneity. We limit our algorithm to in-sample validation, which focuses on constructing a highly descriptive model to replicate the real-world data.

The rest of this paper is organized as follows: Sect. 2 presents the problem formulation of the simulation calibration; Sect. 3 surveys the previous works on simulation calibration; Sect. 4 provides the preliminary machine learning algorithms for the proposed method; Sect. 5 introduces the proposed calibration framework and Sect. 6 illustrates the analysis of the proposed method using two examples; finally, Sect. 7 explains assumptions and limitations of the proposed method and Sect. 8 concludes the paper.

## 2 Problem formulation

In simulation verification and validation, parameter calibration is a mean to validate the model by estimating the optimal parameter, which maximizes the similitude of the simulation result to the real-world observation. In this Section, we present the procedural view and the mathematical view of the simulation calibration.

### 2.1 Procedural definition of simulation calibration

Simulation verification and validation is at the core of the modeling process for the system of interest [38, 45, 82]. The procedure of simulation verification and validation varies according to the simulation model, we adopt the view of [66] (Fig. 1). The verification is a process of ensuring the accuracy of the implementation of the computerized model. The validation is a procedure of verifying that the modeled stochastic process is a good representation of the real-world phenomena under study. The validation consists of two processes: conceptual model validation and operational validation. The conceptual model validation guarantees that the conceptual model represents the underlying mechanism of the real world authentically. The operational validation is a process of determining if the simulation model's output sufficiently satisfies the accuracy required for the model's intended purpose. An iterative process of verification and validation is required to develop a valid simulation model.

The parameter calibration is a key technique used in the agent-based model validation. The simulation outcome is sensitive on the input parameter, because a small change in the parameter values may amplify the nonlinear feedback in the system of interest. Calibration consists of the qualitative calibration, which involves fitting the stylized facts, and the quantitative calibration, which involves fitting both the stylized facts and the numerical output trend [66]. In this study, we focus on the quantitative calibration, a sub-process of the operational validation, wherein the model parameters are adjusted through the quantitative calibration to minimize the discrepancy between the real-world observation and its counterpart derived from the simulation. The set of parameters to be calibrated needs to

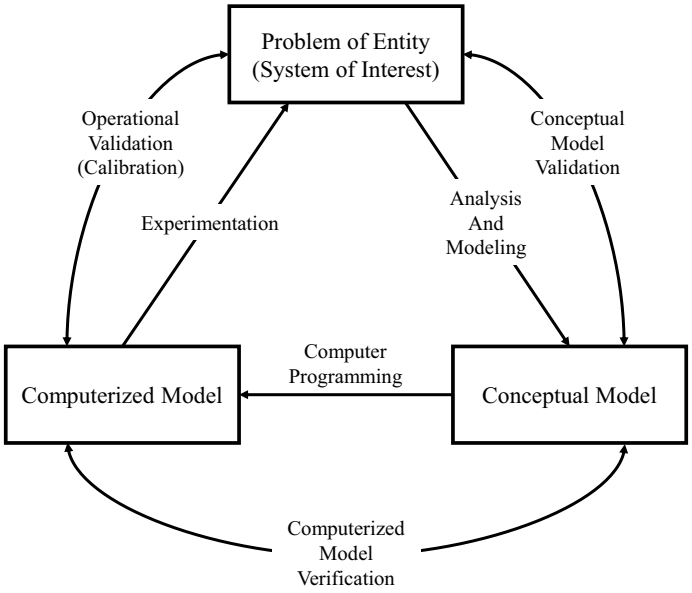



**Fig. 1** Model development process (from [66])

be selected before calibration. The calibration often requires several iterations to find the optimal parameters.

The discussion on whether the calibration process is subsumed under verification or validation originates from much deeper fundamental arguments. If we consider calibration as a part of verification, we regard the calibrated parameters as a part of implementation on the model behavior program. Otherwise, if we consider it as a part of validation, we see the calibrated parameters as a part of the scenario drawn from the real world. The latter perspective is taken in this study, because we intend to vary the parameters based on the variations of validation datasets. Underneath this usage, we hypothesize that a modeler can build a generalized model to be applied to diverse real-world scenarios, if the fundamentally modeled dynamics are the same.

## 2.2 Mathematical formulation of simulation calibration

To eliminate unnecessary information that causes over-fitting from the observed empirical data, we pre-process the observation into reduced-form summary statistics. The choice of the summary statistics varies from the purpose of simulation; for example, Guerini and Moneta [28] extracts the reduced-form stylized facts by using structural vector autoregression [11, 13] to analyze the domain-specific stylized facts, and Heppenstall et al. [33] chooses the spatial average (first moment) as the summary statistics for comparison. Throughout the paper, $x_{obs}$ denotes the pre-processed summary statistics of the empirical data, and the simulation result $\mathcal{M}(\theta;\omega)$ is the corresponding summary statistics, see Table 1 for the nomenclatures.

To determine an objective function to be optimized, we introduce a measure $d$ that calculates the discrepancy between the simulation and observation: for instance, Barde and

**Table 1** Description of notations

| Notation | Description |
|---|---|
| *Collected data* | |
| $x_{obs}$ | Empirical validation summary statistics. $x_{obs} \in \mathbb{R}^{P \times T}$ |
| $\mathcal{D}_{agent}$ | Agent-level simulation result to cluster agents. $\mathcal{D}_{agent} = \{x_a \vert a = 1, ..., A\}$, where $x_a \in \mathbb{R}^{Q \times T}$ |
| $\mathcal{D}_{lat}$ | Representational compressed data. $\mathcal{D}_{lat} = \{z_a \vert a = 1, ..., A\}$, where $z_a \in \mathbb{R}^L$ |
| $\mathcal{D}_{1:n}$ | Input (heterogeneous parameters) and output (discrepancy) pairs up to $n$ calibration iterations |
| *Simulation model* | |
| $\mathcal{M}$ | Agent-based model |
| $\Theta$ | Space of simulation calibration parameter, $\theta \in \Theta \subseteq \mathbb{R}^d$ |
| $\Omega$ | Space of nuisance variable that determines the sample path, $\omega \in \Omega$ |
| $P$ | Dimension of summary statistics, per a timestep, of the simulation model output variables |
| $Q$ | Dimension of simulation agent-level state variables per a timestep |
| $T$ | Number of simulation timesteps |
| $A$ | Number of simulation agents |
| *Overall calibration* | |
| $d$ | Simulation discrepancy measure |
| $\hat{d}$ | Monte-Carlo estimation of the discrepancy function $d$ |
| $J$ | Number of simulation replications in estimating the discrepancy function |
| $N$ | Number of calibration iterations |
| *Dynamic calibration* | |
| $N^t$ | Number of consecutive dynamic calibration iterations |
| $\theta^t$ | Dynamically varying simulation parameters to calibrate |
| $D^t$ | Number of dynamically varying parameters to calibrate |
| $\theta_{n,l}^{t,i}$ | $l$-th dynamic parameter value at $n$-th calibration iteration, $t$-th time, and $i$-th particle |
| $\overline{\theta}_l$ | Upper bound of search space of $l$-th dynamic parameter |
| $\underline{\theta}_l$ | Lower bound of search space of $l$-th dynamic parameter |
| $\mathtt{I}$ | Number of particles (candidate hypotheses) in estimating the parameter posterior |
| $\hat{\mu}_n^{t,i}$ | Empirical average of$i$-th particle simulation result at $n$-th calibration iteration and $t$-th time |
| $\hat{\Sigma}_n^{t,i}$ | Empirical covariance of $i$-th particle simulation result at $n$-th calibration iteration $t$-th time |
| $L_n^{t,i}$ | Likelihood of $x_{obs}$ being observed as simulation outcome at $n$-th iteration, $t$-th time, and $i$-th particle |
| $O_n^{t,i}$ | Observable variable of Hidden Markov Model at $t$-th time and $i$-th particle |
| $\mathcal{R}_n^{t,i}$ | Unobservable hidden regime of $t$-th time, $i$-th particle and $n$-th iteration |
| $K^t$ | Number of hidden regimes in HMM |
| $M$ | Number of merged regimes along particles |
| $\Gamma_n^m$ | Merged regime for the separation of simulated timesteps |
| $\alpha_{m,l}, \beta_{m,l}$ | Shape parameters for parameter posterior distribution at $m$-th merged regime and $l$-th parameter |
| *Hetero-geneous calibration* | |
| $N^k$ | Number of consecutive heterogeneous calibration iterations |

**Table 1** (continued)

| Notation | Description |
|---|---|
| $\theta^k$ | Agent-specifically varying simulation parameters to calibrate |
| $D^k$ | Number of heterogeneously switching parameters to calibrate |
| $L$ | Dimension of hidden representation of agent state variables |
| $z_a$ | Latent representation ($L$-dimensional) of $a$-th agent state variables $x_a$ |
| $K^k$ | Number of agent sub-populations |
| $\boldsymbol{\pi}$ | Mixture probability of clustering algorithm |
| $\boldsymbol{c}$ | Agent cluster assignments $\boldsymbol{c} = \{c_a\}_{a=1}^A$, where $c_a \in \{1, ..., K^k\}$ |
| $\mathsf{K}$ | Kernel function used in surrogate modeling |
| $\nu$ | Differentiability of the kernel function |
| $\mathbb{H}$ | Parameter search space of Bayesian optimization |
| $\hat{\underline{d}}_{1:n}$ | Minimum of the estimated discrepancy in $\mathcal{D}_{1:n}$: $\hat{\underline{d}}_{1:n} = \min_{1:n} \hat{d}(\theta_i)$ |
| $\mu_d(\theta \mid \mathcal{D}_{1:n})$ | Average of predictive distribution in Gaussian process regression estimated with $\mathcal{D}_{1:n}$ |
| $\sigma_d(\theta \mid \mathcal{D}_{1:n})$ | Standard deviation of predictive distribution in Gaussian process regression estimated with $\mathcal{D}_{1:n}$ |
| $w_n$ | Exploration rate of weighted Expected Improvement |
| $\xi_1$ | Probability of choosing next parameter randomly |
| $\xi_2$ | Probability of choosing next parmaeter based on the posterior variance $\sigma_d$ |
| $\xi_3$ | Probability of choosing next parmaeter based on the posterior mean $\mu d$ |
| $\underline{d}^c$ | Minimum discrepancy $\underline{d}^c = \min_{\theta^k} d^c(\theta^k)$ with given agent cluster assignments $\boldsymbol{c} = \{c_a\}_{a=1}^A$ |

Van Der Hoog [3] utilizes the Markov information criterion [2] as the discrepancy measure, and Lamperti et al. [40] adopts the mean square error to measure the discrepancy. The scalar-valued discrepancy measure $d(\mathcal{M}(\theta;\omega), x_{obs})$ is a function of the simulation input parameter $\theta$ and the nuisance variable $\omega$ that determines the stochastic outcome path. To eliminate the effect of the random nuisance variable $\omega$, we select the average over simulation replications $\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)]$. The discrepancy function $d(\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)], x_{obs})$ captures the difference between the observation and the underlying dynamics given input $\theta$, and this objective function is stabilized with respect to the variability arising from the nuisance variable $\omega$. We use the abuse of notation to denote $d(\theta)$ as the discrepancy measure $d(\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)], x_{obs})$ throughout the paper.

In this study, the parameter calibration is defined as the optimization problem of estimating the optimal parameters that yield the most consistent result with real-world observations:

$$\theta^* = \arg\min_{\theta\in\Theta} d(\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)], x_{obs}), \tag{1}$$

## 3 Literature review on simulation calibration

This section presents a literature review on the simulation calibration, and this survey was conducted from two viewpoints: from the algorithms applied in the simulation calibration and from the related research on agent-based model calibration.

### 3.1 Calibration methodologies on simulations

This section analyzes techniques that have been applied to the simulation calibration, such as derivative-based algorithms, heuristic algorithms, Bayesian inference algorithms, and Bayesian optimization algorithms (please refer to Table 2).

*Derivative-based algorithms* The derivative-based calibrations minimize the discrepancy $d(\theta)$ using the gradient descent methods under the derivative $\nabla_\theta d(\theta) = \nabla_\theta d(\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)], x_{obs})$. The derivative calculation $\nabla_\theta d(\theta)$ requires the inner derivative $\nabla_\theta \mathcal{M}(\theta;\omega)$, which is unattainable, because the agent-based simulations $\mathcal{M}$ depend on the discrete event-based state changes. Therefore, the derivative-based calibration requires derivative estimation techniques, such as the finite difference [64] or the infinitesimal perturbation analysis (IPA) [77]. A variant of the finite difference method suffers from the high evaluation cost required to calculate the derivative value at a point $\theta$ [64] . Meanwhile, the IPA [77] approximates the derivative using a single shot of evaluation; however, the high variance of the approximated derivative value makes the IPA too volatile in simulation calibration.

*Heuristic algorithms* Heuristic algorithms have been extensively applied to the simulation calibration in practice. Although the heuristic algorithm often provides no guarantee of the global optimality, some with simple and intuitive ideas still improves the simulation fitness to the real-world validation including the high-dimensional calibration [75]. Nguyen et al. [57] presented several features of the heuristic algorithm which lead to this success in the simulation calibration: (1) it is robust on the task of global optimization for highly nonlinear multi-modal discontinuous functions; (2) the evolution algorithm is designed to align well with parallel computations in multi-processor computers, and, furthermore, the parallelization has a favor in a computationally expensive simulation optimization; (3) it is efficient to the scalability, which means that it is effective in the calibration of agent-based models with many parameters to optimize; (4) it can be applied to both continuous and discrete parameters, which enables both the simulation parameter $\theta$ and the simulation structure (out of finite number of alternatives) to be optimized simultaneously.

By manipulating such advantages, for example, simulated annealing [32] and evolutionary algorithms, such as particle swarm optimization [58], genetic algorithm [75], and evolution strategy [31] were applied to the simulation calibration. Moreover, with the concept of natural selection, the evolution strategy shows an efficiency in the optimization of the continuous agent-based model parameters [31]. It creates the next generation by randomly perturbing the best individuals of the iteration according to a perturbing continuous distribution, and the perturbation distribution is updated for every iteration to improve the generation efficiency. Among various update schemes, for example, the covariance matrix adaptation [31] updates the covariance in order to maximize the likelihood of the previously successful individuals.

*Bayesian inference algorithms* The bottom-up approach of the agent-based model does not permit the attainment of the analytic solution with respect to the response variables. In the absence of knowledge on the analytic solution, the likelihood $p(x_{obs}|\theta)$ is intractable. Here, given that the likelihood is calculated as $p(x_{obs}|\theta) = \int p(x_{obs}|\theta,\omega)p_\Omega(\omega)d\omega = \int \delta\big(d(\mathcal{M}(\theta;\omega), x_{obs})\big)p_\Omega(\omega)d\omega$ (where $p_\Omega$ is the distribution of the nuisance variable $\omega$ that decides the sample path), the exact likelihood formula is not derivable because the distribution $p_\Omega$ and the set $\{\omega|d(\mathcal{M}(\theta;\omega), x_{obs}) = 0\}$ are unknown. The likelihood estimation through the Monte Carlo method is also

**Table 2** Previous research on simulation parameter calibration

| Algorithms | Search method | Global optimality | Advantage | Disadvantage |
|---|---|---|---|---|
| *Derivative-based algorithms* | | | | |
| Gradient Descent with finite difference [64] | Gradient Descent | No | Model-free algorithm | Multiple evaluation required for a single update |
| Gradient Descent with IPA [77] | Gradient Descent | No | Single evaluation required for an update | Limited applicability, high variance |
| *Heuristic algorithms* | | | | |
| Simulated Annealing [32] | Local search with cooling mechanism | No | Able to escape from local minimum | Need to control the cooling rate |
| Evolutionary Algorithm [31, 58, 75] | Update individual's genes based on the principle of biological evolution | No | Model-free, Robust on non-convex optimization surface, parallelizability in optimization reduces optimization time dramatically [57] | No guarantee on the convergence to the global optimum, performance heavily depends on the choice of hyperparameters |
| *Bayesian inference algorithms* | | | | |
| Rejection ABC [78] | Rejection Sampling | No | Simple inference procedure | Not scalable; No convergence due to $\epsilon > 0$ |
| Monte-Carlo Markov Chain ABC [46] | Random walk from the previous evaluated particles | No | Much faster than rejection ABC | Highly correlated samples, Slow convergence rate [18, 49] |
| Sequential Monte Carlo ABC [72] | Sampling from a learned proposal distribution | No | Avoid sampling inefficiency through learning of intermediate distribution, less prone to get stuck in regions of low probability | Choice of the number of particles and the forward-backword kernels affect the performance |
| Synthetic Likelihood [91] | Impose an explicit distribution on the summary statistics | No | Relatively less number of particles required than ABCs | Naive assumption on summary statistics; No prior knowledge on which assumption to impose |

**Table 2** (continued)

| Algorithms | Search method | Global optimality | Advantage | Disadvantage |
|---|---|---|---|---|
| *Bayesian optimization algorithms* | | | | |
| GPR-based Bayesian Optimization [84] | GPR as a surrogate modeling, Bayesian optimization as a parameter search method | Depends on conditions | Analysis on the convergence in special conditions [14, 50, 86], fast convergence, robust on stochastic function optimization [73] | No standard criteria on the selection of kernel shape and acquisition function |
| ***Proposed methods*** | | | | |
| ***Dynamic calibration*** | Sampling from a set of posterior distributions inferred from SMC with Synthetic Likelihood for each merged regimes | No | Calibration on dynamic parameter; Regulate the number of the optimization dimension by HMM | No theoretical convergence; Mean-field assumptions |
| ***Heterogeneous calibration*** | GPR-based Bayesian optimization | Depends on conditions | Calibration on the agent-specifically diversified parameter; Regulate the number of optimization dimension by Cluster-based diversification | No joint relation on the clustering algorithm and the optimization algorithm |

prohibitive because the delta distribution is a discrete, singular measure. The approximate Bayesian computation (ABC) [46, 72, 78] is a class of Bayesian inference algorithms that infer the posterior $p(\theta|x_{obs})$ of the simulation models when the likelihood $p(x_{obs}|\theta)$ is intractable. The ABC estimates the intractable likelihoods by approximating the discrete-typed delta distribution into one of continuous-typed distributions with its mass centered at the origin.

The rejection ABC [78] uses the box distribution to approximate the delta distribution, and estimate the approximate likelihood by counting the number of accepted particles via the rejection sampling, where the box distribution is a uniform distribution on a $\epsilon$-cube. However, the number of point evaluation (simulation) required to infer the approximate posterior in the rejection ABC is exponentially blowing up as the dimension increases. In addition, the rejection ABC is not robust on the choice of $\epsilon$. The Monte-Carlo Markov Chain (MCMC) ABC [46] improves the rejection ABC to boost the speed of the posterior inference by orders of magnitude. MCMC-ABC makes use of the MCMC to choose the next sample, and the use of MCMC dramatically reduces the number of samples rejected by the algorithm.

The sequential Monte Carlo (SMC) ABC [72] approximates the posterior distribution through the particle filter approach. First, it samples $\mathtt{I}$ number of initial particle parameters $\{\theta_0^i|i=1,...,\mathtt{I}\}$ from the initial distribution $\mu_1$. Then, it evaluates the simulation using the candidate particles $\theta_0^i$ for $i=1,...,\mathtt{I}$, following which it calculates the likelihood $p(x_{obs}|\theta_0^i)$ of each particle. The likelihood is approximated from the Monte Carlo estimation through the assumption that the $\epsilon_0$-rejection distribution is the substitute of the delta distribution: $p(\theta_0^i|x_{obs}) = \int \delta(d(\mathcal{M}(\theta_0^i;\omega),x_{obs}))p_\Omega(\omega)d\omega \approx \frac{1}{J}\sum_{j=1}^{J} \mathbb{1}(d(\mathcal{M}(\theta_0^i;\omega_j),x_{obs}) < \epsilon_0)$. The next particles $\{\theta_1^i|i=1,...,\mathtt{I}\}$ are sampled from the weighted initial samples, where the weight of the $i$-th particle is given as $\mathtt{w}_1^i = \frac{p(\theta_0^i|x_{obs})}{\mu_1(\theta_0^i)}$. To provide the randomness on the sampling procedure, we perturb $\theta_1^i$ by the kernel $K$. At the $n$-th iteration, the simulation evaluates the randomly perturbed samples $\theta_n^i$ and the likelihoods are approximated by the $\epsilon_n$-rejection distribution: $p(\theta_n^i|x_{obs}) \approx \frac{1}{J}\sum_{j=1}^{J} \mathbb{1}(d(\mathcal{M}(\theta_n^i;\omega_j),x_{obs}) < \epsilon_n)$. Then, the weights for each particle is calculated by $\mathtt{w}_n^i = \frac{p(\theta_n^i|x_{obs})}{\sum_{i'=1}^{I} w_{n-1}^{i'} \mathsf{K}(\theta_n^i,\theta_{n-1}^{i'})}$. SMC-ABC can be seen as the nonparametric method to estimate the intermediate posterior distributions, because the denominator term $\sum_{i'=1}^{I} \mathtt{w}_{n-1}^{i'} \mathsf{K}(\cdot,\theta_{n-1}^{i'})$ is the nonparametric kernel estimation.

While SMC approximates the likelihood nonparametrically by approximating the delta distribution into a uniform kernel that uses the boxcar function, the synthetic likelihood (SL) [91] is a parametric approach to detour the likelihood intractability by imposing an explicit assumption on the summary statistics. For example, [91] assumes the Gaussian distribution to estimate the likelihood via $p(x_{obs}|\theta) \approx \mathcal{N}(x_{obs}|\hat{\mu}(\theta),\hat{\Sigma}(\theta))$, where $\hat{\mu}(\theta)$ and $\hat{\Sigma}(\theta)$ are the empirical mean and the standard deviation of the $J$ replications of the summary statistics $\{\mathcal{M}(\theta;\omega_j)|j=1,...,J\}$. Although the explicit distribution can be a source of bias in the Bayesian inference, the SL approach, with its fast inference due to explicit assumption, can be seen as an alternative of the ABC approach.

From the modeler's perspective, utilizing the overall landscape of the posterior distribution $p(\theta|x_{obs})$ would take a significant advantage in the calibration process; however, it also requires a tremendous number of simulation executions to estimate the posterior distribution even in a case of simple system dynamics models with few parameters. Therefore, the Bayesian inference is only applicable in agent-based models with limited evaluation cost and number of parameters.

*Bayesian optimization algorithms* Because the closed form of the expectation $\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)]$ is unattainable, the discrepancy $d(\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)],x_{obs})$ can only be

estimated through the Monte Carlo sampling of $\omega \in \Omega$. Hence, the unwanted noise is injected in the process of estimating the discrepancy that requires the population expectation $\mathbb{E}_{\omega \in \Omega}$. This sample path can only be estimated through a Monte-Carlo sampler, since the simulation attains no closed-form solution, except for the simplest forms. Therefore, the estimation noise yields response surface to have a highly rugged landscape, and the optimization techniques, developed under deterministic settings, fail to search the global optimum in such a stochastic setting. The Bayesian optimization (BO) is a probabilistic approach to handle the observation noise, so it is robust in optimizing a stochastic function [73]. In the BO, a stochastic function is approximated by a surrogate function, inferred by the Gaussian process regression (GPR) [84], and the BO samples a new evaluation point from the inferred surrogate function. Unlike the heuristic algorithms that are memoryless except at the current point, the Bayesian optimization uses the entire sample history to compute a posterior of the stochastic function. Theoretical analysis on the BO has been studied extensively [50, 86], and this analysis provides the convergence of the stochastic function with some conditions [14].

Compared to the above methods, the proposed one utilizes both dynamic calibration, which is based on the sampling from a set of posterior distributions inferred from SMC with Synthetic Likelihood for each merged regimes, and heterogeneous calibration, which is based on GPR-based Bayesian optimization. Table 2 illustrates the detail comparisons between the proposed method and the previous works.

### 3.2 Previous research on agent-based model calibration

Before getting into the details on the agent-based model calibration, let us begin with a brief introduction of Agent-Based Model(ABM) for better understanding. The ABM describes a system of interest through the interactions among the heterogeneous agents residing in the system, and each intends to approach their own goals [62]. Such a bottom-up modeling perspective allows a close fit between the model with the real-world, in particular, where many non-intuitive real-world phenomena arise from the decentralized micro-level agent behaviors [20]. From such characteristics, the ABM has been applied to battle-level military [17, 44, 69], microscopic or mesoscopic economic [80, 81], biological [15], and disaster evacuation [1] domains.

