# Optimizing SAP Predictive Analytics with Automated Hyperparameter Tuning Using Differentiable Optimization

Ashwini Chandrakumara[1]

Capgemini Technology Services India Limited, Bangalore, India
ashwini.chanda@hotmail.com

**Abstract.** As enterprises increasingly rely on SAP Predictive Analytics for forecasting, anomaly detection, and operational decision-making, optimizing model performance remains a critical challenge. Traditional methods for hyperparameter tuning in SAP AI Core and SAP HANA Predictive Analytics Library (PAL) involve manual tuning or open-loop optimization, both of which can lead to inefficiencies, suboptimal performance, or increased computational costs. To address this, I introduce SAP-PHT (SAP Predictive Hyperparameter Tuning), an automated hyperparameter optimization framework based on differentiable optimization and backpropagation techniques. SAP-PHT leverages data-driven predictive control (DeePC) methods to optimize model parameters dynamically, ensuring robust closed-loop performance without manual intervention. By integrating iterative learning strategies and adaptive parameter tuning, SAP-PHT enhances model accuracy, stability, and resilience against data variations. Experimental evaluations within SAP S/4HANA and SAP Business Technology Platform (BTP) demonstrate significant improvements in predictive model performance, reducing errors in demand forecasting, supply chain optimization, and financial risk assessments. The results highlight the potential of SAP-PHT in streamlining SAP AI-based analytics, improving business decision-making, and reducing operational costs.

**Keywords:** SAP Predictive Analytics · Hyperparameter Tuning · Differentiable Optimization · Data-Enabled Predictive Control (DeePC) · Backpropagation · Automated Model Optimization · Demand Forecasting · Supply Chain Optimization · Financial Risk Assessment · SAP S/4HANA · SAP Business Technology Platform (BTP)

## 1 Introduction

In recent years, direct data-driven control methodologies have gained renewed attention, primarily due to advancements in machine learning techniques and the increased availability of large datasets. Unlike conventional control approaches, which involve a two-step process consisting of system identification followed by


R. Nagariya et al. (eds.), *Proceedings of the International Conference on Sustainable Business Practices and Innovative Models (ICSBPIM-2025)*, Advances in Economics, Business and Management Research 349,
https://doi.org/10.2991/978-94-6463-872-1_36

model-based control, direct data-driven techniques determine control actions directly from data. By leveraging principles from behavioral system theory and convex optimization, these methods offer an alternative to traditional model-based control strategies while reducing implementation complexity and addressing inherent modeling uncertainties.

Among the various methodologies that have emerged within this domain, Data-enabled Predictive Control (DeePC) has demonstrated considerable effectiveness as a direct data-driven control algorithm. DeePC operates similarly to Model Predictive Control (MPC) by utilizing a receding-horizon optimization framework [1]. However, unlike MPC, which requires an explicit state-space model, DeePC constructs a Hankel matrix using past input/output measurements, thereby eliminating the need for explicit system identification. Although stability guarantees for DeePC have been established only under specific conditions, empirical evaluations have shown its effectiveness in handling nonlinear systems subject to disturbances. The robustness of DeePC can largely be attributed to the inclusion of regularization techniques [2], with its theoretical basis further strengthened by distributionally robust optimization [3]. However, the performance of DeePC is highly dependent on the choice of regularization parameters, making hyperparameter selection a critical challenge, particularly in applications where physical experiments are impractical, expensive, or hazardous.

Currently, two primary approaches are utilized for tuning regularization parameters in DeePC: analytical methods designed for open-loop scenarios and manual tuning through trial-and-error. Analytical methods [4], while useful, tend to be overly conservative as they do not fully account for DeePC's ability to dynamically replan within a closed-loop system. In contrast, manual tuning methods prioritize optimizing closed-loop performance but require extensive experimentation on the actual system, a process that may not be feasible in many practical applications. Given the limitations of open-loop tuning methods and the risks associated with manual tuning, an automated hyperparameter selection strategy is necessary to ensure optimal performance while maintaining reliability and safety.

