STEERING WHAT SGD LEARNS: SPECTRAL CONTROL OF OPTIMIZATION THROUGH DYNAMIC DATA TRANSFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Out-of-distribution (OOD) generalization is challenging because the optimization dynamics induced by the training distribution may favor features that are poorly aligned with the target domain. We propose *Dynamic Gradient Covariance Steering* (DGC), a data-centric approach that steers the implicit optimization preference of SGD by dynamically reweighting training samples, without modifying the optimizer. Our key idea is to characterize an OOD-relevant target through its gradient-feature covariance and adjust the sampling distribution so that the resulting covariance approaches this target. We establish that, under a piecewise-linear training model and an OOD-relevant subspace sufficiency condition, the desired covariance is sufficient to recover the optimal OOD solution, while covariance steering provably reduces the learning time of OOD-relevant directions and accelerates the corresponding excess-risk decay. We further establish finite-width stability and quantify the error introduced by approximate covariance matching. Experiments on standard domain generalization benchmarks and controlled code understanding under covariate shift show consistent improvements over existing methods and demonstrate that the gains arise from dynamic sampling rather than simply adding synthetic data.

1 INTRODUCTION

A fundamental challenge in machine learning is out-of-distribution (OOD) generalization: models trained on one distribution must perform well on a differing test distribution, as in covariate shift (invariant $P(y \mid \mathbf{x})$, shifted $P(\mathbf{x})$) or domain generalization (target environment entirely unavailable) [2; 27; 34]. In both settings, the training distribution provides only an imperfect signal for which features to prioritize. Prior work addresses this by learning invariant or transferable representations across source environments [19; 18; 8], or by matching distributions/moments between source and target [21; 22; 33]. Such methods treat the training distribution as an object to be corrected or aligned, overlooking that SGD itself does not treat all feature directions equally: the sampling distribution determines the gradient statistics encountered during optimization, and hence which directions are learned earlier. Recent gradient-matching approaches suggest optimization dynamics themselves offer a useful lens for cross-domain generalization [28; 24].

This motivates a different question: rather than transforming the training distribution to resemble a target one, can we transform it so SGD learns OOD-relevant features more efficiently? We characterize this via the gradient-feature covariance induced by the training distribution, whose eigenspectrum reflects SGD's instantaneous optimization preference. The desired covariance need not come from the full target distribution: for covariate shift it can be estimated from unlabeled target inputs, and for domain generalization from structure shared across source environments. This relates to prior work exploiting shared cross-domain structure, including invariant representations, common-specific decompositions, and gradient-based domain matching [19; 23; 28], but differs in directly steering the covariance governing the optimization signal rather than enforcing representation invariance or modifying the optimizer. We thus cast OOD adaptation as *dynamic gradient covariance steering*: reweighting the sampling distribution so its induced covariance approaches an OOD-relevant target, while leaving the SGD optimizer unchanged.

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

We propose **Dynamic Gradient Covariance Steering (DGC)**, an iterative data transformation framework for OOD generalization. At each intervention step, DGC identifies the low-dimensional OOD-relevant subspace of the desired gradient covariance, computes sampling weights whose induced covariance approximates the desired one within this subspace, and trains the model with standard SGD under the resulting distribution, periodically recomputing the gradient features and weights. DGC thus changes the data presented to the optimizer rather than the optimization algorithm itself.

Theoretically, we show that an OOD-relevant covariance induces a corresponding optimization strength on the OOD-relevant subspace, with larger strength yielding faster OOD risk decay and, under a subspace sufficiency condition, convergence to the minimum OOD risk within the hypothesis class. We further show this guarantee is stable under finite-width feature drift and degrades only gracefully under approximate weight optimization.

We instantiate DGC for both covariate shift, using unlabeled target inputs to construct the desired covariance, and domain generalization, using the optimization structure shared across source environments. Across vision and code-understanding benchmarks, DGC consistently outperforms strong baselines—achieving 72.7% average accuracy on DomainBed (vs. 70.8% for Fishr and 68.7% for ERM) and improving over UniXcoder on most of reported code clone-detection and code-search metrics—demonstrating that dynamically steering the optimization preference through data transformation is an effective alternative to directly correcting the input distribution.

Our contributions are summarized as follows:

- **Dynamic Gradient Covariance Steering.** We provide a new perspective on OOD data transformation by viewing it as a mechanism for steering the optimization preference of SGD, rather than merely augmenting the training distribution. Based on this perspective, we propose *Dynamic Gradient Covariance Steering* (DGC), which dynamically estimates the desired gradient-feature covariance and determines sample weights to steer SGD toward OOD-relevant directions without modifying the underlying optimizer.
- **Theory of covariance steering.** Under a piecewise-linear training assumption, we show that an appropriate target covariance over an OOD-relevant subspace is sufficient for OOD optimality. We further prove that steering toward this covariance reduces the learning time of relevant directions and accelerates their excess-risk decay, with additional finite-width and approximate-matching guarantees.
- **Empirical evaluation.** DGC consistently improves performance across standard domain generalization benchmarks. In a controlled code understanding setting, counterfactual data are used to explicitly construct an OOD shift; comparisons with the same synthetic data without DGC weighting further demonstrate that the improvement comes from steering the optimization process rather than merely increasing data diversity.

2 Related Work

Out-of-distribution generalization. Out-of-distribution (OOD) generalization seeks predictors that remain effective when the test distribution differs from training [27; 34]. One line of work imposes invariance across source environments, e.g., Invariant Risk Minimization (IRM), V-REx, and domain-adversarial or correlation-alignment methods that enforce invariant representations [1; 14; 16; 18]. Another line addresses covariate shift via distribution alignment, using importance weighting with density-ratio estimation (e.g., KMM, KLIEP) or, more recently, matching moments or feature-space statistics directly (e.g., CORAL, MMD) [22; 7; 17]. These approaches typically modify the predictor, representation, or training distribution to make source and target more invariant or aligned. In contrast, our approach studies how the training distribution shapes the optimization dynamics of SGD, constructing a sampling distribution that induces an optimization preference favorable to OOD-relevant features.

Gradient-based OOD generalization. A growing line of work studies OOD generalization through the gradients of the training loss: gradient agreement methods encourage compatible optimization directions across domains [28], while Fishr matches domain-level gradient variances, showing that second-order statistics of per-sample gradients connect to the local loss landscape and are informative

for OOD generalization [24]. Fishr achieves this by adding a regularization term that constrains gradient statistics during optimization to make gradient variances invariant across source domains, whereas DGC neither adds a gradient regularizer nor modifies the optimizer; instead, it uses gradient features to identify OOD-relevant optimization directions and dynamically reweights the sampling distribution so that SGD itself preferentially learns these directions, allowing the same optimization-centric framework to address both domain generalization and controlled covariate shift.

Implicit bias and feature learning. Prior work shows that SGD learns predictive features at different rates, often favoring simpler or stronger features earlier [13; 30], which can cause reliance on shortcuts under distribution shift—suggesting generalization depends not only on the data or model class but also on the optimization trajectory [5; 26]. We build on this by treating SGD's optimization preference as controllable through data sampling: rather than removing this implicit bias, DGC redirects it toward the OOD-relevant subspace, using the training distribution to control not just which samples are observed but which features are learned earlier.

