CYBERNETICS AND PHYSICS, VOL. 14, NO. 1, 2025, 67–73

DECENTRALIZED SPSA-BASED CORRECTION OF TASK TIME PREDICTIONS IN ADAPTIVE MULTI-AGENT SYSTEMS

Elizaveta Tarasova
Saint Petersburg State University
Russia
elizaveta.tarasova@spbu.ru

Ekaterina Moseiko
Saint Petersburg State University
Russia
e.moseyko@spbu.ru

Article history:
Received 09.06.2025, Accepted 21.06.2025

Abstract

We propose a decentralized method for adaptive prediction of task processing times in dynamic environments. Each controller independently estimates the expected duration of incoming tasks using a gradient-free update rule based on the Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm. To ensure coherence across the system, controllers synchronize their models through consensus over a time-varying communication graph. This approach enables efficient learning under limited observability and noisy feedback, without requiring access to gradients or global information.

We provide theoretical guarantees for convergence under bounded noise, drawing on recent results for distributed SPSA with consensus. Simulations demonstrate the method's resilience to abrupt changes in task behavior ("drift") and show that it outperforms baseline methods in terms of prediction accuracy and inter-controller consistency. We further illustrate how the SPSA+Cons approach can be deployed in a modular multi-agent AI platform to align operational parameters across heterogeneous agents under uncertainty.

The proposed solution is lightweight, fully decentralized, and suitable for a variety of settings where centralized control is infeasible or costly. Potential applications include collaborative scheduling, sensor coordination, and adaptive task routing in large-scale systems.

Key words

Decentralized learning, SPSA, consensus algorithms, task duration prediction, gradient-free optimization, distributed systems

1 Introduction

We focus on the problem of decentralized task allocation in dynamic and uncertain environments, where agents must operate without centralized coordination. Such environments include robotic fleets, service systems, and distributed computational platforms. In these settings, tasks arrive continuously, and individual components must make decisions based on local observations, partial information, and noisy feedback.

Our approach focuses on distributed controllers that analyze incoming tasks and predict their expected processing time. Once the prediction is made, the task is forwarded to a decentralized marketplace, where peer components self-organize to select an executor using a multi-agent negotiation mechanism. This setup allows the system to remain flexible and scalable, without requiring a global scheduler.

The core contribution of this work is a decentralized method for synchronizing prediction models across controllers. Each controller updates its model locally based on recent task outcomes, while also periodically exchanging information with its neighbors to improve accuracy and maintain consistency. We use a gradient-free optimization technique based on Simultaneous Perturbation Stochastic Approximation (SPSA), which enables adaptation under noise, delays, and limited observability.

This approach is model-agnostic and supports any parametric prediction method. For example, a controller may use a simple linear model or a tree-based ensemble; our synchronization mechanism ensures that even under drift or data sparsity, the prediction models converge.

Simulation results show that our method maintains accurate task-time predictions and remains robust under distributional shifts and partial feedback. The system demonstrates good scalability and low coordination overhead compared to classical optimization-based baselines.

Beyond theoretical motivation and synthetic benchmarks, the proposed method can be directly applied in distributed AI platforms composed of interacting agents (e.g., MLOps, DevOps, resource managers). In such environments, coordination of noisy, partially observable decisions becomes essential. We demonstrate this in Section 8, where SPSA+Cons is used for cross-agent synchronization and drift adaptation under live constraints.

2 Related Work

Decentralized task allocation has been widely studied in multi-agent systems, with popular approaches including auction-based protocols [Braquet and Bakolas, 2021; Andreev et al., 2007] and evolutionary heuristics [Patel et al., 2020]. While effective in some scenarios, these methods often rely on global task visibility or fixed coordination strategies, making them less suitable in environments with uncertainty or partial observability.