Although the bottom-up approach presents a highly descriptive model, ABMs are asked for the lack of theoretic explanations on their results. As an effort for resolving this, modelers proves the validity of their ABMs through either a qualitative alignment to the well-known emergent behaviors [16, 76] or a quantitative comparison with the real-world empirical data [3, 33, 40] (please refer to Table 3).

As an example of the qualitative methods, Calvez and Hutzler [16] investigates if an ant-foraging ABM generates the well-known emergent behaviors in ecology, so they defined a domain-knowledge-incorporated fitness function that measures the degree of emergence for the collective behavior of interest. With the fitness function, they evaluated a number of emergent behaviors such as the formation of the ant lines and the total time taken to return all the food to the nest.

Other than a heuristic approach that is largely lack of theoretical analysis [33], the quantitative method moves toward a probabilistic surrogate meta-model approach that provides the theoretic guarantee and the estimation of the global optimal point of an irregular response surface. For example, [3] estimates the surrogate meta-model with Gaussian process regression, and Lamperti et al. [40] infers the meta-model by using

**Table 3** Previous research on agent-based model calibration

| Qualitative calibration | Surrogate model | Optimization algorithm | Input | Fitness measure of desired emergent behaviors |
|---|---|---|---|---|
| Calvez and Hutzler [16] | × | Genetic Algorithm | – | Quantity of food brought back for given period/Time to bring all the food back/Number of ant lines |
| Stonedahl and Wilensky [76] | × | Genetic Algorithm | Binary encoded input parameters | Variance of velocity over agents/Variance of velocity over time/Veeness (v-shaped formation) |
| Quantitative Calibration | Surrogate Model | Optimization Algorithm | Output Summary Statistics | Discrepancy measure between observation and simulation |
| Barde and Van Der Hoog [3] | GPR | Latin Hypercube | Extracted transition matrix from CTW [89] | Markov Information Criterion [2] |
| Heppenstall et al. [33] | × | Evolution Algorithm | Spatial average of oil price | Standardized Root Mean Squared Error |
| Lamperti et al. [40] | XGBoost | Sobol Sampling [53]/Uncertainty Sampling [] | Average time-series macro quantities | $F_1$ measure as binary discrepancy/Mean squared error as real-valued discrepancy |
| ***Proposed Dynamic Calibration*** | × | Sequential Monte Carlo [72] | Average number of transactions/price indices | Likelihood approximated by Synthetic Likelihood |
| ***Proposed Heterogeneous Calibration*** | GPR | Mixture of Bayesian Optimization [34] | Average number of transactions/price indices | Mean Absolute Percentage Error |

the XGBoost algorithm. However, the benefits of flexible meta-model comes at the cost of the increased number of simulation execution. In particular, the number of simulation run increases exponentially by dimension. Therefore, it is prohibitive to use the nearly-orthogonal Latin hypercube sampling, the sobol sampling [53], or the uncertainty sampling [41] for the next point selection algorithms. We want to recall that the purpose of calibration is not to discover the entire shape of the surrogate function, and the calibration should incorporate more efficient and robust optimization techniques, such as the Bayesian optimization.

Moreover, considering the structure of ABM, the expression power with only the static parameters may not be sufficient to capture the underlying non-stationary dynamics in the real world. In such cases, It is required that some well-designed ABMs utilize dynamically and/or heterogeneously varying parameters to express the complex behavior. The introduction of such various types of parameters would boost the modeling power by expanding the degree of freedom. However, the above previous works barely concerned on these traits in ABM, which eventually may lead to a question on their generality. Also, Considering such conditions, it would be indispensable that both more efficient and robust optimization techniques be applied to the ABM calibration.

The proposed framework incorporates dynamic and heterogeneous parameter calibrations to resolve the above considerations. Specifically, the dynamic calibration aims to minimize the temporal divergence between the simulation and real world using a similar concept of SMC [72]. Also, the heterogeneous calibration aims to overcome a possible disadvantage occurring from the link between the macroscopic aggregate behaviors and the microscopic agent-level variables, using the mixture of Bayesian optimization [34]. By the iterative process using the dynamic and the heterogeneous calibrations, the proposed framework utilizes the hidden structures extracted from the simulation for improving the calibration results. Furthermore, since the proposed method also causes a significant increase in the optimization dimension, the proposed framework adopts machine learning techniques to restrict the rate at which the optimization dimension increases. Table 3 provides the detail comparisons of the proposed method and the previous ABM calibration works.

## 4 Preliminary

In this section, we introduce the machine learning algorithms that we used in our calibration framework to extract the hidden structures. These machine learning algorithms were used as components in our calibration framework. We presented a set of clustering models, a latent representation model, and a Bayesian optimization method with the response surface fitting model.

*Clustering model for temporal regime detection* The hidden Markov model (HMM) [7, 36] is a statistical graphical model that estimates the optimal regime indices $\{\mathcal{R}^t\}_{t=1}^T$ from the time-dependent observation data $\{O^t\}_{t=1}^T$. The latent regime $\mathcal{R}^t$ at time $t$ is a scalar-valued regime index that is determined from the latent regime $\mathcal{R}^{t-1}$ of the previous timestep and the transition probability $\boldsymbol{\pi}$. The observation $O^t$ at time $t$ is a multi-dimensional vector that is assumed to be generated from the emission distribution $F$, with emission parameter $\phi_{\mathcal{R}^t}$. below is the generative process of the HMM.

$$\begin{aligned} \mathcal{R}^1 &\sim Mult(\boldsymbol{\pi}_{init}) \\ \mathcal{R}^t|\mathcal{R}^{t-1} &\sim Mult(\boldsymbol{\pi}_{\mathcal{R}^{t-1}}) \\ O^t|\mathcal{R}^t &\sim F(\boldsymbol{\phi}_{\mathcal{R}^t}) \end{aligned} \tag{2}$$

$\boldsymbol{\pi}_{init}$ is the multinomial probability of assigning the cluster index $\mathcal{R}^1$ on the initial timestep, $\pi_{\mathcal{R}^{t-1},\mathcal{R}^t} = p(\mathcal{R}^t|\mathcal{R}^{t-1})$ is the probability of transiting from $\mathcal{R}^{t-1}$-th regime to $\mathcal{R}^t$-th regime, and *Mult* denotes a multinomial distribution. In this study, the emission distribution is Gaussian distribution, and the emission parameters are the mean and variance.

Regime detection, using the HMM, requires estimating both the regime indices and the HMM parameters. First, the HMM parameters $\boldsymbol{\pi}_{init}$, $\boldsymbol{\pi}_v$, and $\boldsymbol{\phi}$ are estimated using the Baum Welch algorithm [61], a variant of the expectation-maximization algorithm [6], given observation $\{\mathcal{O}^t\}_{t=1}^T$. Second, the Viterbi algorithm [61], a dynamic programming algorithm [5], infers the optimal regime indices $\{\mathcal{R}^t\}_{t=1}^T$, given trained HMM parameters and observations. Note that to maximize the likelihood, the similar observations will be grouped together.

*Clustering models for agent heterogeneity* In this subsection, we introduce the probabilistic clustering algorithms required for heterogeneous calibration: a parametric Gaussian mixture model (GMM) [7] and a nonparametric Dirichlet process mixture model (DPMM) [79]. The idea is to group agents that share similarities in their state variables. Similar to the HMM, the clustering algorithms have time independent cluster assignment variables $\{c_a\}_{a=1}^A$ as latent variables (subscript $a$ denotes for the agents), and emission distributions $F$ with emission parameters $\boldsymbol{\phi}_{c_a}$. The emission distribution is Gaussian distribution for both the GMM and DPMM, and the emission parameters are the mean and the variance.

The GMM estimates the observation true density through the combination of the Gaussian distribution. Given training dataset $\mathcal{D}_{lat} = \{z_a|a = 1, ..., A\}$ ($z_a \in \mathbb{R}^L$ is the latent vector of the agent state trajectory $x_a \in \mathbb{R}^{Q\times T}$, see Variational Autoencoder), the generative model of the GMM is as follows:

$$\begin{aligned} c_a &\sim Categorical(\boldsymbol{\pi}) \\ z_a|c_a &\sim \mathcal{N}(\mu_{c_a}, \Sigma_{c_a}) \end{aligned} \tag{3}$$

The likelihood $p(z_a|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ of a single instance $z_a$ is the mixture of the Gaussian distribution is given by $\sum_{k=1}^K \pi_k \mathcal{N}(\mu_k, \Sigma_k)$. The full likelihood $p(\mathcal{D}_{lat}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multiplication $\prod_{a=1}^A p(z_a|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$, so the exponentially increasing terms cause the full likelihood to be intractable. Therefore, the EM algorithm [52], an iterative algorithm for maximum likelihood estimation, estimates both the optimal maximum likelihood parameters and the optimal cluster assignments. The algorithm iterates, to calculate the likelihood, and update the parameters.

Although the GMM is advantageous for its simplicity and fast estimation, it is affected by the pre-selected number of clusters $K$, which requires an additional model selection procedure to choose data-adaptive $K$. Having the observation that a common prior for $\boldsymbol{\pi}$ in the Bayesian GMM is the Dirichlet distribution, the DPMM selects a data-adaptive $K$ by assuming the Dirichlet process [79] as the prior for the mixture weights $\boldsymbol{\pi}$, where the Dirichlet process is the infinite-dimensional generalization of the Dirichlet distribution. A mixture weight $\boldsymbol{\pi} = \{\pi_k\}_{k=1}^\infty$, sampled from Dirichlet process, is constructed through the stick-breaking process [70], or GEM distribution [21]; the weight $\pi_k$ is equal to $\pi'_k \prod_{k'=1}^{k-1}(1 - \pi'_{k'})$, with $\pi'_k \sim Beta(1, \gamma)$, where the hyperparameter $\gamma$ controls the magnitude of $K$. The generative process of DPMM is as follows:

$$\begin{aligned} \boldsymbol{\pi} &\sim GEM(\gamma) \\ c_a &\sim Mult(\boldsymbol{\pi}) \\ \boldsymbol{\phi}_k &\sim G_0 \\ z_a|c_a &\sim F(\boldsymbol{\phi}_{c_a}) \end{aligned} \tag{4}$$

The DPMM assumes the prior distribution for the mixture weights $\boldsymbol{\pi}$ and the emission parameters $\boldsymbol{\phi}$. The emission distribution of the DPMM is the Gaussian distribution, and the emission parameters are the mean $\mu_k$ and the variance $\Sigma_k$. The prior distribution $G_0$ of $\phi_k = (\mu_k, \Sigma_k)$ is the Gaussian-Inverse-Wishart distribution [54]. The parameters $\boldsymbol{\pi}$ and $\boldsymbol{\phi}$ of DPMM are inferred by maximizing the posterior distribution $p(\boldsymbol{\pi}, \boldsymbol{\phi}|\{z_a\}_{a=1}^{A})$. However, the posterior distribution in the DPMM is intractable; therefore, the maximum a posteriori parameters are estimated through a Gibbs sampling algorithm [55] in this paper.

*Latent representation learning model* Agent-level state variables often have a large dimension, which causes failure in extracting meaningful clusters when applying the above mixture models, due to the curse of dimension. The failure arises from the characteristic of the Gaussian distribution. The $(Q \times T)$-variate Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ has most of its mass on the ellipsoidal shell of the Mahalanobis radius [43] $\sqrt{\text{trace}(\Sigma)}$. When sampling from $\mathcal{N}(\mu, \Sigma)$, $2^{O(Q\times T)}$ number of samples are required to obtain a point within the radius $\sqrt{\text{trace}(\Sigma)}/2$ from $\mu$ [19], which means the Gaussian distribution in high dimension cannot represent the density of clustered data appropriately. This phenomena leads agents assigned in an identical cluster to have distant agent-level state variables. Therefore, we introduced a latent representation learning algorithm to compress the high-dimensional data into low dimensional data.

The variational autoencoder (VAE) [37] is a probabilistic latent variable model that extracts the latent representation $\mathcal{D}_{lat} = \{z_a|a = 1, ..., A\}$ from the agent-level states data $\mathcal{D}_{agent} = \{x_a|a = 1, ..., A\}$, where $z_a \in \mathbb{R}^L$ and $x_a \in \mathbb{R}^{Q\times T}$. The structure of the VAE consists of two neural networks: an encoder network $\text{Enc} : \mathbb{R}^{Q\times T} \to \mathbb{R}^L$ and a decoder network $\text{Dec} : \mathbb{R}^L \to \mathbb{R}^{Q\times T}$. The hidden latent representation $z_a$ of the original data $x_a$ is the output of the encoder network $z_a = \text{Enc}(x_a)$. The extracted representation $z_a$ is subsequently propagated to the decoder network to regenerate the original data as $\tilde{x}_a = \text{Dec}(\text{Enc}(x_a))$. The vanilla version of the VAE assumes the encoder output distribution has a standard $L$-variate Gaussian distribution $\mathcal{N}(0, I)$ as a prior.

To make the reconstructed output $\tilde{x}_a$ similar to the original input $x_a$, the VAE maximizes the log-likelihood $\log p(x)$. However, because the log-likelihood is intractable to calculate, an alternative loss function, the evidence lower bound (ELBO), is proposed as a supporting lower bound function. In detail, the formula below shows the ELBO.

$$\begin{aligned} \log p(x) &= \log \int_z p(x|z)p(z)dz \\ &= \log \int_z q(z|x)\frac{p(x|z)p(z)}{q(z|x)}dz \\ &\geq \int_z q(z|x) \log \frac{p(x|z)p(z)}{q(z|x)}dz \\ &= \mathbb{E}_q[\log p(x|z)] - KL(q||p) \end{aligned} \tag{5}$$

The reconstruction term $\mathbb{E}_q[\log p(x|z)]$ calculates the cross entropy between the reconstructed data $\tilde{x}_a$ and the original data $x_a$. The regularization term, KL divergence $KL(q||p)$, measures how much information is lost by using the encoder distribution $q(z|x)$ to represent

the prior distribution $p(z)$. Using a stochastic gradient descent method, the neural network parameters are learned. Note that maximizing the ELBO is equivalent to minimizing the KL divergence between the approximate parametrized posterior $q(z|x)$ and the true posterior $p(z|x)$, i.e. the VAE estimates the posterior distribution $p(z|x)$ using the proposal posterior distribution $q(z|x)$ estimated through amortized inference.

*Gaussian process regression-based Bayesian optimization* In this subsection, we introduce a Bayesian method to optimize a stochastic objective function, which consists of two phases: function estimation phase and decision phase for the future evaluations. The Gaussian process regression (GPR) [8, 63] is a function estimation phase that estimates the predictive posterior of the stochastic function under the given dataset $\mathcal{D}_{1:n} = \{(\theta_1, \hat{d}_1), ..., (\theta_n, \hat{d}_n)\}$ where $\hat{d}_n$ is the Monte Carlo estimation $d(\frac{1}{J}\sum_{j=1}^{J}\mathcal{M}(\theta_n;\omega_j), x_{obs})$. The explanatory variables are the multi-dimensional parameters $\theta$, and the response variable is the scalar-valued discrepancy function $d$.

The objective function $d$ is distributed according to the Gaussian process prior $GP(0, \mathsf{K})$, where $\mathsf{K}$ is the covariance of the prior. The GPR estimates the predictive distribution of a random variable with index $\theta$ assumed as Gaussian distribution, as below.

$$p(d(\theta)|\mathcal{D}_{1:n}) \sim \mathcal{N}(\kappa^T cov^{-1} F, \mathsf{K}_{\theta,\theta} - \kappa^T cov^{-1}\kappa) = \mathcal{N}(\mu_d(\theta|\mathcal{D}_{1:n}), \sigma_d^2(\theta|\mathcal{D}_{1:n})), \quad (6)$$

Here, $\kappa = [\mathsf{K}_{\theta,\theta_1}, ..., \mathsf{K}_{\theta,\theta_n}]^T$, $F = \left[\hat{d}_1, ..., \hat{d}_n\right]^T$, and $(cov)_{i,j} = \mathsf{K}_{\theta_i,\theta_j} + \frac{1}{\beta}\delta_{\{i=j\}}$, where $\mathsf{K}_{\theta,\theta'}$ is the kernel distance between $\theta$ and $\theta'$. The magnitude of $\beta$ represents the variance for the Monte Carlo method to calculate the discrepancy $d$. The kernel function is chosen based on the joint consideration of the response curve shape and the meaning of each dimension of $\theta$. The training procedure for the GPR optimizes the kernel function hyperparameters that determine the shape of the kernel function $\mathsf{K}$.

The decision phase uses Bayesian optimization [12, 60, 67, 73]. The Bayesian optimization forms an acquisition function, and chooses the next evaluation point $\theta_{n+1}$ by optimizing the acquisition function $f$.

$$\theta_{n+1} = \arg\max_{\theta\in\mathbb{H}} f(\theta|\mathcal{D}_{1:n}) \quad (7)$$

Here, the search space $\mathbb{H}$ is defined by $\mathbb{H} = \left\{\theta \in \Theta \middle| \|\theta - \theta_i\|_2 \geq r \text{ for all } \theta_i \in \mathcal{D}_{1:n}\right\}$ with the given radius $r \geq 0$. The strictly positive $r > 0$ forces the newly generated parameter $\theta_{n+1}$ to not be sampled from algready explored region, so it prevents optimization being stuck in a local optimum [25, 42].

An acquisition function embeds the exploration-exploitation selection criteria. A fully exploiting acquisition function is $f(\theta|\mathcal{D}_{1:n}) = -\mu_d(\theta|\mathcal{D}_{1:n})$, and the other extreme of acquisition function is $f(\theta|\mathcal{D}_{1:n}) = \sigma_d(\theta|\mathcal{D}_{1:n})$ so it fully explores the unvisited region in the search space.

The expected improvement (EI) [63, 73] acquisition function with $f(\theta|\mathcal{D}_{1:n}) = \int_0^\infty P(d(\theta) \leq \underline{\hat{d}}_{1:n} - m)dm$, where $\underline{\hat{d}}_{1:n} = \min_{1:n} \hat{d}_i$, automatically trades off the exploration and the exploitation ratio. The closed-form EI formula is given below.

$$f(\theta|\mathcal{D}_{1:n}) = (\underline{\hat{d}}_{1:n} - \mu_d(\theta|\mathcal{D}_{1:n}))\boldsymbol{\Phi}\left(\frac{\underline{\hat{d}}_{1:n} - \mu_d(\theta|\mathcal{D}_{1:n})}{\sigma_d(\theta|\mathcal{D}_{1:n})}\right) + \sigma_d(\theta|\mathcal{D}_{1:n})\phi\left(\frac{\underline{\hat{d}}_{1:n} - \mu_d(\theta|\mathcal{D}_{1:n})}{\sigma_d(\theta|\mathcal{D}_{1:n})}\right) \quad (8)$$

Here, $\phi$ and $\boldsymbol{\Phi}$ are the probability density function and the cumulative density function of the standard Gaussian distribution $\mathcal{N}(0, 1)$, respectively. The first term in Eq. 8 is the exploration term, that is, the predicted difference between the minimum of the simulation errors

$\hat{d}_{1:n}$ and the predicted mean $\mu_d(\theta|\mathcal{D}_{1:n})$, penalized by the probability of improvement $\Phi\left(\frac{\hat{d}_{1:n}-\mu_d(\theta|\mathcal{D}_{1:n})}{\sigma_d(\theta|\mathcal{D}_{1:n})}\right)$. The second term forces the Bayesian optimization to explore the parameter space based on the uncertainty term $\sigma_d(\theta|\mathcal{D}_{1:n})$, penalized by $\phi\left(\frac{\hat{d}_{1:n}-\mu_d(\theta|\mathcal{D}_{1:n})}{\sigma_d(\theta|\mathcal{D}_{1:n})}\right)$. For this reason, the EI is a balanced exploration and exploitation method in its nature. It gained popularity, as a result of both theoretical analyses [14, 27, 50, 65, 86] and experimental successes [42, 68, 73].

Besides the use of the EI in its fundamental form, the ratio of the exploitation term and the exploration term can be controlled by adding weight $w_n$, as below.

$$wEI(\theta|\mathcal{D}_{1:n}, w_n) = (1-w_n)(\hat{d}_{1:n} - \mu_d(\theta|\mathcal{D}_{1:n}))\Phi\left(\frac{\hat{d}_{1:n} - \mu_d(\theta|\mathcal{D}_{1:n})}{\sigma_d(\theta|\mathcal{D}_{1:n})}\right) + w_n\sigma_d(\theta|\mathcal{D}_{1:n})\phi\left(\frac{\hat{d}_{1:n} - \mu_d(\theta|\mathcal{D}_{1:n})}{\sigma_d(\theta|\mathcal{D}_{1:n})}\right) \tag{9}$$

The above weighted EI (wEI) [74] acquisition function is equivalent to the EI, when $\omega_n = 0.5$.

## 5 Proposed calibration methodology

In this Section, we describe our calibration framework for the ABM with a validation dataset $x_{obs}$. The description starts from the overall illustration of the calibration framework. Then, we describe two detailed calibration components, which are *dynamic calibration* and *heterogeneous calibration*.

### 5.1 Overall calibration framework

Our calibration framework is designed to maximize the fitness between simulation and observation by expanding the parameter types to be calibrated in two different ways, temporal expansion and heterogeneous expansion. When the optimization parameter $\theta$ is $d$-dimensional, we optimize $D^t$ number of temporally varying parameters and $D^k = d - D^t$ number of agent-specifically varying parameters. Dynamic calibration extends the optimization dimensionality from $D^t$ dimension to $D^t \times M$ dimension, where $M$ is the degree of freedom for the dynamic parameters. Similarly, heterogeneous calibration extends the dimension from $D^k$ into $D^k \times K$, where $K$ is the number of the agent sub-populations. Note that we only include the parameters to be calibrated in the set of $\theta$. Figure 2 illustrates the calibration process overview, and the details of the algorithm is presented under Algorithm 1.

Before the iterative calibration steps, the agents in the model are clustered, see Line 2 in Algorithm 1. After obtaining the agent sub-populations, we commence the calibration iterations from the randomly initialized parameters. The effect of the choice of the initial parameters are examined based on experiments in Sect. 6.1.3.

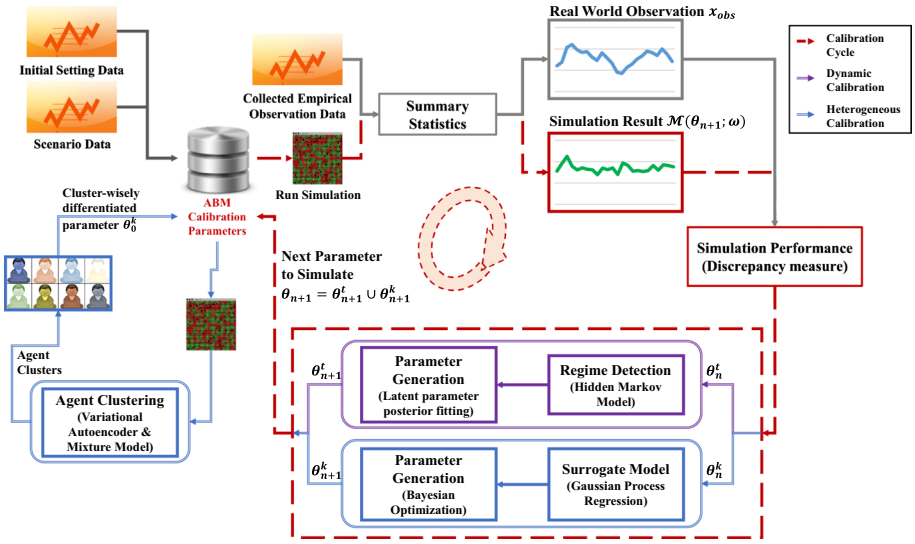



**Fig. 2** The suggested calibration framework overview. To initiate the iterative calibration cycle (red dotted line) to optimize the objective function, the framework partitions the agent population through *Agent Clustering Component* (blue double line) at the beginning of the calibration procedure. Each iteration adjusts either the dynamic parameters by dynamic calibration (purple double line) or the heterogeneous parameters by heterogeneous calibration (blue double line) in the iteration cycle (Color figure online)

A calibration iteration consists of one performance evaluation step and two calibration components.