3 Problem Formulation

We consider OOD generalization under a training distribution P_{train} and a potentially different test distribution P_{test}. Given a model f_θ and loss function ℓ, the standard training objective is

$$\min_\theta \ \mathcal{L}_{\text{train}}(\theta) = \mathbb{E}_{(\mathbf{x},y)\sim P_{\text{train}}} \left[\ell(f_\theta(\mathbf{x}), y)\right]. \tag{1}$$

Our goal is to improve the target risk $\mathcal{L}_{\text{test}}(\theta)$ by modifying the training distribution, while keeping the optimization algorithm unchanged. The key observation is that changing the sampling distribution also changes the optimization preference of SGD.

At optimization step t, we define the *gradient feature* of sample i as $\boldsymbol{\phi}_i^t = \nabla_\theta \ell(f_{\theta_t}(\mathbf{x}_i), y_i)$. For sampling probabilities $\mathbf{p} = (p_1, \ldots, p_N)$, the corresponding gradient covariance is

$$\boldsymbol{\Sigma}_t(\mathbf{p}) = \sum_{i=1}^{N} p_i \boldsymbol{\phi}_i^t (\boldsymbol{\phi}_i^t)^\top. \tag{2}$$

The eigenspectrum of $\boldsymbol{\Sigma}_t$ characterizes the instantaneous optimization preference of SGD: directions with larger eigenvalues receive stronger optimization signals and tend to be learned earlier. We therefore seek to construct a desired, potentially time-dependent covariance $\boldsymbol{\Sigma}_t^*$ that captures the optimization preference relevant to OOD generalization.

Covariate shift. With unlabeled target samples available, we consider the covariate-shift setting where $P_{\text{test}}(y \mid \mathbf{x}) = P_{\text{train}}(y \mid \mathbf{x})$ but $P_{\text{test}}(\mathbf{x}) \neq P_{\text{train}}(\mathbf{x})$. Rather than directly requiring access to the target labels, we assume that the task-relevant capability can be specified through a data editing operation that modifies the feature spectrum of the training data toward the desired target capability. Let $\widetilde{P}^t_{\text{target}}$ denote the resulting edited distribution. We define the ideal optimization preference as

$$\boldsymbol{\Sigma}_t^* = \mathbb{E}_{\mathbf{x}\sim \widetilde{P}^t_{\text{target}}} \left[\boldsymbol{\phi}_t(\mathbf{x}) \boldsymbol{\phi}_t(\mathbf{x})^\top\right]. \tag{3}$$

The concrete construction of $\widetilde{P}^t_{\text{target}}$ depends on the target capability and can be implemented by task-specific data editing. We provide an example based on controlled feature editing and synthetic data generation in Section 7.

Domain generalization. When the target distribution is unavailable, we instead observe multiple source environments $\{P_1, \ldots, P_E\}$. Let

$$\boldsymbol{\Sigma}_t^e = \mathbb{E}_{(\mathbf{x},y)\sim P_e} \left[\boldsymbol{\phi}_t^e(\mathbf{x}, y) \boldsymbol{\phi}_t^e(\mathbf{x}, y)^\top\right] \tag{4}$$

denote the gradient covariance of environment e. We construct the desired optimization preference from the directions that are consistently supported across environments. Specifically, let $\mathbf{U}_{t,r}$ contain the r directions with the largest cross-environment stability, where stability is measured by the variance of the corresponding projected covariance strength across environments. We then define

$$\boldsymbol{\Sigma}_t^* = \mathbf{U}_{t,r} \boldsymbol{\Lambda}_{t,r} \mathbf{U}_{t,r}^\top, \tag{5}$$

where $\mathbf{\Lambda}_{t,r}$ contains the corresponding average covariance strengths across environments. This construction extracts a low-dimensional optimization structure that is shared across source environments, rather than requiring access to the unseen target distribution. The detailed procedure for estimating this stable subspace is described in Section 7.

4 OOD-Relevant Optimization Preference

The objective of covariance steering is not merely to match a prescribed covariance matrix. Its purpose is to modify the optimization preference of SGD so that OOD-relevant feature directions are learned more efficiently. We formalize this connection under the piecewise-linear view of neural network optimization.

Let $\mathcal{U}_t^* \subseteq \mathbb{R}^d$ denote the OOD-relevant subspace at optimization segment t, with $\dim(\mathcal{U}_t^*) = r \ll d$. We define the optimization strength induced by a gradient covariance $\mathbf{\Sigma}_t$ on this subspace as

$$\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t) = \min_{\substack{\mathbf{u}\in\mathcal{U}_t^* \\ \|\mathbf{u}\|_2=1}} \mathbf{u}^\top \mathbf{\Sigma}_t \mathbf{u}. \tag{6}$$

A larger value of $\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t)$ means that SGD assigns stronger optimization signals to all directions within the OOD-relevant subspace.

Let $\mathbf{\Sigma}_t^*$ denote the ideal gradient covariance defined for the corresponding OOD scenario. Rather than requiring the full covariance to characterize OOD performance, we focus on the subspace that contains the optimization-relevant directions for the OOD task.

Let

$$\mathbf{\Sigma}_t^* = \mathbf{U}_t^* \mathbf{\Lambda}_t^* (\mathbf{U}_t^*)^\top + \mathbf{R}_t^*, \tag{7}$$

where $\mathbf{U}_t^* = [\mathbf{u}_{t,1}^*, \ldots, \mathbf{u}_{t,r}^*] \in \mathbb{R}^{d\times r}$, $r \ll d$, contains the r OOD-relevant optimization directions. We denote their span by $\mathcal{U}_t^* = \mathrm{span}(\mathbf{U}_t^*)$. For any parameter displacement $\mathbf{v}$, let $\mathbf{P}_t^* = \mathbf{U}_t^*(\mathbf{U}_t^*)^\top$ be the orthogonal projection onto $\mathcal{U}_t^*$.

Assumption 4.1 (OOD-Relevant Subspace Sufficiency)**.** For each optimization segment t, let θ_t^* denote an OOD risk minimizer within the considered hypothesis class. There exists a constant $C_t > 0$ and a residual $\varepsilon_t \geq 0$ such that, for parameters θ in the local optimization region,

$$\mathcal{R}_{\mathrm{OOD}}(\theta) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \leq C_t \left\|\mathbf{P}_t^*(\theta - \theta_t^*)\right\|_2^2 + \varepsilon_t. \tag{8}$$

The residual ε_t accounts for the contribution of directions outside $\mathcal{U}_t^*$.

Assumption 4.2 (Piecewise-Linear SGD Dynamics)**.** Within each local optimization segment, the gradient features are fixed up to the local linearization of the model. Let $\mathbf{u}_{t,j}$ be an eigenvector of $\mathbf{\Sigma}_t$ with eigenvalue $\lambda_{t,j}$. The optimization error along this direction satisfies

$$a_{t,j,k+1} = (1 - \eta\lambda_{t,j}) a_{t,j,k}, \tag{9}$$

where $\eta > 0$ is the SGD step size and k indexes the local SGD updates.