Distributed optimization techniques provide more flexibility and robustness, but many of them require access to exact gradients [Nedic and Ozdaglar, 2009], which is not always practical in real systems. A fundamentally different class of methods, based on gradient-free optimization, was pioneered in [Granichin, 1989], where a randomized approximation scheme using input perturbations was proposed. This approach demonstrated consistency even under correlated observation noise. Subsequent developments [Granichin, 1992; Granichin, 2002] extended the framework to broader classes of systems, including those with arbitrary bounded disturbances. These works laid the foundation for the class of methods now known as Simultaneous Perturbation Stochastic Approximation (SPSA) [Spall, 1992; Spall, 1997], where updates require only one or two function evaluations per iteration regardless of the dimension. The asymptotic optimality of gradient-free search procedures was formally established in [Polyak and Tsybakov, 1990], where the authors proved that no other iterative optimization method can outperform the proposed scheme (in terms of convergence rate) over a wide class of stochastic problems. This result provides a strong theoretical underpinning for the use of randomized algorithms in settings with noisy or uncertain feedback.

SPSA-based algorithms have shown strong applicability in tracking tasks [Granichin and Amelina, 2015], including nonstationary optimization problems [Vakhitov et al., 2009] and large-scale consensus control [Erofeeva et al., 2021; Erofeeva et al., 2025]. A comprehensive treatment of randomized approximation under unknown-but-bounded disturbances, including convergence rate analysis and real-world applications, is provided in [Granichin et al., 2015]. These methods have since been applied to control, estimation, and network optimization tasks. Their stochastic structure enables implementation under limited feedback and, in some cases, allows natural extensions to quantum computation frameworks [Vakhitov et al., 2006]. The method proposed in this work builds on these foundations and adapts decentralized SPSA for predictive modeling in collaborative systems under dynamic and partially observable conditions.

From a system perspective, our work is aligned with ideas from cyber-physical systems (CPS), where distributed sensing, computation, and local decision-making are used to adapt to dynamic environments [Rajkumar et al., 2010]. Related concepts include synchronization in networks of oscillators [Arenas et al., 2008] and decentralized control with limited communication [Sandell Jr et al., 1978].

This paper contributes to the ongoing research on decentralized adaptation in cyber-physical systems. Recent studies have addressed distributed optimization under unknown-but-bounded disturbances using projection and separation principles [Kizhaeva and Erofeeva, 2023], as well as randomized multi-agent control of sensor networks under partial observability and dynamic communication topologies [Sergeenko and Granichin, 2022]. The method proposed in this work complements these directions by offering a decentralized, gradient-free strategy for predictive adaptation and synchronization in task allocation scenarios, with theoretical guarantees under bounded uncertainty.

3 System Model

We consider a decentralized system that processes a continuous stream of tasks $T = \{T_1, T_2, \dots\}$. Each task T_i is characterized by observable features such as arrival time r_i, type $type_i$, and estimated complexity $h(type_i)$. Some additional parameters, such as urgency or deadline D_i, may be inferred from historical context.

Each task is initially handled by a distributed controller, which uses a parametric model to predict the expected processing time $\hat{p}_i$ based on the task's feature vector x_i:

$$\hat{p}_i = f(x_i, \theta),$$

where f is a parameterized prediction model (e.g., linear regression or decision-tree ensemble), and θ is the model parameter vector.

After the prediction is made, the task is forwarded to a decentralized marketplace, where agents participate in peer-to-peer assignment using local policies. The controller does not make the final assignment, but only contributes a time estimate used in downstream decision-making.

To adapt to nonstationary environments and delayed feedback, each controller maintains a local history of completed tasks and periodically updates its model parameters based on observed errors. In addition, controllers exchange partial information with their neighbours over a dynamic communication graph $\mathcal{G}_t$, defined by a time-varying symmetric stochastic matrix B_t.

We assume that the average graph $\mathcal{G}_{\text{av}} = \mathbb{E}[\mathcal{G}_t]$ is connected, ensuring long-term information propagation across the network. This setup supports decentralized learning under noise, limited visibility, and temporal drift in task characteristics.