1. Our iteration commence from the simulation evaluation phase. The simulation discrepancy $\hat{d}(\theta_n)$ is measured from the evaluated simulation results $\mathcal{M}(\theta_n;\omega)$, with the parameters $\theta_n$ for the $n$-th iteration, see Line 13 in Algorithm 2 and Line 13 in Algorithm 1. The function $d$ is instantiated for each application case, and $d$ could be either the simple Euclidean distance, KL divergence, or negative log-likelihoods, among others [29, 47, 48]. We specified $d$ for each of our examples in Sects. 5.2 and 5.3.
2. After calculating $d$, we alternately performe *dynamic calibration* or *heterogeneous calibration*, which are the two major components of the calibration framework. The proposed calibration framework alternates each component with predetermined iterations: $N^t$ iterations for the dynamic component and $N^k$ iterations for the heterogeneous component.

    (a) Dynamic calibration performes the regime detection; see Line 14 in Algorithm 2, and generates the next set of the dynamic parameter candidates $\theta_{n+1}^t$; see Line 15 in Algorithm 2.
    (b) Heterogeneous calibration learnes the response surface; see Line 25 in Algorithm 3, and generates the next heterogeneous parameter $\theta_{n+1}^k$; see Line 26 in Algorithm 3.

We wish to comment on the performance (or time) of the proposed method as a function of the model scale, such as the number of agents in the ABM. The whole calibration

process consists of the dynamic and the heterogeneous calibrations. Of the two, only the computational performance (or time) of the heterogeneous process may seem to depend on the number of agents. However, the heterogeneous calibration uses the number of agents only in the process of clustering, and the remaining procedures do not depend on the number of agents anymore. Thus, we can argue that the computational performance (or time) would not be significantly degraded by the number of agents. Having said that, it would need extra experiments for the quantitative analysis, so this issue would be thoroughly covered in future work.

The following subsections illustrate details on the dynamic (Sect. 5.2) and the heterogeneous (Sect. 5.3) calibrations.

**Algorithm 1:** Calibration Framework Algorithm

**input :** Initial parameters $\theta_0$
**output:** Initial heterogeneous parameters $\theta_0^k$

**Function** AgentClustering($\theta_0$): // Clustering
  Run simulation J replications with $\theta_0$ to obtain agent-level simulation result $\mathcal{D}_{agent} = \{x_a | a = 1, ..., A\}$
  Obtain latent representation $\mathcal{D}_{lat}$ of $\mathcal{D}_{agent}$ by applying VAE
  Obtain agent cluster assignments by applying one of mixture models (GMM or DPMM)
  Generate heterogeneous parameters $\theta_0^k$ by assigning random values for each cluster
  **return** $\theta_0^k$

**input :** Input static parameter combination $\theta_0$ // Selected calibration parameters are undivided initially
**output:** Calibrated parameter combination $\theta^*$ // Calibrated parameters are divided according to timestep and clusters

**Function** CalibrationFramework($\theta_0$):
  $\theta_0^k$=AgentClustering($\theta_0$) // Clustering phase
  **for** *n in range(N)* **do** // n : calibration iteration
    **if** $0 \leq n - \left\lfloor \frac{n}{N^t+N^k} \right\rfloor (N^t + N^k) < N^t$ **then** // Dynamic calibration
      **for** *i in range(I)* **do** // i : candidate hypothesis index
        Run simulation $\mathcal{M}(\theta_n^{t,i} \cup \theta_n^k; \omega_j)$ for $j = 1, ..., J$ where $\theta_n^i = \theta_n^{t,i} \cup \theta_n^k$ // Evaluation phase
        Obtain $\hat{\mu}_n^{t,i} = \mathbb{E}_J[\mathcal{M}(\theta_n^{t,i} \cup \theta_n^k; \omega_j)]$ and $\hat{\Sigma}_n^{t,i} = \mathrm{Var}_J[\mathcal{M}(\theta_n^{t,i} \cup \theta_n^k; \omega_j)]$
      $\theta_{n+1}^t$=DynamicCalibration($\theta_n^t, \hat{\mu}_n^{t,i}, \hat{\Sigma}_n^{t,i}$) (see Algorithm 2) // Sampling phase
      $\theta_{n+1}^k = \theta_n^k$
    **else if** $0 \leq n - \left\lfloor \frac{n}{N^t+N^k} \right\rfloor (N^t + N^k) - N^t < N^k$ **then** // Heterogeneous calibration
      Select best dynamic parameter $\theta_n^{t,i^*}$ out of $I$ candidate hypotheses
      Run simulation $\mathcal{M}(\theta_n^{t,i^*} \cup \theta_n^k; \omega_j)$ for $j = 1, ..., J$ // Evaluation phase
      Obtain $\hat{d}_n^k = d(\mathbb{E}_J[\mathcal{M}(\theta_n^{t,i} \cup \theta_n^k; \omega_j)], x_{obs})$
      $\theta_{n+1}^k$=HeterogeneousCalibration($\theta_n^k, \hat{d}_n^k$) (see Algorithm 3) // Sampling phase
      $\theta_{n+1}^t = \theta_n^t$
  Set $\theta^* = \theta_{n^*}^{t,i^*} \cup \theta_{n^*}^k$ to have the lowest discrepancy $d(\theta^*)$
  **return** $\theta^* = \theta_{n^*}^{t,i^*} \cup \theta_{n^*}^k$ // Final calibrated parameters

## 5.2 Dynamic parameter calibration

The dynamic parameter calibration [51] finds the optimal $D^t$ number of dynamically varying parameters $\theta^{t,*} \in \mathbb{R}^{D^t \times T}$ that yielded the most consistent summary statistics $\mathbb{E}_\omega[\mathcal{M}(\theta^{t,*} \cup \theta^k; \omega)]$ with the observation $x_{obs} \in \mathbb{R}^{p \times T}$. Here, $\theta^k$ denotes for the heterogeneous parameters, and the dynamic calibration optimizes the dynamic parameters $\theta^t$ without the joint consideration with $\theta^k$. Formally, the dynamic calibration is the following optimization task.

$$\theta^{t,*} = \arg\min_{\theta^t} d(\mathbb{E}_{\omega \in \Omega}[\mathcal{M}(\theta^t \cup \theta^k; \omega)], x_{obs}) \tag{10}$$

The increment in the parameter numbers is significant, if the simulation becomes a long execution with total timesteps $T$. Therefore, we limit the increment by assuming that the dynamic parameters are identical for the identified temporal regimes over the simulation executions. This assumption regulates the optimization dimension from $D^t \times T$ to $D^t \times M$, by limiting the degree of freedom $M$ for the variability of dynamic parameters.

The degree of freedom in dynamic calibration stems from the temporally clustered regimes. In other words, the dynamic calibration optimizes the regime-wisely diversified dynamic parameters. The regimes in the dynamic calibration differentiates the well-fitted periods from the poorly-fitted periods. This distinction enables the poorly-fitted regimes and the well-fitted ones to explore and exploit the parameter space, separately.

### 5.2.1 Simulation performance in dynamic calibration

Dynamic calibration is a particle-based algorithm in which the total $\mathtt{I}$ number of particles are evaluated for every calibration iterations. Similar to the Sequential Monte Carlo approach introduced in Sect. 3, the dynamic calibration iteratively samples the next set of particles $\{\theta_{n+1}^{t,i} | i = 1, ..., \mathtt{I}\}$ from the intermediate posterior distributions. The dynamic calibration incorporates the regime detection algorithm into the SMC to generate dynamically switching parameters for the next evaluation phase. Further, dynamic calibration estimates the mean-field posterior distributions for each of the detected regimes separately, in which the parameters for the next iteration are sampled from. In this subsection, we introduce a discrepancy measure in the process of inferring the intermediate posterior distribution.

Dynamic calibration calculates the likelihood as a simulation performance measure via the SL approach (see Sect. 3), where SL approach imposes an explicit parametric distribution on the summary statistics. The choice of the explicit assumption can vary for simulation models. An example of the explicit assumption can be the Gaussian distribution. Given that the summary statistics $\mathcal{M}(\theta_n^{t,i} \cup \theta_n^k; \omega)$ is a $(P \times T)$-dimensional multivariate random variable on the nuisance stochastic variable $\omega$, the $J$ ensembles of realized summary statistics $\{\mathcal{M}(\theta_n^{t,i} \cup \theta_n^k; \omega_j) | j = 1, ..., J\}$ has its empirical moments. Given empirical moments, the likelihood $L_n^{t,i}$ is given as $L_n^{t,i} = \mathcal{N}(x_{obs}^t | \hat{\mu}_n^{t,i}, \hat{\Sigma}_n^{t,i})$, where $\hat{\mu}_n^{t,i}$ is the cross-sectional average of the realizations, and $\hat{\Sigma}_n^{t,i}$ is the cross-sectional covariance of the realizations at time $t$. The likelihood value $L_n^{t,i}$ at time $t$ is used as the weight of the $i$-th particle to estimate the posterior distribution for the corresponding regime.

**Algorithm 2:** Dynamic Calibration

**Input** : Dynamic parameters $\theta_n^{t,i}$
**Input** : Average of simulation result $\hat{\mu}_n^{t,i}$
**Output:** Merged regimes $\Gamma_n^m$

**Function** RegimeDetection($\theta_n^{t,i}, \hat{\mu}_n^{t,i}$): // Temporal Clustering
  **for** *i in range(I)* **do**
    Obtain hidden regime $\{\mathcal{R}_n^{t,i} | t = 1, ..., T\}$ for $i$-th candidate, using HMM
  Obtain overall regime detection $\Gamma_n^m$ by merging hidden regimes of candidate hypotheses
  **return** $\Gamma_n^m$,

**Input** : Dynamic parameters $\theta_n^{t,i}$
**Input** : Likelihoods $L_n^{t,i}$
**Input** : Merged regimes $\Gamma_n^m$
**Output:** Next dynamic parameter $\theta_{n+1}^{t,i}$

**Function** ParameterGeneration($\theta_n^{t,i}, L_n^{t,i}, \Gamma_n^m$): // Generate next parameters
  **for** *m in range(M)* **do**
    **for** *l in range($D^t$)* **do** // $D^t$ : number of dynamic parameters
      Estimate beta distribution Beta($\alpha_{m,l}, \beta_{m,l}$)
    Generate next parameter hypotheses $\theta_{n+1}^{t,i}$ from Beta($\alpha_{m,l}, \beta_{m,l}$)
  **return** $\theta_{n+1}^{t,i}$

**Input** : Dynamic parameter $\theta_n^t$, simulation statistics $\hat{\mu}_n^{t,i}, \hat{\Sigma}_n^{t,i}$
**Output:** Next dynamic parameter $\theta_{n+1}^t$ to evaluate

**Function** DynamicCalibration($\theta_n^t, \hat{\mu}_n^{t,i}, \hat{\Sigma}_n^{t,i}$): // Sampling phase
  Obtain likelihoods $L_n^{t,i} = \mathcal{N}(x_{obs}^t | \hat{\mu}_n^{t,i}, \hat{\Sigma}_n^{t,i})$
  $\Gamma_n^m$ =RegimeDetection($\theta_n^{t,i}, \hat{\mu}_n^{t,i}$) (see Section 5.2.2)
  $\theta_{n+1}^{t,i}$ =ParameterGeneration($\theta_n^{t,i}, L_n^{t,i}, \Gamma_n^m$) (see Section 5.2.3)
  **return** $\theta_{n+1}^t$ // Dynamic parameter particles to evaluate on next iteration

### 5.2.2 Regime detection component

Incorporating the regime detection algorithm, the HMM, into the calibration task suppresses the increment in the degree of freedom by dividing the simulation timesteps into a few pieces. We deploy the HMM algorithm (see Sect. 4) for each particle to cluster the well-fitted and poorly-fitted regimes. For the $i$-th particle, the time-dependent deviation $O_n^{t,i} = \hat{\mu}_n^{t,i} - x_{obs}^t$ is defined as the observable variable of the HMM. Also, the temporal regime $\mathcal{R}_n^{t,i} \in \{1, ..., K^t\}$ is the unobservable hidden variable of the HMM at the $t$-th timestep and the $i$-th particle. The crossover period where the deviation between the observation $x_{obs}$ and the simulation $\hat{\mu}_n^{t,i}$ drastically amplifies would be detected by the HMM. In consequence, the before and the after of the crossover period is separated by the HMM. Using the estimated temporal regimes for each particle, we merge the temporal regimes along particles into the merged regimes $\{\Gamma_n^m\}_{m=1}^M$. The $m$-th merged regime $\Gamma_n^m$ consists of timesteps $t_1$ and $t_2$ that satisfy $(\mathcal{R}_n^{t_1,1}, ..., \mathcal{R}_n^{t_1,I}) = (\mathcal{R}_n^{t_2,1}, ..., \mathcal{R}_n^{t_2,I})$. For each merged regime $\Gamma_n^m$, the parameter posterior distribution is estimated, and the next parameter values on the regime $\Gamma_n^m$ is sampled from the posterior. Therefore, the total number of the merged regimes $M$ becomes the degree of freedom for the variability of the next dynamic parameter $\theta_{n+1}^{t,i}$.

### 5.2.3 Parameter generation component

Given a fixed merged regime $\Gamma_n^m$, the next step is to estimate the posterior distribution on the parameter space. We apply two mean-field assumptions to the posterior distribution. The first mean-field assumption assumes that the posterior distribution on different merged

regimes is independent. The second mean-field assumption assumes that the posterior of the different dynamic parameters is independent. We discuss these assumptions further in Sect. 7.

The beta distribution is estimated from the parameter values and the parameter weights. First, the parameter values are normalized to be $\tilde{\theta}^{t,i}_{n,l} = (\theta^{t,i}_{n,l} - \underline{\theta}_l)/(\overline{\theta}_l - \underline{\theta}_l)$, where $\overline{\theta}_l$ and $\underline{\theta}_l$ are the upper bound and the lower bound of the $l$-th dynamic parameter. This normalization restricts all the parameter values in a unit interval [0, 1]. Second, the likelihood of the normalized parameter values $\tilde{\theta}^{t,i}_{n,l}$, with the attached weights $L^{t,i}_n$ being observed in the beta distribution with the shape parameters $\alpha_{m,l}$ and $\beta_{m,l}$, is given as below.

$$p(\{\tilde{\theta}^{t,i}_{n,l}, L^{t,i}_n\}_{t\in\Gamma^m_n, i\in\{1,\dots,\mathrm{I}\}}|\alpha_{m,l}, \beta_{m,l}) = \prod_{t\in\Gamma^m_n}\prod_{i=1}^{\mathrm{I}} L^{t,i}_n \mathrm{Beta}(\tilde{\theta}^{t,i}_{n,l}|\alpha_{m,l}, \beta_{m,l}) \tag{11}$$

The posterior distribution $p(\theta|x_{obs})$ at the $n$-th iteration is approximated as the Beta distribution $Beta(\tilde{\theta}^t_{n,l}|\alpha^*_{m,l}, \beta^*_{m,l})$ ($t \in \Gamma^m_n$), with maximum likelihood estimated shape parameters $\alpha^*_{m,l}$ and $\beta^*_{m,l}$.

We have three options to resample the next dynamic parameters on the inferred intermediate parameter distribution $\mathrm{Beta}(\alpha_{m,l}, \beta_{m,l})$:

1. *Sampling by Time*: For each timestep $t \in \Gamma^m_n$, sample $\hat{\theta}^{t,i}_{n+1,l}$ from $\mathrm{Beta}(\alpha_{m,l}, \beta_{m,l})$, and scale up the sample to generate the next parameter $\theta^{t,i}_{n+1,l} = \underline{\theta}_l + (\overline{\theta}_l - \underline{\theta}_l) \times \hat{\theta}^{t,i}_{n+1,l}$.
2. *Sampling by Regime*: The next parameter $\theta^{t,i}_{n+1,l}$ is constant throughout the merged regime $\Gamma^m_n$ to be a scaled-up sample from $\mathrm{Beta}(\alpha_{m,l}, \beta_{m,l})$.
3. *Mode Selection*: The next parameter $\theta^{t,i}_{n+1,l}$ is fixed throughout the merged regime $\Gamma^m_n$ to be a scaled-up sample from $\left\{\mu^{\mathrm{Beta}}_{m,l} + \left(i - \frac{\mathrm{I}+1}{2}\right)\sigma^{\mathrm{Beta}}_{m,l} \middle| i = 1, \dots, I\right\}$, where $\mu^{\mathrm{Beta}}_{m,l}$ and $\sigma^{\mathrm{Beta}}_{m,l}$ are the mean and the standard deviation of $\mathrm{Beta}(\alpha_{m,l}, \beta_{m,l})$, respectively.

*Sampling by Time* involves searching the most rapidly switching parameters across the three generation processes. However, the small parameter deviation at the initial timesteps affects the simulation dynamics afterwards. This temporal dependency hinders the dynamic calibration from estimating the accurate time-varying parameters subsequently. Furthermore, *Sampling by Time* is expected to estimate the over-fitted dynamic parameters. On the other hand, *Sampling by Regime* involves searching the parameters with an identical value at the identical regimes, which makes the calibrated parameter fluctuate less. Besides the sampling-based methods, *Mode Selection* eradicates the sampling error by resampling the next particle from the selected modes. We experimented on all the parameter-generating rules in Sect. 6.1.

### 5.3 Heterogeneous parameter calibration

Heterogeneous calibration resolves the simulation divergence arising from the agent's heterogeneity by minimizing the distance between the simulation $\mathbb{E}_\omega[\mathcal{M}(\theta^t \cup \theta^k)]$ and the validation data $x_{obs}$. Heterogeneous calibration optimizes $D^k$ number of heterogeneous parameters $\theta^k \in \mathbb{R}^{D^k \times K^k}$. Before introducing the algorithm, we present two assumptions. First, we assume that we have a fixed set of dynamic parameters $\theta^t$, which are the complements of the heterogeneous parameters from the set of calibration parameters $\theta$. Second, the heterogeneous parameters $\theta^k$ are static parameters differentiated by the agent sub-populations. The heterogeneous calibration solves the following problem.

$$\theta^{k,*} = \arg\min_{\theta^k} d(\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta^t \cup \theta^k;\omega)], x_{obs}) \tag{12}$$

The second assumption makes it possible to solve the above optimization problem. If $\theta^k$ is differentiated by the agents, then the number of optimization dimension becomes $D^k \times A$, where $A$ is the number of agents. This high dimensional problem requires innumerous number of simulation executions for calibration. In addition, the expansion of the parameter by the agents could be viewed as over-parametrization, which causes over-fitting. Therefore, differentiating the heterogeneous parameters by agent clusters mediates the intractability of optimization and potential over-fitting issue.

### 5.3.1 Simulation performance in heterogeneous calibration

Heterogeneous calibration assumes that the mean absolute percentage error (MAPE) is the discrepancy measure $d$; however, this selection can change per application case [29]. Heterogeneous calibration adjusts the heterogeneous parameters $\theta^k$ by optimizing the observable discrepancy $\hat{d}(\theta^k) = \left\| \frac{\frac{1}{J}\sum_{j=1}^{J} \mathcal{M}(\theta^t \cup \theta^k;\omega_j) - x_{obs}}{x_{obs}} \right\|_{L^1}$, which is the Monte Carlo estimation of the true discrepancy.

### 5.3.2 Agent clustering component

Heterogeneous calibration divides agents into several clusters using the agent-level simulation results $\mathcal{D}_{agent} = \{x_a\}_{a=1}^{A}$, with $x_a \in \mathbb{R}^{Q\times T}$. However, if the dimension $Q \times T$ is large (i.e. the execution of the simulation is lengthy and the agent-level states at time $t$ is high-dimensional), then the clustering algorithms suffer from the curse of dimension. Therefore, heterogeneous calibration compresses the data $\mathcal{D}_{agent}$ through a latent representation learning algorithm (Line 3 in Algorithm 3) via VAE (see Sect. 4) before applying clustering algorithm.

Then, using one of the mixture models introduced in Eqs. 3 or 4, the heterogeneous calibration divides the agent population into $K^k$ pieces. We test both the parametric mixture model, Eq. 3, and the nonparametric mixture model, Eq. 4, in our experiments. When we use the DPMM, the heavy-tailedness of the agent-level variables in the ABM results in too many number of optimal clusters [23, 39, 59]. Thus, we apply the nearest-neighbor algorithm to the clustering result of the DPMM, to reduce the number of clusters and to regulate the degree of freedom. With the obtained cluster assignments, the heterogeneous calibration differentiates heterogeneous parameters by assigning a random value to each cluster. Note that the heterogeneous parameters were static across the agents, before the agent-clustering phase.

**Algorithm 3:** Heterogeneous Calibration

**Input** : Gaussian process training data $\mathcal{D}_{1:n}$

**Function** ResponseSurface($\mathcal{D}_{1:n}$):
  Obtain optimal kernel parameters in Gaussian process regression to fit collected data $\mathcal{D}_{1:n}$

**Output:** Next heterogeneous parameters $\theta^k_{n+1}$

**Function** ParameterGeneration(): // Generate next parameter
  **if** $n < N_0$ **then** // Random search for burn-in iterations
    Randomly select $\theta^k_{n+1}$
  **else if** $n \geq N_0$ **then** // Bayesian optimization search
    Sample $\xi$ from uniform distribution on $[0, 1]$ // Exploration-exploitation selection probability
    **if** $\xi < \xi_1$ **then**
      Randomly select $\theta^k_{n+1}$
    **else if** $\xi_1 \leq \xi < \xi_1 + \xi_2$ **then** // Fully exploration
      $\theta^k_{n+1} = \arg\max_{\theta^k} \sigma_d(\theta^k|\mathcal{D}_{1:n})$
    **else if** $\xi_1 + \xi_2 \leq \xi < \xi_1 + \xi_2 + \xi_3$ **then** // Fully exploitation
      $\theta^k_{n+1} = \arg\min_{\theta^k} \mu_d(\theta^k|\mathcal{D}_{1:n})$
    **else if** $\xi \geq \xi_1 + \xi_2 + \xi_3$ **then** // Mixture of exploration-exploitation
      $\theta^k_{n+1} = \arg\max_{\theta^k} wEI(\theta^k|\mathcal{D}_{1:n}, w_n)$
  **return** $\theta^k_{n+1}$

**Input** : Heterogeneous parameter $\theta^k_n$
**Output:** Next heterogeneous parameter $\theta^k_{n+1}$

**Function** HeterogeneousCalibration($\theta^k_n, \hat{d}^k_n$): // Sampling phase
  Add input-output pair $\mathcal{D}_{1:n} = \mathcal{D}_{1:n-1} \cup \{(\theta^k_n, \hat{d}^k_n)\}$
  ResponseSurface($\mathcal{D}_{1:n}$) (see Section 5.3.3)
  $\theta^k_{n+1}$ =ParameterGeneration() (see Section 5.3.4)
  **return** $\theta^k_{n+1}$ // Next parameter to evaluate

### 5.3.3 Response surface learning component

Function optimization (Eq. 12) requires two phases: function approximation phase and decision phase for the future evaluations. The GPR is a probabilistic approach to infer the unknown discrepancy function using a given noisy dataset $\mathcal{D}_{1:n} = \{(\theta^k_1, \hat{d}^k_1), ..., (\theta^k_n, \hat{d}^k_n)\}$, where the empirical discrepancy $\hat{d}^k_n$ is assumed to sample from a Gaussian distribution. The Gaussian process prior GP(0, K) represents the space of functions that the true discrepancy function can lie on, where K is the prior covariance function or the kernel function. As described in Sect. 4 in detail, the GPR infers the posterior distribution of the stochastic function with $\mathcal{N}(\mu_d(\theta^k_{new}|\mathcal{D}_{1:n}), \sigma_d(\theta^k_{new}|\mathcal{D}_{1:n})) = \mathcal{N}(\kappa^T cov^{-1} F, \mathsf{K}_{\theta^k_{new}, \theta^k_{new}} - \kappa^T cov^{-1} \kappa)$. The parameters of the kernel function are point estimated by maximizing the likelihoods.

### 5.3.4 Parameter generation component

In this subsection, we describe the decision phase for the future evaluations in the heterogeneous calibration. Based on the surrogate function inferred by the GPR, the Bayesian optimization infers the discrepancy function $d$ by adaptively selecting the heterogeneous parameters $\theta^k_{n+1}$ for the next evaluation. The selection of the next parameter comes from the inner optimization problem $\theta^k_{n+1} = \arg\max_{\theta^k \in \mathbb{H}} f(\theta^k|\mathcal{D}_{1:n})$ of optimizing the acquisition function $f$. Here, the search space $\mathbb{H} = \left\{\theta^k \middle| \|\theta^k - \theta^k_i\|_2 \geq r_0 \text{ for all } \theta^k_i \text{ in } \mathcal{D}_{1:n}\right\}$ forces the search algorithm to avoid the $r_0$-neighborhood of the previously evaluated parameters. We used the L-BFGS-B algorithm to optimize the acquisition function, with $r_0 = 0$.