To address this challenge, Data-enabled Predictive Control Hyperparameter Tuning via Differentiable Optimization (DeePC-Hunt) is introduced as a framework for automating the hyperparameter tuning process in DeePC. This approach conceptualizes DeePC as a control policy and considers its hyperparameters as tunable parameters within this framework. By incorporating an approximate model of the system dynamics, DeePC-Hunt utilizes backpropagation [5] to iteratively optimize the hyperparameters toward a locally optimal configuration. This methodology employs a constrained variation of the resilient backpropagation algorithm [6] alongside widely used automatic differentiation techniques, enabling direct optimization of DeePC's regularization parameters based on closed-loop performance metrics derived from the approximate system model.

Results indicate that the integration of DeePC with DeePC-Hunt enhances robustness against model uncertainties compared to conventional model-based control approaches. Furthermore, this improvement is achieved without requiring extensive manual hyperparameter tuning, making the proposed methodology particularly beneficial in scenarios where real-world experimental verification is challenging or costly. By facilitating automated hyperparameter selection, DeePC-Hunt presents a practical solution for improving the adaptability and reliability of direct data-driven control systems.

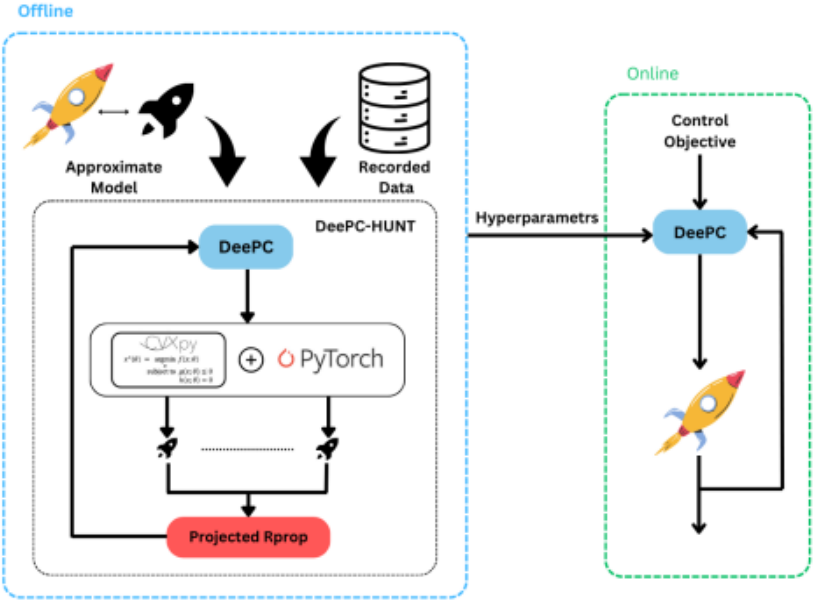



Fig. 1: DeePC-Hunt optimizes control using CvxpyLayers, PyTorch, and projected resilient backpropagation.

### 1.1 Related Work

DeePC-Hunt builds upon policy optimization algorithms that iteratively adjust control policies to minimize cumulative costs, typically using variants of projected gradient descent. Unlike traditional MPC or DeePC, DeePC-Hunt formulates policies as differentiable mappings from states to actions, enabling advanced optimization for improved adaptability in complex systems.

Central to DeePC-Hunt is the use of the implicit function theorem [7] to differentiate the solution map of Convex Optimization Control Policies via KKT conditions, facilitating gradient-based hyperparameter tuning. Prior works [8,9] similarly differentiate KKT conditions for MPC policies but rely on explicit

state-space models, limiting their scope. Broader differentiable optimization approaches [10] demonstrate effectiveness in control but depend heavily on model-based assumptions, risking performance degradation under model mismatch.

DeePC-Hunt addresses these limitations by combining real-world empirical data with synthetic data from approximate models, enhancing robustness and closed-loop performance despite model uncertainties.

### 1.2 Contributions

This work introduces:

- DeePC-Hunt, a novel framework for automated hyperparameter tuning in DeePC policies, reducing manual intervention.
- A tailored resilient backpropagation algorithm [6] with box constraints, leveraging automatic differentiation for stable and efficient optimization.
- Empirical validation on a challenging Vertical Takeoff and Landing (VTVL) vehicle landing task using a high-fidelity Gym simulation [11], demonstrating improved performance over model-mismatched MPC.

### 1.3 Paper Organization

Section II reviews differentiable convex optimization layers, DeePC, and the adapted resilient backpropagation method. Section III details DeePC-Hunt's theoretical framework and implementation. Section IV presents empirical results on the VTVL task, comparing with model-mismatched MPC. Section V concludes and outlines future research directions.