Under this assumption, the optimization error along each spectral direction contracts independently:

$$|a_{t,j,k}| = (1 - \eta\lambda_{t,j})^k |a_{t,j,0}|. \tag{10}$$

Thus, the gradient covariance directly determines the rate at which different feature directions are learned.

Definition 4.3 (OOD Feature Learning Time)**.** For an OOD-relevant direction $\mathbf{u}_{t,j} \in \mathcal{U}_t^*$, define its ϵ-learning time as

$$\tau_{t,j}(\epsilon) = \min\left\{k : |a_{t,j,k}| \leq \epsilon\right\}. \tag{11}$$

Under Assumption 4.2, $\tau_{t,j}(\epsilon) \leq \frac{\log(|a_{t,j,0}|/\epsilon)}{-\log(1-\eta\lambda_{t,j})}$. For sufficiently small $\eta\lambda_{t,j}$,

$$\tau_{t,j}(\epsilon) \approx \frac{\log(|a_{t,j,0}|/\epsilon)}{\eta\lambda_{t,j}}. \tag{12}$$

Hence, a larger covariance eigenvalue corresponds to a shorter optimization time for the associated feature direction.

Theorem 4.4 (OOD Optimality under Sufficient Optimization Preference)**.** *Suppose Assumptions 4.1 and 4.2 hold. Let $\theta_{t,k}$ be obtained by k local SGD updates and let θ_t^* be the OOD risk minimizer in the considered hypothesis class. Then*

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \leq C_t \left(1 - \eta\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t)\right)^{2k} \|\mathbf{P}_t^*(\theta_{t,0} - \theta_t^*)\|_2^2 + \varepsilon_t. \tag{13}$$

In particular, if the ideal covariance satisfies

$$\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t^*) = \lambda_t^* > 0, \tag{14}$$

then the optimization process induced by $\mathbf{\Sigma}_t^*$ satisfies

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t,k}^*) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \leq C_t(1 - \eta\lambda_t^*)^{2k} \|\mathbf{P}_t^*(\theta_{t,0} - \theta_t^*)\|_2^2 + \varepsilon_t. \tag{15}$$

Consequently,

$$\limsup_{k\to\infty} \left[\mathcal{R}_{\mathrm{OOD}}(\theta_{t,k}^*) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*)\right] \leq \varepsilon_t. \tag{16}$$

Thus, when the OOD-relevant subspace is sufficient ($\varepsilon_t = 0$), the optimization dynamics induced by the ideal gradient covariance asymptotically attain the minimum OOD risk within the considered hypothesis class.

Corollary 4.5 (Reduced OOD Feature Learning Time)**.** *Under the conditions of Theorem 4.4, let $\tau_t^{\mathrm{steer}}(\epsilon)$ and $\tau_t^{\mathrm{src}}(\epsilon)$ denote the maximum learning time over the OOD-relevant subspace under the steered and original sampling distributions, respectively. Then*

$$\tau_t^{\mathrm{steer}}(\epsilon) \leq \frac{\log(C_t/\epsilon)}{2\eta\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t^{\mathrm{steer}})}, \tag{17}$$

while

$$\tau_t^{\mathrm{src}}(\epsilon) \leq \frac{\log(C_t/\epsilon)}{2\eta\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t^{\mathrm{src}})}. \tag{18}$$

Therefore, a larger steered optimization strength implies a shorter learning time: $\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t^{\mathrm{steer}}) > \lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t^{\mathrm{src}}) \Longrightarrow \tau_t^{\mathrm{steer}}(\epsilon) < \tau_t^{\mathrm{src}}(\epsilon)$.

4.1 Covariate Shift

Under covariate shift, let $\mathcal{U}_t^{\mathrm{target}}$ be its leading r-dimensional subspace and let $\mathbf{\Sigma}_{t,r}^{\mathrm{target}}$ denote the corresponding rank-r approximation.

Corollary 4.6 (Covariate-Shift Optimization Preference)**.** *Suppose the target gradient covariance is approximately concentrated on $\mathcal{U}_t^{\mathrm{target}}$, with residual spectral mass bounded by δ_t. If a sampling distribution induces*

$$\lambda_{\mathrm{OOD}}\left(\mathbf{\Sigma}_t\right) \geq \lambda_{\mathrm{OOD}}\left(\mathbf{\Sigma}_t^{\mathrm{src}}\right) + \gamma_t, \qquad \gamma_t > 0, \tag{19}$$

then the resulting SGD trajectory admits an OOD excess-risk bound whose exponential convergence rate improves by

$$2\eta\gamma_t, \tag{20}$$

up to the residual term $\delta_t + \varepsilon_\perp$.

4.2 Domain Generalization

When the target distribution is unavailable, the OOD-relevant subspace is estimated from the structure shared across multiple source environments $\{P_1, \ldots, P_E\}$ with gradient covariances $\mathbf{\Sigma}_t^e$. Let $\mathcal{U}_t^{\mathrm{inv}}$ denote an r-dimensional subspace whose optimization strength $\lambda_t^{\mathrm{inv}} = \min_{\mathbf{u}\in\mathcal{U}_t^{\mathrm{inv}}, \|\mathbf{u}\|_2=1} \mathbf{u}^\top \mathbf{\Sigma}_t^e \mathbf{u}$ is stable across environments, with cross-environment variation bounded by δ_t.

Corollary 4.7 (Domain-Generalization Optimization Preference)**.** *Suppose an unseen environment satisfies the same invariant-subspace condition with deviation at most δ_t. If the sampling distribution induces an optimization strength $\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t) \geq \lambda_t^{\mathrm{inv}}$, then the OOD excess risk on the unseen environment satisfies*

$$\mathcal{R}_{\mathrm{OOD}}(\theta_k) - \mathcal{R}_{\mathrm{OOD}}^* \leq C_t \exp\left[-2\eta k \left(\lambda_t^{\mathrm{inv}} - \delta_t\right)\right] + \varepsilon_\perp. \tag{21}$$

Thus, the guarantee depends on the optimization strength of the cross-environment stable subspace and degrades gracefully with its environmental variation.

Algorithm 1 Dynamic Gradient Covariance Steering

Require: Training dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, initial parameters θ_0, OOD-relevant covariance Σ_t^*, intervention interval K, total training steps T, SGD learning rate η, regularization coefficient λ.