4 Decentralized Adaptive Controllers

Each task T_i is initially handled by a local controller, which is responsible for predicting the expected processing time before the task is released to the decentralized assignment mechanism. Controllers operate independently, using local data and partial feedback from previously processed tasks.

4.1 Task Time Prediction

For each incoming task T_i, the controller extracts a feature vector $x_i \in \mathbb{R}^d$ describing relevant attributes such as task type, estimated complexity, and contextual metadata. The controller then predicts the expected processing time $\hat{p}_i$ using a parametric model of the form:

$$\hat{p}_i = f(x_i, \theta), \tag{1}$$

where $\theta \in \mathbb{R}^d$ is the current parameter vector. The prediction model f may take different forms, such as a linear regression model or a decision-tree ensemble. Controllers may use different model architectures, but they share the same parameter structure and synchronize θ using a decentralized optimization mechanism.

4.2 Model Synchronization via Decentralized SPSA

To ensure consistency across distributed controllers and adapt to the dynamics of the task environment, we apply a decentralized, gradient-free optimization method based on Simultaneous Perturbation Stochastic Approximation (SPSA) with consensus. We follow the asynchronous formulation proposed in [Granichin et al., 2021], where parameter updates occur every two time steps.

Let $\hat{\theta}^i_{2k-2}$ denote the model parameters of the controller C_i at iteration $2k-2$, and let $\Delta^i_k \in \{-\frac{1}{\sqrt{d}}, +\frac{1}{\sqrt{d}}\}^d$ be a random perturbation vector. The perturbed parameter vectors are defined as:

$$\begin{aligned} x^{i,+}_{2k-1} &= \hat{\theta}^i_{2k-2} + \beta_k \Delta^i_k, \\ x^{i,-}_{2k} &= \hat{\theta}^i_{2k-2} - \beta_k \Delta^i_k, \end{aligned} \tag{2}$$

where $\beta_k > 0$ is the perturbation magnitude.

Each controller observes the processing time p^{real}_t of a task T^i_t and evaluates its prediction model f on the current task feature vector x^i_t. The prediction error is normalized by the estimated complexity $h(T^i_t)$, and the observed loss is given by:

$$F^i_t(\theta) = \left(\frac{f(x^i_t, \theta) - p^{\text{real}}_t}{h(T^i_t) \cdot p^{\text{real}}_t + \epsilon} \right)^2, \tag{3}$$

where $\epsilon > 0$ is a regularization constant to prevent division by small values, $h(T_i)$ is a complexity estimate derived from historical duration statistics grouped by type and urgency. Errors in $h(T_i)$ may lead to biased learning, but are mitigated through regularization and consensus averaging.

Based on two consecutive evaluations, we define the SPSA gradient estimate:

$$g^i_k := \Delta^i_k \cdot \frac{F^i_{2k-1}(x^{i,+}_{2k-1}) - F^i_{2k}(x^{i,-}_{2k})}{2\beta_k}. \tag{4}$$

The parameter update with consensus is then performed as:

$$\hat{\theta}^i_{2k} = \hat{\theta}^i_{2k-2} - \alpha_k \left(g^i_k + \gamma_k \sum_{j \in \mathcal{N}^i_k} b^{ij}_k (\hat{\theta}^i_{2k-2} - \hat{\theta}^j_{2k-2}) \right), \tag{5}$$

where $\alpha_k > 0$ is the learning rate, $\gamma_k > 0$ is the consensus gain, b^{ij}_k are the weights of the communication graph at iteration k, and $\mathcal{N}^i_k$ is the set of neighbours of the controller i.

This approach enables each controller to improve its predictive performance based on local feedback, while gradually aligning its model with others in the network through pairwise exchanges. The algorithm requires no explicit gradient computations and remains stable under noisy observations and limited communication.

In practical deployments, each controller executes the SPSA update and consensus step in regular intervals, typically aligned with monitoring or task assignment cycles (e.g., every 30–60 seconds in systems with 5-second SLA windows). The gradient-free update (Step 1) and local evaluation (Step 2) are performed independently and asynchronously across nodes, while consensus averaging (Step 3) requires occasional communication but can tolerate delays or partial participation without breaking convergence guarantees.