The emergent behavior in the large-scale ABM prevents the response curve from being smooth [40]. The Bayesian optimization with EI often fails to find a global minimum in practical applications where the response curve attains non-differentiability [8]. One the

other hand, the Latin hypercube sampling [3] is prohibitive because heterogeneous calibration increases degree of freedom. Therefore, we generalize of using a single criterion such as EI or Latin hypercube by mixing up the exploration methods with the exploitation methods [34]. The following strategy is the mixture of the acquisition functions introduced in Sect. 4.

- Choose random parameters for the first $N_0$ iterations
- For the next iterations,
  - ∗ With probability $\xi_1$, choose random parameters
  - ∗ With probability $\xi_2$, choose $\theta^k_{n+1} = \arg\max_\theta \sigma_d(\theta^k|\mathcal{D}_{1:n})$
  - ∗ With probability $\xi_3$, choose $\theta^k_{n+1} = \arg\min_\theta \mu_d(\theta^k|\mathcal{D}_{1:n})$
  - ∗ With probability $1 - \sum_i \xi_i$, choose $\theta^k_{n+1} = \arg\max_\theta wEI(\theta|\mathcal{D}_{1:n}, w_n)$

The value $\xi_1$ is the probability of choosing the next heterogeneous parameter randomly. The value $\xi_2$ is the probability of searching the region where the model is most uncertain, and this corresponds to the full exploration search. The value $\xi_3$ is the probability of searching the minimum point with current surrogate function, so it corresponds to the full exploitation search. If the surrogate function approximates the unobservable target response curve closely enough, the full exploitation is likely to find the global minimum of the response curve. Lastly, the value $\xi_4$ is the probability for the weighted EI acquisition function, which is the exploration-exploitation search algorithm.

## 6 Experimental result

In Sect. 6, we introduce the experimental results of the suggested calibration framework in two test cases. First, we tested the feasibility of our algorithm on the wealth distribution model [88]. We synthesize a validation dataset with an arbitrarily chosen parameter set, and this artificially generated dataset limits the source of the simulation divergence to random effects and the predetermined parameters set. Second, we experimente the real-world applicability on a real estate market model, which uses a real validation dataset with elaborately designed model structures and real input scenarios. In such a realistic case, additional sources of divergence exist, including the imperfect modeling and highly noisy socio-economics data.

Both ABMs experiment the calibration framework proposed in Algorithm 1. To investigate the effects of each component in the framework, we also experiment with dynamic calibration and heterogeneous calibration, separately. We evaluate the dynamic calibration without adjusting the heterogeneous parameters, and evaluate the heterogeneous calibration without modifying the dynamic parameters. The code used in the experiments is released in public.[1]

[1] https://github.com/Kim-Dongjun/Automatic-Calibration-of-Dynamic-and-Heterogeneous-Parameters-in-Agent-based-Models.

## 6.1 Test case 1: wealth distribution ABM

### 6.1.1 Model description

The wealth distribution model [88], adapted from the sugarscape model [20], is an agent-based model for investigating the macroscopic wealth distribution via microscopic agent behaviors. In the model, a grid provides its wealth equally to agents located on it. At the end of each simulation timestep, the agents consume their wealth to survive, and move toward the wealthiest neighboring grid to maximize their wealth income. Further, the grids recover their wealth capacity fully at the end of each timestep for future provision. The wealth capacity varies proportionally to the dynamic parameter, *Wealth Income*, and agents consume their wealth, in proportion to the heterogeneous parameter, *Wealth Consumption*. The agents' initial wealth is initialized randomly between 10 to 20, and the agent's initial position is created randomly on the 10 by 10 grid.

### 6.1.2 Virtual experimental design

**6.1.2.1 Synthetic parameter setting** The wealth distribution ABM assumes the artificially synthesized parameter setting as the true unobservable parameters. The synthetic parameter is generated as follows (Table 4). The dynamic parameter *Wealth Income* is differentiated every ten simulation timesteps. From the first timestep to the 10-th timestep, the parameter value is 1.5, and the parameter value of the next ten timestep is 0.5. Alternating the values between 1.5 and 0.5 for ten durations, the dynamic parameter is synthesized. The heterogeneous parameter is differentiated by the agent clusters, with a value of 0.9 for the first cluster, and 0.1 for the second cluster. The first cluster consists of agents with wealth in the top 50 percent at the initial timestep. The bottom 50 percent of the agents consists of the second cluster.

**6.1.2.2 Summary statistics** Wealth inequality is the main interest in this model; thus, the validation data was composed of High Class Wealth Average, Middle Class Wealth Average, Low Class Wealth Average, and Gini Index, as in Table 5. Here, the High Class Wealth Average is the average wealth of the top 33% agents, and Gini Index [26] is an index for measuring wealth inequality. In wealth distribution ABM, the agent cluster is assigned prior to the calibration stage because the purpose of wealth distribution ABM is on testing the performance of the suggested surrogate model-based Bayesian optimization.

**6.1.2.3 Experimental cases** We conduct six experiments in the wealth distribution ABM: obtaining validation dataset, synthetic parameter experiment, random search experiment, dynamic calibration, heterogeneous calibration, and the whole calibration framework experiment. The first experiment obtains the validation dataset $x_{obs}$ by averaging the simulation results of the $J = 1000$ replications for the same set of synthetic parameter. The second experiment measures the variability of the discrepancy estimation by performing simulation with the synthetic parameter, using $J = 10$. The synthetic parameter experiment is replicated $R = 30$ times with the same setting, to obtain the statistics on the variability. The third experiment randomly searches the parameter space.

The fourth experiment (dynamic calibration) evaluates and compares the three parameter updating schemes: *Sampling by Time*, *Sampling by Regime*, and *Mode Selection*. The

**Table 4** The wealth distribution model parameters

| Parameter | Name | Type | Description | Default value |
|---|---|---|---|---|
| Unoptimizable Model Parameters | Grid Wealth | Static | Capacity of grid wealth. Each grid has its basic capacity ranging from 2 to 15. Every timestep, the grid reproduces its full capacity, which is the multiplication of dynamic parameters on its basic capacity. | 0–30 |
| | Number of Grid in X-axis | Static | Number of grids in latitudinal axis. | 10 |
| | Number of Grid in Y-axis | Static | Number of grids in longitudinal axis. | 10 |
| | Number of Agent | Static | Number of agents in the sugarscape model. | 100 |
| | Simulation Time | Static | Number of simulation timesteps in the model. | 50 |
| | Agent Vision Distance | Static | Observable search range to maximize wealth possession. Agent only moves toward four directions. | 1 |
| | Migration Probability | Static | Probability to move toward the best position | 1 |
| | Life Expectancy | Static | Agent life expectancy | 100 |
| Parameter | Name | Type | Description | Search Range |
| Calibration Parameters | *Wealth Income* (WI) | Dynamic | Grid wealth capacity at each time is proportional to this parameter | 0–2 |
| | *Wealth Consumption* (WC) | Heterogeneous | Wealth consumption rate out of earning wealth at each time | 0–1 |

**Table 5** The wealth distribution model output summary statistics

| Variable | Name | Description |
|---|---|---|
| Output macro variables to validate | High Class Wealth Average | Average wealth of top 1/3 agents |
| | Middle Class Wealth Average | Average wealth of middle 1/3 agents |
| | Low Class Wealth Average | Average wealth of bottom 1/3 agents |
| | Gini Index [26] | The area ratio of the Lorenz curve to measure the wealth inequality |

heterogeneous parameters in the fourth experiment are fixed throughout the calibration procedure as the synthetic (true) heterogeneous parameters. The fifth experiment (heterogeneous calibration) applies the surrogate-based calibration to known clustering assignments. In the heterogeneous calibration, the dynamic parameter is fixed to be the synthetic parameter throughout the experiment.

The last experiment calibrates both the dynamic and heterogeneous parameters to investigate the effects of the interaction of two components. The last experiment tests the entire calibration framework with slightly different hyperparameters for the case *a* and *b*. The case *a* tests the calibration framework with $(N^t, N^k) = (2, 3)$, and the second case *b* tests it with $(N^t, N^k) = (20, 30)$, to investigate the effects of the predetermined hyperparameters on the calibration framework.

For the third, fourth, and fifth experiments, the optimizable calibration parameters are randomly initialized at the initial iteration. The list of the experimental variables in each experiment is presented in Table 6. Furthermore, all the experiments assume there were 100 agents ($A = 100$) and 50 timesteps for the model execution ($T = 50$).

**6.1.2.4 Performance measure** Because we know the synthetic parameters, we proceed to calculate the mean absolute error (MAE), to measure the distance from the estimated dynamic parameter to the synthetic dynamic parameter. We also calculate the Euclidean error to measure the distance from the estimated heterogeneous parameter to the synthetic heterogeneous parameter. To compare the simulation output, we apply the MAPE to measure the distance from the simulation result to the validation data in all the experiments.

### 6.1.3 Dynamic calibration results

The overall mechanism of the dynamic calibration is presented in Fig. 3. Figure 3a is a single dimension of the summary statistics, Gini Index, of a parameter candidate hypothesis. (b) is the result of the HMM, $\mathcal{R}_n^{t,i}$ (Algorithm 2-Line 3), following the division of the well-fitted regime, in green points, by the poorly-fitted regime, in red and blue points. The joint likelihoods, $L_n^{t,i}$ (Algorithm 2-Line 13), are presented in (c). Figure 3d presents the regime detection results for all three candidate hypotheses, $\Gamma_n^m$ (Algorithm 2-Line 14). The estimated beta distribution of a single merged regime, $\text{Beta}(\alpha_{m,0}, \beta_{m,0})$ (Algorithm 2-Line 9), is presented in (e). Figure 3f presents one of the next candidate hypotheses, $\theta_{n+1}^{t,i}$ (Algorithm 1-Line 10), generated from the estimated beta distributions. The current dynamic parameter candidates ($\theta_n^{t,i}$) are plotted as the gray lines, and the next dynamic parameter ($\theta_{n+1}^{t,i}$) is plotted as the black line, with different colors representing different merged regimes.

**Table 6** The wealth distribution model experimental design

| | Parameter type | Calibration parameter | Update rule | $N$ | $N^t$ | $N^k$ | $K^t$ | $K^{k1}$ | $J^2$ | $I$ | Rep.[3] | Cases |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Validation | – | – | – | 1 | – | – | – | – | 1000 | – | 1 | 1 |
| Synthetic parameter | – | – | – | 1 | – | – | – | – | 10 | – | 30 | 30 |
| Random search | Dynamic | WI | RS[4] | 100 | – | – | – | – | 10 | – | 30 | 30 |
| | Heterogeneous | WC | | | | | | | | | | |
| Dynamic calibration | Dynamic | WI | ST/SR/MS[5] (3 cases) | 100 | 100 | 0 | 3 | 2 | 10 | 3 | 30 | 3×30 |
| | Heterogeneous | –[6] | – | | | | | | | | | |
| Heterogeneous calibration | Dynamic | –[7] | – | 100 | 0 | 100 | 1 | 2 | 10 | 1 | 30 | 30 |
| | Heteregeneous | WC | BO[8] | | | | | | | | | |
| Calibration framework[a] | Dynamic | WI | MS | 200 | 2 | 3 | 3 | 2 | 10 | 3 | 30 | 30 |
| | Heterogeneous | WC | BO | | | | | | | | | |
| Calibration framework[b] | Dynamic | WI | MS | 200 | 20 | 30 | 3 | 2 | 10 | 3 | 30 | 30 |
| | Heterogeneous | WC | BO | | | | | | | | | |

[1] The wealth distribution model assumed that the agent clusters were pre-assigned, to investigate the surrogate model-based Bayesian optimization.

[2] The expectation of the simulation $\mathbb{E}_{\omega\in\Omega}[\mathcal{M}(\theta;\omega)]$ in the objective function prevented the optimization from searching in the wrong direction. $J$ is the number of Monte Carlo samples for estimating the expectation. See 7.1 in Sect. 7.1.

[3] Each experiment was replicated 30 times, to evaluate the statistics of the calibration performance

[4] RS: Random Search

[5] ST: Sampling by Time, SR: Sampling by Regime, MS: Mode Sampling

[6] The synthetic heterogeneous parameter was assumed throughout the experiment

[7] The synthetic dynamic parameter was assumed throughout the experiment

[8] BO: The mixture of Bayesian optimization (BO) with $(\xi_1, \xi_2, \xi_3, \xi_4) = (0.15, 0.2, 0.05, 0.6)$ and $N_0 = 5$. See 7.1 in Sect. 7.1

**Fig. 3** Dynamic parameter calibration mechanism is plotted. **a–g** Presents the calibration procedure in a single dynamic calibration iteration, and **h** presents a final calibrated simulation result. **a** Presents the simulated and validation Gini index. **b** Presents a regime detection result for the third particle, in Algorithm 2-Line 14. **c** Presents the normalized likelihoods $\mathcal{L}_0^{t,i}$, in Algorithm 2-Line 13. **d** Presents the merged regimes, in Algorithm 2-Line 4. **e** Presents the parameter posterior distribution, in Algorithm 2-Line 9. **f** Presents the parameter calibration result after the first iteration, in Algorithm 2-Line 10

The well-fitted regime in Fig. 3b makes the parameter estimation at the early period as successful as the red dots in (f). This is because the third candidate hypothesis in the well-fitted regime has high likelihoods of the next parameter being generated near the third hypothesis. The simulation fails to recover the validation statistics afterwards because the regime is over-detected up to simulation time 13. As iteration proceeds, the regime detection algorithm iteratively updates the regime information to partition simulation time correctly. As the dynamic regimes are optimized by iterations, subsequently, the calibrated parameter value settles down to the optimal value (see Fig. 4a). Figure 3g is a simulation result, $\mathcal{M}(\theta_1^{t,0} \cup \theta_1^k; \omega)$ (Algorithm 1-Line 6), evaluated using the estimated parameter in Fig. 3f. Figure 3h illustrates the Gini coefficient of the simulation after the calibration process eventually fits the synthetically generated validation data $\mathcal{M}(\theta_{n^*}^{t,i^*} \cup \theta_{n^*}^k; \omega)$ (Algorithm 1-Line 16).

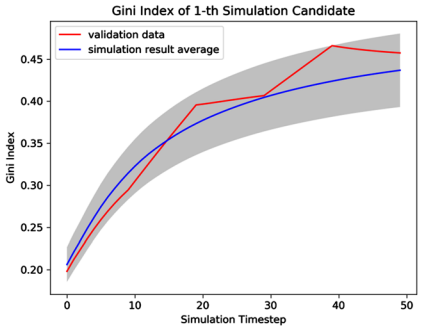
Figure 4 presents the best estimated parameters with two updating rules: *Mode Selection* and *Random Search*. *Mode Selection* in (a) estimates the plausible parameter that shares a tendency with the synthetic parameter. However, *Random Search* in (b) finds the highly fluctuating parameter, which is uninterpretable. Figure 5a plots the parameter MAE for the four updating rules. *Mode Selection* and *Sampling by Regime* outperforms the other updating rules in terms of the parameter error. However, Fig. 5b shows that the updating rule *Sampling by Time* exhibits the best performance in terms of the simulation error.

Table 7 shows the experimental statistics of the dynamic calibration. *Sampling by Time* yields the best performance in terms of the simulation error, but *Sampling by Time* has the highest parameter error, which implies that it estimates the over-fitted parameter. On the other hand, the performance of *Sampling by Regime* is not significantly different from that of the *Random Search*. Lastly, *Mode Selection* is the best updating rule in terms of the parameter error reduction.

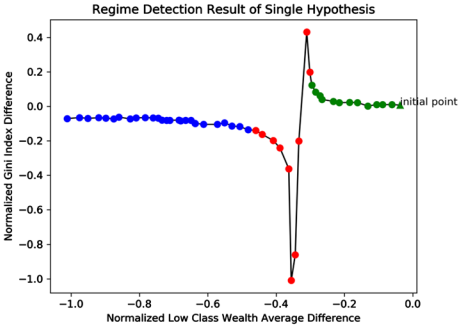
Figure 6 compares the parameter value initializing methods between random initialization and selective initialization. The selective initialization assigns equi-distant parameter values for each of particles. For example, in our setting, the three particles of dynamic parameters are initialized statically with values 0.5, 1.0, and 1.5. We believe that the selective initialization is efficient in inferring the parameter distribution because the synthetic dynamic parameter alternates between 0.5 and 1.5 (see Fig. 4a for the true dynamic parameter). As shown in Fig. 6, the selective initialization saturates faster than the random initialization. However, Fig. 6 indicates that the random initialization is as effective as the selective initialization after iterations.

#### 6.1.4 Heterogeneous calibration result

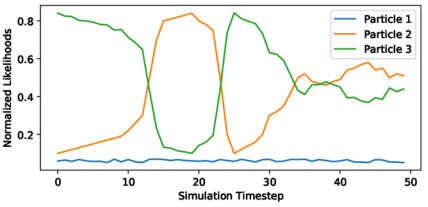
Figure 7a–c illustrate the landscape of the discrepancy surrogate model inferred through GPR with 10, 50, and 100 evaluation points, respectively. The surrogate function becomes gradually sophisticated as the iteration increased. Figure 7d–f present the contour plot of the surrogate model that contains the sampling path of the suggested BO. The initial



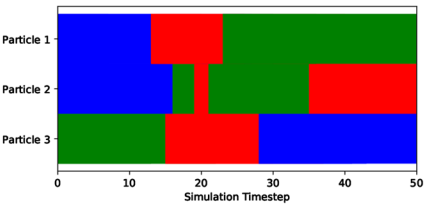

**(a)** Gini Index *before* Calibration



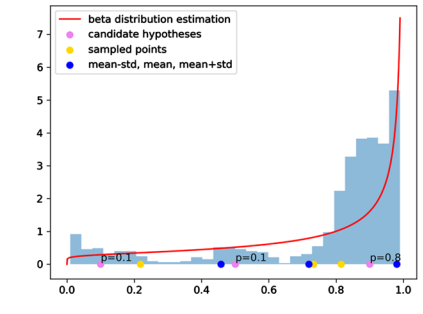

**(b)** Detected Regime $\{R_0^{t,3}\}_{t=1}^{50}$



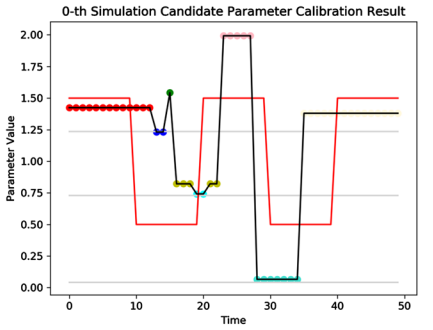

**(c)** Weights $L_0^{t,i}$



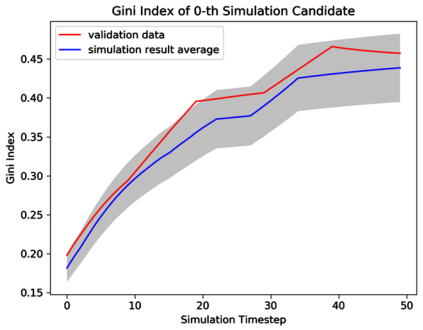

**(d)** Merge Regime $\{I_0^m\}_{m=1}^M$



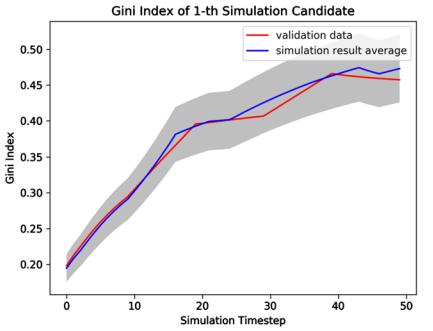

**(e)** Posterior Beta Distribution




**(f)** Calibrated Parameter *after* First Calibration Iteration

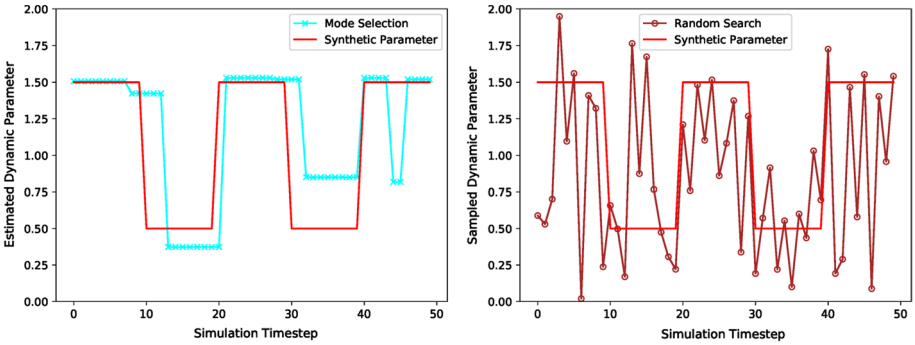



**(g)** Gini Index *after* First Calibration Iteration



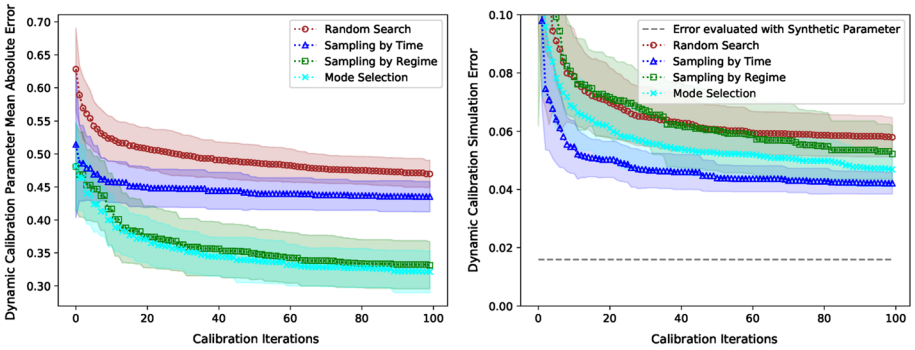

**(h)** Gini Index *after* Calibration Finishes

simulation result in Fig. 7g is transformed to be as close to the validation data as the best simulation result in Fig. 7h.




**(a)** Dynamic Calibration **(b)** Random Search

**Fig. 4** **a** The estimated optimal dynamic parameter, x marked line, is plotted. Dynamic calibration generates a dynamic parameter regime-wisely to avoid over-fitting in *Mode Selection* update rule. **b** The sampled optimal dynamic parameter, o marked line, is plotted. *Random Search* finds an over-fitted dynamic parameter, which only fits the given validation data, without following the synthetic dynamic parameter trend




**(a)** Parameter Error by Iteration **(b)** Simulation Error by Iteration

**Fig. 5** **a** The calibrated dynamic parameter mean absolute errors are plotted. Parameter generation methods *Sampling by Regime* and *Mode Selection* find parameters closer to the synthetic parameter than the other methods. **b** Dynamic calibration simulation error is plotted by iterations. Parameter generation method *Sampling by Time* performs the best in terms of simulation MAPE

Figure 8a presents the parameter Euclidean error of the heterogeneous calibration. The parameter Euclidean error of the heterogeneous calibration saturates at 0.021, and this non-zero saturation occurs because of the noisy discrepancy $\hat{d}$. In terms of the simulation error, Fig. 8b presents that the heterogeneous calibration converges to the optimal error, which is obtained from simulating with the synthetic parameter.