Initialize model parameters θ_0 and sampling distribution $\mathbf{p} = (p_1, \ldots, p_N)$.
for $t = 0, 1, \ldots, T-1$ **do**
  if $t \bmod K = 0$ **then**
    Compute the gradient feature of each training sample: $\boldsymbol{\phi}_i^t = \nabla_\theta L(f_{\theta_t}(\mathbf{x}_i), y_i)$.
    Obtain the r-dimensional OOD-relevant subspace: $\mathbf{U}_t^* = \mathrm{Top}_r(\Sigma_t^*)$.
    Project each gradient feature onto the OOD-relevant subspace: $\mathbf{z}_i^t = (\mathbf{U}_t^*)^\top \boldsymbol{\phi}_i^t$.
    Compute the projected covariance contribution: $\mathbf{A}_i^t = \mathbf{z}_i^t (\mathbf{z}_i^t)^\top$.
    Compute the target covariance in the OOD-relevant subspace: $\mathbf{P}_t^* = (\mathbf{U}_t^*)^\top \Sigma_t^* \mathbf{U}_t^*$.
    Construct the covariance contribution matrix: $\mathbf{A}_t = [\mathrm{vec}(\mathbf{A}_1^t), \ldots, \mathrm{vec}(\mathbf{A}_N^t)]$.
    Solve the sampling-weight optimization problem: $\mathbf{w}_t = \arg\min_{\mathbf{w} \in \Delta_N} \|\mathbf{A}_t \mathbf{w} - \mathrm{vec}(\mathbf{P}_t^*)\|_2^2 + \lambda \|\mathbf{w} - \mathbf{p}\|_2^2$.
  end if
  Construct the training distribution using the latest sampling weights: $\mathcal{D}_t = T_t(\mathcal{D}; \mathbf{w}_t)$.
  Update model parameters using SGD: $\theta_{t+1} = \theta_t - \eta \nabla_\theta L_{\mathcal{D}_t}(f_{\theta_t})$.
end for
return trained model f_{θ_T}.

5 Dynamic Gradient Covariance Steering

Since the optimization preference relevant to OOD generalization is determined by the gradient covariance restricted to the OOD-relevant subspace, we steer the sampling distribution to match the ideal covariance only on this subspace rather than the full high-dimensional covariance. The overall procedure is summarized in Algorithm 1.

At intervention step t, each sample's gradient feature is projected onto the subspace, $\mathbf{z}_i^t = (\mathbf{U}_t^*)^\top \boldsymbol{\phi}_i^t \in \mathbb{R}^r$, giving the sample-wise covariance contribution $\mathbf{A}_i^t = \mathbf{z}_i^t (\mathbf{z}_i^t)^\top$. Let $\mathbf{P}_t^* = (\mathbf{U}_t^*)^\top \boldsymbol{\Sigma}_t^* \mathbf{U}_t^*$ denote the target covariance restricted to this subspace. Given sampling weights $\mathbf{w}_t$, the induced covariance is $\widehat{\mathbf{P}}_t(\mathbf{w}_t) = \sum_{i=1}^N w_i^t \mathbf{A}_i^t$.

The weights are obtained by solving the covariance approximation problem

$$\min_{\mathbf{w}_t} \left\| \sum_{i=1}^N w_i^t \mathbf{A}_i^t - \mathbf{P}_t^* \right\|_F^2 + \lambda \|\mathbf{w}_t - \mathbf{p}\|_2^2, \tag{22}$$

subject to $w_i^t \geq 0$, $\sum_i w_i^t = 1$, where $\mathbf{p}$ is the original sampling distribution and $\lambda \geq 0$ penalizes deviation from it.

Vectorizing $\mathbf{A}_t = [\mathrm{vec}(\mathbf{A}_1^t), \ldots, \mathrm{vec}(\mathbf{A}_N^t)]$, the problem becomes quadratic, and ignoring the simplex constraint admits the closed-form solution

$$\mathbf{w}_t = \left(\mathbf{A}_t^\top \mathbf{A}_t + \lambda \mathbf{I}\right)^{-1} \left(\mathbf{A}_t^\top \widehat{\mathbf{p}}_t + \lambda \mathbf{p}\right). \tag{23}$$

The weights are then normalized and projected onto the simplex $\Delta_N = \{\mathbf{w} \in \mathbb{R}^N : w_i \geq 0, \sum_i w_i = 1\}$ when necessary.

As detailed in Algorithm 1, the model is then optimized via SGD under the updated sampling distribution $\mathbf{w}_t$; after a fixed number of steps, the gradient features and target subspace are recomputed. Dynamic Gradient Covariance Steering thus continuously adapts the sampling distribution to the current OOD-relevant subspace, rather than applying a fixed reweighting throughout training.

6 Theoretical Analysis

6.1 Finite-Width Stability

The preceding analysis assumes the gradient covariance accurately reflects the local optimization geometry. In finite-width networks, gradient features evolve during training, so the covariance used

for sampling is only a local approximation of the covariance governing the next optimization segment. We quantify how this feature drift affects the OOD optimization guarantee.

Let $\boldsymbol{\phi}_i^t$ denote the gradient feature of sample i at intervention step t, and define the local feature drift and boundedness condition

$$\delta_t = \max_i \left\|\boldsymbol{\phi}_i^{t+1} - \boldsymbol{\phi}_i^t\right\|_2, \qquad \max_{i,t} \left\|\boldsymbol{\phi}_i^t\right\|_2 \le M. \tag{24}$$

Proposition 6.1 (Finite-Width Covariance Stability)**.** *Under the bounded feature condition, the covariance induced by the same sampling distribution at two consecutive intervention steps satisfies*

$$\|\boldsymbol{\Sigma}_{t+1} - \boldsymbol{\Sigma}_t\|_2 \le 2M\delta_t + \delta_t^2. \tag{25}$$

This shows the local covariance remains stable whenever features evolve slowly, without requiring an infinite-width or fixed-feature assumption. Writing $\lambda_t^{\mathrm{OOD}} = \lambda_{\mathrm{OOD}}(\boldsymbol{\Sigma}_t)$, the standard spectral perturbation bound $|\lambda_{t+1}^{\mathrm{OOD}} - \lambda_t^{\mathrm{OOD}}| \le \|\boldsymbol{\Sigma}_{t+1} - \boldsymbol{\Sigma}_t\|_2$ combined with Proposition 6.1 gives

$$\left|\lambda_{t+1}^{\mathrm{OOD}} - \lambda_t^{\mathrm{OOD}}\right| \le 2M\delta_t + \delta_t^2, \tag{26}$$

i.e., feature evolution perturbs the OOD optimization strength only by an amount controlled by the local drift.

Theorem 6.2 (Finite-Width OOD Optimization Stability)**.** *Suppose the conditions of Theorem 4.4 hold and the feature drift satisfies Eq. equation 24. Let $\bar{\delta}_t = 2M\delta_t + \delta_t^2$. Then*

$$\lambda_{t+1}^{\mathrm{OOD}} \ge \lambda_t^{\mathrm{OOD}} - \bar{\delta}_t. \tag{27}$$

Consequently, for step size η with $0 < \eta(\lambda_t^{\mathrm{OOD}} - \bar{\delta}_t) < 1$, the OOD excess risk after k local updates satisfies

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t+1,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_{t+1}^*) \le C_{t+1}\left(1 - \eta\left(\lambda_t^{\mathrm{OOD}} - \bar{\delta}_t\right)\right)^{2k} E_{t+1}^2 + \varepsilon_{t+1}, \tag{28}$$

where $E_{t+1} = \|\mathbf{P}_{t+1}^*(\theta_{t+1,0} - \theta_{t+1}^*)\|_2$.