## 2 Background

Optimization is pivotal in machine learning and control for decision-making and model improvement. This section summarizes key methods underlying data-driven control: differentiable convex optimization layers, the DeePC algorithm, and projected resilient backpropagation (PRBP).

### 2.1 Differentiable Convex Optimization Layers

Differentiable convex optimization layers enable integration of convex problem-solving within neural networks while preserving end-to-end differentiability. Using implicit differentiation, gradients of the optimal solution $\mathbf{x}^*$ with respect to parameters $\theta$ are computed as [7]:

$$\frac{d\mathbf{x}^*}{d\theta} = -(\nabla^2_{\mathbf{xx}} f(\mathbf{x}^*))^{-1} \nabla^2_{\mathbf{x}\theta} f(\mathbf{x}^*). \tag{1}$$

These layers facilitate structured decision-making in reinforcement learning and model-based control with computational efficiency.

### 2.2 Data-Enabled Predictive Control (DeePC)

DeePC is a direct data-driven control approach that bypasses explicit system identification by leveraging historical input-output data to predict control actions. Its core problem is formulated as:

$$\min_{g} \|Y_g - Y_{ref}\|_2^2 + \lambda \|g\|_2^2, \tag{2}$$

where $Y_g$ is the predicted output, $Y_{ref}$ the reference, and $\lambda$ a regularization parameter controlling overfitting [2]. While DeePC demonstrates robustness to disturbances, its performance is sensitive to hyperparameter tuning, motivating automated optimization methods like DeePC-Hunt.

### 2.3 Projected Resilient Backpropagation (PRBP)

PRBP is an optimization algorithm designed for constrained problems that adjusts learning rates adaptively while projecting updates onto feasible sets:

$$w_{t+1} = P_{\mathcal{C}}(w_t - \eta_t \nabla f(w_t)), \tag{3}$$

with $P_{\mathcal{C}}$ enforcing constraints [6]. PRBP improves convergence stability and prevents oscillations, making it well-suited for tuning DeePC hyperparameters efficiently.

Together, these methods underpin DeePC-Hunt's automated hyperparameter tuning framework, enhancing robustness and scalability in data-driven control.

## 3 DeePC-Hunt

DeePC-Hunt automates tuning of DeePC's regularization parameter $\lambda$, critical for controller stability and robustness. By using a surrogate model to approximate system dynamics, it exploits the observation that optimal parameters generalize across systems with similar behaviors, enabling efficient hyperparameter optimization without altering the original model.

The surrogate model is mathematically represented as:

$$\tilde{x}_{k+1} = \tilde{f}(\tilde{x}_k, \tilde{u}_k), \tag{4}$$

$$\tilde{y}_k = \tilde{h}(\tilde{x}_k, \tilde{u}_k), \tag{5}$$

where $\tilde{u} \in \mathbb{R}^m$, $\tilde{x} \in \mathbb{R}^n$, and $\tilde{y} \in \mathbb{R}^p$. The functions $\tilde{f}$ and $\tilde{h}$ serve as approximations of the actual system's input-output behavior, often derived from simplified physical models or model reduction techniques [12].

To determine an optimal value of $\lambda$, the DeePC-Hunt algorithm minimizes an approximate closed-loop cost function:

$$\tilde{C}_{\pi_\lambda^{\mathrm{wd}}}(w_{\mathrm{ini},1}, w_{\mathrm{ref}}) = \sum_{i=1}^{N} \|\tilde{y}_i - y_{\mathrm{ref},i}\|_Q^2 + \|\tilde{u}_i - u_{\mathrm{ref},i}\|_R^2, \tag{6}$$

where $\tilde{u}_i$ is the control action determined by policy $\pi_\lambda^{\mathrm{wd}}$, and $\tilde{y}_i$ represents the corresponding output. The variable $w_{\mathrm{ini},i+1}$ updates iteratively by incorporating new control inputs and output data while discarding outdated measurements.

