A Dynamic Penalization Framework for Online rank-1 Semidefinite Programming Relaxations

Ahmad Al-Tawaha ATAWAHA@VT.EDU
Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061, USA

Javad Lavaei LAVAEI@BERKELEY.EDU
Industrial Engineering and Operations Research, University of California, Berkeley, CA 94720, USA

Ming Jin JINMING@VT.EDU
Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061, USA

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

We propose a unified framework for solving sequences of semidefinite programs (SDPs) with rank-one constraints—critical in domains such as combinatorial optimization and power systems. Enforcing rank-1 feasibility in SDP relaxations is especially challenging in dynamic environments where problem parameters evolve across time. To address this, our method operates on two complementary levels. At the per-task level, we introduce a two-phase optimization scheme: the decision phase solves a penalized SDP using a task-specific penalty matrix to approximate a rank-1 solution, while the learning phase updates this penalty matrix via subgradient-based feedback to progressively enforce rank-1 feasibility. At the meta level, we introduce a meta-learning framework that accelerates optimization across tasks by predicting effective initializations for the penalty matrix in each new task. The meta-model leverages historical solutions to produce informed penalty initializations, reducing the number of inner iterations needed per task. We provide theoretical guarantees showing that the task-averaged regret decreases with the number of tasks, with rates that improve under higher task similarity. Empirical results in real-world rank-constrained applications, including the Max-Cut problem and Optimal Power Flow (OPF), demonstrate that our method consistently recovers rank-1 solutions.

Keywords: Online Optimization, Rank-Constrained Semidefinite Programming, Dynamic Penalization, Meta-Learning.

1. Introduction

Semidefinite programming has emerged as a powerful framework for addressing nonlinear and nonconvex optimization problems, particularly in polynomial optimization and quadratically constrained quadratic programs (QCQPs). A wide range of optimization problems, including discrete optimization problems, can be reformulated or approximated as polynomial optimization problems, which are further transformable into QCQPs using auxiliary variables and constraints (Madani et al., 2020; Wang, 2022). By relaxing the nonconvex QCQP constraints into semidefinite problem, SDPs enable exact solutions or tight lower bounds on the objective (Boyd et al., 1994; Boyd and Vandenberghe, 2004). However, enforcing rank-1 constraints in these relaxations remains challenging. Convex SDP relaxations yield low-rank solutions efficiently, but achieving exact rank-1 solutions often requires penalty terms that drive nonzero eigenvalues to zero, promoting rank-1 feasibility (Pataki, 1998; Sojoudi and Lavaei, 2014).

This challenge is further compounded in sequential optimization settings, where evolving problem parameters require solving a series of similar instances, as seen in power systems (Dall'Anese et al., 2017), communication networks (Chen and Lau, 2011), and online learning (Mokhtari et al., 2016). Solving each instance independently is computationally prohibitive, even with modern advances (Zavala and Anitescu, 2010). Traditional SDP-based methods with pre-tuned penalty parameters for static problems (Zohrizadeh et al., 2018; Madani et al., 2015; Liu et al., 2017) fail in online tasks, where re-optimizing penalty terms for each new instance adds significant overhead.

A promising direction to mitigate these computational challenges further lies in meta-learning techniques (Hospedales et al., 2021), which accelerate optimization by leveraging knowledge from previously solved similar problems. This approach could enable faster convergence through intelligent initialization of key parameters, such as the penalty term parameter in penalized SDPs. However, applying meta-learning to sequential rank-1 constrained SDPs requires addressing unique theoretical and practical challenges, particularly in guaranteeing rank-1 recovery while maintaining computational efficiency.

The fundamental challenge in sequential relaxed SDPs is achieving computationally efficient rank-1 recovery in dynamic environments. Determining penalty terms that consistently enforce rank-1 solutions is inherently complex (Mezura-Montes and Coello, 2011). Although static (Hsieh et al., 2015), dynamic (Liu et al., 2016), and adaptive penalty methods (Wang et al., 2021; Krohling and dos Santos Coelho, 2006; Fan and Yan, 2012; Huang et al., 2007; Wang et al., 2023) exist for constrained optimization, most rely on stochastic adjustments and are designed for offline scenarios. Moreover, traditional SDP relaxations use pre-tuned penalty terms for static problems, which become infeasible in evolving settings, as they require re-optimization for every new instance. These inefficiencies highlight the need for adaptive and automated frameworks that dynamically adjust penalty parameters within single and across sequential tasks.

Two key challenges arise in solving sequences of SDPs: (i) dynamically enforcing rank-1 structure across evolving tasks, and (ii) leveraging similarities between tasks to reduce per-task optimization cost. We address these challenges by proposing a unified framework that integrates dynamic penalization with meta-learning for SDP optimization. Our main contributions are summarized below:

- **Dynamic Penalization for rank-1 Recovery:** We develop a dynamic penalization method that updates a penalty matrix $W_{t,k}$ at each iteration k of task t. These updates are computed by differentiating through the relaxed SDP, allowing the optimization to adaptively guide the solution toward rank-1 feasibility. The framework leverages established convergence guarantees, demonstrating a sublinear convergence rate.

- **Meta-Initialization for Multi-Task SDP Optimization:** To accelerate convergence, we introduce a meta-learning strategy that predicts the initialization $W_{t,1}$ for each task. This reduces the number of required iterations K_t, particularly when task instances exhibit structural similarity.