5 Execution Agent Behavior

Execution agents E_j are responsible for processing tasks assigned through the task exchange mechanism. Once a task is assigned, the agent executes it without interruption. Each agent maintains its current status $s_j(t)$, which indicates whether the agent is busy or idle at global time t.

The estimated time of availability is computed based on the agent's current workload. If the agent j is currently executing task T_i, which started at the time τ_i and has predicted processing time $\hat{p}_i$, then the expected time at which the agent becomes available is:

$$t^{\text{ready}}_j = \tau_i + \hat{p}_i. \tag{6}$$

If the agent is idle at the time t, we assume $t^{\text{ready}}_j = t$. This value is used by controllers to estimate when the agent can receive new tasks.

After completing a task, each agent provides feedback to the issuing controller, including the actual processing time p^{real}, the start time τ_i, and the finish time. This feedback is used to improve prediction accuracy and adapt the model to observed execution dynamics.

6 Theoretical Foundation

Each controller updates its local prediction model using a gradient-free Simultaneous Perturbation Approximation (SPA) method combined with consensus-based synchronization. Despite perturbations in observed processing times, our update rule remains stable due to the boundedness (but not randomness) of observation noise.

The theoretical guarantees follow from the convergence results established in [Granichin et al., 2021], specifically Theorem 1, which considers distributed gradient-free optimization over a connected communication graph and assumes that the optimization is subject to piecewise-smooth loss functions and non-random but bounded perturbations.

Let B_t denote the communication matrix at iteration t, and let $B_{\text{av}} = \mathbb{E}[B_t]$ be the expected matrix. The second-smallest eigenvalue of the corresponding Laplacian matrix $L = I - B_{\text{av}}$ is denoted $\bar{\lambda}_2 = \lambda_2(L)$.

Theorem. Let each controller update its local model using the decentralized SPSA algorithm with consensus under bounded input perturbations and constant step size $\alpha > 0$. Suppose the following conditions are satisfied:

1. each local loss function is convex and has a shared minimizer;
2. the regularized cost functions admit Lipschitz-continuous subgradients;
3. the squared norm of the gradient estimate is uniformly bounded;
4. the iterates exhibit bounded drift, and local function evaluations vary smoothly;
5. the communication matrices B_t are symmetric, stochastic, i.i.d., and their expectation defines a connected graph.

Then, under these assumptions, the decentralized optimization method converges in the mean-square sense, provided that the step size satisfies

$$\alpha < \frac{\bar{\lambda}_2}{c_1 + \sqrt{c_2 c_m}},$$

where $\bar{\lambda}_2 = \lambda_2(L(\mathbb{E}[B_t]))$ is the second smallest eigenvalue of the Laplacian of the expected communication graph, and the constants c_1, c_2, and c_m depend on problem parameters and noise bounds.

Moreover, under this condition, the following performance bound holds:

$$\limsup_{k\to\infty} \mathbb{E}\left[\|\theta_{2k}^{(c)} - \theta^*\|^2\right] \le C,$$

where C is a constant whose value depends on system-specific parameters, step-size, communication topology, and noise bounds.

This result follows from Theorem 1 in [Granichin et al., 2021]. In our case, where the observation noise is unknown but bounded rather than stochastic, the constants c_2 and c_m are determined using worst-case bounds rather than expectations. Verification of the five required assumptions is provided in Appendix A.

As an illustrative example, consider a fully connected communication graph where every controller communicates with each other with fixed probability $p > 0$ at each step. The normalized matrix B_t assigns weights $b_t^{ij} = \frac{1}{m-1}$ for $i \neq j$ when active, and selects diagonal entries to ensure row sums equal 1. Then the Laplacian is $L = I - B_{\text{av}}$ with

$$\bar{\lambda}_2 = 1 - \frac{1}{m-1}.$$

For $m = 5$ controllers, we obtain $\bar{\lambda}_2 = \frac{3}{4}$, yielding an explicit upper bound on α.