The optimal regularization parameter is identified by solving the constrained, non-convex optimization problem:

$$\min_{\lambda \in \Lambda} \mathbb{E}_{w_{\mathrm{ini},1} \sim D(W_p)} \left[ \tilde{C}_{\pi_\lambda^{\mathrm{wd}}}(w_{\mathrm{ini},1}, w_{\mathrm{ref}}) \right], \tag{7}$$

subject to the surrogate system constraints:

$$\tilde{x}_{i+1} = \tilde{f}(\tilde{x}_i, \tilde{u}_i), \tag{8}$$

$$\tilde{y}_i = \tilde{h}(\tilde{x}_i, \tilde{u}_i), \tag{9}$$

$$\tilde{u}_i = \pi_\lambda^{\mathrm{wd}}(w_{\mathrm{ini},i}, w_{\mathrm{ref}}). \tag{10}$$

Since $\pi_\lambda^{\mathrm{wd}}$ results from an optimization problem, this formulation is inherently a bi-level optimization problem. Differentiation of $\pi_\lambda^{\mathrm{wd}}$ is executed using CvxpyLayers [7]. To efficiently determine $\lambda^*$, projected resilient backpropagation techniques [13] are employed.

As $D(W_p)$ represents an empirical distribution of recorded trajectories, stochastic gradient descent (SGD) is utilized to approximate the solution across large datasets. The optimization is further refined using Monte Carlo sampling techniques, ensuring computational tractability. Algorithm 1 formally outlines the DeePC-Hunt procedure.

This structured framework ensures that DeePC-Hunt effectively refines $\lambda$ in a manner that enhances DeePC's closed-loop performance while maintaining computational efficiency.

## 4 Numerical Simulation: VTVL Vehicle Landing

### 4.1 Rocket Lander Gym Environment

The **DeePC-Hunt** framework is evaluated on controlling the descent and landing of a **Vertical Takeoff and Vertical Landing (VTVL)** rocket on an ocean platform. This task involves precise vertical touchdown while respecting nonlinear dynamics, disturbances, and control limits. Fig. 2 shows the rocket's free-body diagram and governing dynamics.

### 4.2 Dynamical Model

The rocket's motion is described by:

$$m\ddot{x} = F_s \cos(\theta) - F_E \sin(\phi + \theta) l_1, \tag{11}$$

$$m\ddot{y} = F_s \cos(\theta) - F_E \sin(\phi + \theta) l_1 - mg, \tag{12}$$

$$I\ddot{\theta} = F_E \sin(2\pi - \phi) l_1 - F_s l_2, \tag{13}$$

where $x, y$ are positions, $\theta$ is pitch angle, $F_s, F_E$ are side and main thrusts, $\phi$ is thrust direction, $m$ mass, $I$ moment of inertia, and $l_1, l_2$ are engine lever arms.

**Algorithm 1** DeePC-Hunt Algorithm

**Require:** Batch size $b$, time horizon $N$, policy $\pi^{\mathrm{wd}}$, reference trajectory $w_{\mathrm{ref}}$, training iterations $t_{\max}$, learning rate parameters $(\eta_{\max}, \eta_{\min}, \beta, \alpha)$, and approximate dynamics $(\tilde{f}, \tilde{h})$.

**Ensure:** Optimized regularization parameters $\lambda$

**Initialize:** Regularization parameters $\lambda_0$, learning rates $\eta_0$, and gradients $\gamma_0$.
**for** $k = 1$ to $t_{\max}$ **do**
  **for** $j = 1$ to $b$ **do** {Parallel execution}
    Sample initial condition $w_{\mathrm{ini},1,j} \sim D(W_p)$
    Compute initial state estimate $\tilde{x}_j^0 = \Psi(w_{\mathrm{ini},1,j})$
    **for** $i = 0$ to $N-1$ **do**
      Compute control input $\tilde{u}_j^i \leftarrow \pi_{\lambda_k}^{\mathrm{wd}}(w_{\mathrm{ini},i,j}, w_{\mathrm{ref}})$
      Compute system response $\tilde{y}_j^i \leftarrow \tilde{h}(\tilde{x}_j^i, \tilde{u}_j^i)$
      Update state estimate $\tilde{x}_j^{i+1} \leftarrow \tilde{f}(\tilde{x}_j^i, \tilde{u}_j^i)$
      Update historical data $w_{\mathrm{ini},i+1,j}$
    **end for**
  **end for**
  Compute batch cost:

$$J(\lambda_k) \leftarrow \frac{1}{b} \sum_{j=1}^{b} \tilde{C}_{\pi_\lambda^{\mathrm{wd}}}(w_{\mathrm{ini},j}, w_{\mathrm{ref}})$$