- **Task-Averaged Regret Bound:** We provide a theoretical bound on the task-averaged regret $\bar{R}(K,T)$, which measures the average optimization effort per task (Def. 8). Specifically, we show that: $\bar{R}(K,T) \leq \mathcal{O}\left(\frac{1}{\sqrt{K}}\left(V_T^{1/3}T^{-1/3} + V_{\text{similar}}\right)\right)$, where V_T is the path length over tasks and V_{similar} quantifies the maximum gap between the task-specific solution and its best possible initialization. This bound improves over single-task optimization when the number of tasks T is large or when V_T and V_{similar} are small (the tasks are sufficiently similar).

The organization of the paper is as follows: Sec. 2 introduces the problem formulation, Secs. 3 and 4 describe our method for single-task and multi-task scenarios, respectively, Sec. 5 presents the experimental evaluations, and Sec. 6 concludes the paper. Due to space limitations, all proofs and additional experimental details are provided in the online supplement (Al-Tawaha et al., 2025).

2. Problem Formulation

We consider a sequence of optimization tasks $\{\mathcal{T}_t\}_{t=1}^{T}$, where t indexes each task in the sequence. Each polynomial optimization task involves minimizing a polynomial objective subject to polynomial constraints. Since general polynomial optimization is NP-hard, we adopt a standard approach that approximates it as a QCQP through variable lifting and substitution. Accordingly, we represent each task t as the following QCQP:

$$\underset{x_t \in \mathbb{R}^n}{\text{minimize}}\; x_t^\top M_{0,t} x_t \quad \text{subject to} \quad x_t^\top M_{i,t} x_t \leq a_{i,t},\; i = 1, \ldots, p, \quad x_t^\top N_{j,t} x_t = b_{j,t},\; j = 1, \ldots, q, \tag{1}$$

where $M_{0,t}, M_{i,t}, N_{j,t} \in \mathbb{S}^n$ representing the coefficients of the quadratic objective and constraint functions for task t, and $a_{i,t}, b_{j,t} \in \mathbb{R}$ representing the constraint thresholds. To tackle the non-convex QCQP (1), we introduce an auxiliary variable $X_t = x_t x_t^\top \in \mathbb{S}^n$. This reformulation lifts the original vector variable x_t into a rank-1 matrix, resulting in the **rank-constrained SDP**:

$$\begin{aligned}
&\underset{X_t \in \mathbb{S}^n}{\text{minimize}} && f_{0,t}(X_t) && \text{(2a)}\\
&\text{subject to} && f_{i,t}(X_t) \leq a_{i,t}, \quad i = 1, \ldots, p, && \text{(2b)}\\
& && h_{j,t}(X_t) = b_{j,t}, \quad j = 1, \ldots, q, && \text{(2c)}\\
& && X_t \succeq 0, && \text{(2d)}\\
& && \operatorname{rank}(X_t) = 1, && \text{(2e)}
\end{aligned}$$

where $f_{i,t}, h_{j,t} : \mathbb{S}^n \to \mathbb{R}$ are twice continuously differentiable convex functions for $i = 0, \ldots, p$ and $j = 1, \ldots, q$. While this formulation captures the original non-convex structure, solving it directly is computationally intractable due to the non-convex rank-one constraint in (2e). To make the problem tractable, a common approach is to relax the rank constraint and solve the resulting **standard SDP relaxation**, which retains all constraints except (2e). However, this can result in high-rank solutions that are infeasible or suboptimal for the original problem. To mitigate this, we augment the standard formulation by introducing a rank-promoting penalty term. The result is a **penalized SDP relaxation**, where the objective includes a strongly convex penalty function $g\left(X_t; W_t\right)$, with $W_t \in \mathbb{S}^n$ denoting the penalty matrix to be learned. This term encourages low-rank structure while preserving the convexity of the problem. The penalized SDP for task t is formulated as:

$$\underset{X_t \in \mathbb{S}^n}{\text{minimize}}\; f_{0,t}(X_t) + g(X_t; W_t) \quad \text{subject to} \quad \begin{aligned} &f_{i,t}(X_t) \leq a_{i,t},\; i = 1, \ldots, p,\\ &h_{j,t}(X_t) = b_{j,t},\; j = 1, \ldots, q,\\ &X_t \succeq 0. \end{aligned} \tag{3}$$

The penalty function $g : \mathbb{S}^n \times \mathbb{S}^n \to \mathbb{R}$ is defined to guide X_t toward being approximately $\operatorname{rank}-1$ while maintaining the convexity of the problem. For large-scale instances, imposing full positive semidefinite constraints on the matrix X_t is computationally prohibitive. To reduce computational complexity, we follow the approach in Madani et al. (2015) and replace the global PSD constraint $X_t \succeq 0$ with localized PSD constraints on a collection of principal submatrices of X_t. Specifically, we define a set of index subsets $\Omega_t = \{\mathcal{G}_{s,t}\}_{s=1}^{S_t}$, where each $\mathcal{G}_{s,t} \subseteq \{1, \ldots, n\}$ identifies a subset of variables. Instead of enforcing global semidefiniteness, we require that each corresponding submatrix $X_t[\mathcal{G}_{s,t}, \mathcal{G}_{s,t}]$ is positive semidefinite, i.e., $X_t[\mathcal{G}_{s,t}, \mathcal{G}_{s,t}] \succeq 0$ for all s. This leads to a **reduced SDP formulation**, where the PSD constraint is applied only to these structured subsets. Similarly, we impose the rank-one condition locally, requiring that each submatrix satisfies $\operatorname{rank}(X_t[\mathcal{G}_{s,t}, \mathcal{G}_{s,t}]) = 1$. These local conditions collectively promote a globally low-rank structure in X_t, while enabling more efficient optimization.