### 6.1.5 Calibration framework result

As shown in Table 7, the calibration framework[b], with $C_{dyn} = 20, C_{het} = 30$, outperforms the calibration framework[a], in terms of the simulation error. Although the calibration framework[a] has the least dynamic parameter error among all the experiments in the Test Case 1, the simulation error is large, due to the inaccuracy in the estimation of the

**Table 7** The wealth distribution model performance

| Experiment | Parameter update rule | | High Class Wealth Average MAPE | Middle Class Wealth Average MAPE | Low Class Wealth Average MAPE | Gini Index MAPE | Total MAPE | Parameter mean absolute error | Parameter euclidean error |
|---|---|---|---|---|---|---|---|---|---|
| | Dynamic | Heterogeneous | | | | | | | |
| *Test case 1: wealth distribution ABM* | | | | | | | | | |
| Synthetic parameter experiment | – | – | 0.012 | 0.016 | 0.020 | 0.015 | 0.016 | 0 | 0 |
| | | | (±0.009) | (±0.011) | (±0.013) | (±0.009) | (±0.007) | (±0) | (±0) |
| Random search | RS | – | 0.096 | 0.060 | 0.046 | 0.024 | 0.057 | 0.549 | 0 |
| | | | (±0.015) | (±0.008) | (±0.008) | (±0.007) | (±0.007) | (±0.037) | (±0) |
| | – | RS | 0.052 | 0.031 | 0.035 | 0.028 | 0.036 | 0 | 0.034 |
| | | | (±0.041) | (±0.022) | (±0.027) | (±0.027) | (±0.018) | (±0) | (±0.020) |
| | RS | RS | 0.140 | 0.089 | 0.077 | 0.041 | 0.087 | 0.623 | 0.339 |
| | | | (±0.031) | (±0.018) | (±0.024) | (±0.015) | (±0.013) | (±0.060) | (±0.238) |
| Dynamic Calibration | ST | – | 0.072 | 0.045 | 0.034 | 0.018 | 0.042* | 0.601 | 0 |
| | | | (±0.008) | (±0.006) | (±0.004) | (±0.004) | (±0.004) | (±0.076) | (±0) |
| | SR | – | 0.088 | 0.055 | 0.043 | 0.021 | 0.052* | 0.537 | 0 |
| | | | (±0.016) | (±0.013) | (±0.010) | (±0.005) | (±0.010) | (±0.104) | (±0) |
| | MS | – | 0.080 | 0.048 | 0.039 | 0.021 | 0.047* | 0.500 | 0 |
| | | | (±0.011) | (±0.007) | (±0.007) | (±0.005) | (±0.006) | (±0.109) | (±0) |
| Heterogeneous calibration | – | BO | 0.025 | 0.015 | 0.018 | 0.013 | 0.018* | 0 | 0.021 |
| | | | (±0.018) | (±0.010) | (±0.012) | (±0.007) | (±0.008) | (±0) | (±0.011) |
| Calibration framework[a] | MS | BO | 0.124 | 0.085 | 0.074 | 0.044 | 0.082 | 0.462 | 0.110 |
| | | | (±0.017) | (±0.020) | (±0.031) | (±0.021) | (±0.021) | (±0.044) | (±0.020) |
| Calibration framework[b] | MS | BO | 0.113 | 0.070 | 0.073 | 0.048 | 0.076* | 0.501 | 0.092 |

**Table 7** (continued)

| Experiment | Parameter update rule | | High Class Wealth Average MAPE | Middle Class Wealth Average MAPE | Low Class Wealth Average MAPE | Gini Index MAPE | Total MAPE | Parameter mean absolute error | Parameter euclidean error |
|---|---|---|---|---|---|---|---|---|---|
| | Dynamic | Heterogeneous | | | | | | | |
| | | | (±0.011) | (±0.007) | (±0.021) | (±0.021) | (±0.010) | (±0.089) | (±0.015) |

Each experiment replicates 30 times to yield the performance mean and standard deviation. One tailed Welch's t-test on each experiment is implemented with the baseline random search calibration with its performance ($0.087 \pm 0.013$)

* $P < 0.05$

heterogeneous parameter. The calibration framework[b], on the other hand, outperforms the calibration framework[a], because the calibration framework[b] estimates the heterogeneous parameter more closely than the calibration framework[a]. Figure 9 presents the experimental result of the calibration framework[b].

## 6.2 Test case 2: real estate market ABM

### 6.2.1 Model description

The real estate market model is an agent-based model for predicting housing price and the number of housing transactions. The model consists of three types of agents: household, external supplier, and realtor. The presented model does not assume any physical environment, which means that agents can interact with any other ones without limitations. Figure 10 visualizes the interactions between those agents. A household buys or sells houses to meet the residential needs. Further, the household registers the owned houses in *List Process* and decides to buy a listed house in *Buy Processes* based on internal decision making process.

In *List Process*, a household registers its vacant houses on the market through a realtor agent. A house is vacant, if the lease contract[2] is over, and no other bargain is made. When no household lives in a vacant house, and the house owner does not move into the house, then the householder puts up the house on a market for a leaseholder or a buyer. The external supplier acts as an exogeneous house supplier, and the newly built houses are registered in the housing market automatically.

In *Buy Process*, a household purchases houses in three steps: participation step, selection step, and contract step. First, the household makes the decision to participate in the market. A household without any house to reside engages unconditionally in the housing market, looking for a new residence. A household with a residence participates with probability *Market-Participation-Rate*, searching for a new house for the purpose of investing. Second, a household in the market selects a house out of the registered houses, according to its preferences. The preference of a household is based on the area, contract type, and house type. A household first decides between the capital area and out-of-capital area for a new house. Then, it decides upon the contract type, between lease and purchase, following which it decides upon the house type, among a house, apartment, and condo. Leaseholders with subsisting contracts always choose to purchase a house, if they make a deal with other house owners. Others, including (1) house owners living in their house and (2) leaseholders with contract termination, purchase or lease a house with probability *Purchase-Rate* or $(1 - \textit{Purchase-Rate})$. A realtor receives the participating agents' preferences, and sends the candidate houses to each household with matching preferences. Last, in the contract step, a household makes a deal on a house within its budget from the house list. If another household contracts the house first, the household chooses the other house. During these interactions between the realtor agent and the household agents, the realtor agent stochastically selects a sampling on the houses being offered by the household agents and randomly perturbs the ordering of the interaction between seller and buyer agents. The heterogeneous parameter, *Willingness-to-Pay*, controls the budget, which is the maximum ratio out of the total asset an agent would raise, using savings and loan services.

[2] Lease contract includes *Jeonse* and rental contract, where *Jeonse* is the lump-sum housing lease that is a Korean-specific residential form of living for two years; it is secured by paying a large amount of deposit at the initiation of the contract, which is returned at the end of the contract period, instead of paying rental fee monthly.

*Update Process* updates all the required simulation state variables. A household saves the rest of its monthly salary after deducting the expenses, such as consumption, taxes, rental fee, and loan repayment. Housing price is decreased to

$$100(1 - \textit{Market-Price-Decrease-Rate} + \text{Inflation Rate})\%, \tag{13}$$

up to ten times, if it is listed on the market; however, no agent signs a contract for the house at the timestep. The housing price is increased to

$$100(1 + \textit{Market-Price-Increase-Rate} + \text{Inflation Rate})\%, \tag{14}$$

if other houses with the same conditions are favored at the timestep. After updating the state variables, an external supplier supplies new houses in the market based on the supply rate parameters released by the Korea Appraisal Board (KAB); see Table 8 for detailed description. A newly generated house attains its state variables sampled from the state variable generating distribution. This generating distribution is created from the real-world data collected from *The Survey of Household Finances and Living Conditions* released by Statistics Korea every five years. We utilize the dataset released in 2015 to initialize agents' state variables in afterward timesteps. Further information on the real estate market model can be found from the previous paper [92].

### 6.2.2 Virtual experimental design

**6.2.2.1 Calibration parameters** The real estate market ABM has three dynamic parameters and two heterogeneous parameters: *Market-Participation-Rate* (MPR), *Market-Price-Increase-Rate* (MPIR), and *Market-Price-Decrease-Rate* (MPDR) as the dynamic parameters; and *Willingness-to-Pay* (WP) and *Purchase-Rate* (PR) as the heterogeneous parameters. Based on the bullish and bearish periods being well-known phenomena that control the seasonal economic trend, the model's purpose is to generate the market dynamics by modeling the supply and demand process. After the scenario-tailored supply process, the demand process is modeled based on two main parameters: MPR and WP. The unobservable parameter, MPR, models the dynamically varying amount of demand in the real estate market. The remaining unobservable parameter, WP, models the demand from the perspective of heterogeneity, because the shift of the individual investment portfolio affects the transaction amount. The magnitude of the price variability, which comes from the change in the supply and demand, is modeled by other dynamic parameters such as the MPIR and MPDR. The residential form in the process of signing the contract, between the buying position and Jeonse position[3], is modeled on the parameter PR, which is heterogeneously differentiated.

**6.2.2.2 Summary statistics** The main function of this model is to predict the housing price and number of housing transactions closely to the validation dataset. The official government data on the price index and the transaction number is released by the KAB. We consider the Jevons index [71] to be the housing price index, in conformity to how the KAB calculates their indices. We scale-up the simulation transaction numbers to make the simulation ($10^4$ #households) compatible with the real-world ($2 \times 10^7$ #households) transaction amounts.

[3] Jeonse is the lump-sum housing lease form of living for two years. The tenant pay large amount of deposit, instead of paying monthly rental fee.

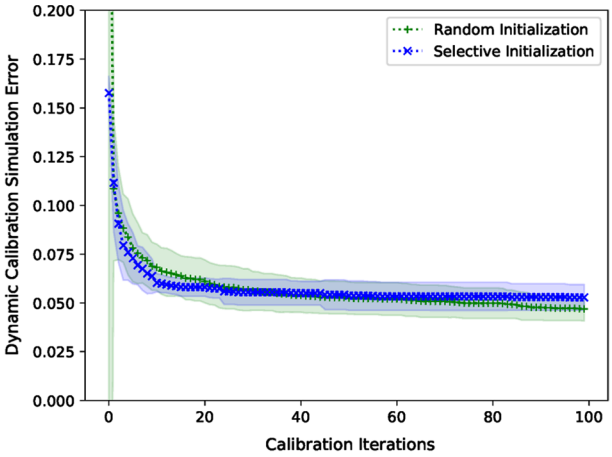


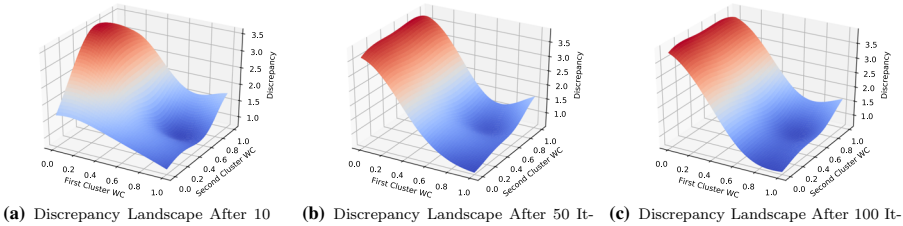

**Fig. 6** Dynamic calibration simulation errors with different initializations are plotted. The + marked line is the experimental result of the random initialization and the x marked line is that of the selective initialization, with equally distributed initial static parameters, having values 0.5, 1.0, and 1.5

The KAB releases 24 ($= 2 \times 3 \times 2 \times 2$) types of summary statistics, with each type having the following different housing characteristics: two house regions (Capital/Out-of-Capital), three house types (Detached/Apartment/Multiplex), two transaction types (Sales/Lease), and two summary statistics types (Index/Transaction number). We select the eight apartment summary statistics to validate the apartment-related real-world validation data, as listed in Table 9 for experiments, except for Heterogeneous Calibration$^z$. The experimental case Heterogeneous Calibration$^z$ validates the weighted eight summary statistics, which are the weighted sum of the 24 statistics in the dimension of House Type. Note that apartments constitute 70% of the housing transactions in Korea.

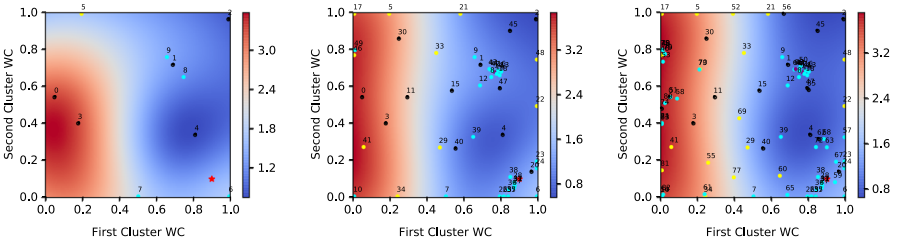
In agent clustering, all the agent-level state variables are considered, except the experimental cases Heterogeneous Calibration$^{A,\xi,K^k,d^k}$, where the sensitivity analysis of the varying hyperparameters are conducted. The agents have the following state variables: (1) Living Region distinguishes agents by their living region, according to whether they are capital or out-of-capital area; (2) Savings, Income and Loan variables are normalized to have values ranging from zero to one; (3) House Type and Living Type takes the one-hot encoded form, where the one-hot encoding is a data pre-processing method for transforming the integer index $i$ into a multi-dimensional zero vector, which however has one vector at the $i$-th dimension.

**6.2.2.3 Experiments** We conduct five experiments on the real estate market ABM: human calibration, random search, dynamic calibration, heterogeneous calibration, and the entire calibration framework. The experimental design setting is presented in the Table 10. The first and the second experiments are the Human Calibration and the Random Search. In both experiments, all the parameters are optimized, which included dynamic parameters (MPR, MPIR and MPDR) and heterogeneous parameters (WP and PR). Heterogeneous parameters (WP and PR) are undiversified by clusters because there is no information on the cluster assignment *before* agent clustering.

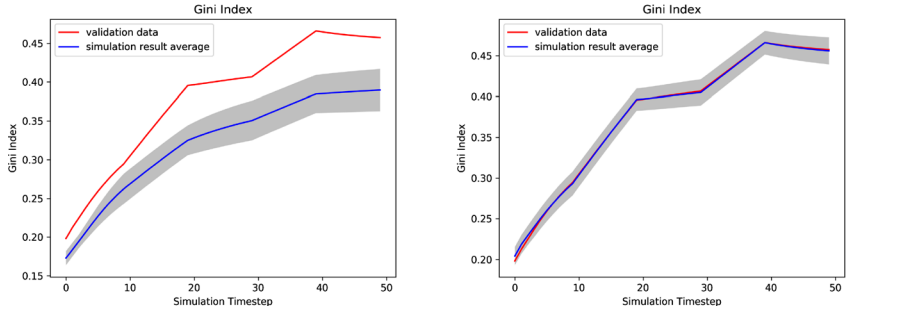
The Human Calibration performs 100 iterations of manual search (Table 10). At each iteration, the modelers divide the regime based on the combination of their prior knowledge and the simulated result, and the modelers adjusts the dynamic parameter based on the intuitive guess. The Random Search involves randomly searching all the parameters. The dynamic parameters in the Random Search are diversified by every timesteps.




**(a)** Discrepancy Landscape After 10 Iterations **(b)** Discrepancy Landscape After 50 Iterations **(c)** Discrepancy Landscape After 100 Iterations




**(d)** Discrepancy Level Curve After 10 Iterations **(e)** Discrepancy Level Curve After 50 Iterations **(f)** Discrepancy Level Curve After 100 Iterations




**(g)** Gini Index *before* Calibration **(h)** Gini Index *after* Calibration

**Fig. 7** Heterogeneous parameter calibration mechanism is plotted. **a–c** Illustrates the discrepancy surrogate model landscape (Algorithm 3-Line 25) after 10, 50 and 100 iterations of calibration, respectively. **d–f** presents the corresponding discrepancy surrogate model level curve. The Bayesian optimization eventually finds the global optimum as we expected. **g** and **h** compares the before and after of a single dimension of summary statistics (Gini index)

The third experiment (dynamic calibration) is composed of five cases. The first case of the dynamic calibration optimizes the first parameter MPR, which is believed to have the most significant effect on the macro output. The second and the third cases calibrates the price control parameters, MPIR and MPDR, respectively. The fourth case calibrates the first and the second parameters, i.e. MPR and MPIR, to investigate if the combination of the transaction number control parameter and the price index control parameter works. In the last case, the dynamic calibration optimizes all the dynamic parameters. All five cases assumes the manually calibrated heterogeneous parameters to be unchangeable.

The fourth experiment investigates the heterogeneous calibration using several cases. The first case is calibrating the first heterogeneous parameter (WP), the second case is on the second parameter (PR) and the third case calibrates all the heterogeneous parameters (WP and PR). They are denoted as HETEROGENEOUS CALIBRATION$^a$, i.e. HETEROGENEOUS CALIBRATION$^a$ investigates the effect with respect to the target calibration parameter. In addition,

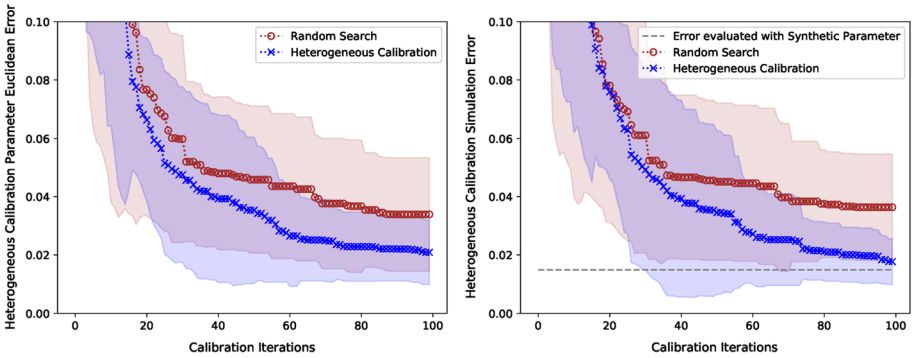



**Fig. 8** **a** The calibrated heterogeneous parameter Euclidean errors are plotted. **b** Heterogeneous calibration simulation MAPE is plotted. Suggested calibration converges to the optimal lower bound, which is the simulation error with the synthetic parameters as input parameters. Note that the optimal lower bound is not 0 since the simulation is stochastic

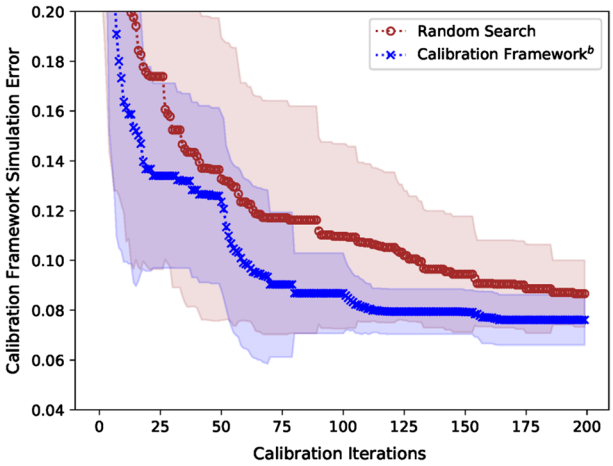



**Fig. 9** The calibration framework$^b$ simulation MAPE is plotted. The calibration framework$^b$ first updates the dynamic parameter for 20 iterations, with fixed heterogeneous parameter, and next updates the heterogeneous parameter for 30 iterations, with previously estimated optimal dynamic parameter

Heterogeneous Calibration$^a$ studies the final performance with respect to the number of clusters. Heterogeneous Calibration$^a$ uses every dimension of the agent-level state variables to obtain the agent sub-populations, and fits the apartment-related eight summary statistics.

Meanwhile, the next four cases Heterogeneous Calibration$^{A,\xi,K^k,d^k}$ are for the sensitivity analysis. The four cases utilizes six agent-level state variables (all except House Type) to cluster and fit the apartment-related eight summary statistics. The case with the superscript $A$ varies the number of agents to figure out the effect of the number of agents in the heterogeneous calibration. The case with $\xi$ examines the choice of the hyperparameters that control the mixture rate of acquisition functions. The case with $K^k$ investigates whether the increament of degree of freedom indeed guarantees the simulation result to be closer to the target. The case with $d^k$ designs to examine the choice of the heterogeneous parameters to calibrate.

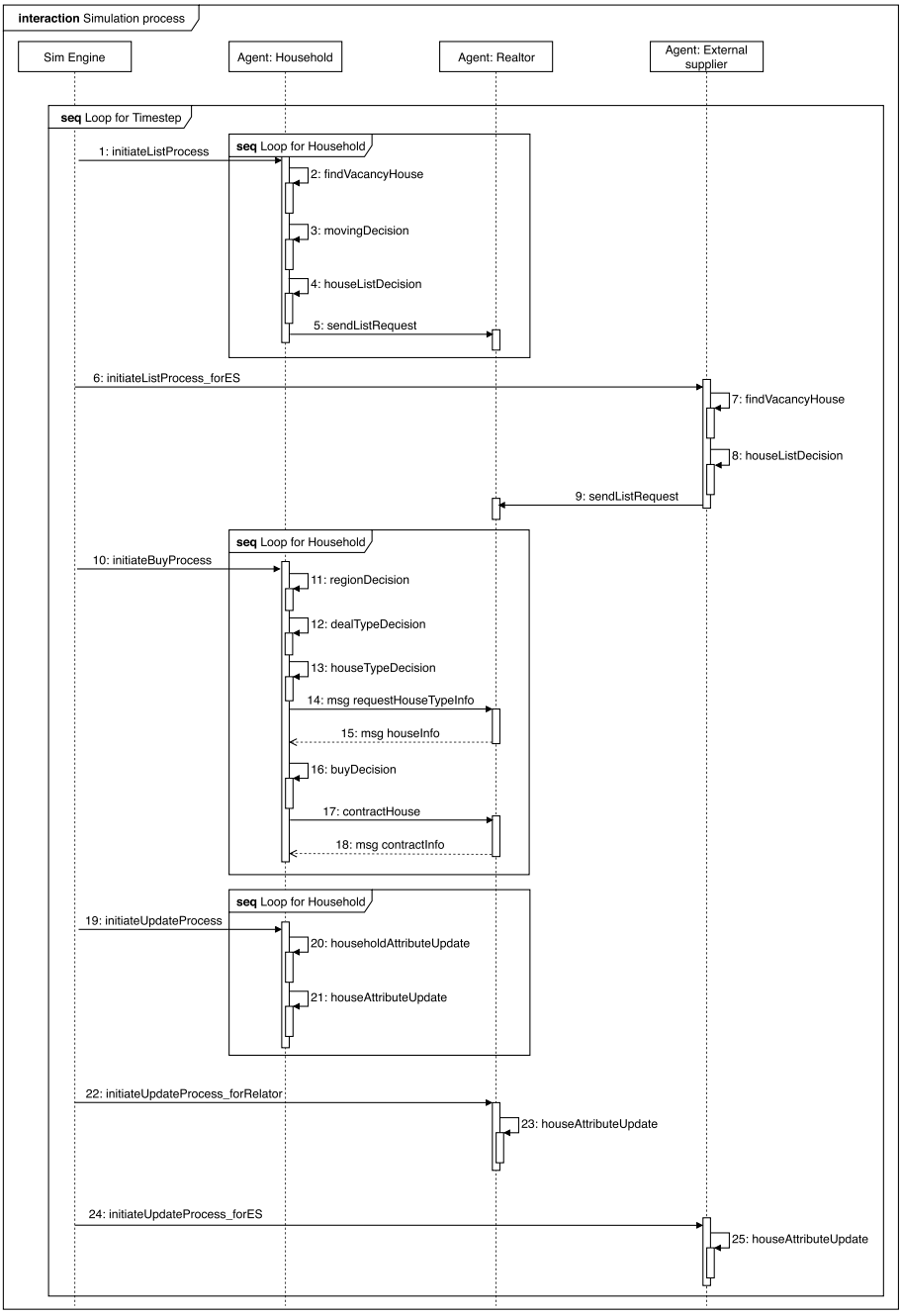



**Fig. 10** The real estate market model flowchart with the three sequential processes (*List Process*, *Buy Process* and *Update Process*) for each timestep

**Table 8** The real estate market model parameters

| Parameter | Name | Type | Description | Source[1] |
|---|---|---|---|---|
| Observable Model Parameters (Unchangable) | Housing Supply Rate in Capital | Static | Housing supply rate in the capital area | KAB |
| | Housing Supply Rate in Out-of-Capital | Static | Housing supply rate in out-of-capital area | KAB |
| | Mortgage Loan Maturity | Static | Maturity value of mortgage loans | KHFC |
| | Interest Rate | Dynamic | Interest rate of loan | BOK |
| | Inflation Rate | Dynamic | Inflation rate calculated based on consumer price index | SK |
| | Jeonse[2] Price Exchange Rate | Dynamic | Ratio of Jeonse price out of purchase price | KB |
| | Rent-Jeonse Exchange Rate | Dynamic | Exchange rate of rend and Jeonse deposit | KAB |
| | House Type Proportion | Dynamic | Proportions of detached/apartment/multiplex | SK |
| | House Price per Area | Dynamic | House price per unit area of each house type | KB |
| | Region Moving Rate in Capital | Dynamic | Probability of migration from capital to out-of-capital area | SK |
| | Region Moving Rate in Out-of-Capital | Dynamic | Probability of migration from out-of-capital to capital area | SK |
| | New Housing Supply Count in Capital | Dynamic | New housing supply rate of the capital area | KMOLIT |
| | New Housing Supply Count in Out-of-Capital | Dynamic | New housing supply rate of the out-of-capital area | KMOLIT |