  **for** $i = 1$ to $r$ **do** {Parallel execution}
    Update gradient estimate $\gamma_{k+1}^i$
    Adjust learning rate using resilient backpropagation
  **end for**
  Update $\lambda_{k+1}$ using projected gradient descent
**end for**=0

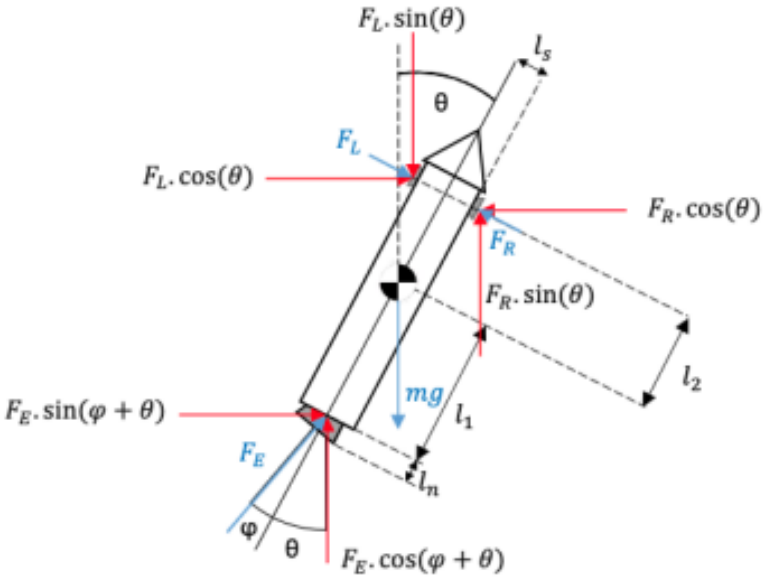



Fig. 2: Free-body diagram of the VTVL rocket and equations of motion (adapted from Ferrante, 2017).

### 4.3 Control Inputs and Constraints

Control input vector is $u = (F_E, F_s, \phi) \in \mathbb{R}^3$, with all states and inputs constrained within predefined bounds to ensure safety and stability:

$$x(t) \in X, \quad y(t) \in Y, \quad \theta(t) \in \Theta, \quad u(t) \in U.$$

### 4.4 Model Predictive Control (MPC)

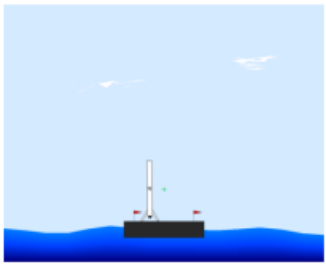
MPC is implemented by linearizing and discretizing the system, solving:

$$\min_{x,u} \sum_{i=1}^{T_f} \|x_i - r\|_Q^2 + \|u_i\|_R^2, \tag{14}$$

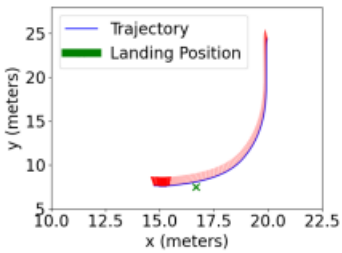
subject to system dynamics and state-input constraints. Two models are tested: Model A (accurate) and Model B (with parameter perturbations). Results are illustrated in Fig. 3.

### 4.5 DeePC-Hunt Policy

DeePC-Hunt trains policies using optimal regularization parameters $\lambda$ found via its hyperparameter optimization, respecting the same constraints. Key settings include:

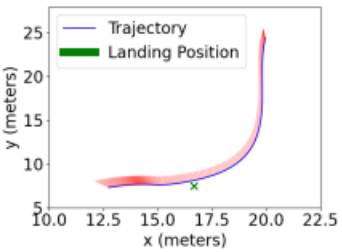


(a) Final position of the VTVL vehicle using the parameters $m = 530.41\text{kg}$, $l_1 = 2.85\text{m}$, $l_2 = 2.14\text{m}$




(b) Trajectory of VTVL vehicle using the correct parameters $m = 530.41\text{kg}$, $l_1 = 2.85\text{m}$, $l_2 = 2.14\text{m}$

(c) Final position of the VTVL vehicle using the incorrect parameters $m = 265.201\text{kg}$, $l_1 = 2.14\text{m}$, $l_2 = 2.85\text{m}$




(d) Trajectory of the VTVL vehicle using the incorrect parameters $m = 265.201\text{kg}$, $l_1 = 2.14\text{m}$, $l_2 = 2.85\text{m}$

Fig. 3: MPC policy performance under nominal and perturbed models.