The quality of the solution can be characterized through a hierarchy of relaxations. Let f^*be the optimal value of the original rank-constrained problem in Eq. (2). Let f^*_{SDP} and $f^*_{\text{r}-\text{SDP}}$ represent the optimal values of the standard and reduced SDP relaxations, respectively, when the rank constraint is relaxed and no penalization is used. Furthermore, let $f^*_{\text{pen, SDP}}$ and $f^*_{\text{pen, r-SDP}}$ denote the optimal values of the penalized SDP formulations in Eq. (3), for the full and reduced cases, respectively. Then, following the analysis in Sojoudi and Lavaei (2014); Madani et al. (2015), we have: $f^*_{\text{r}-\text{SDP}} \le f^*_{\text{SDP}} \le f^* \le f^*_{\text{pen,SDP}} \le f^*_{\text{pen,r}-\text{SDP}}$. This chain of inequalities allows us to quantify proximity to global optimality. Based on this, we define a **global optimality guarantee:**

$$G_{\text{opt}} = \frac{f^*_{\text{SDP}}}{f^*_{\text{pen,SDP}}}. \tag{4}$$

3. Dynamic Penalization for Single-Task rank-1 Enforcement in SDP

For each task t in the sequence $\{\mathcal{T}_t\}_{t=1}^{T}$, our framework operates in two interconnected phases: a **learning phase** and a **decision phase**. In the learning phase, the goal is to refine the penalty parameters W_t to enforce rank-1 in the solution X_t. This is achieved by minimizing a loss function $\mathcal{L}(X_t, W_t)$ that penalizes deviations from the rank-1. The optimization problem for updating W_t is formulated as:

$$\underset{W_t \in \mathbb{S}^n}{\text{minimize}}\; G(W_t) = \mathcal{L}(X_t^*(W_t), W_t), \tag{5}$$

where X_t^* is the optimal solution function obtained in the decision phase for a given penalty matrix W_t. In the decision phase, we solve the following penalized semidefinite program to obtain $X_t^*(W_t)$:

$$X_t^*(W_t) = \underset{\bar{X}_t \in \mathbb{S}^n}{\arg\min}\; f_{0,t}(\bar{X}_t) + g(\bar{X}_t; W_t) \quad \text{subject to} \quad \begin{aligned} f_{i,t}(\bar{X}_t) &\le a_{i,t}, \quad i = 1, \ldots, p, \\ h_{j,t}(\bar{X}_t) &= b_{j,t}, \quad j = 1, \ldots, q, \\ \bar{X}_t &\succeq 0. \end{aligned} \tag{6}$$

We assume that $X_t^*(W_t)$ exists and satisfies the Linear Independence Constraint Qualification (LICQ) at the optimal point. For instance, this condition holds in optimal power flow problem (Hauswirth et al., 2018), which is the focus of our experimental validation.

At iteration k of task t, we denote the current penalty as $W_{t,k}$ and the descent direction as $g_{t,k}$. If $G(W_t)$ is differentiable at $W_{t,k}$, then $g_{t,k} = \nabla G(W_{t,k})$. Otherwise, we follow the strategy in

Xu and Zhu (2023), where $g_{t,k}$ is chosen as the vector with the smallest norm in the convex hull of the approximate generalized gradient set $\partial G\left(W_{t,k}, \varepsilon_k\right)$, with ε_k a tolerance parameter. This ensures well-defined descent directions even when G is non-differentiable.

We update $W_{t,k}$ using a line-search procedure that ensures sufficient descent at each step. Assuming the objective $G\left(W_t\right)$ is lower-bounded by its value at a local stationary point W_t^*, the following sublinear convergence guarantee can be established based on the analysis in Xu and Zhu (2023):

$$\frac{1}{K}\sum_{k=1}^{K}\|g_{t,k}\| \leq \sqrt{\frac{G\left(W_{t,1}\right)-G\left(W_t^*\right)}{\beta\sigma_{\min}K}} \tag{7}$$

where $\beta \in (0,1)$ is the line-search parameter and $\sigma_{\min} > 0$ is a lower bound on the step size. This result implies a sublinear convergence rate of $\mathcal{O}(K^{-1/2})$ for the average gradient norm highlights how the iterative procedure steadily refines $W_{t,k}$, guiding $X_t^*\left(W_t\right)$ toward a near rank-1 solution in practice. For details of the line-search analysis, see Xu and Zhu (2023).