**Table 8** (continued)

| Parameter | Name | Type | Description | Source[1] |
|---|---|---|---|---|
| | Loan-To-Value (LTV) Regulation Limit in Capital | Dynamic | Loan-To-Value regulation limit in capital area | FSC |
| | Loan-To-Value Regulation Limit in Out-of-Capital | Dynamic | Loan-To-Value regulation limit in out-of-capital area | FSC |
| | Debt-To-Income (DTI) Regulation Limit in Capital | Dynamic | Debt-To-Income regulation limit in capital area | FSC |
| | Debt-To-Income Regulation Limit in Out-of-Capital | Dynamic | Debt-To-Income regulation limit in out-of-capital area | FSC |
| | Consumption Rate | Heterogeneous | Household consumption rate by income quintiles | SK |

| Parameter | Name | Type | Description | Default | Range |
|---|---|---|---|---|---|
| Unobservable Model Parameters (Adjustable) | EMA Discount Rate | Static | Discount parameter values in EMA | 0.3 | Fixed |
| | Priority Threshold | Static | Threshold value of the house price increase process | 0.1 | Fixed |
| | *Market-Participation-Rate* (MPR) | Dynamic | Market participation rate of agents who holds a house to live | 0.005 | 0-0.05 |
| | *Market-Price-Increase-Rate* (MPIR) | Dynamic | House price increase rate | 0.01 | 0-0.012 |
| | *Market-Price-Decrease-Rate* (MPDR) | Dynamic | House price decrease rate | 0.01 | 0-0.01 |
| | *Willing-to-Pay* (WP) | Heterogeneous | Maximum rate for purchase out of the total asset an agent could raise | 0.7 | 0.3-0.9 |

**Table 8** (continued)

| Parameter | Name | Type | Description | Default | Range |
|---|---|---|---|---|---|
| | *Purchase-Rate* (PR) | Heterogeneous | Purchase rate of the leaseholders to contract a new house to reside | 0.6 | 0.3-0.9 |

The observable variables are unchangable throughout the calibration process, and the subset of the adjustable unobserved parameters are the calibration target parameters

[1]KAB: Korea Appraisal Board, KHFC: Korea Housing Finance Corporation, BOK: Bank of Korea, SK: Statistics Korea, KB: Kookmin Bank, KMOLIT: Korea Ministry of Land, Infrastructure and Transport, FSC: Financial Services Commission

[2] Jeonse is a long-term housing lease contract that only exists in South Korea

**Table 9** The real estate market model state variables

| Variable | Name | Description | Value |
|---|---|---|---|
| Macro-level Summary Statistics for Validation[1] | Apartment Sales Price Index in Capital | Jevons price index of Apartment sales price | Housing Price is converted into a percentage, with base value as 100 at the initial timestep. |
| | Apartment Sales Price Index in Out-of-Capital | | |
| | Apartment Lease Price Index in Capital | Jevons price index of Apartment lease price. | |
| | Apartment L ease Price Index in Out-of-Capital | | |
| | Apartment Sales Transaction Number in Capital | Transaction numbers of Apartment sales. | Simulation transaction number is scaled up to be compatible with the validation transaction number. |
| | Apartment Sales Transaction Number in Out-of-Capital | | |
| | Apartment Lease Transaction Number in Capital | Transaction numbers of Apartment lease. | |
| | Apartment Lease Transaction Number in *Out*-of-Capital | | |
| Micro-level State Variables for Agent Clustering[2] | Living Region | Agent living region between capital/out-of-capital area. | 1: Capital, 0: Out-of-Capital |
| | Savings | Total savings. | 1 unit/1000 KRW |
| | Income | Sum of the labor income and transfer income. | 1 unit/1000 KRW |
| | Loan | Total amount of money agent have borrowed from bank. | 1 unit/1000 KRW |
| | House Type | Type of house where an agent lives. | 1: Detached House, 2: Apartment, 3: Multiplex House, 4: No House |
| | Living Type | Type of living where an agent lives | 1: Owner, 2: Lease, 3: No House |
| | Number of Houses | Number of houses agent owns | 1 unit/1 House |

The macro-level summary statistics are the validation target variables, and the micro-level state variables are utilized in the agent clustering phase

[1] Data source: Korea Appraisal Board

[2] Micro-level real-world data is generally not accessible in many of agent-based models. Real estate market model assumes that we gather the snapshot of micro data at the initial timestep. After the initial timestep, the model is not accessible on the micro-level data. The micro-level data are collected from *The Survey of Household Finances and Living Conditions* hosted from Statistics Korea. The agents' initial state variables are randomly generated from the distribution of collected real-world data

**Table 10** Experimental variables of each experiment in the second test case are listed

| Experiment | Parameter Type | Calibration Parameter | $N$ | $N^t$ | $N^k$ | $K^t$ | $A$ | $K^{k1}$ | $\xi$ | $J^2$ | $I$ | Rep. | Cases |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Human Calibration | Dynamic | MPR+MPIR +MPDR | 100 | – | – | – | 10000 | 1 | – | 10 | 1 | 30 | 30 |
| | Static | WP+PR | | | | | | | | | | | |
| Random Search | Dynamic | MPR+MPIR +MPDR | 100 | – | – | – | 10000 | 1 | – | 10 | 1 | 30 | 30 |
| | Static | WP+PR | | | | | | | | | | | |
| Dynamic Calibration | Dynamic | MPR/MPIR/ MPDR/ MPR+MPIR/ All[3](5 cases) | 100 | 100 | 0 | 3 | 10000 | 1 | – | 10 | 3 | 30 | 5×30 |
| | Heterogeneous | –[4] | | | | | | | | | | | |
| Heterogeneous Calibration[a] | Dynamic | –[5] | 100 | 0 | 100 | 1 | 10000 | 1/2/4/8/DPMM (5 cases) | (0.15,0.2,0.05,0.6) | 10 | 1 | 30 | 3×5×30 |
| | Heterogeneous | WP/PR/All[6] (3cases) | | | | | | | | | | | |
| Heterogeneous Calibration[A] | Dynamic | – | 200 | 0 | 200 | 1 | 2000/5000/100 00/20000/40000 (5 cases) | 8 | (0.15,0.2,0.05,0.6) | 5 | 1 | 10 | 5×10 |
| | Heterogeneous | WP | | | | | | | | | | | |
| Heterogeneous Calibration[ξ] | Dynamic | – | 200 | 0 | 200 | 1 | 10000 | 8 | (0,0,0,1)/ (0.1,0.05,0.05, 0.8)/ (0.05,0.3,0.05,0.6)/ (0.15,0.2,0.05,0.6) (4 cases) | 5 | 1 | 10 | 4×10 |
| | Heterogeneous | WP | | | | | | | | | | | |
| Heterogeneous Calibration[$K^k$] | Dynamic | – | 200 | 0 | 200 | 1 | 10000 | 1/3/4/8 (4 cases) | (0.15,0.2,0.05,0.6) | 5 | 1 | 10 | 3×10 |
| | Heterogeneous | WP | | | | | | | | | | | |

**Table 10** (continued)

| Experiment | Parameter Type | Calibration Parameter | $N$ | $N^t$ | $N^k$ | $K^t$ | $A$ | $K^{k1}$ | $\xi$ | $J^2$ | $I$ | Rep. | Cases |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heterogeneous Calibration[dk] | Dynamic | – | 200 | 0 | 200 | 1 | 10000 | 4 | (0.15,0.2,0.05,0.6) | 5 | 1 | 10 | 3×10 |
| | Heterogeneous | WP/PR/All (3 cases) | | | | | | | | | | | |
| Calibration Framework[a] | Dynamic | All[7] | 200 | 2 | 3 | 3 | 10000 | 2/DPMM (2 cases) | (0.15,0.2,0.05,0.6) | 10 | 3 | 30 | 2×30 |
| | Heterogeneous | | | | | | | | | | | | |
| Calibration Framework[b] | Dynamic | All | 200 | 20 | 30 | 3 | 10000 | 2/DPMM (2 cases) | (0.15,0.2,0.05,0.6) | 10 | 3 | 30 | 2×30 |
| | Heterogeneous | | | | | | | | | | | | |

Dynamic calibration calibrates the dynamic parameters, with fixed human calibrated heterogeneous parameters. Heterogeneous calibration calibrates the heterogeneous parameters, with fixed human calibrated dynamic parameters. The calibration framework calibrates all the dynamic and the heterogeneous parameters, with two cases as in this table

[1] Assumption 1 in Sect. 7 holds that the heterogeneous parameters were differentiated pre-assigned agent clusters. To use the pre-assigned agent clusters in the simulation replications, the initial agent state variables were fixed throughout the simulation executions, as discussed in Assumption 4. See detailed discussion in Sect. 7.

[2] Assumption 7 in Sect. 7 holds that the simulation execution was replicated multiple times, to eradicate the nondesirable noise injected in the landscape of the response surface, which eventually caused the failure of the surrogate function. See more discussion in Sect. 7.

[3] Three dynamic parameters (MPR+MPIR+MPDR) were calibrated

[4] Heterogeneous parameters were fixed as manually calibrated parameters throughout the experiment

[5] Dynamic parameters were fixed as manually calibrated parameters throughout the experiment

[6] Two heterogeneous parameters (WP+PR) were calibrated

[7] Five dynamic/heterogeneous parameters (MPR, MPIR, MPDR, WP, and PR) were calibrated

The fifth experiment examines the entire calibration framework. It consisted of two cases, which investigate the effect of the number of calibration components continuation, $N^t$ and $N^k$. The first case involves calibrating dynamic parameters for 2 continuous iterations, and adjusting heterogeneous parameters for 3 continuous iterations, i.e. $N^t = 2$ and $N^k = 3$. In the next case, the investment optimizes all the parameters using $N^t = 20$ and $N^k = 30$.

All the experiments (except *Human Calibration*) initializes the parameters randomly. Further, every experiments (except *Heterogeneous Calibration*$^A$) all take $A = 10000$ households as the number of agents. Every experiments assume $T = 24$ timesteps for the model duration.

### 6.2.3 Dynamic calibration result

The simulating output is more sensitive on transaction numbers than the price indices. Having that the MPR determines the magnitude of the amount of transaction numbers by controlling the market demand, calibrating MPR is expected to generate a realistic simulation results. The first dynamic experiment of calibrating MPR indeed demonstrates the above argument by reducing Total MAPE significantly, from Human Calibration. In addition, Table 11 shows that the calibration with the housing price parameters (MPIR and/or MPDR) is ineffective in minimizing Total MAPE, as we expected. The inferior performance of calibrating either MPIR or MPDR mainly arise from the fact that those price parameters are not the key parameters for the number of transactions. For example, the highest error dimension (Apartment Sales Transaction Number in Capital MAPE) is diminished when we calibrate the MPR parameter, but remained unfitted when we calibrate other price-related parameters. We conclude that the selection of the parameter to be calibrated is the major factor on the calibration performance.

To investigate the effect of the number of parameters on the calibration performance, we calibrate both the MPR and MPIR. As shown in Table 11, the calibration performance differs insignificantly from calibraing MPR parameter, i.e. the experiment yields an equivalent performance (average 0.168) with the calibration experiment for the MPR parameter (average 0.167). Furthermore, the experiment in which all the parameters are calibrated yields a similar performance (average 0.174). The experiment on the number of dynamic parameters implies that the calibration performance is mainly driven by a small set of parameters. Therefore, a selection criteria for the choice of calibration parameter would influence the validation process significantly.

The o-marked trajectories in Fig. 11 indicates the baseline Human Calibration experiment. The Human Calibration fails to optimize the parameters because the dynamic calibration search space is too extensive to manually adjust. Figure 11a, d show that Human Calibration has a higher level of transaction numbers, compared to the validation data. Meanwhile, it yields a lower level of transaction numbers in Fig. 11b, c. The overly extensive search space of dynamic parameter inhibits the simulated dimensions from moving in the opposite direction of the trajectories of the dimensions (a) and (d) move toward downward, and the trajectories of the dimensions (b) and (c) shift upwardly, through manual calibration.

Meanwhile, the dynamic calibration automatically search parameter space with an adaptive regime detection algorithm. Experimental results of the dynamic calibration

**Table 11** Performance evaluations of the proposed calibration models in test case 2 are listed

| Experiment | Calibration Parameters | Number of Clusters ($K^k$) | Apartment Sales Price Index in Capital MAPE | Apartment Lease Price Index in Capital MAPE | Apartment Sales Price Index in Out-of-Capital MAPE | Apartment Lease Price Index in Out-of-Capital MAPE | Apartment Sales Transaction Number in Capital MAPE | Apartment Lease Transaction Number in Capital MAPE | Apartment Sales Transaction Number in Out-of-Capital MAPE | Apartment Lease Transaction Number in Out-of-Capital MAPE | Total MAPE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *Test case 2: real estate market ABM* | | | | | | | | | | | |
| Human Calibration | All Calibration Parameters | 1 | 0.012 | 0.020 | 0.007 | 0.003 | 0.765 | 0.159 | 0.284 | 0.463 | 0.214 |
| | | | (±0.001) | (±0.001) | (±0.000) | (±0.000) | (±0.027) | (±0.012) | (±0.016) | (±0.017) | (±0.005) |
| Random Search | All Calibration Parameters | 1 | 0.025 | 0.011 | 0.009 | 0.008 | 0.554 | 0.291 | 0.338 | 0.397 | 0.204 |
| | | | (±0.010) | (±0.006) | (±0.005) | (±0.004) | (±0.181) | (±0.085) | (±0.105) | (±0.127) | (±0.008) |
| Dynamic Calibration | MPR | 1 | 0.005 | 0.025 | **0.003** | 0.005 | **0.266** | 0.277 | 0.343 | **0.409** | **0.167*** |
| | | | (±0.001) | (±0.002) | (±0.000) | (±0.001) | (±0.015) | (±0.012) | (±0.016) | (±0.018) | (±0.003) |
| | MPIR | 1 | 0.006 | **0.016** | 0.005 | 0.004 | 0.740 | 0.167 | 0.277 | 0.472 | 0.211 |
| | | | (±0.003) | (±0.005) | (±0.002) | (±0.006) | (±0.022) | (±0.012) | (±0.015) | (±0.010) | (±0.004) |
| | MPDR | 1 | **0.002** | 0.017 | 0.003 | **0.004** | 0.770 | **0.162** | **0.240** | 0.435 | 0.204 |
| | | | (±0.002) | (±0.006) | (±0.003) | (±0.002) | (±0.028) | (±0.015) | (±0.012) | (±0.011) | (±0.003) |
| | MPR+MPIR | 1 | 0.009 | 0.024 | 0.005 | 0.006 | 0.273 | 0.280 | 0.337 | 0.410 | 0.168* |
| | | | (±0.005) | (±0.008) | (±0.003) | (±0.002) | (±0.034) | (±0.018) | (±0.023) | (±0.018) | (±0.003) |
| | MPR+MPIR +MPDR | 1 | 0.015 | 0.020 | 0.008 | 0.012 | 0.281 | 0.272 | 0.335 | 0.452 | 0.174* |
| | | | (±0.009) | (±0.010) | (±0.003) | (±0.004) | (±0.027) | (±0.018) | (±0.017) | (±0.029) | (±0.004) |

**Table 11** (continued)

| Experiment | Calibration Parameters | Number of Clusters ($K^k$) | | Apartment Sales Price Index in Capital MAPE | Apartment Lease Price Index in Capital MAPE | Apartment Sales Price Index in Out-of-Capital MAPE | Apartment L ease Price Index in Out-of-Capital MAPE | Apartment Sales Transaction Number in Capital MAPE | Apartment Lease Transaction Number in Capital MAPE | Apartment Sales Transaction Number in Out-of-Capital MAPE | Apartment Lease Transaction Number in *Out*-of-Capital MAPE | Total MAPE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heterogeneous Calibration[a] | WP | Undivided | 1 | 0.012 | **0.020** | 0.007 | **0.003** | 0.667 | **0.142** | 0.279 | 0.469 | 0.200* |
| | | | | (±0.001) | (±0.001) | (±0.001) | (±0.000) | (±0.080) | (±0.018) | (±0.056) | (±0.066) | (±0.004) |
| | | Parametric | 2 | 0.006 | 0.027 | **0.004** | 0.006 | **0.259** | 0.255 | **0.166** | **0.150** | 0.109* |
| | | | | (±0.002) | (±0.004) | (±0.001) | (±0.002) | (±0.012) | (±0.010) | (±0.011) | (±0.020) | (±0.003) |
| | | | 4 | 0.006 | 0.026 | 0.004 | 0.005 | 0.266 | 0.244 | 0.202 | 0.161 | 0.114* |
| | | | | (±0.002) | (±0.002) | (±0.001) | (±0.001) | (±0.030) | (±0.005) | (±0.020) | (±0.015) | (±0.002) |
| | | | 8 | **0.004** | 0.029 | 0.004 | 0.006 | 0.263 | 0.164 | 0.178 | 0.208 | **0.107*** |
| | | | | (±0.003) | (±0.007) | (±0.003) | (±0.003) | (±0.099) | (±0.044) | (±0.057) | (±0.051) | (±0.013) |
| | | Nonparametric | 15.3 | 0.005 | 0.026 | 0.004 | 0.005 | 0.265 | 0.164 | 0.357 | 0.177 | 0.125* |
| | | | (±1.4) | (±0.000) | (±0.000) | (±0.000) | (±0.000) | (±0.181) | (±0.129) | (±0.064) | (±0.146) | (±0.020) |
| | PR | Undivided | 1 | 0.012 | 0.020 | 0.007 | **0.003** | 0.764 | 0.218 | 0.249 | 0.360 | 0.204* |
| | | | | (±0.001) | (±0.000) | (±0.000) | (±0.000) | (±0.039) | (±0.033) | (±0.021) | (±0.057) | (±0.004) |
| | | Parametric | 2 | 0.013 | **0.019** | 0.007 | 0.003 | 0.835 | **0.104** | **0.184** | 0.256 | 0.178* |
| | | | | (±0.001) | (±0.000) | (±0.000) | (±0.000) | (±0.038) | (±0.029) | (±0.020) | (±0.044) | (±0.009) |
| | | | 4 | 0.013 | 0.020 | 0.007 | 0.003 | 0.827 | 0.137 | 0.201 | 0.255 | 0.183* |
| | | | | (±0.003) | (±0.002) | (±0.001) | (±0.001) | (±0.099) | (±0.164) | (±0.038) | (±0.067) | (±0.023) |

**Table 11** (continued)

| Experiment | Calibration Parameters | Number of Clusters ($K^k$) | | Apartment Sales Price Index in Capital MAPE | Apartment Lease Price Index in Capital MAPE | Apartment Sales Price Index in Out-of-Capital MAPE | Apartment L ease Price Index in Out-of-Capital MAPE | Apartment Sales Transaction Number in Capital MAPE | Apartment Lease Transaction Number in Capital MAPE | Apartment Sales Transaction Number in Out-of-Capital MAPE | Apartment Lease Transaction Number in *Out*-of-Capital MAPE | Total MAPE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 8 | **0.011** | 0.021 | 0.006 | 0.003 | 0.567 | 0.164 | 0.185 | **0.205** | 0.145* |
| | | | | (±0.006) | (±0.015) | (±0.008) | (±0.008) | (±0.097) | (±0.113) | (±0.041) | (±0.080) | (±0.013) |
| | | Nonparametric | 15.9 | 0.012 | 0.020 | **0.006** | 0.003 | **0.414** | 0.144 | 0.192 | 0.211 | **0.125*** |
| | | | (±1.2) | (±0.001) | (±0.000) | (±0.000) | (±0.000) | (±0.102) | (±0.085) | (±0.038) | (±0.058) | (±0.021) |
| | WP+PR | Undivided | 1 | 0.008 | **0.012** | **0.004** | 0.004 | 0.398 | 0.255 | 0.424 | 0.461 | 0.196* |
| | | | | (±0.002) | (±0.003) | (±0.001) | (±0.002) | (±0.126) | (±0.078) | (±0.072) | (±0.098) | (±0.006) |
| | | Parametric | 2 | 0.011 | 0.021 | 0.006 | **0.003** | 0.303 | **0.127** | 0.200 | **0.179** | **0.106*** |
| | | | | (±0.001) | (±0.001) | (±0.000) | (±0.000) | (±0.092) | (±0.028) | (±0.028) | (±0.054) | (±0.015) |
| | | | 4 | 0.009 | 0.023 | 0.005 | 0.004 | 0.289 | 0.241 | **0.168** | 0.181 | 0.115* |
| | | | | (±0.003) | (±0.002) | (±0.001) | (±0.001) | (±0.054) | (±0.106) | (±0.039) | (±0.077) | (±0.017) |
| | | | 8 | **0.007** | 0.025 | 0.004 | 0.005 | 0.243 | 0.180 | 0.218 | 0.230 | 0.114* |
| | | | | (±0.006) | (±0.015) | (±0.008) | (±0.008) | (±0.097) | (±0.113) | (±0.041) | (±0.080) | (±0.013) |
| | | Nonparametric | 15.5 | 0.011 | 0.021 | 0.007 | 0.003 | **0.232** | 0.199 | 0.198 | 0.186 | 0.107* |
| | | | (±1.2) | (±0.002) | (±0.001) | (±0.000) | (±0.000) | (±0.198) | (±0.151) | (±0.065) | (±0.176) | (±0.030) |
| Calibration Framework[a] | All Calibration Parameters | Parametric | 2 | 0.028 | **0.016** | 0.020 | 0.021 | 0.286 | **0.178** | 0.232 | 0.202 | 0.123* |
| | | | | (±0.012) | (±0.015) | (±0.009) | (±0.013) | (±0.073) | (±0.067) | (±0.055) | (±0.048) | (±0.016) |

**Table 11** (continued)

| Experiment | Calibration Parameters | Number of Clusters ($K^k$) | | Apartment Sales Price Index in Capital MAPE | Apartment Lease Price Index in Capital MAPE | Apartment Sales Price Index in Out-of-Capital MAPE | Apartment L ease Price Index in Out-of-Capital MAPE | Apartment Sales Transaction Number in Capital MAPE | Apartment Lease Transaction Number in Capital MAPE | Apartment Sales Transaction Number in Out-of-Capital MAPE | Apartment Lease Transaction Number in *Out*-of-Capital MAPE | Total MAPE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Nonparametric | 15.6 | 0.031 | 0.022 | 0.019 | 0.018 | 0.324 | 0.183 | 0.215 | 0.264 | 0.135* |
| | | | (±1.3) | (±0.018) | (±0.016) | (±0.015) | (±0.014) | (±0.108) | (±0.033) | (±0.063) | (±0.116) | (±0.019) |
| Calibration Framework[b] | All Calibration Parameters | Parametric | 2 | **0.016** | 0.016 | **0.005** | **0.004** | **0.219** | 0.195 | 0.252 | **0.136** | **0.105*** |
| | | | | (±0.001) | (±0.001) | (±0.003) | (±0.001) | (±0.007) | (±0.018) | (±0.049) | (±0.000) | (±0.002) |
| | | Nonparametric | 15.4 | 0.024 | 0.021 | 0.009 | 0.011 | 0.320 | 0.197 | **0.195** | 0.265 | 0.130* |
| | | | (±1.1) | (±0.023) | (±0.012) | (±0.006) | (±0.005) | (±0.084) | (±0.031) | (±0.004) | (±0.083) | (±0.001) |

Numbers are the mean and the standard deviation of experiments by replicating 30 times. The boldface indicates the smallest error in each experimental case. One tailed Welch's t-tests on the suggested calibration methodologies are implemented with the baseline human calibration results