Lipschitz-continuous convex losses arise naturally in many real-world systems, for example, when penalties increase smoothly after SLA violations or when cost functions grow linearly with delay or load. These structures are typical for applications involving bounded resources, latency-sensitive operations, or adaptive service control.

7 Simulation and Comparative Analysis

To evaluate the proposed method (referred to as **SPSA+Cons**), we conducted 1,000 simulation runs with time-varying task behavior. In each scenario, a sudden change ("drift") occurred halfway through the execution, altering the underlying processing time pattern. This tests the ability of different methods to adapt under changing conditions and noisy feedback.

We compared SPSA+Cons against the following baselines:

NoSPSA-NoCons (F). Each controller updates its parameters independently using basic local error minimization, without communication or coordination. This baseline reflects isolated learning.

NoSPSA-NoCons (T). Same as above, but includes an additional stabilization term in the update rule. However, no inter-controller interaction is applied.

DistSubgrad (partial). A classical distributed subgradient approach that performs parameter averaging between neighboring controllers. In this setup, each node observes only partial and noisy information, which limits the quality of gradient estimates.

SPSA+Cons combines gradient-free updates based on observed processing times with consensus-based synchronization across the network. It does not require access to explicit gradients or centralized coordination.

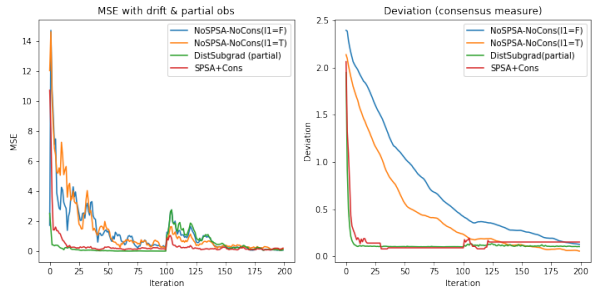

Figure 1. Comparative results under a drifting scenario. *Left:* average prediction error over time. SPSA+Cons (red) adapts quickly and maintains low error. Baselines struggle to recover after the drift. *Right:* parameter consistency across controllers. SPSA+Cons keeps models aligned, while baselines without consensus diverge. Results are averaged over 1,000 random seeds.

Figure 1 shows that SPSA + Cons achieves accurate prediction and consistent controller states, even after abrupt changes in system dynamics. The comparison confirms that local updates alone are insufficient, and unreliable gradients reduce the stability of distributed subgradient methods.

8 Application of SPSA+Cons in a Multi-Agent AI Platform

In the context of a multilayer, agent-based AI platform (the AIoT Constructor), where different agents are responsible for functions such as resource management, MLOps, DevOps, logging, and so on, SPSA+Cons can be used to adaptively align and optimize their operational parameters.

An illustrative use case is the dynamic allocation of compute resources between model-serving agents (MLOps agents) and monitoring agents (resource manager agents) under variable platform load.

The technical challenge arises from the fact that inference latency and data-stream processing times can change abruptly (drift caused by new sensor events, load fluctuations, node failures), while feedback from agents (latency and throughput metrics) is noisy aand only partially visible due to the trade-off between high-quality monitoring (which consumes additional compute) and allocating those same resources to task execution.

The SPSA+Cons solution for this problem works as follows:

1. **Local parameter vectors.** Each agent maintains its own vector of parameters (e.g., queue-priority weights, batch sizes, mode-switch thresholds).
2. **SPSA update cycle.** In each SPSA cycle, an agent perturbs its parameters in small random directions, measures the resulting change in overall latency (or another quality metric) and performs a stochastic approximation of the gradient.
3. **Consensus averaging.** Periodically, all agents exchange their current parameter estimates through a consensus protocol and average them according to the platform's communication topology (e.g., agent-to-agent, broker-mediated, or neighborhood-based exchange). This alignment prevents disparate or conflicting configurations.
4. **Rapid adaptation after drift.** Following a drift event (e.g. latency doubling due to a node failure), SPSA+Cons enables quick recovery: local SPSA steps re-tune parameters and consensus ensures the entire agent pool converges toward a new, shared optimal configuration.