4. Meta-Learning for Multi-Task rank-1 Enforcement in SDP

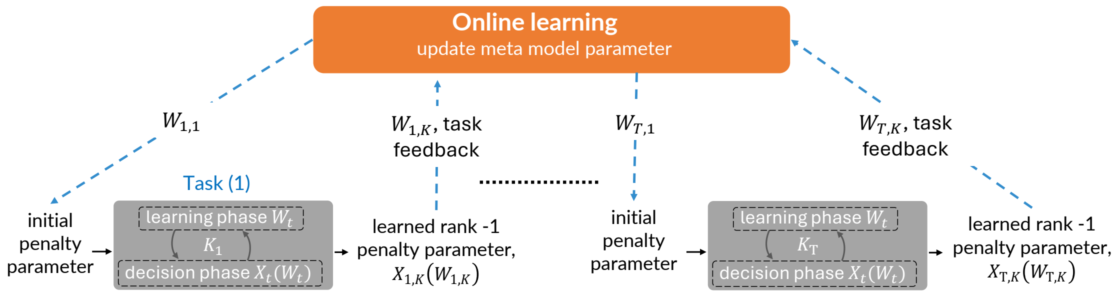
Online learning
update meta model parameter
$W_{1,1}$
$W_{1,K}$, task feedback
$W_{T,1}$
$W_{T,K}$, task feedback
Task (1)
initial penalty parameter
learning phase W_t
K_1
decision phase $X_t(W_t)$
learned rank -1 penalty parameter, $X_{1,K}(W_{1,K})$
initial penalty parameter
learning phase W_t
K_T
decision phase $X_t(W_t)$
learned rank -1 penalty parameter, $X_{T,K}(W_{T,K})$

Figure 1: Meta-learning framework for multi-task rank-one enforcement in SDP. The meta-model predicts an effective initialization $W_{t,1}$, accelerating convergence and progressively enforcing the rank-one constraint.

Solving semidefinite programs with rank-one constraints is computationally demanding—particularly in multi-task settings, where each task may require many iterations to converge. To reduce this overhead, we propose a meta-learning framework that leverages structural similarities across tasks to accelerate optimization.

Our approach uses past optimization history to predict effective initializations $W_{t,1}$ for each new task t, reducing the number of iterations needed in the learning and decision phases. By exploiting shared task structure, the meta-model improves both convergence speed and solution quality, particularly when tasks exhibit statistical or structural similarity. To formalize the efficiency of the meta-initialization strategy, we define the task-averaged stationarity regret, which quantifies the average first-order optimality gap across all tasks:

Definition 1 *For each task $t \in \{1,\ldots,T\}$, suppose a within-task algorithm produces iterates $\{W_{t,k}\}_{k=1}^{K}$. Denote by $g_{t,k}$ the (sub)gradient or stationarity measure for task t at iteration k. We* *define the task-averaged regret* $\bar{R}(K,T)$ *as:*

$$\bar{R}(K,T) = \frac{1}{T}\sum_{t=1}^{T}\left[\frac{1}{K}\sum_{k=1}^{K}\|g_{t,k}\|\right]. \tag{8}$$

This metric captures how well the meta-model improves initialization quality across tasks. We now provide a regret bound that demonstrates how the average optimization effort decreases with task count.

From the single-task convergence result in (7), we observe that the task-averaged regret can be upper bounded by terms involving the initial penalty matrix $W_{t,1}$. This insight motivates our key idea: to design a meta-algorithm that sequentially updates $W_{t,1}$ by performing online learning on a surrogate loss. Specifically, we interpret the right-hand side of (7) as a task-specific meta-loss:

$$\ell_t(W_{t,1}) = \sqrt{\frac{G\left(W_{t,1}\right) - G(W_t^*)}{\beta\sigma_{\min}K}}.$$

This formulation directly links the task-averaged stationarity regret in (8) to the sequence of meta-losses $\ell_t\left(W_{t,1}\right)$. By aggregating these bounds across all tasks and normalizing by T, we obtain:

$$\bar{R}(K,T) \leq \frac{1}{T}\sum_{t=1}^{T}\ell_t\left(W_{t,1}\right). \tag{9}$$

This bound follows directly by substituting the single-task convergence guarantee (7) into the definition of task-averaged regret in (8). Note that, we have transformed the original problem of bounding the task average regret into a relaxed problem of bounding the right-hand side of (9), which can be formulated as a standard online learning problem. In particular, we can treat ℓ_t as a loss function, which is revealed after the completion of each task. We update the initialization $W_{t,1}$ using an online learning algorithm, yielding $W_{t+1,1}$ for the next task. As illustrated in Figure 1, this framework consists of two nested phases: the inner loop optimizes W_t to minimize rank-1 deviation within each task, while the outer loop updates the initialization using the meta-loss feedback across tasks. We apply Online Gradient Descent (OGD) for convex meta-losses (Besbes et al., 2015; Al-Tawaha and Jin, 2024) and Follow-the-Perturbed-Leader (FTPL-D) for non-convex cases (Xu and Zhang, 2024). This iterative procedure enables effective transfer of knowledge across tasks, reducing computational cost while maintaining high solution quality. The complete meta-learning procedure is formalized in Algorithm 1

Algorithm 1 Meta-Learning Framework for Initialization

```
Require: Number of problem instances $T$, initial penalty matrix $W_{t,1}$
1: for $t = 1$ to $T$ do
2:     Receive task $t$.
3:     Solve problem (5) starting from $W_{t,1}$ using the method from Xu and Zhu (2023), and obtain
       $W_{t,K}, X_{t,K}$ after $K$ iterations.
4:     Compute the empirical meta-loss: $\hat{\ell}_t\left(W_{t,1}\right) = \sqrt{\frac{G(W_{t,1})-G(W_{t,K})}{\beta\sigma_{\min}K}}$.
5:     Update the meta parameters $W_{t+1,1}$ using an online learning algorithm to minimize the meta loss.
6: end for
```

Note that, in practice, we do not have direct access to the local stationary point W_t^* when evaluating task specific loss. Instead, we rely on an empirical task loss $\hat{\ell}\left(W_{t,1}\right)$, which substitutes

W_t^* with the final iterate $W_{t,K}$ obtained after K learning steps. This empirical loss provides a practical alternative to measure optimization progress and serves as a surrogate objective in our meta-learning framework.