* $P < 0.05$

[a] Experiment with $C_{dyn} = 2$, and $C_{het} = 3$

[b] Experiment with $C_{dyn} = 20$, and $C_{het} = 30$

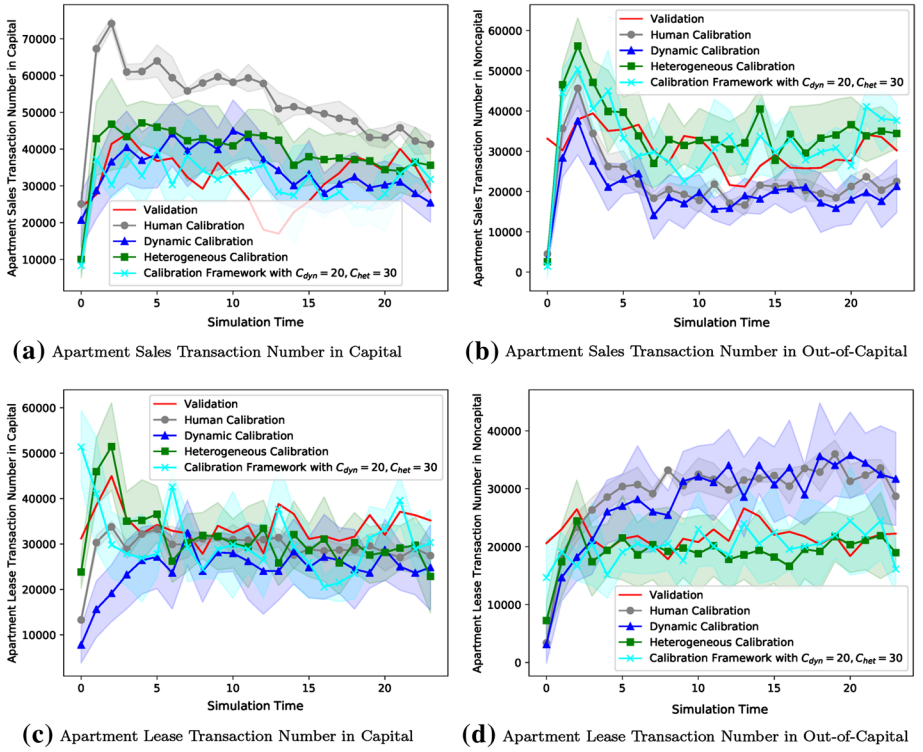


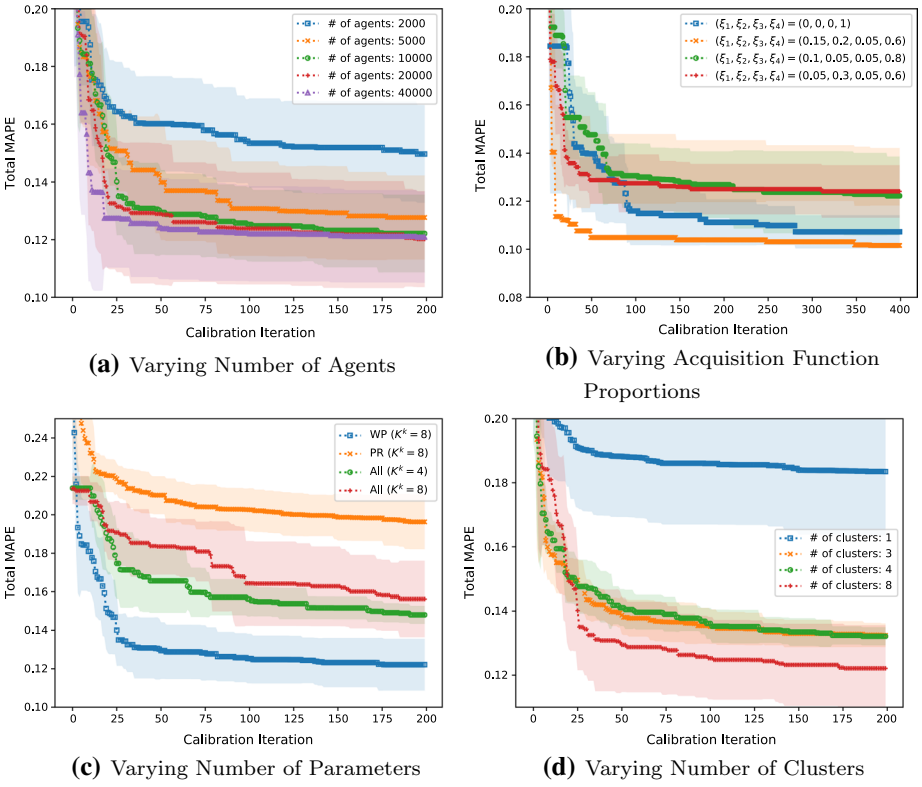

**Fig. 11** Dynamic calibration (triangle marked, all dynamic parameters calibration), heterogeneous calibration (square marked, all heterogeneous parameters calibration), the calibration framework[b] (x marked, all parameters calibration) experiments are illustrated, with the human calibration result (circle marked) for comparison. **a** and **b** are the apartment transaction numbers of the capital area for sales and lease, respectively. **c** and **d** are the apartment transaction numbers of the out-of-capital area for sales and lease, respectively. All calibration methodologies outperforms the human calibration result in particular in the dimension of (**a**). Heterogeneous calibration outperforms dynamic calibration in all (**b**–**d**). The calibration framework[b] outperforms heterogeneous calibration in (**a**) and (**d**)

demonstrates that it is successful on finding a better parameter that push simulated trajectories in an opposite direction for (a, d)-statistics and (b, c)-statistics.

### 6.2.4 Heterogeneous calibration result

The results of the calibration without the agent clustering procedure ($K^k = 1$) are presented in Table 11; there is no significant improvement in TOTAL MAPE, compared to HUMAN CALIBRATION. Meanwhile, when the agents are divided into a number of sub-populations ($K^k > 1$), HETEROGENEOUS CALIBRATION$^a$ achieves a significant performance gain in Table 11.

Similar to the dynamic calibration, the selection of the parameter to calibrate is crucial to the calibration performance. The parameter WP determines the level of demand in the housing market, and this parameter is the key parameter to control the magnitude of the transaction number. When the WP value is high, both the lease transaction number and sales transaction number becomes high. In contrast, the parameter PR determines whether




**(a)** Varying Number of Agents

**(b)** Varying Acquisition Function Proportions

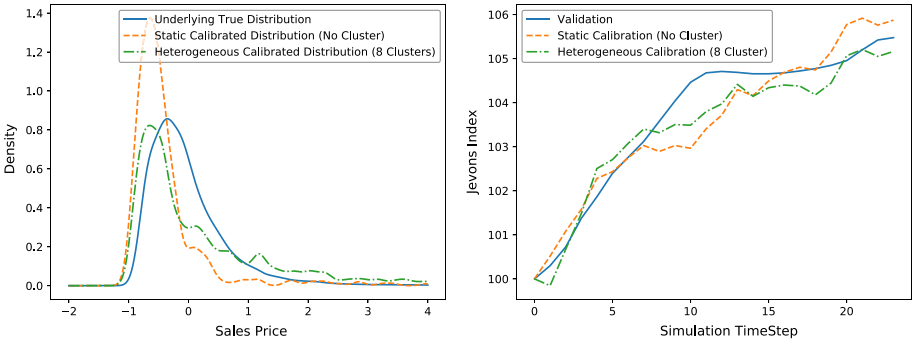
**(c)** Varying Number of Parameters

**(d)** Varying Number of Clusters

**Fig. 12** Sensitivity analysis on hyperparameters. Lower is the better. **a** The number of agents affect to the heterogeneous calibration performance since the clustering performance depends on the number of agents. Note that the Variational Autoencoder prefers more agents for training. **b** We experiment diverse acquisition function settings by varying the hyperparameters of the acquisition functions. **c** The choice of calibration target heterogenous parameters out of WP and PR affects to the performance. **d** As the number of clusters increases, the MAPE performs better

a housing contract is a lease or sales, which results in the sales transaction number and the lease transaction number moving in opposite directions. In other words, the high PR value means the number of sales transaction is increased while the number of the lease transaction is decreased. Because the overall magnitude of the transaction number is controlled by the first parameter WP, the calibration with WP performs better than that with PR. When all the heterogeneous parameters are calibrated, the calibration performances with varying number of clusters ($K^k$) are less oscillating.

Nearly half clusters in Heterogeneous Calibration$^a$ consists of agents who do not reside in apartment. This occurs because Heterogeneous Calibration$^a$ captures agent subpopulations with *every* dimensions of micro-level state variables. The parameter values of clusters with agents who do not live in apartment are irrelevant to fitting the target observation. Therefore, we eliminate the House Type agent state variable in the agent clustering phase to investigate how the agent partitioning affects to the internal dynamics. For the purpose, experiments Heterogeneous Calibration$^{A,\xi,K^k,d^k}$ assumes that the agents that did not live in apartments are grouped as the last cluster. Figure 15 illustrates the exemplary cluster assignment case that puts every agents without apartment in the last cluster.




**(a)** Comparison of Normalized Sales Price Distribution

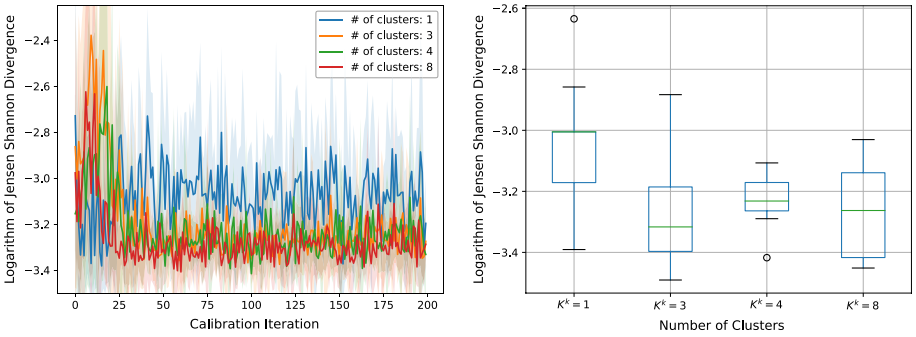
**(b)** Comparison of Sales Price Indices

**Fig. 13** The match of summary statistics (price index) does not guarantees the micro distributions (price) to align. **a** Heterogeneous calibration fits better than static calibration without agent clustering by expanding the degree of freedom. **b** Heterogeneous calibration and static calibration performs similar in sense of validation performance

Figure 12 presents the sensitivity analysis of the heterogeneous calibration. Figure 12a shows the calibration performance with different numbers of agents. The calibration with 2000 agents performs the worst, due to two reasons. The first reason is that the number of instances in $\mathcal{D}_{agent}$ is insufficient to train the machine learning models (e.g. VAE) for the calibration. The machine learning models with a limited dataset does not guarantee the originally targetted functionality. The second reason is the limited degree of freedom to represents the interactions between the agents. The future research on the automatic calibration shall be to investigate the comparisons of limitation sources. As we increase the number of the agents, the validation error decreases to a certain convergence point (0.12 of MAPE in Fig. 12a). Here, the non-zero convergence can be attributed to (1) the difficulty of the validation target dataset, (2) the realism of the simulation model, and (3) the parameter space exploration capability of the calibration method.

Figure 12b illustrates the calibration result by diversifying the hyperparameter $\xi$ (the acquisition function proportions) of the BO. The square depicts the pure weighted EI performance, and the other lines describe the performance trajectories according to varied probability proportions. The mixture of a number of the acquisition function originates from the frequent failure of finding the global minimum in the practical applications of the EI algorithm. Figure 12b supports two potential advantages of using the mixed strategies. First, the performance of the selective mixed strategy drops significantly on the front iterations. Second, the selective mixed strategy converges to a performance level ($\approx 0.10$) that is better than the performance ($\approx 0.11$) of the pure weighted EI strategy.

Figure 12c illustrates the performance with varying number of clusters. The squared blue line represents for the calibration of WP with eight clusters, the green line represents for the calibration of all heterogeneous parameters with four clusters, and the orange line represents for the calibration of PR with eight clusters. Even though the number of parameters to calibrate are identical for three lines, the calibration with WP exceeds the calibration with PR on performance. In addition, the calibration of both parameters is inferior to that of WP with eight clusters because of the lower degree of freedom. Lastly, the red line represents the calibration of both WP and PR with eight clusters, so the parameter to calibrate is doubled. With 16 adjustable parameters, the chart indicates that the 200 iterations is not




**(a)** Jensen Shannon Divergence of Sales Price Distributions

**(b)** Calibrated Box Plot of Jensen Shannon Divergence

**Fig. 14** Jensen Shannon divergence results. Lower the better. **a** Heterogeneous calibration perform better than the static calibration without agent clustering. **b** Snapshot box plot of the Jensen Shannon divergence at the minimum iteration with respect to the TOTAL MAPE

sufficient to reach to the saturation performance. In agent-based models with expensive evaluation cost, the selection of the number of clusters with many heterogeneous parameters could be further examined.

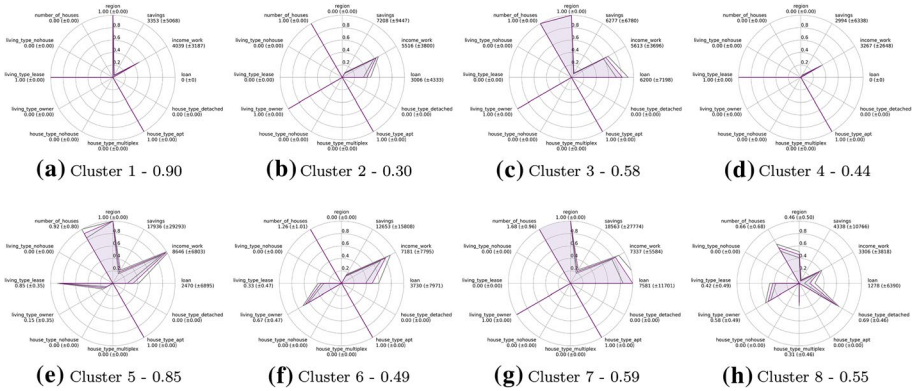
Figure 12d shows the calibration performance when the number of clusters varies. The static calibration with the undivided parameter underperforms (blue line), in comparison to any of heterogeneous calibrations with $K^k > 1$. Figure 12d illustrates the tendency of the calibration performance to improve as the degree of freedom (i.e. the number of agent clusters) increases. Figure 13 further investigate the degree of freedom in the perspective of variable distribution (underlying micro-level distribution). Figure 13a compares the underlying true and the simulated distributions of the housing sales price of static calibration (orange, $K^k = 1$) and heterogeneous calibration (green, $K^k = 8$). When the degree of freedom is unidimensional ($K^k = 1$), the distribution largely deviated from the underlying true distribution. As the degree of freedom expands to be multi-dimensional ($K^k = 8$), the shape of the distribution fits closer to the underlying true distribution. Although Fig. 13b indicates that the overall price indices behave similarly between $K^k = 1$ and $K^k = 8$, there was a significant change in the microscopic agent's behavior. It should be noted that the micro-level distribution is not counted as a validation measure.

Figure 14 illustrates the quantitative measure of the divergence between the simulated and real-world micro distributions. Figure 14a illustrates the Jensen-Shannon divergence of the sales price distribution by the calibration iteration. The static parameter calibration without the agent clustering ($K^k = 1$) performs a higher Jensen-Shannon divergence, compared to the heterogeneous parameter calibration with $K^k > 1$. On the other hand, the distinction for the varying degrees of freedom where $K^k > 1$ does not appear clearly. Figure 14b is a snapshot of the Jensen-Shannon performance. The snapshot is captured at the calibration iteration where the performance is minimized for each of experiment.

*6.2.4.1 Discussion on extracted sub-population*

In this subsection, we introduce a brief interpretation of the agent clustering results in Fig. 15. The characteristics of the agent clusters and the corresponding interpretations are summarized in Table 12.

Clusters 1, 3, 5 and 7 consists of agents who live in apartments in the capital region, and clusters 2, 4 and 6 consists of agents who live in apartments in the out-of-capital region.



**(a)** Cluster 1 - 0.90 **(b)** Cluster 2 - 0.30 **(c)** Cluster 3 - 0.58 **(d)** Cluster 4 - 0.44

**(e)** Cluster 5 - 0.85 **(f)** Cluster 6 - 0.49 **(g)** Cluster 7 - 0.59 **(h)** Cluster 8 - 0.55

**Fig. 15** Statistics of agent state variables for each cluster are illustrated. The subcaptions contain the cluster index with the calibrated parameter value *Willing-to-Pay* assigned to the corresponding cluster. Agents who live in apartments are divided by seven clusters and the agents who do not reside in apartments are grouped as the last cluster

The agents who live in the detached house or the multiplex house are grouped together in the last cluster, regardless of where they live. The agent clustering makes it possible to expand the unidimensional degree to the eight-dimensional degree of freedom.

We analyze the agent sub-populations based on three cluster types: first-time-buyers, participants-by-demand, and speculators. In the capital area, cluster 1 contains the first-time buyers, and cluster 5 consists the participants-by-demand. The calibrated parameter values for clusters 1 and 5 is higher than that for cluster 3 and 7, and this calibration result coincides with the empirical study of [10]. The cluster 3 has a relatively lower *Willingness-to-Pay* value, because they have already purchased a house for their residential purposes. The additional demand in Cluster 3 is intended for housing speculation. Similarly, Cluster 7 consisted of speculators who participate in the housing market with the intention to invest, which leads to the level of WP for this cluster being lower than that of the first-time-buyers and/or participants-by-demand.

In the out-of-capital area, cluster 4 is made up of the first-time buyers, and cluster 2 consists of the participants-by-demand. Because the out-of-capital housing prices are lower than the capital area, the households living in the out-of-capital region could afford their favorite house with a small proportion of their affordable budget. Therefore, the overall WP level of clusters 2 and 4 is lower than the corresponding clusters with the same type in the capital region. At the same time, the speculators in the out-of-capital area (cluster 6) have a slightly lower *Willingness-to-Pay* value than cluster 7, because cluster 6 speculates on cheaper out-of-capital houses.

### 6.2.5 Calibration framework result

While we discuss the ablation studies in the above Sections, in this subsection, we describe the experiments conducted on the whole calibration framework. The calibration framework is originally designed as the combination of the dynamic calibration and the heterogeneous calibration. The two experiments conduct on the calibration framework are presented in Table 11; one (Calibration Framework$^a$) is for $(N^t, N^k) = (2, 3)$,

**Table 12** Interpretation of agent sub-populations in Fig. 15

| Cluster | House type | Living region | Income level | Number of houses | Holding loan amount | Interpretation | *Willing-to-Pay* (Calibrated) |
|---|---|---|---|---|---|---|---|
| 1 | Apartment | Capital | Low | $\leq 1$ | Low | First-Time-Buyer | 0.90 |
| 2 | Apartment | Out-of-Capital | Medium | $\leq 1$ | Low | Participant-by-Demand | 0.30 |
| 3 | Apartment | Capital | Medium | $\leq 1$ | High | Participant-by-Demand | 0.58 |
| 4 | Apartment | Out-of-Capital | Low | $\leq 1$ | Low | First-Time-Buyer | 0.44 |
| 5 | Apartment | Capital | High | $\leq 1$ | Medium | Participant-by-Demand | 0.85 |
| 6 | Apartment | Out-of-Capital | High | $> 1$ | Medium | Speculator | 0.49 |
| 7 | Apartment | Capital | High | $> 1$ | High | Speculator | 0.59 |
| 8 | Detached/Multiplex | – | – | – | – | Agents who does not live in Apartment | 0.55 |

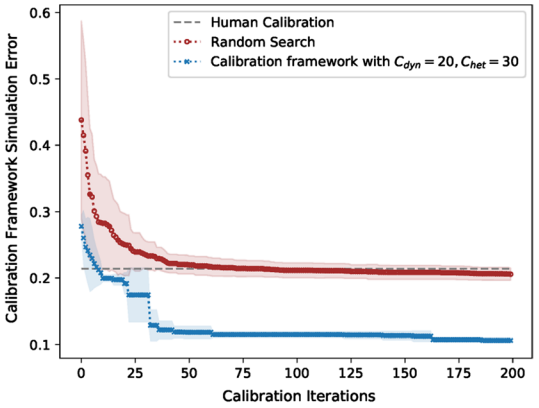



**Fig. 16** The calibration framework[b] simulation MAPE is plotted in the x marked line. The calibration framework[b] first calibrates all three dynamic parameters for 20 iterations, with fixed heterogeneous parameters, and next calibrates all two heterogeneous parameters for 30 iterations, with previously calibrated optimal dynamic parameters. The random search reaches to the human calibration error after 80 iterations, and saturates on the afterward iterations. The calibration framework achieves the human calibration error in short iterations, and saturates to nearly half of the human calibration error

and the other (Calibration Framework$^b$) is for $(N^t, N^k) = (20, 30)$. All the dynamic heterogeneous parameters are calibrated in both experiments.

The framework is designed such that when the dynamic parameters converges as the iteration increases, the heterogeneous parameters are expected to find the optimal parameter setting. However, the experimental result $(0.123 \pm 0.016)$ in Table 11 for the setting $(N^t, N^k) = (2, 3)$ does not yield a better performance than the heterogeneous calibration result $(0.106 \pm 0.015)$. We suspect that this poor performance occurs because the dynamic parameters are updated so often that the response surface of the heterogeneous calibration does not capture the global minimum. In other words, too rapidly switching dynamic parameters would cause the rapid fluctuation of the response curve for the heterogeneous calibration, in which this fluctuation eventually results the degraded performance. When we design the hyperparameters $(N^t, N^k) = (20, 30)$, the rapid fluctuation on the response curve does not occur. Therefore, with this set of hyperparameters, the calibration framework outperforms both the dynamic calibration ($N^k = 0$) and heterogeneous calibration ($N^t = 0$), as shown in Table 11.

Figure 16 presents the simulation errors of the calibration framework$^b$ based on calibration iterations. First, nearly 80 iterations are required before the random search matches the human calibration result, and no significant improvement is made on the later iterations. Second, the calibration framework$^b$ outperforms the human calibration within 20 iterations, on average. After superseding the human calibration, the calibration framework$^b$ saturates speedily. After 20 iterations of dynamic calibration and ten iterations of heterogeneous calication, the calibration performance saturates.

Figure 11 illustrates each dimension of the transaction numbers, where Fig. 11 presents the result of the experiment for Heterogeneous Calibration$^a$ ($K^k = 2$) with all the heterogeneous parameters to be calibrated, and for Calibration Framework$^b$ ($K^k = 2$) with all the parameters to be calibrated. As shown in Fig. 11a and Table. 11, the overall magnitude of Calibration Framework$^b$ best fits the validation. The Calibration Framework$^b$ performs stably for all the output dimension with relatively less

errors, specifically for dimensions (a) and (d), which can not be fitted through HUMAN CALIBRATION.

## 7 Assumptions and limitations

This Section provides the assumptions and the limitations of the suggested calibration framework provided by Algorithm 1.

### 7.1 Assumptions for calibration framework

Below is the enumeration of the assumptions we made in the proposed algorithms and subsequent models from the machine learning field. An exhaustive examination would result in the assumption list being too long, and the possibility of the reader not differentiating among the importance of assumptions. Here, we particularly selects the assumptions that may affect the calibration performance significantly.

(A1) We assume that the heterogeneous parameters are differentiated by the agent clusters, rather than by the individual agents. This assumption is applied in Line 2 of Algorithm 1, to reduce the time complexity. Furthermore, this assumption circumvents the over-parametrization problem by limiting the number of parameters (degree of freedom) to be calibrated.

(A2) We assume that both the dynamic and heterogeneous parameters are bounded. The invalid parameter region is eliminated before the calibration stage. This assumption is applied to Line 9 of Algorithm 2, in estimating the posterior distribution as a beta distribution Beta($\alpha_{m,l}, \beta_{m,l}$) with bounded input.

(A3) We use a mean-field assumption, which indicates the posterior shape parameters $\alpha_{m,l}$ are independent on $m$ and $l$ and $\beta_{m,l}$ are independent on $m$ and $l$. This assumption is applied to Lines 7-9 of Algorithm 2.

(A4) We use the fixed set of the initial agent state variables in heterogeneous calibration. If the agent-level state variables at the initial step is not given from the real-world dataset, the fixed random seed would lead to the initial agent state variables being identical for all the simulation executions. Even though the initial agent state variables are identical, the simulation path deviates for the replications as the simulation time proceeds. This assumption is applied to Lines 2 of Algorithm 1.

(A5) We assume that the target optimization parameters $\theta$ are selected among unobservable model parameters before the calibration process.

(A6) We assume that running the simulation model several times would eliminate the output stochasticity. This assumption is applied in Line 6 and 12 of Algorithm 1. A discussion of the noisy environment under the use of the evolution algorithm can be found in [16].