A key advantage of this approach is that it does not require an explicit model of the performance gradient of the system. Simulation results indicate strong robustness to noisy metrics and partial telemetry loss, while preserving parameter consistency across agents, which is critical for coordinated actions (for example, synchronized 'training'/'inference' mode switches). Moreover, both horizontal and vertical scalability are supported, since consensus exchanges can occur only among logically adjacent agents without a central coordinator.

This methodology can also be extended to coordinate hyperparameter selection in distributed training, dynamically balance data channels between Streaming agents and Analytics agents, and adaptively manage SLA thresholds in the DevOps agent based on real-time traffic and performance.

9 Conclusion

We presented a decentralized method for adaptive task duration prediction based on gradient-free SPSA updates and consensus-based synchronization between controllers. The approach is robust to noisy feedback and sudden changes in task characteristics, making it suitable for systems with partial observability and limited prior knowledge.

Theoretical analysis guarantees convergence under bounded noise and connectivity assumptions on the communication graph. Experimental results confirm that the method maintains low prediction error and consistency across controllers, even in environments with drift.

We further demonstrated how the proposed approach can be integrated into a modular multi-agent AI platform, where it supports dynamic coordination and alignment of operational parameters across heterogeneous agents under uncertainty. This use case highlights the method's broader applicability in decentralized optimization scenarios beyond task prediction.

Future work will explore asynchronous updates, real-time deployment in physical systems, and integration with other forms of distributed decision-making. Potential application areas include collaborative scheduling, smart manufacturing, and sensor-based monitoring systems. A related method for change tracking based on input-perturbed gradient-free optimization was recently

proposed by the authors in [Akinfiev et al., 2025], where theoretical convergence and tracking properties were established in a simpler setting. The present work extends those ideas to a distributed, consensus-driven architecture suitable for multi-agent task coordination.

Acknowledgements

The research was carried out at Saint Petersburg University, project no. 121061000159-6.

References

Akinfiev, I., Granichin, O., and Tarasova, E. (2025). A gradient-free optimization method with input perturbations for tracking parameter changes under bounded noise. *Automation and Remote Control*, **86** (8).

Andreev, M., Rzevski, G., Skobelev, P., Shveykin, P., Tsarev, A., and Tyugashev, A. (2007). Adaptive planning for supply chain networks. pp. 215–224.

Arenas, A., Díaz-Guilera, A., Kurths, J., Moreno, Y., and Zhou, C. (2008). Synchronization in complex networks of nonlinear oscillators. *Physics Reports*, **469** (3), pp. 93–153.

Braquet, M. and Bakolas, E. (2021). Greedy decentralized auction-based task allocation for multi-agent systems∗∗this work was supported in part by a fellowship of the belgian american educational foundation. *IFAC-PapersOnLine*, **54** (20), pp. 675–680. Modeling, Estimation and Control Conference MECC 2021.

Erofeeva, V., Granichin, O., Granichina, O., Proskurnikov, A., and Sergeenko, A. (2021). Weighted spsa-based consensus algorithm for distributed cooperative target tracking. pp. 1074–1079.

Erofeeva, V., Granichin, O., and Uzhva, D. (2025). Meso-scale coalitional control in large-scale networks. *Automatica*, **177**, pp. 112276.

Granichin, O. (1989). Stochastic approximation with input perturbation under dependent observation noises. *Vestnik Leningrad University. Series 1: Mathematics, Mechanics, Astronomy*, (4), pp. 27–31.

Granichin, O. (1992). Procedure of stochastic approximation with disturbances at the input. *Automation and Remote Control*, **53** (2), pp. 232–237.