Our meta-learning framework reduces per-task optimization cost by exploiting shared structure across tasks and adapting initialization strategies online. To evaluate its adaptability to changing environments, we analyze performance using dynamic regret. This metric compares the loss incurred by our online meta-initializations $\{W_{t,1}\}_{t=1}^T$ to that of the optimal sequence $\{W_{t,1}^*\}_{t=1}^T$ selected in hindsight. Bounding this quantity characterizes how well the meta-model adapts to evolving tasks and establishes a link to task-averaged regret. The following lemma provides a formal guarantee.

Lemma 2 *Consider a sequence of T optimization tasks. For both convex and non-convex meta-loss functions, when $W_{t,1}$ is updated using OGD for the convex case (Besbes et al., 2015), or FTPL-D variant for the non-convex case (Xu and Zhang, 2024), the task-averaged regret is bounded as:*

$$\bar{R}(K,T) \leq \mathcal{O}\left(\frac{1}{\sqrt{K}}\left(V_T^{1/3}T^{-1/3} + V_{\text{similar}}\right)\right), \tag{10}$$

where $V_T = \sum_{t=2}^T \left\| W_{t,1}^* - W_{t-1,1}^* \right\|$ is the path length capturing changes in the optimal meta-initializations, and $V_{\text{similar}} := \max_{t=1,\dots,T} \sqrt{\frac{G(W_{t,1}^*)-G(W_t^*)}{\beta\,\sigma_{\min}}}$ is a similarity term reflecting the alignment between the best initialization $W_{t,1}^*$ and the task-specific solution W_t^*. Here, $\beta \in (0,1)$ is the line search parameter and $\sigma_{\min} > 0$ is a lower bound on the step size.

Our result shows that the average optimization effort per task decreases as the number of tasks increases, provided that tasks evolve smoothly over time. This is captured by the path length V_T, which quantifies how much the optimal initialization changes between tasks. When V_T grows sublinearly with T, the meta-model can effectively transfer knowledge across tasks, leading to improved initialization and reduced within-task iterations. However, if the environment changes arbitrarily, the optimal initialization can shift significantly, making sublinear task-averaged regret unachievable without further assumptions.
The second key factor is task similarity, captured by the term V_{similar}, this term captures how close the best possible initialization $W_{t,1}^*$ is to the final task-specific solution W_t^*. When tasks are similar, this gap in objective values remains small across all tasks. Hence, the meta-model can predict initializations that start closer to the optimum, reducing the number of gradient steps needed to reach stationarity. The smaller this term, the less adaptation is needed within each task.

5. Experiments

5.1. Dynamic Penalization for Single-Task SDP: A Max-Cut Case Study

The Max-Cut problem seeks to partition the vertices of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ into two sets such that the sum of the weights of edges between the sets is maximized. The standard SDP relaxation of this problem replaces the binary constraints $x_i \in \{-1, 1\}$ with a semidefinite constraint on the Gram matrix X :

$$\begin{array}{ll} \max_{X\in\mathbb{S}^n} & \frac{1}{4}\operatorname{Tr}(LX) \\ \text{s.t.} & X \succeq 0, \quad X_{ii} = 1, \forall i \in \mathcal{V}, \end{array} \tag{11}$$

where L is the Laplacian matrix of the graph $\mathcal{G}$, defined as $L = D - C$, with D being the degree matrix and C the adjacency matrix containing the edge weights c_{ij}.

Interestingly, the Max-Cut SDP formulation demonstrates differentiable behavior in the mapping $X(W)$, enabling the use of `cvxpylayer` to compute gradients. Our framework iteratively refines penalty parameters W to enforce rank-1 feasibility. Each iteration consists of a learning phase, which updates W to minimize:

$$G(W) = \lambda_1\big(n^2 - \|X^*(W)\|_F^2\big) + \lambda_2 \operatorname{Tr}\big(L X^*(W)\big).$$

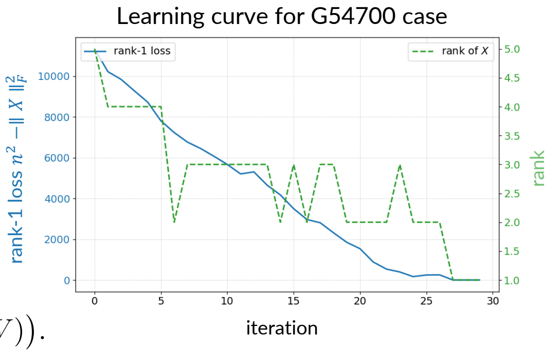

Figure 2: Learning curve for G54700 illustrating the effective rank of the solution across iterations.