Thus the OOD optimality result established in the piecewise-linear setting is stable under finite-width feature evolution, with the degradation appearing explicitly through $2M\delta_t + \delta_t^2$: slow feature evolution implies small degradation of the OOD rate. More generally, if $\sum_{t=0}^{T-1}(2M\delta_t + \delta_t^2) \le \Delta_T$, the cumulative perturbation over T steps is bounded by

$$\lambda_T^{\mathrm{OOD}} \ge \lambda_0^{\mathrm{OOD}} - \Delta_T. \tag{29}$$

Hence the ideal OOD optimization guarantee remains effective as long as the cumulative feature drift stays sufficiently smaller than the initial OOD optimization strength.

6.2 Approximate Covariance Steering

In practice, the sampling weights $\widehat{\mathbf{w}}_t$ only approximately solve the covariance steering problem, with error $\epsilon_{w,t} = \|\widehat{\mathbf{w}}_t - \mathbf{w}_t^*\|_2$ relative to the exact solution $\mathbf{w}_t^*$. Letting $\mathbf{A}_t$ collect the vectorized sample-wise covariance contributions with $\|\mathbf{A}_t\|_2 \le B_t$, the resulting covariance error decomposes as

$$\left\|\widehat{\mathbf{P}}_t - \mathbf{P}_t^*\right\|_F \le \delta_t^{\mathrm{opt}} + B_t\epsilon_{w,t}, \tag{30}$$

where $\delta_t^{\mathrm{opt}} = \|\mathbf{A}_t\mathbf{w}_t^* - \widehat{\mathbf{p}}_t\|_2$ is the irreducible error from restricting to the feasible weighting set, and $B_t\epsilon_{w,t}$ is the additional error from inexact optimization.

Theorem 6.3 (Approximate OOD Optimization Guarantee)**.** *Let λ_t^* be the ideal OOD optimization strength from Theorem 4.4. Then*

$$\lambda_{\mathrm{OOD}}(\widehat{\boldsymbol{\Sigma}}_t) \ge \lambda_t^* - \Delta_t^{\mathrm{opt}}, \qquad \Delta_t^{\mathrm{opt}} := \delta_t^{\mathrm{opt}} + B_t\epsilon_{w,t}. \tag{31}$$

Consequently, if $\Delta_t^{\mathrm{opt}} < \lambda_t^*$, the OOD excess risk after k local SGD updates satisfies

$$\mathcal{R}_{\mathrm{OOD}}(\widehat{\theta}_{t,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \le C_t\left[1 - \eta\left(\lambda_t^* - \Delta_t^{\mathrm{opt}}\right)\right]^{2k} E_t^2 + \varepsilon_t, \tag{32}$$

with $E_t = \|\mathbf{P}_t^*(\widehat{\theta}_{t,0} - \theta_t^*)\|_2$.

Thus, imperfect weight optimization degrades the ideal rate λ_t^* by at most Δ_t^{opt}. In particular, as $\epsilon_{w,t} \to 0$ and (when the feasible set admits an exact covariance match) $\delta_t^{\mathrm{opt}} \to 0$, we recover $\lambda_{\mathrm{OOD}}(\widehat{\boldsymbol{\Sigma}}_t) \to \lambda_t^*$.

Table 1: OOD generalization performance on seven benchmark datasets. We report the mean accuracy and standard deviation over 5 runs, with the average computed across all benchmarks. The best result on each benchmark and the overall average are highlighted in bold.

Algorithm	CMNIST	RMNIST	VLCS	PACS	OfficeHome	TerraInc	DomainNet	Avg
ERM	57.8 ± 0.2	97.8 ± 0.1	77.6 ± 0.3	86.7 ± 0.3	66.4 ± 0.5	53.0 ± 0.3	41.3 ± 0.1	68.7
IRM [1]	67.7 ± 1.2	97.5 ± 0.2	76.9 ± 0.6	84.5 ± 1.1	63.0 ± 2.7	50.5 ± 0.7	28.0 ± 5.1	66.9
GroupDRO [25]	61.1 ± 0.9	97.9 ± 0.1	77.4 ± 0.5	87.1 ± 0.1	66.2 ± 0.6	52.4 ± 0.1	33.4 ± 0.3	67.9
Mixup [31]	58.4 ± 0.2	98.0 ± 0.1	78.1 ± 0.3	86.8 ± 0.3	68.0 ± 0.2	54.4 ± 0.3	39.6 ± 0.1	69.0
MLDG [15]	58.2 ± 0.4	97.8 ± 0.1	77.5 ± 0.1	86.8 ± 0.4	66.6 ± 0.3	52.0 ± 0.1	41.6 ± 0.1	68.7
CORAL [29]	58.6 ± 0.5	98.0 ± 0.0	77.7 ± 0.2	87.1 ± 0.5	68.4 ± 0.2	52.8 ± 0.2	41.8 ± 0.1	69.2
MMD [7]	63.3 ± 1.3	98.0 ± 0.1	77.9 ± 0.1	87.2 ± 0.1	66.2 ± 0.3	52.0 ± 0.4	23.5 ± 9.4	66.9
DANN [4]	57.0 ± 1.0	97.9 ± 0.1	79.7 ± 0.5	85.2 ± 0.2	65.3 ± 0.8	50.6 ± 0.4	38.3 ± 0.1	67.7
CDANN [16]	59.5 ± 2.0	97.9 ± 0.0	79.9 ± 0.2	85.8 ± 0.8	65.3 ± 0.5	50.8 ± 0.6	38.5 ± 0.2	68.2
MTL [6]	57.6 ± 0.3	97.9 ± 0.1	77.7 ± 0.5	86.7 ± 0.2	66.5 ± 0.4	52.2 ± 0.4	40.8 ± 0.1	68.5
SagNet [20]	58.2 ± 0.3	97.9 ± 0.0	77.6 ± 0.1	86.4 ± 0.4	67.5 ± 0.2	52.5 ± 0.4	40.8 ± 0.2	68.7
ARM [32]	63.2 ± 0.7	**98.1** ± 0.1	77.8 ± 0.3	85.8 ± 0.2	64.8 ± 0.4	51.2 ± 0.5	36.0 ± 0.2	68.1
V-REx [14]	67.0 ± 1.3	97.9 ± 0.1	78.1 ± 0.2	87.2 ± 0.6	65.7 ± 0.3	51.4 ± 0.5	30.1 ± 3.7	68.2
RSC [12]	58.5 ± 0.5	97.6 ± 0.1	77.8 ± 0.6	86.2 ± 0.5	66.5 ± 0.6	52.1 ± 0.2	38.9 ± 0.6	68.2
Fish	61.8 ± 0.8	97.9 ± 0.1	77.8 ± 0.6	85.8 ± 0.6	66.0 ± 2.9	50.8 ± 0.4	43.4 ± 0.3	69.1
Fishr	68.8 ± 1.4	97.8 ± 0.1	78.2 ± 0.2	86.9 ± 0.2	68.2 ± 0.2	53.6 ± 0.4	41.8 ± 0.2	70.8
DGC	**70.2** ± 0.6	97.9 ± 0.1	**80.6** ± 0.5	**88.1** ± 0.4	**69.9** ± 0.5	**56.7** ± 0.6	**45.8** ± 0.2	**72.7**

7 EXPERIMENT

7.1 DOMAINNET

Benchmark. We evaluate DGC on DomainNet [22] following the DomainBed protocol [8]. DomainNet is a challenging multi-domain image classification benchmark containing six visually distinct domains: *clipart*, *infograph*, *painting*, *quickdraw*, *real*, and *sketch*. It contains 345 object categories and approximately 586K images, making it substantially more diverse than smaller domain generalization benchmarks. Following the standard leave-one-domain-out setting, we use one domain as the held-out test domain and train on the remaining domains. We report classification accuracy on the unseen domain and follow the same evaluation protocol as Fishr for direct comparison.