Granichin, O. (2002). Randomized algorithms for stochastic approximation under arbitrary disturbances. *Automation and Remote Control*, **63** (2), pp. 209–219.

Granichin, O. and Amelina, N. (2015). Simultaneous perturbation stochastic approximation for tracking under unknown but bounded disturbances. *IEEE Transactions on Automatic Control*, **60**, pp. 1653–1658.

Granichin, O., Erofeeva, V., Ivanskiy, Y., and Jiang, Y. (2021). Simultaneous perturbation stochastic approximation-based consensus for tracking under unknown-but-bounded disturbances. *IEEE Transactions on Automatic Control*, **66** (8), pp. 3710–3717.

Granichin, O., Volkovich, V., and Toledano-Kitai, D. (2015). *Randomized Algorithms in Automatic Control and Data Mining*, vol. 67 of *Intelligent Systems Reference Library*. Springer.

Kizhaeva, N. and Erofeeva, V. (2023). Partially observed distributed optimization under unknown–but–bounded disturbances. *Cybernetics and Physics*, **12** (1), pp. 16–22.

Nedic, A. and Ozdaglar, A. (2009). Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, **54** (1), pp. 48–61.

Patel, A., Rudnick-Cohen, D., Azarm, S., Otte, M., Xu, H., and Herrmann, J. W. (2020). Decentralized task allocation in multi-agent systems using a decentralized genetic algorithm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11061–11067.

Polyak, B. T. and Tsybakov, A. B. (1990). Optimal accuracy orders of search algorithms in stochastic optimization problems. *Problems of Information Transmission*, **26** (2), pp. 45–53.

Rajkumar, R., Lee, I., Sha, L., and Stankovic, J. (2010). Cyber-physical systems: The next computing revolution. *Proceedings of the IEEE*, **100** (Special Centennial Issue), pp. 834–850.

Sandell Jr, N. R., Varaiya, P., Athans, M., and Safonov, M. G. (1978). Survey of decentralized control methods for large scale systems. *IEEE Transactions on Automatic Control*, **23** (2), pp. 108–128.

Sergeenko, A. and Granichin, O. (2022). Sensor network control based on randomized and multi-agent approaches. *Cybernetics and Physics*, **11** (2), pp. 94–105.

Spall, J. (1992). Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, **37** (3), pp. 332–341.

Spall, J. (1997). A one-measurement form of simultaneous perturbation stochastic approximation. *Automatica*, **33** (1), pp. 109–112.

Vakhitov, A., Granichin, O., and Gurevich, L. (2009). Algorithm for stochastic approximation with trial input perturbation in the nonstationary problem of optimization. *Automation and Remote Control*, **70** (11), pp. 1827–1835.

Vakhitov, A. T., Granichin, O. N., and Sysoev, S. S. (2006). A randomized stochastic optimization algorithm: Its estimation accuracy. *Automation and Remote Control*, **67** (4), pp. 589–597.

Appendix A. Convergence Verification under Bounded Perturbations

We verify that the decentralized SPSA method with consensus update satisfies the assumptions required for convergence in the presence of *unknown but bounded* noise. The local objective at controller C_c is:

$$F^{(c)}(p) = \mathbb{E}_{i \in \mathcal{H}_k^{(c)}} \left[\left(\max \left\{ \left| \hat{p}_i + x_i^\top p - p_i^{\text{real}} \right| - \delta,\ 0 \right\} \right)^2 \right], \tag{7}$$

where $p \in \mathbb{R}^d$ is the parameter vector. The perturbations in observations are deterministic, unknown, but satisfy $|\xi_k^{\pm(c)}| \leq \xi_{\max}$.