The weights $\lambda_1, \lambda_2 \geq 0$ control the trade-off between rank-1 enforcement and cut quality, with $\lambda_1 \gg \lambda_2$ to emphasize the Frobenius penalty. The decision phase then solves a penalized SDP to compute X:

$$X^*(W) \in \arg\max_{\bar{X} \in \mathbb{S}^n} \ \frac{1}{4}\operatorname{Tr}(L\bar{X}) - \big(\operatorname{Tr}(W\bar{X}) + \mu\|\bar{X}\|_F^2\big), \text{ s.t. } \quad \bar{X} \succeq 0, \quad \bar{X}_{ii} = 1, \ \forall i \in \mathcal{V}.$$

The process is repeated for K iterations, with feedback from the decision phase guiding updates in the learning phase. We evaluate our method on 10 small graphs from Set 2 of the Optsicom Project benchmark[1], where each graph contains $n = 125$ nodes and $m = 375$ edges. As shown in Table 1, our approach consistently achieves near-optimal or optimal solutions compared to existing methods. Using the G54700 graph as an example, Figure 2 illustrates how the rank-1 loss $n^2 - \|X\|_F^2$ and the effective rank improve over successive iterations. Unlike traditional SDP methods, our approach produces rank-1 solutions without randomization—a key advantage for differentiable optimization, as it enables end-to-end gradient flow without stochastic rounding, making it well-suited for deep learning pipelines.

Table 1: Max-Cut results on Optsicom graphs. We compare our dynamic penalized SDP with RUN-CSP Toenshoff et al. (2021), Khalil et al. Khalil et al. (2017), and a Greedy baseline.

Graphs	Dyn. Penalized SDP	RUN-CSP	Khalil et al.	Greedy Search	Opt
G54100	**110**	**110**	108	80	110
G54200	108	**112**	108	90	112
G54300	**106**	**106**	104	86	106
G54400	**112**	**112**	108	96	114
G54500	110	**112**	**112**	94	112
G54600	**110**	**110**	**110**	88	110
G54700	**112**	110	108	88	112
G54800	**108**	106	**108**	76	108
G54900	**108**	**108**	**108**	88	110
G541000	**110**	**110**	108	80	112

5.2. Dynamic Penalization for Multi-Task SDP: Optimal Power Flow Case Study

As a practical case study, we apply the proposed approach to the OPF problem for multi-task online setting, where each task t corresponds to a unique operational condition. The goal is to minimize