(A7) We assume that the heterogeneous calibration generates the next parameter to be evaluated based on four criteria. The hyperparameters $\{\xi_i\}_{i=1}^4$ determine which acquisition function to utilize among 1) random search, 2) predictive variance, 3) negative predictive mean and 4) weighted EI. This assumption is applied to Lines 13-21 of Algorithm 3.

### 7.2 Limitations from the assumptions and framework

In the previous Section, we list the assumptions of the proposed framework; these assumptions impose certain limitations of the calibration process. We conjecture that some of the following limitations originate from the enumerated assumptions. Further, we suggest that other limitations arise from either the simulation settings or calibration variable selections.

*Over-parametrization* If a simulation has too many parameters, the automatic calibration algorithm would be likely to over-fit the parameters, and the calibration simply memorizes the observation dataset. Therefore, the automatic calibration could result in unrealistic parameters to optimize the validation error with excessive number of parameters. Hence, the automatic calibration users need to limit the number of calibration parameters by weighing in on the degree of freedom from the parameters and the validation data availability.

When the simulation divergence arise based on time and heterogeneity, the proposed calibration framework would reduce the divergence between the simulation and real world. However, the increment in the optimization dimension would be significant if we calibrate the dynamically varying parameters or the agent-specific varying parameters. Hence, the proposed calibration is designed to limit the degree of freedom by extracting the hidden structures, using machine learning algorithms. The hidden regimes $K^t$ regulate the parameter variability in the dynamic calibration, and the hidden agent clusters $K^k$ restrain the optimization infeasibility. In this aspect, with carefully selected machine learning hyperparameters $K^t$ and $K^k$, over-parametrization [87] can be partially mitigated in the proposed calibration framework.

*Over-fitting* Besides of over-parametrization, another source of over-fitting is on the dataset. First, the over-fitting arises when there are insufficient validation data instances. Many ABMs (in particular, socio-economics case) target the phenomena that are not repeatable or that are expensive to replicate in an identical scenario, and this practical limitation only permits the calibration task to have a few-shot instances. In this paper, we assume the available dataset is only a single instance $x_{obs}$, so the calibration becomes particularly vulnerable to the data quantity. With a single instance, calibration is likely to memorize that instance, instead of capturing the true hidden dynamics lying beneath $x_{obs}$.

When the calibrated simulation mimics the observation, which is just a randomly selected sample path out of the various real-world possibilities, then the estimated parameters will be incorrect. Therefore, we limit our calibration framework to in-sample validation, with the aim of maximizing the agreement between the simulation and the collected real-world dataset. To expand the applicability of the proposed framework, the regularization technique or Bayesian approach to calculate the discrepancy by $\mathbb{E}_{x\in\mathcal{N}(x_{obs},\mathbb{I})}[d(\mathbb{E}_{\omega}[\mathcal{M}(\theta;\omega)],x)]$ is needed in the future work.

Second, the over-fitting occurs when the gathered data is noisy. Since we are not aware of the potential observation noise, wealth distribution ABM defines its validation dataset $x_{obs}$ as the average of 1000 simulation replications with synthetic parameter. This synthetic dataset restricts the source of observation noise as either the simulation stochasticity and the pre-defined synthetic parameter. Other source of observation noise is eradicated in wealth distribution ABM. This is the reason of testing the suggested framework on such toy ABM.

In addition, agent-based model calibration often pre-process the observed dataset to generate a set of summary statistics to minimize the unnecessary noise. For instance,

[28] extracts the structural reduced-form summary statistics to minimize the obsdesrvation noise embedded in the collected dataset. The reduced-form summary statistics only attain the qualitative structural information of the original observation, and the statistics discard the details of dynamic fluctuations in the original raw observation. Guerini and Moneta [28] confirms whether the simulation attains the stylized facts of interest by quantitatively comparing the structural reduced-forms of the observation and simulation. Here, the stylized facts are the macroscopic structural information.

*Automatic calibration opacity* In future, if the automatic calibration becomes a part of simulation models, the automatic calibration can be another black-box wrapping a simulation model. However, in the current stage, automatic calibration is a separate algorithm that is applied to existing simulation models. Moreover, the calibrated parameters are explicitly provided to the calibration algorithm users; therefore, currently, the calibration model does not increase the opacity.

*Mean-field assumption* When adjusting dynamic parameters, the mean-field assumption described in Lines 7-9 of Algorithm 2 updates the parameter values separately without the joint consideration of the effects of the other parameters and other merged regimes. This mean-field assumption limits the usability of the suggested dynamic calibration in the ABMs with many unobservable dynamic parameters.

*Number of agents* The applicability of the heterogeneous calibration mainly lies in the ABM with many agents, because the learning process of the VAE requires a number of data instances (number of agents). The research on estimating the number of agents required to learn the latent representation is another field of machine learning [85]. The required number of agents depends on two factors: the dimension $Q \times T$ of the agent states $x_a$ and the simulation complexity. As the dimension increases and the simulation complexifies, the encoder and the decoder require deep networks and large amounts of neural network parameters, to extract the valid representations. Then, it is required to have many agents (data instances) to estimate the neural network parameters properly.

*Agent clustering* With a fixed number of clusters $K^k$, let us assume that $\underline{d}^{\boldsymbol{c}} = \min_{\theta^k} d^{\boldsymbol{c}}(\theta^k)$ is the minimum discrepancy with agent cluster assignments $\boldsymbol{c} = \{c_a\}_{a=1}^{A}$, where $c_a \in \{1, ..., K^k\}$. Then, there exists an optimal cluster assignment $\boldsymbol{c}^* = \{c_a^*\}$, with $c_a^* \in \{1, ..., K^k\}$, that yields the lowest minimum discrepancy $\underline{d}^{\boldsymbol{c}^*} = \min_{\boldsymbol{c}} \underline{d}^{\boldsymbol{c}}$. The joint consideration of the agent clustering process with the optimization problem to obtain the optimum cluster $c^*$, is not considered in heterogeneous calibration. Because $\underline{d}^{\boldsymbol{c}}$ depends heavily on how $\boldsymbol{c}$ is assigned, future work should embed the agent clustering process within the optimization procedure.

*Kernel choice* The function space that the GPR can regress is given as $\{g(x) = \sum_{i=1}^{n} \alpha_i \mathrm{K}(x, x_i) | n \in \mathbb{N}, \alpha_i \in \mathbb{R}\}$ [63]. This indicates that the kernel function affects the differentiability of the inferred surrogate function. Specifically, the Matern kernel with hyperparameter $\nu$ represents the function space consisting of $(\lceil \nu \rceil - 1)$-times differentiable functions. However, the regularity of the discrepancy cannot be uncovered for generic ABMs, because the effect of the internal structure and the parameters on the simulation output is not known [22]. Under such limited insight into the discrepancy function, selection of the hyperparameter $\nu$ could lead to failure in approximating the discrepancy function. An alternative (but not interpretable) way to impose kernel is the spectral kernel [90], which implicitly trains the Fourier dual of the kernel function.

*Bayesian optimization* The surrogate function may not represent the possibly non-smooth discrepancy function, due to the wrong regularity hyperparameter $\nu$ in kernel. Because a single acquisition function in such case fails to capture the global optimum, we suggest a mixed strategy for selecting an acquisition function. The suggested mixed

strategy has a major disadvantage, in that the hyperparameters $\xi_1$, $\xi_2$, and $\xi_3$ are manually chosen.

*Compact parameter space* Bounding the proper parameter space crucially affects the calibration performance. The search domain is manually selected from the whole feasible parameter space, and this manual setting can restrict the applicability of the suggested framework.

*Underdeterminedness* Formulating the calibration using an optimization problem, as in Eq. 1, may have multiple global minimum parameters; i.e., $\Theta^* = \{\theta^* | \arg\min_{\theta \in \Theta} d(\mathbb{E}_{\omega \in \Omega}[\mathcal{M}(\theta;\omega)], x_{obs})\}$ is not a singleton. We acknowledge that this is a crucial problem, especially in the calibration task with sparse dataset, because there is no guide to select the optimal parameter among $\Theta^*$, except the performance, as measured from $x_{obs}$. The domain expert can mitigate this underdetermined system by implementing calibration multiple times and choosing the optimal parameter out of the calibrated set of parameters by using domain knowledge.

Currently, there is another line of research in the simulation calibration community: ABC. The ABC solves the calibration by inferring the posterior distribution of the parameter $p(\theta|x_{obs})$, and this Bayesian inference provides the overall distribution of how the parameter is likely to generate the observed data. Therefore, the Bayesian inference algorithms can resolve the underdeterminedness.

*Calibration of neither dynamic nor heterogeneous parameters* In this study, we focus on the calibration of the dynamic and heterogeneous parameters. The parameters that are neither dynamically varying nor heterogeneously shifting are calibrated in the process of the heterogeneous calibration, because the heterogeneous calibration, excluding the agent clustering phase, is basically the process of surrogate-based model calibration. The surrogate-based model calibration using the BO and GPR can calibrate both continuous-valued and discrete-valued generic parameters.

## 8 Conclusion

In this paper, we propose an automatic data-driven calibration framework for the dynamic and heterogeneous parameters of a simulation world by extracting the hidden structures from the simulation outputs. The dynamic calibration controls the dynamically switching parameters by extracting the hidden dynamic regimes, and treating each regime separately. The heterogeneous calibration adjusts the agent cluster-wise heterogeneous parameters by extracting the hidden agent-subpopulations. The experimental results for both types of calibration demonstrate that the proposed separate calibrations, with plausible estimated parameters, reduced the simulation error. The calibration framework, the combination of the dynamic and heterogeneous calibration yielded the expected results in both test cases, with both the keep-it-simple-and-stupid (KISS) model, as well as the elaborate and complex model of housing markets.

It should be noted that the proposed automatic calibration was specifically designed for the multi-agent systems, as one of the subroutines of the calibration included the clustering on the agent population for the segmented values of the heterogeneous parameters. If we generalize this agent-specific routine to represent the general simulation model's building blocks, this proposed calibration could be one special example of a general calibration framework.

**Acknowledgements** This work was supported by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00225).

## References

1. Bae, J. W., Lee, S., Hong, J. H., & Moon, I. C. (2014). Simulation-based analyses of an evacuation from a metropolis during a bombardment. *Simulation, 90*(11), 1244–1267.
2. Barde, S. (2017). A practical, accurate, information criterion for nth order Markov processes. *Computational Economics, 50*(2), 281–324.
3. Barde, S., & van der Hoog, S. (2017). An empirical validation protocol for large-scale agent-based models. Technicalreport, School of Economics, University of Kent.
4. Beisbart, C., & Saam, N. J. (2018). *Computer simulation validation: Fundamental concepts, methodological frameworks, and philosophical perspectives*. Berlin: Springer.
5. Bertsekas, D. P., Bertsekas, D. P., Bertsekas, D. P., & Bertsekas, D. P. (1995). *Dynamic programming and optimal control* (Vol. 1). Belmont, MA: Athena Scientific.
6. Bilmes, J. A., et al. (1998). A gentle tutorial of the EM algorithm and its application to parameter estimation for gaussian mixture and hidden Markov models. *International Computer Science Institute, 4*(510), 126.
7. Bishop, C. M. (2006). *Pattern recognition and machine learning*. Berlin: Springer.
8. Bodin, E., Kaiser, M., Kazlauskaite, I., Campbell, N. D., & Ek, C. H. (2019). Modulated bayesian optimization using latent gaussian process models. arXiv preprint arXiv:190611152
9. Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences, 99*(suppl 3), 7280–7287.
10. Bracke, P. (2015). How much do investors pay for houses? *Real Estate Economics*, *49*(S1), 41–73.
11. Brischetto, A., Voss, G., et al. (1999). *A structural vector autoregression model of monetary policy in Australia*. Sydney, NSW: Reserve Bank of Australia Sydney.
12. Brochu, E., Cora, V. M., & De Freitas, N. (2010). A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. arXiv preprint arXiv:10122599
13. Buigut, S. K., & Valev, N. T. (2005). Is the proposed east African monetary union an optimal currency area? A structural vector autoregression analysis. *World Development, 33*(12), 2119–2133.
14. Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research, 12*(Oct), 2879–2904.
15. Byrne, H., & Drasdo, D. (2009). Individual-based and continuum models of growing cell populations: A comparison. *Journal of Mathematical Biology, 58*(4–5), 657.
16. Calvez, B., & Hutzler, G. (2005). Automatic tuning of agent-based models using genetic algorithms. In *Proceedings of the 6th international conference on multi-agent-based simulation* (pp. 41–57). Berlin, Heidelberg: Springer.
17. Cares, J. R. (2002). The use of agent-based models in military concept development. In *Proceedings of the winter simulation conference* (Vol. 1, pp. 935–939). IEEE.
18. Chen, Y., & Gutmann, M. U. (2019). Adaptive gaussian copula ABC. arXiv preprint arXiv:190210704.
19. Dasgupta, S. (1999). Learning mixtures of gaussians. In *40th annual symposium on foundations of computer science (Cat. No. 99CB37039)* (pp. 634–644). IEEE.
20. Epstein, J. M. (2006). *Generative social science: Studies in agent-based computational modeling*. Princeton: Princeton University Press.
21. Ewens, W. J. (1990). Population genetics theory-the past and the future. In *Mathematical and statistical developments of evolutionary theory* (pp. 177–227). Dordrecht: Springer.
22. Fehler, M., Klügl, F., & Puppe, F. (2004). Techniques for analysis and calibration of multi-agent simulations. In *International workshop on engineering societies in the agents world* (pp. 305–321). Berlin, Heidelberg: Springer.
23. Forbes, F., & Wraith, D. (2014). A new family of multivariate heavy-tailed distributions with variable marginal amounts of tailweight: Application to robust clustering. *Statistics and Computing, 24*(6), 971–984.
24. Fox, E. B., Sudderth, E. B., Jordan, M. I., & Willsky, A. S. (2008). An HDP-HMM for systems with state persistence. In *Proceedings of the 25th international conference on Machine learning* (pp. 312–319). ACM.
25. Frazier, P. I. (2018). A tutorial on Bayesian optimization. *arXiv preprint*arXiv:180702811

26. Gini, C. (1936). On the measure of concentration with special reference to income and statistics. *Colorado College Publication, General Series, 208*, 73–79.
27. Grünewälder, S., Audibert, J. Y., Opper, M., & Shawe-Taylor, J. (2010). Regret bounds for gaussian process bandit problems. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 273–280).
28. Guerini, M., & Moneta, A. (2017). A method for agent-based models validation. *Journal of Economic Dynamics and Control, 82*, 125–141.
29. Gutmann, M. U., Dutta, R., Kaski, S., & Corander, J. (2014). Statistical inference of intractable generative models via classification. arXiv preprint arXiv:14074981
30. Hamilton, J. D. (2016) Regime switching models. *The new Palgrave dictionary of economics* pp. 1–7
31. Hansen, N., & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation, 9*(2), 159–195.
32. Hara, S., Kita, H., Ikeda, K., & Susukita, M. (2013). Configuring agents' attributes with simulated annealing. In *Agent-based approaches in economic and social complex systems VII* (pp. 45–59). Tokyo: Springer.
33. Heppenstall, A. J., Evans, A. J., & Birkin, M. H. (2007). Genetic algorithm optimisation of an agent-based model for simulating a retail market. *Environment and Planning B: Planning and Design, 34*(6), 1051–1070.
34. Hoffman, M., Brochu, E., & de Freitas, N. (2011). Portfolio allocation for Bayesian optimization. In *Proceedings of the twenty-seventh conference on uncertainty in artificial intelligence* (pp. 327–336). AUAI Press.
35. Hosseinali, F., Alesheikh, A. A., & Nourian, F. (2013). Agent-based modeling of urban land-use development, case study: Simulating future scenarios of Qazvin city. *Cities, 31*, 105–113.
36. Jurafsky, D. (2000). *Speech & language processing*. London: Pearson Education India.
37. Kingma, D. P., & Welling, M. (2013). Auto-encoding variational Bayes. *arXiv preprint*arXiv:13126114.
38. Kleijnen, J. P. (1995). Verification and validation of simulation models. *European Journal of Operational Research, 82*(1), 145–162.
39. Kosmidis, I., & Karlis, D. (2016). Model-based clustering using copulas with applications. *Statistics and Computing, 26*(5), 1079–1099.
40. Lamperti, F., Roventini, A., & Sani, A. (2018). Agent-based model calibration using machine learning surrogates. *Journal of Economic Dynamics and Control, 90*, 366–389.
41. Lewis DD, Gale WA (1994) A sequential algorithm for training text classiers. In: SIGIR'94, Springer, pp 3–12
42. Lizotte, D. J. (2008). *Practical Bayesian optimization*. Alberta: University of Alberta.
43. Mahalanobis, P. C. (1936). *On the generalized distance in statistics*. New Delhi: National Institute of Science of India.
44. Malhotra, A. (2009). Agent-based modeling in defence. *DRDO Science Spectrum.,* 60–65.
45. Manson, S. (2003). Validation and verification of multi-agent models for ecosystem management. Complexity and ecosystem management: The theory and practice of multi-agent approaches. (pp. 63–74)
46. Marjoram, P., Molitor, J., Plagnol, V., & Tavaré, S. (2003). Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences, 100*(26), 15324–15328.
47. Marks, R. E. (2013). Validation and model selection: Three similarity measures compared. *Complexity Economics, 2*(1), 41–61.
48. Marks, R. E. (2019). Validation metrics: A case for pattern-based methods. In *Computer simulation validation* (pp. 319–338). Springer.
49. Meeds, E., & Welling, M. (2014). GPS-ABC: Gaussian process surrogate approximate Bayesian computation. arXiv preprint arXiv:14012838
50. Mockus, J. (2012). *Bayesian approach to global optimization: Theory and applications*. Berlin: Springer.
51. Moon, I. C., Kim, D., Yun, T. S., Bae, J. W., Kang, D. O., & Paik, E. (2018). Data-driven automatic calibration for validation of agent-based social simulations. In *2018 IEEE international conference on systems, man, and cybernetics (SMC)* (pp. 1605–1610). IEEE.
52. Moon, T. K. (1996). The expectation-maximization algorithm. *IEEE Signal Processing Magazine, 13*(6), 47–60.
53. Morokoff, W. J., & Caflisch, R. E. (1994). Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing, 15*(6), 1251–1279.
54. Murphy, K. P. (2007). Conjugate Bayesian analysis of the gaussian distribution. def 1(22):16

55. Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. Cambridge: MIT Press.
56. Naiem, A., Reda, M., El-Beltagy, M., & El-Khodary, I. (2010). An agent based approach for modeling traffic flow. In *2010 The 7th international conference on informatics and systems (INFOS)* (pp. 1–6). IEEE.
57. Nguyen, A. T., Reiter, S., & Rigo, P. (2014). A review on simulation-based optimization methods applied to building performance analysis. *Applied Energy, 113*, 1043–1058.
58. Noel, M. M., & Jannett, T. C. (2004). Simulation of a new hybrid particle swarm optimization algorithm. In *Proceedings of the Thirty-sixth southeastern symposium on system theory, 2004* (pp. 150–153). IEEE.
59. Peel, D., & McLachlan, G. J. (2000). Robust mixture modelling using the t distribution. *Statistics and Computing, 10*(4), 339–348.
60. Preuss, R., & von Toussaint, U. (2018). Global optimization employing gaussian process-based Bayesian surrogates. *Entropy, 20*(3), 201.
61. Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE, 77*(2), 257–286.
62. Railsback, S. F., & Grimm, V. (2019). *Agent-based and individual-based modeling: A practical introduction*. Princeton: Princeton University Press.
63. Rasmussen, C., & Williams, C. (2006). *Gaussian Processes for Machine Learning. Adaptive Computation and Machine Learning*.
64. Recchioni, M. C., Tedeschi, G., & Gallegati, M. (2015). A calibration procedure for analyzing stock price dynamics in an agent-based framework. *Journal of Economic Dynamics and Control, 60*, 1–25.
65. Ryzhov, I. O. (2016). On the convergence rates of expected improvement methods. *Operations Research, 64*(6), 1515–1528.
66. Sargent, R. G. (2010). Verification and validation of simulation models. In *Proceedings of the 2010 winter simulation conference* (pp. 166–183). IEEE.
67. Sasena, M. J., Papalambros, P., & Goovaerts, P. (2002). Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization, 34*(3), 263–278.
68. Schonlau, M. (1997). Computer experiments and global optimization. PhD thesis, University of Waterloo, Waterloo, Ont., Canada, Canada, aAINQ22234.
69. Scogings, C., & Hawick, K. (2012). An agent-based model of the battle of Isandlwana. In *Proceedings of the 2012 winter simulation conference (WSC)* (pp. 1–12). IEEE.
70. Sethuraman, J. (1994). A constructive definition of dirichlet priors. *Statistica Sinica*, *4*, 639–650.
71. Silver, M., & Heravi, S. (2007). Why elementary price index number formulas differ: Evidence on price dispersion. *Journal of Econometrics, 140*(2), 874–883.
72. Sisson, S. A., Fan, Y., & Tanaka, M. M. (2007). Sequential Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences, 104*(6), 1760–1765.
73. Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems* (pp. 2951–2959). Lake Tahoe: Harrahs and harveys.
74. Sóbester, A., Leary, S. J., & Keane, A. J. (2005). On the design of optimization strategies based on global response surface approximation models. *Journal of Global Optimization, 33*(1), 31–59.
75. Stonedahl, F., & Wilensky, U. (2010a). Evolutionary robustness checking in the artificial anasazi model. In *2010 AAAI fall symposium series*.
76. Stonedahl, F., & Wilensky, U. (2010b). Finding forms of flocking: Evolutionary search in abm parameter-spaces. In *International workshop on multi-agent systems and agent-based simulation* (pp. 61–75). Berlin, Heidelberg: Springer.
77. Suri, R. (1987). Infinitesimal perturbation analysis for general discrete event systems. *Journal of the ACM (JACM), 34*(3), 686–717.
78. Tavaré, S., Balding, D. J., Griffiths, R. C., & Donnelly, P. (1997). Inferring coalescence times from DNA sequence data. *Genetics, 145*(2), 505–518.
79. Teh, Y. W. (2010). Dirichlet process. *Encyclopedia of Machine Learning* (pp 280–287).
80. Tesfatsion, L. (2002). Agent-based computational economics: Growing economies from the bottom up. *Artificial Life, 8*(1), 55–82.
81. Tesfatsion, L. (2006). Agent-based computational economics: A constructive approach to economic theory. *Handbook of Computational Economics, 2*, 831–880.
82. Thacker, B. H., Doebling, S. W., Hemez, F. M., Anderson, M. C., Pepin, J. E., & Rodriguez, E. A. (2004). Concepts of model verification and validation. Tech. rep., Los Alamos National Lab.
83. Thiele, J. C., Kurth, W., & Grimm, V. (2014). Facilitating parameter estimation and sensitivity analysis of agent-based models: A cookbook using NetLogo and R. *Journal of Artificial Societies and Social Simulation, 17*(3), 11.

84. Tresidder, E., Zhang, Y., & Forrester, A. (2012). Acceleration of building design optimisation through the use of kriging surrogate models. *Proceedings of building simulation and optimization* (pp. 1–8). Loughborough, UK.
85. Valiant, L. (2013). Probably Approximately Correct: NatureÕs Algorithms for Learning and Prospering in a Complex World. Basic Books (AZ)
86. Vazquez, E., & Bect, J. (2010). Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference, 140*(11), 3088–3095.
87. Whittaker, G., Confesor, R., Di Luzio, M., Arnold, J., et al. (2010). Detection of overparameterization and overfitting in an automatic calibration of swat. *Transactions of the ASABE, 53*(5), 1487–1499.
88. Wilensky, U. (1998). Netlogo wealth distribution model. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. https://ccl.northwestern.edu/netlogo/models/WealthDistribution.
89. Willems, F. M., Shtarkov, Y. M., & Tjalkens, T. J. (1995). The context-tree weighting method: Basic properties. *IEEE Transactions on Information Theory, 41*(3), 653–664.
90. Wilson, A., & Adams, R. (2013). Gaussian process kernels for pattern discovery and extrapolation. In *International conference on machine learning* (pp. 1067–1075).
91. Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature, 466*(7310), 1102.
92. Yun, T. S., Moon, I. C., et al. (2020). Housing market agent-based simulation with loan-to-value and debt-to-income. *Journal of Artificial Societies and Social Simulation, 23*(4), 1–5.