A1. Monotonicity

Define

$$f_i(p) = \left(\max\left\{\left|x_i^\top p + \hat{p}_i - p_i^{\text{real}}\right| - \delta,\ 0\right\}\right)^2.$$

Let $z_i(p) := x_i^\top p + \hat{p}_i - p_i^{\text{real}}$. The subgradient is

$$\nabla f_i(p) = \begin{cases} 2(|z_i| - \delta) \cdot \operatorname{sign}(z_i)\, x_i, & \text{if } |z_i| > \delta, \\ 0, & \text{otherwise.} \end{cases}$$

Since f_i is convex, for any p, p^*:

$$\langle p - p^*,\ \nabla f_i(p) \rangle \geq 0,$$

and hence

$$\langle p - p^*,\ \mathbb{E}_i[\nabla f_i(p)] \rangle \geq 0.$$

Assumption A1 is satisfied.

A2. Lipschitz Gradient

Let $p_1, p_2 \in \mathbb{R}^d$. For $z_1 = z_i(p_1)$ and $z_2 = z_i(p_2)$:

$$\|\nabla f_i(p_1) - \nabla f_i(p_2)\| \leq 4\|x_i\|^2 \cdot \|p_1 - p_2\|.$$

Thus, f_i has Lipschitz gradient with $M_i = 4\|x_i\|^2$, and $M = \max_i M_i$.

A3. Bounded Gradient Norm

Let $z_i = x_i^\top p_k + \hat{p}_i - p_i^{\text{real}}$. Then:

$$\|\nabla f_i(p_k)\|^2 \leq \begin{cases} 4(|z_i| - \delta)^2 \cdot \|x_i\|^2, & |z_i| > \delta, \\ 0, & \text{otherwise.} \end{cases}$$

Since $|z_i| \leq \|x_i\| \cdot \|p_k\| + |\hat{p}_i - p_i^{\text{real}}|$, we obtain:

$$\|\nabla f_i(p_k)\|^2 \leq 4\|x_i\|^2 \left(\|x_i\| \cdot \|p_k\| + |\hat{p}_i - p_i^{\text{real}}| - \delta\right)^2.$$

Let

$$g_2^2 := \max_i 4\|x_i\|^2 \left(\|x_i\| \cdot \|p_k\| + |\hat{p}_i - p_i^{\text{real}}| - \delta\right)^2.$$

Then $\mathbb{E}\|\nabla f_i(p_k)\|^2 \leq g_2^2$, and A3 is satisfied.

A4. Bounded Step Size Drift

From the SPSA-consensus update:

$$p_{k+1}^{(c)} = p_k^{(c)} - \alpha \left(\hat{g}_k^{(c)} + \gamma \sum_{j \in \mathcal{N}_c} b_{cj} (p_k^{(j)} - p_k^{(c)}) \right)$$

we analyse the magnitude of the update using the standard SPSA gradient approximation.

$$\begin{aligned} &\hat{g}_k^{(c)} = \frac{F^{(c)}(p^+) - F^{(c)}(p^-)}{2\theta_k} \cdot \Delta_k^{(c)} + \eta_k^{(c)}, \\ &|F(p^+) - F(p^-)| \leq 2L\theta_k \sqrt{d}, \\ &|\eta_k^{(c)}| \leq \frac{2\xi_{\max}}{\theta_k} \sqrt{d}. \end{aligned} \tag{8}$$

Using these bounds 8, we obtain the following estimate on the update step:

$$\begin{gathered} \|p_{k+1}^{(c)} - p_k^{(c)}\| \leq \alpha \left(\left(L + \frac{\xi_{\max}}{\theta_k} \right) d + \gamma R \right), \\ R = \max_{j \in \mathcal{N}_c} \|p_k^{(j)} - p_k^{(c)}\|. \end{gathered} \tag{9}$$

Bounded for fixed α, so A4 holds.

A5. Communication Matrix

Let $B_k = [b_k^{ij}]$ be the symmetric stochastic matrix for controller graph. Assume:

$$b_k^{ii} = 1 - \sum_{j \neq i} b_k^{ij}, \quad \sum_j b_k^{ij} = 1,$$

and the expected graph is connected with $\lambda_2(I - \mathbb{E}[B_k]) > 0$. Then B_k satisfies symmetry, boundedness, and connectivity. A5 is satisfied.