1. http://grafo.etsii.urjc.es/optsicom/maxcut/

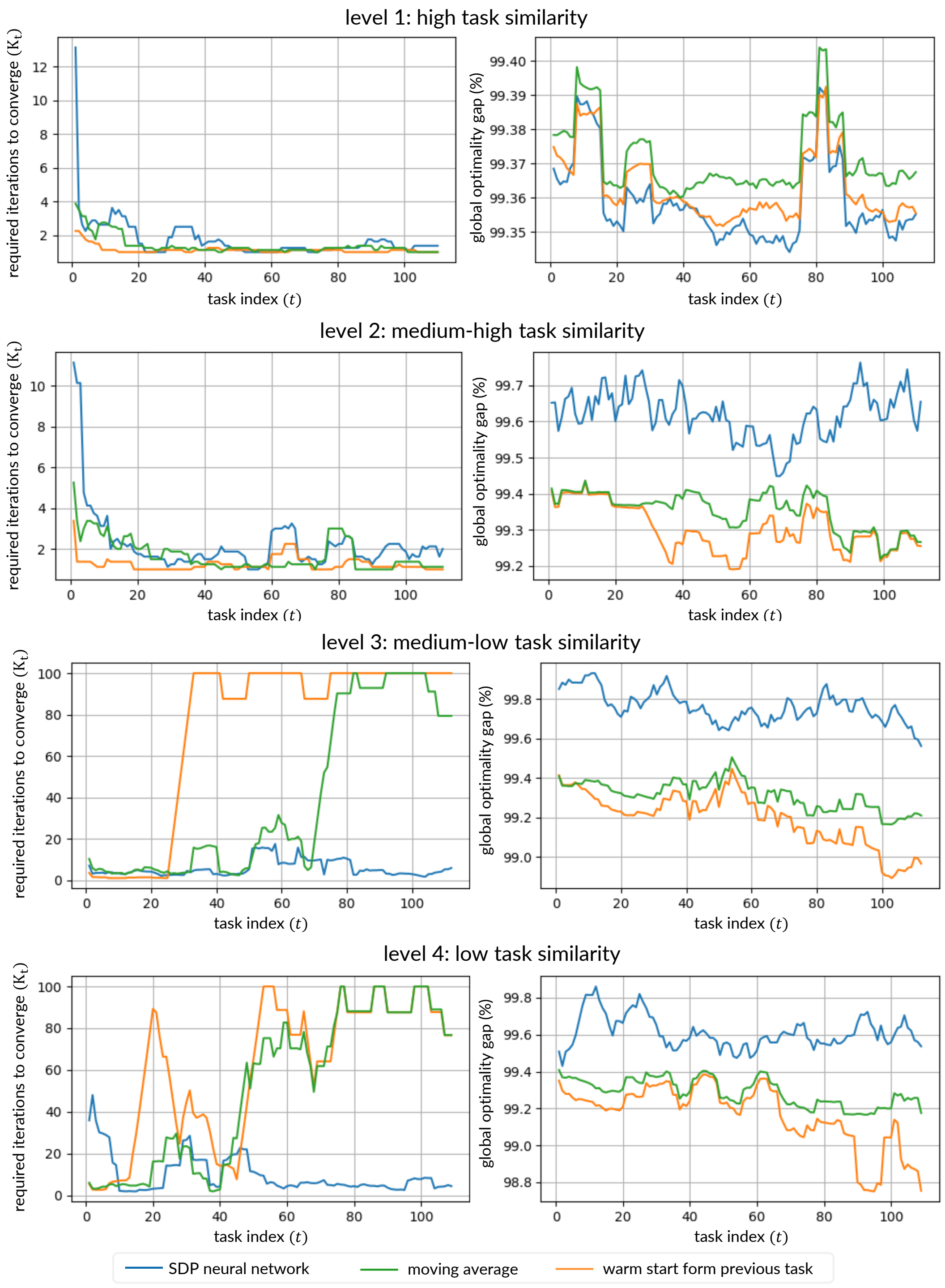

Figure 3: Comparison of initialization strategies across task similarity levels (level 1: high similarity to level 4: low similarity). Left: iterations to rank−1 convergence. Right: global optimality gap. Strategies include SDP neural network, moving average, and warm start from the previous task.

generation costs while satisfying operational constraints such as power balance, voltage limits, and line flow capacities. To improve scalability, we use the reduced SDP relaxation, where rank-1 feasibility is achieved when all submatrices corresponding to selected subgraphs have rank-1. For each task t, the objective of the penalized reduced SDP formulation as:

$$\min_{X_t \in \mathbb{S}^n} \sum_{i \in \mathcal{N}_g} c_{i,t}\left(P_{i,t}\right) + \operatorname{Tr}\left(W_t X_t\right),$$

where $\mathcal{N}_g$ is the set of generator buses, $P_{i,t}$ denotes the active power output of generator i in task t, and $c_{i,t}$ is the associated generation cost. The penalty term $\operatorname{Tr}\left(W_t X_t\right)$ enforces rank-1 solutions. The loss function used in the learning phase is:

$$G\left(W_t\right) = \left\|X_t^*\left(W_t\right)\right\|_* - \left\|X_t^*\left(W_t\right)\right\|_2,$$

where $\|X\|_* = \sum_i \sigma_i(X)$ the nuclear norm (sum of singular values) and $\|X\|_2 = \sigma_{\max}(X)$ the spectral norm. The expression vanishes when X_t^* is rank-1, thereby measuring deviation from the desired rank-1 feasibility. We incorporate a meta-model $\mathcal{M}_{\theta_t}$, designed to predict an effective initialization $W_{t,1}$ for each task t. Parameterized as an SDP neural network, the meta-model uses task-specific features and leverages prior optimization history. It is updated online after each task using the observed loss, allowing the model parameters θ_t to adapt and improve across tasks. Additional implementation details are provided in Al-Tawaha et al. (2025).

Figure (3) illustrates the performance of our dynamic penalization framework across a sequence of OPF tasks with varying similarity levels. These levels (Levels 1–4) are defined by sampling operational conditions with progressively increasing standard deviations—higher levels indicate greater variation between tasks. The left-hand plots report the number of iterations required to reach rank-1 feasibility.

For highly similar tasks (Level 1), the meta-model performs on par with baseline strategies such as warm-start and moving average, as expected. However, as similarity decreases, the meta-model demonstrates clear advantages. Baseline methods frequently fail to converge within the allowed iteration budget, while the meta-model consistently achieves rank-1 solutions with fewer iterations. This reflects its ability to generalize and adapt across tasks, reinforcing our theoretical guarantees.

The right-hand plots show the global optimality gap, as defined in (4). Across all levels of task similarity, our framework consistently achieves an optimality of between 99.3% and 100%, demonstrating its ability to maintain high-quality solutions that closely approximate the global optimum. Notably, even for less similar tasks—where baseline methods diverge or fail to converge—the meta-model reliably preserves both feasibility and optimality.

6. Conclusion

We proposed a meta-learning framework for solving rank-one constrained SDPs arising in dynamic optimization tasks. Our method combines a dynamic penalization strategy that iteratively refines the penalty matrix W_t within each task, with data-driven meta-initialization scheme that predicts effective starting points $W_{t,1}$ using task-specific features and past experience. To quantify learning efficiency, we introduced a task-averaged regret metric and proved a theoretical bound: $\bar{R}(K,T) \leq \mathcal{O}\left(\frac{1}{\sqrt{K}}\left(V_T^{1/3} T^{-1/3} + V_{\text{similar}}\right)\right)$. This demonstrates the framework's ability to transfer knowledge and reduce per-task optimization effort in dynamic settings.

Acknowledgments

A. Al-Tawaha and M. Jin are supported in part by the National Science Foundation under grants ECCS-233177 and IIS-2312794, the Amazon-Virginia Tech Initiative for Efficient and Robust Machine Learning, and the Commonwealth Cyber Initiative
Javad Lavaei was supported by the U. S. Army Research Laboratory and the U. S. Army Research Office under Grant W911NF2010219, Office of Naval Research under Grant N000142412673, AFOSR, and NSF.