Implementation details. We use the same ResNet-50 architecture and ImageNet initialization as the standard DomainBed experiments. The backbone is fine-tuned while batch normalization statistics are kept fixed. Rather than requiring access to the held-out domain, the desired covariance Σ_t^* is constructed from the training environments only. Specifically, we extract the optimization directions that are consistently supported across the source domains and retain their low-dimensional shared subspace. The resulting covariance within this subspace serves as the target optimization structure for the subsequent covariance-steering step.

Results. Table 1 summarizes the results across DomainBed benchmarks. DGC achieves an average accuracy of 72.7% across the seven benchmarks, compared with 68.7% for ERM and 70.8% for Fishr. The improvement is consistent with the proposed optimization view: instead of explicitly enforcing invariance of gradient statistics across domains, DGC uses the shared source-domain optimization structure to identify directions that are useful for OOD learning and dynamically increases their sampling-induced optimization strength. On DomainNet, which contains six substantially different visual domains, the larger gain suggests that dynamically steering the optimization preference can be particularly beneficial when the domain shift is strong and heterogeneous.

7.2 CROSS-MODAL CODE REPRESENTATION UNDER CONTROLLED COVARIATE SHIFT

Tasks and datasets. Following the understanding-task evaluation protocol of UniXcoder [10], we evaluate DGC on both clone detection and code search. Clone detection measures whether the learned code representation can capture semantic similarity between programs, while code search evaluates cross-modal alignment between natural-language descriptions and code.

For clone detection, we report MAP@R on POJ-104 and Recall, Precision, and F1-score on BigCloneBench. For code search, we report MRR on CosQA [11], AdvTest [3], and CodeSearchNet

Table 2: Results on code understanding tasks. We follow the evaluation protocol of UniXcoder. "Synthetic Data Only" uses the same counterfactual synthetic data as DGC but removes the proposed DGC sampling weights.

Model	Clone Detection				Code Search		
	POJ-104	BigCloneBench					
	MAP@R	Recall	Precision	F1-score	CosQA	AdvTest	CSN
RoBERTa	76.67	**95.1**	87.8	91.3	60.3	18.3	61.7
CodeBERT	82.67	94.7	93.4	94.1	65.7	27.2	69.3
GraphCodeBERT	85.16	94.8	95.2	95.0	68.4	35.2	71.3
SyNCoBERT	88.24	-	-	-	-	38.3	74.0
CodeT5-base	88.65	94.8	94.7	95.0	67.8	39.3	71.5
UniXcoder	90.52	92.9	**97.6**	95.2	70.1	41.3	74.4
DGC	**91.77**	**95.8**	96.7	**96.2**	**72.3**	**43.5**	**75.1**
Synthetic Data Only	90.83	93.3	96.4	94.8	69.4	41.1	74.7

(CSN) [9]. These benchmarks cover both code-to-code and text-to-code retrieval, providing complementary evaluation of the learned code representations.

Training setting. We use UniXcoder as the backbone and follow its standard code understanding setup. We augment the original training data with counterfactual code generated through program-graph interventions, such as removing selected `CALLS` relations, thereby creating a controlled covariate shift with structural changes. We use UniXcoder as the backbone and follow its standard code understanding setup. We augment the original training data with counterfactual code generated through program-graph interventions, such as removing selected `CALLS` relations, thereby creating a controlled covariate shift with structural changes. The training details can be found at B.1.

At each DGC intervention step, we extract gradient features from the current model, estimate Σ_t^* from the synthesized target distribution, and solve the covariance approximation problem to obtain sample-wise sampling weights. To isolate the effect of covariance steering, we compare against a *Synthetic Data Only* variant that uses the identical synthetic data without DGC weighting, directly measuring the benefit of dynamic sampling.

Results. Table 2 reports the results. DGC consistently outperforms the corresponding Synthetic Data Only variant across most of seven evaluation metrics. The consistent improvement over Synthetic Data Only is important for isolating the effect of DGC. Since both methods use the same counterfactual data, the performance gap cannot be attributed simply to increased data diversity. Instead, it indicates that dynamically reweighting the synthetic examples according to their gradient-covariance contribution provides a more effective optimization signal for the downstream code understanding tasks.

DGC also improves over the original UniXcoder results on most of reported metrics. The gains are particularly visible on clone detection and cross-modal code search, suggesting that steering the optimization toward the target structural covariance can improve both code-to-code semantic representation and text-to-code alignment.

8 Conclusion

We propose Dynamic Gradient Covariance Steering (DGC), an optimization-centric framework for OOD generalization that dynamically reweights training samples to steer SGD toward OOD-relevant optimization directions. Our theory establishes the connection between the desired gradient covariance, feature learning dynamics, and OOD excess risk, and further provides finite-width and approximate covariance-matching guarantees. Experiments on domain generalization benchmarks and controlled code understanding under covariate shift demonstrate that DGC consistently improves OOD performance, while comparisons with synthetic data without DGC weighting show that the gains arise from optimization steering rather than simply increasing data diversity.

9 ICLR CHECKS

AI USE STATEMENT

In this work, we used generative AI tools to aid or polish writing. We have not used generative AI tools for other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for drafting sections of the paper. We have reviewed all AI-assisted work. Specifically, all text and code snippets generated or refined by the AI were manually verified by the authors to ensure factual accuracy, logical consistency, and correct citation attribution. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

We do not involve human subjects in our experiments, nor do we collect any private user data. Our research relies on publicly available benchmark datasets, which are widely used in the community and have been vetted for standard ethical considerations. We believe our proposed method does not introduce new societal risks or potential for misuse beyond those inherent to the general field. All code and models will be released under an open-source license to promote reproducibility and transparency.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our results, we have made significant efforts to document our methodology and share our resources.

- **Code and Models:** We will release the complete source code, including training scripts and pre-trained model checkpoints, in an anonymous repository upon publication (link provided in supplementary materials).
- **Datasets:** All datasets used in this study are publicly available. We provide detailed descriptions of data preprocessing steps and hyperparameter configurations in Appendix.
- **Experimental Setup:** We report the mean and standard deviation over multiple random seeds in Section 7 to ensure statistical reliability.

We believe these measures allow other researchers to faithfully reproduce our findings.

REFERENCES

[1] Martín Arjovsky, L. Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *ArXiv*, abs/1907.02893, 2019.

[2] Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. Analysis of representations for domain adaptation. In *NIPS*, volume 19, 2007.

[3] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, et al. Codebert: A pre-trained model for programming and natural languages. In *Findings of the association for computational linguistics: EMNLP 2020*, pp. 1536–1547, 2020.