- Prediction horizon $T_f = 10$, initialization horizon $T_{ini} = 1$,
- Initial regularization $\lambda_0 = (50, 50, 1000)$,
- Data collected with Pseudo-Random Binary Sequence (PRBS),
- Optimization hyperparameters: $t_{\max} = 100$, $r = 3$, $N = 20$, $B = 1$, learning rates $(\eta_{\max}, \eta_{\min}) = (10^2, 10^{-3})$, decay parameters $(\beta, \alpha) = (1.2, 0.5)$,
- Regularization search space $\Lambda = \{\lambda_i \in [10^{-5}, 10^5]\}$.

Table 1 compares DeePC and MPC outcomes, showing that DeePC-Hunt achieves higher landing success under model uncertainty, while MPC yields lower cost with accurate models.

## 5 Conclusion

This work introduces *DeePC-Hunt*, a novel framework that automates hyperparameter tuning for the Data-Enabled Predictive Control (DeePC) algorithm

Table 1: Comparison of DeePC and MPC Policies

| **Method** | **Cost** ($\times 10^3$) | **Success Rate (%)** |
|---|---|---|
| MPC (A) | 11.228 | 46 |
| DeePC (A) | 16.678 | 56 |
| MPC (B) | 8.213 | 22 |
| DeePC (B) | 12.165 | 66 |

using gradient-based optimization. By systematically refining regularization parameters offline, DeePC-Hunt eliminates manual tuning and enhances closed-loop control performance, even when system models are unknown or inaccurate.

Compared to traditional Model Predictive Control (MPC), DeePC-Hunt maintains robustness against model uncertainties and reduces reliance on precise system identification. Its use of a surrogate model enables efficient parameter estimation with minimal direct system interaction, making it suitable for real-time applications with computational constraints.

The framework's scalability and adaptability position it well for complex control domains such as aerospace, robotics, and industrial automation. Future work may explore extensions to multi-agent settings, adaptive tuning under dynamic conditions, and integration with reinforcement learning to further improve policy optimization and reduce computational demands.

In summary, DeePC-Hunt offers an automated, efficient, and scalable solution for DeePC hyperparameter optimization, enhancing data-driven control performance without extensive prior knowledge of system dynamics.

## References

1. F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2017.
2. J. Coulson, J. Lygeros, and F. Dörfler, "Regularized and distributionally robust data-enabled predictive control," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 2696–2701.
3. ——, "Distributionally robust chance constrained data-enabled predictive control," *IEEE Transactions on Automatic Control*, vol. 67, no. 7, pp. 3289–3304, 2022.
4. A. Chiuso, M. Fabris, V. Breschi, and S. Formentin, "Harnessing the final control error for optimal data-driven predictive control," 2023.
5. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
6. M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The rprop algorithm," in *IEEE International Conference on Neural Networks*, 1993, pp. 586–591.
7. B. Amos and J. Z. Kolter, "Optnet: Differentiable optimization as a layer in neural networks," 2021.
8. B. Amos, I. D. J. Rodriguez, J. Sacks, B. Boots, and J. Z. Kolter, "Differentiable mpc for end-to-end planning and control," 2019.
9. R. Zuliani, E. C. Balta, and J. Lygeros, "Bp-mpc: Optimizing the closed-loop performance of mpc using backpropagation," 2024.

10. A. Agrawal, S. Barratt, S. Boyd, and B. Stellato, "Learning convex optimization control policies," in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, vol. 120. PMLR, Jun 2020, pp. 361–373.
11. G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016, https://arxiv.org/abs/1606.01540.
12. A. C. Antoulas, *Approximation of large-scale dynamical systems*. SIAM, 2005.
13. E. K. Ryu and W. Yin, *Large-Scale Convex Optimization: Algorithms Analyses via Monotone Operators*. Cambridge University Press, 2022.