References

Ahmad Al-Tawaha and Ming Jin. Does online gradient descent (and variants) still work with biased gradient and variance? In *2024 American Control Conference (ACC)*, pages 3570–3575. IEEE, 2024.

Ahmad Al-Tawaha, Javad Lavaei, and Ming Jin. A dynamic penalization framework for online rank-1 semidefinite programming relaxations-online document. 2025. URL https://ahmad-tawaha.webador.com/publication.

Omar Besbes, Yonatan Gur, and Assaf Zeevi. Non-stationary stochastic optimization. *Operations research*, 63(5):1227–1244, 2015.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear matrix inequalities in system and control theory*. SIAM, 1994.

Junting Chen and Vincent KN Lau. Convergence analysis of saddle point problems in time varying wireless systems—control theoretical approach. *IEEE Transactions on Signal Processing*, 60(1): 443–452, 2011.

Emiliano Dall'Anese, Swaroop S Guggilam, Andrea Simonetto, Yu Christine Chen, and Sairaj V Dhople. Optimal regulation of virtual power plants. *IEEE Transactions on Power Systems*, 33 (2):1868–1881, 2017.

Qinqin Fan and Xuefeng Yan. Differential evolution algorithm with co-evolution of control parameters and penalty factors for constrained optimization problems. *Asia-Pacific Journal of Chemical Engineering*, 7(2):227–235, 2012.

Adrian Hauswirth, Saverio Bolognani, Gabriela Hug, and Florian Dörfler. Generic existence of unique lagrange multipliers in ac optimal power flow. *IEEE control systems letters*, 2(4):791–796, 2018.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9): 5149–5169, 2021.

Yi-Chih Hsieh, Yung-Cheng Lee, and Peng-Sheng You. Solving nonlinear constrained optimization problems: An immune evolutionary based two-phase approach. *Applied Mathematical Modelling*, 39(19):5759–5768, 2015.

Fu-zhuo Huang, Ling Wang, and Qie He. An effective co-evolutionary differential evolution for constrained optimization. *Applied Mathematics and computation*, 186(1):340–356, 2007.

Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30, 2017.

Renato A Krohling and Leandro dos Santos Coelho. Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 36(6):1407–1416, 2006.

Jianjun Liu, Kok Lay Teo, Xiangyu Wang, and Changzhi Wu. An exact penalty function-based differential search algorithm for constrained global optimization. *Soft Computing*, 20:1305–1313, 2016.

Tian Liu, Bo Sun, and Danny HK Tsang. Rank-one solutions for sdp relaxation of qcqps in power systems. *IEEE Transactions on Smart Grid*, 10(1):5–15, 2017.

Ramtin Madani, Morteza Ashraphijuo, and Javad Lavaei. Promises of conic relaxation for contingency-constrained optimal power flow problem. *IEEE Transactions on Power Systems*, 31(2):1297–1307, 2015.

Ramtin Madani, Mohsen Kheirandishfard, Javad Lavaei, and Alper Atamtürk. Penalized semidefinite programming for quadratically-constrained quadratic optimization. *Journal of Global Optimization*, 78:423–451, 2020.

Efrén Mezura-Montes and Carlos A Coello Coello. Constraint-handling in nature-inspired numerical optimization: past, present and future. *Swarm and Evolutionary Computation*, 1(4):173–194, 2011.

Aryan Mokhtari, Shahin Shahrampour, Ali Jadbabaie, and Alejandro Ribeiro. Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 7195–7201. IEEE, 2016.

Gábor Pataki. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of operations research*, 23(2):339–358, 1998.

Somayeh Sojoudi and Javad Lavaei. Exactness of semidefinite relaxations for nonlinear optimization problems with underlying graph structure. *SIAM Journal on Optimization*, 24(4):1746–1778, 2014.

Jan Toenshoff, Martin Ritzert, Hinrikus Wolf, and Martin Grohe. Graph neural networks for maximum constraint satisfaction. *Frontiers in artificial intelligence*, 3:580607, 2021.

Alex L Wang. *On Quadratically Constrained Quadratic Programs and their Semidefinite Program Relaxations*. PhD thesis, University of Waterloo, 2022.

Bing-Chuan Wang, Han-Xiong Li, Yun Feng, and Wen-Jing Shen. An adaptive fuzzy penalty method for constrained evolutionary optimization. *Information Sciences*, 571:358–374, 2021.

Bing-Chuan Wang, Jing-Jing Guo, Pei-Qiu Huang, and Xian-Bing Meng. A two-stage adaptive penalty method based on co-evolution for constrained evolutionary optimization. *Complex & Intelligent Systems*, 9(4):4615–4627, 2023.

Siyuan Xu and Minghui Zhu. Efficient gradient approximation method for constrained bilevel optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12509–12517, 2023.

Zhipan Xu and Lijun Zhang. Online non-convex learning in dynamic environments. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Victor M Zavala and Mihai Anitescu. Real-time nonlinear optimization as a generalized equation. *SIAM Journal on Control and Optimization*, 48(8):5444–5467, 2010.

Fariba Zohrizadeh, Mohsen Kheirandishfard, Edward Quarm Jnr, and Ramtin Madani. Penalized parabolic relaxation for optimal power flow problem. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 1616–1623. IEEE, 2018.