[4] Yaroslav Ganin, E. Ustinova, Hana Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35, 2016.

[5] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[6] Muhammad Ghifary, W. Kleijn, M. Zhang, and D. Balduzzi. Domain generalization for object recognition with multi-task autoencoders. *ICCV*, pp. 2551–2559, 2015.

[7] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[8] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2021.

[9] Daya Guo, Shuo Ren, Shuai Lu, Zhangyin Feng, Duyu Tang, Shujie Liu, Long Zhou, Nan Duan, Alexey Svyatkovskiy, Shengyu Fu, et al. Graphcodebert: Pre-training code representations with data flow. *arXiv preprint arXiv:2009.08366*, 2020.

[10] Daya Guo, Shuai Lu, Nan Duan, Yanlin Wang, Ming Zhou, and Jian Yin. Unixcoder: Unified cross-modal pre-training for code representation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7212–7225, 2022.

[11] Junjie Huang, Duyu Tang, Linjun Shou, Ming Gong, Ke Xu, Daxin Jiang, Ming Zhou, and Nan Duan. Cosqa: 20,000+ web queries for code search and question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 5690–5700, 2021.

[12] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *ECCV*, volume 2, 2020.

[13] Dimitris Kalimeris, Gal Kaplun, Preetum Nakkiran, Benjamin Edelman, Tristan Yang, Boaz Barak, and Haofeng Zhang. Sgd on neural networks learns functions of increasing complexity. *Advances in neural information processing systems*, 32, 2019.

[14] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *ICML*, pp. 5815–5826, 2021.

[15] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018.

[16] Haoliang Li, Sinno Jialin Pan, S. Wang, and A. Kot. Domain generalization with adversarial feature learning. In *CVPR*, pp. 5400–5409, 2018.

[17] Y. Li and N. Vasconcelos. Efficient multi-domain learning by covariance normalization. *CVPR*, pp. 5419–5428, 2019.

[18] Ya Li, Mingming Gong, Xinmei Tian, Tongliang Liu, and Dacheng Tao. Domain generalization via conditional invariant representations. In *AAAI*, 2018.

[19] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, 2013.

[20] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *CVPR*, pp. 8690–8699, 2021.

[21] Sinno Jialin Pan, I. Tsang, James T. Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE TNN*, 22:199–210, 2011.

[22] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. *ICCV*, pp. 1406–1415, 2019.

[23] Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Efficient domain generalization via common-specific low-rank decomposition. In *ICML*, pp. 7728–7738, 2020.

[24] Alexandre Rame, Corentin Dancette, and Matthieu Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. *arXiv preprint arXiv:2109.02934*, 2021.

[25] Shiori Sagawa, Pang Wei Koh, T. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *ICLR*, 2020.

[26] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in neural information processing systems*, 33:9573–9585, 2020.

[27] Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.

[28] Yuge Shi, Jeffrey Seely, Philip HS Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *ICLR*, 2022.

[29] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*, pp. 443–450, 2016.

[30] Haihan Zhang, Yimu Zhang, and Cong Fang. Strong correlations induce cause only predictions in transformer training. In *ICLR*, volume 2026, pp. 111966–112005, 2026.

[31] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[32] Marvin Zhang, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn. Adaptive risk minimization: A meta-learning approach for tackling group distribution shift. *arXiv preprint arXiv:2007.02931*, 2020.

[33] Yuchen Zhang, T. Liu, Mingsheng Long, and Michael I. Jordan. Bridging theory and algorithm for domain adaptation. In *ICML*, 2019.

[34] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization in vision: A survey. *arXiv:2103.02503*, 2021.

A Appendix

A.1 Proof of Theorem 4.4

Proof. Let

$$\mathbf{e}_{t,k} = \mathbf{P}_t^*(\theta_{t,k} - \theta_t^*) \tag{33}$$

denote the optimization error projected onto the OOD-relevant subspace.

Under the piecewise-linear SGD assumption, within a fixed local linear region, the update along an eigen-direction $\mathbf{u}_{t,j}$ of $\mathbf{\Sigma}_t$ satisfies

$$a_{j,k+1} = (1 - \eta\lambda_{t,j})\, a_{j,k} + \eta\xi_{j,k}, \tag{34}$$

where $a_{j,k}$ is the coefficient of the optimization error along $\mathbf{u}_{t,j}$ and $\xi_{j,k}$ denotes the stochastic update noise. For the OOD-relevant subspace, let

$$\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t) = \min_{\mathbf{u}\in\mathcal{S}_t^*,\, \|\mathbf{u}\|_2=1} \mathbf{u}^\top \mathbf{\Sigma}_t \mathbf{u}. \tag{35}$$

Ignoring the stochastic residual, the projected error therefore satisfies

$$\|\mathbf{e}_{t,k}\|_2 \le (1 - \eta\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t))^k \, \|\mathbf{e}_{t,0}\|_2 \,, \tag{36}$$

provided that $0 < \eta\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t) < 1$.

The OOD-subspace sufficiency assumption implies that the OOD excess risk is controlled by the optimization error inside $\mathcal{S}_t^*$. Consequently, for some constant $C_t > 0$,

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \le C_t \, \|\mathbf{e}_{t,k}\|_2^2 + \varepsilon_t, \tag{37}$$

where ε_t accounts for the approximation error outside the OOD-relevant subspace and the local linearization error. Combining the preceding two inequalities gives

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \le C_t \, (1 - \eta\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t))^{2k} \, \|\mathbf{P}_t^*(\theta_{t,0} - \theta_t^*)\|_2^2 + \varepsilon_t, \tag{38}$$

which proves the first claim.

For the ideal covariance $\mathbf{\Sigma}_t^*$, suppose that

$$\lambda_{\mathrm{OOD}}(\mathbf{\Sigma}_t^*) = \lambda_t^* > 0. \tag{39}$$

The same local SGD dynamics then give

$$\left\|\mathbf{e}_{t,k}^*\right\|_2 \le (1-\eta\lambda_t^*)^k \left\|\mathbf{e}_{t,0}^*\right\|_2. \tag{40}$$

Substituting this bound into the OOD excess-risk inequality yields

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t,k}^*) - \mathcal{R}_{\mathrm{OOD}}(\theta_t^*) \le C_t\,(1-\eta\lambda_t^*)^{2k} \left\|\mathbf{P}_t^*(\theta_{t,0}-\theta_t^*)\right\|_2^2 + \varepsilon_t. \tag{41}$$

Therefore, a covariance that provides sufficient optimization strength along the OOD-relevant subspace yields geometric decay of the OOD excess risk. □

A.2 Proof of Theorem 6.2

Proof. Let $\mathcal{S}_t^*$ denote the OOD-relevant subspace and let $\mathbf{P}_t^*$ be its orthogonal projection matrix. By the finite-width feature drift condition in Eq. equation 24, for every training sample we have

$$\left\|\boldsymbol{\phi}_{t+1}(\mathbf{x}) - \boldsymbol{\phi}_t(\mathbf{x})\right\|_2 \le \delta_t. \tag{42}$$

Define

$$\mathbf{\Delta}_t(\mathbf{x}) = \boldsymbol{\phi}_{t+1}(\mathbf{x}) - \boldsymbol{\phi}_t(\mathbf{x}). \tag{43}$$

Then $\|\mathbf{\Delta}_t(\mathbf{x})\|_2 \le \delta_t$. The change in the sample-wise covariance contribution is

$$\boldsymbol{\phi}_{t+1}\boldsymbol{\phi}_{t+1}^\top - \boldsymbol{\phi}_t\boldsymbol{\phi}_t^\top = \boldsymbol{\phi}_t\mathbf{\Delta}_t^\top + \mathbf{\Delta}_t\boldsymbol{\phi}_t^\top + \mathbf{\Delta}_t\mathbf{\Delta}_t^\top. \tag{44}$$

Since $\|\boldsymbol{\phi}_t(\mathbf{x})\|_2 \le M$, the spectral norm of the covariance perturbation satisfies

$$\left\|\boldsymbol{\phi}_{t+1}\boldsymbol{\phi}_{t+1}^\top - \boldsymbol{\phi}_t\boldsymbol{\phi}_t^\top\right\|_2 \le 2M\delta_t + \delta_t^2 = \bar{\delta}_t. \tag{45}$$

Taking the expectation over the training distribution gives

$$\left\|\mathbf{\Sigma}_{t+1} - \mathbf{\Sigma}_t\right\|_2 \le \bar{\delta}_t. \tag{46}$$

For any unit vector $\mathbf{u} \in \mathcal{S}_t^*$, we therefore have

$$\mathbf{u}^\top\mathbf{\Sigma}_{t+1}\mathbf{u} \ge \mathbf{u}^\top\mathbf{\Sigma}_t\mathbf{u} - \left\|\mathbf{\Sigma}_{t+1} - \mathbf{\Sigma}_t\right\|_2. \tag{47}$$

Using the definition of the OOD optimization strength,

$$\lambda_t^{\mathrm{OOD}} = \min_{\mathbf{u}\in\mathcal{S}_t^*,\,\|\mathbf{u}\|_2=1} \mathbf{u}^\top\mathbf{\Sigma}_t\mathbf{u}, \tag{48}$$

we obtain

$$\lambda_{t+1}^{\mathrm{OOD}} \ge \lambda_t^{\mathrm{OOD}} - \bar{\delta}_t, \tag{49}$$

which proves Eq. equation 27.

We next consider the local SGD dynamics at step $t+1$. By Theorem 4.4, the optimization error restricted to the OOD-relevant subspace contracts according to the effective optimization strength. From the preceding bound, this strength is at least $\lambda_t^{\mathrm{OOD}} - \bar{\delta}_t$. Hence, after k local SGD updates we have

$$\left\|\mathbf{P}_{t+1}^*(\theta_{t+1,k} - \theta_{t+1}^*)\right\|_2 \le \left(1 - \eta\left(\lambda_t^{\mathrm{OOD}} - \bar{\delta}_t\right)\right)^k E_{t+1}, \tag{50}$$

where

$$E_{t+1} = \left\|\mathbf{P}_{t+1}^*(\theta_{t+1,0} - \theta_{t+1}^*)\right\|_2. \tag{51}$$

Finally, Assumption 4.1 implies that the OOD excess risk is controlled by the optimization error in the OOD-relevant subspace. Thus,

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t+1,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_{t+1}^*) \le C_{t+1}\left\|\mathbf{P}_{t+1}^*(\theta_{t+1,k} - \theta_{t+1}^*)\right\|_2^2 + \varepsilon_{t+1}. \tag{52}$$

Substituting the contraction bound gives

$$\mathcal{R}_{\mathrm{OOD}}(\theta_{t+1,k}) - \mathcal{R}_{\mathrm{OOD}}(\theta_{t+1}^*) \le C_{t+1}\left(1 - \eta\left(\lambda_t^{\mathrm{OOD}} - \bar{\delta}_t\right)\right)^{2k} E_{t+1}^2 + \varepsilon_{t+1}, \tag{53}$$

which proves Eq. equation 28. □

B EXPERIMENT DETAILS

B.1 CODE DATA CONSTRUCTION AND COUNTERFACTUAL SYNTHESIS

Source data. Our code experiments are based on a collection of Python functions represented by local program graphs. The resulting counterfactual dataset contains 55,453 records. Each record is associated with a unique function identifier and contains the original code, a counterfactual version, and the corresponding program graphs. In particular, the original graph contains the complete local program structure, while the counterfactual graph is obtained by removing one selected `CALLS` edge. The corresponding source code is then regenerated from the modified graph.

For a given function, multiple counterfactual examples can be generated by removing different `CALLS` edges. Consequently, examples generated from the same function share the same original code while differing in the removed dependency. This structure allows us to construct batches in which the model must distinguish closely related programs that differ in a specific structural dependency.

Counterfactual synthesis. The counterfactual generation process consists of three steps. First, we extract a local program graph centered on the target function, including function and module nodes together with their structural and dependency relations. Second, we select a `CALLS` edge as the intervention target and remove this edge from the graph. Third, a code generation model is used to regenerate the function conditioned on the modified graph. The resulting code preserves the surrounding program structure while removing the selected function-call dependency.

Let G denote the original program graph and let $e = (u, v)$ be a selected `CALLS` edge. The counterfactual graph is defined as $G^{\mathrm{cf}} = G \setminus e$. We denote the corresponding generated code by x^{cf}. The resulting pair (x, x^{cf}) therefore represents a controlled structural intervention rather than an unconstrained code perturbation.

Controlled target distribution. The counterfactual transformation provides an explicit mechanism for defining a target distribution. Instead of treating all possible code changes as equally relevant, we specify the target structural property through the intervention operator used to construct G^{cf}. The resulting synthetic target samples emphasize the corresponding class of program dependencies and provide the samples from which the target gradient covariance Σ_t^* is estimated.

At each DGC intervention step, gradient features are extracted from the synthetic target samples using the current model parameters. We estimate their empirical covariance and retain the leading OOD-relevant subspace. This produces a dynamic target covariance Σ_t^* that changes as the model representation evolves. Importantly, the target samples are used only to specify the controlled target optimization structure; DGC does not change the SGD update rule itself.

Training and evaluation split. The counterfactual dataset is used for training and optimization analysis, whereas CodeSearchNet is used only as an independent evaluation benchmark. The CodeSearchNet validation set contains 23,107 Python docstring–code pairs. No natural-language queries are used during counterfactual training. We evaluate the final code representations by retrieving code with natural-language queries and computing MRR@10.

Training objective. The retriever encodes each code fragment using a RoBERTa encoder followed by masked mean pooling and ℓ_2 normalization. For the counterfactual training objective, examples generated from the same function are grouped together so that the original function is contrasted against counterfactual variants associated with different removed dependencies. This construction prevents the model from solving the task using unrelated function identity and instead encourages it to encode the structural difference introduced by the graph intervention.

We use the covariance-steering weights produced by DGC to determine the sampling probability of each training example. The weighted training distribution is updated periodically as the gradient features evolve. All evaluation on CodeSearchNet is performed using the text-only code encoder, so the retrieval benchmark measures the quality of the learned code representation rather than the auxiliary graph representation used during training.