ASPIRE: MAKE SPECTRAL GRAPH COLLABORATIVE FILTERING GREAT AGAIN VIA ADAPTIVE FILTER LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph filter design is central to spectral collaborative filtering, yet most existing methods rely on manually tuned hyperparameters rather than fully learnable filters. We show that this challenge stems from a bias in traditional recommendation objectives, which induces a spectral phenomenon termed low-frequency explosion, thereby fundamentally hindering the effective learning of graph filters. To overcome this limitation, we propose a novel Adaptive Spectral graPh collaboratIve filteRing framEwork (ASPIRE) based on a bi-level optimization objective. Guided by our theoretical analysis, we disentangle the filter learning objective, which in turn leads to excellent recommendation performance, spectral adaptivity, and training stability in practice. Extensive experiments show our learned filters match the performance of carefully engineered task-specific designs. Furthermore, ASPIRE is equally effective in LLM-powered collaborative filtering. Our findings demonstrate that graph filter learning is viable and generalizable, paving the way for more expressive graph neural networks in collaborative filtering.

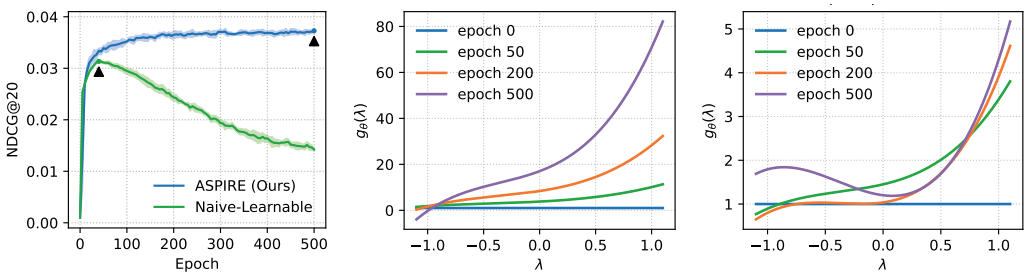
1 INTRODUCTION

Graph Neural Networks (GNNs) (Kipf & Welling, 2016; Bruna et al., 2013) have become a significant paradigm in Collaborative Filtering (CF), where user-item interactions are modeled as bipartite graphs and representation propagation serves as implicit preference aggregation Gilmer et al. (2017); Wang et al. (2019); Ju et al. (2024). From a spectral perspective, many successful models can be interpreted as applying graph filters that emphasize specific frequency components of the interaction graph (Shen et al., 2021; Peng et al., 2022; Aiolli, 2013). This spectral view provides a principled understanding of why low-frequency smoothing improves recommendation accuracy and has inspired a series of filter-based designs (Peng et al., 2024; Liu et al., 2023; Zhu & Zhang, 2024).

Despite this theoretical clarity, a striking gap remains between expectation and practice. Intuitively, learnable spectral filters should generalize well to diverse signal distributions with minimal hyperparameter tuning, mitigate over-smoothing (Liu et al., 2021; Wu et al., 2024; Li et al., 2018), and enable more expressive graph neural networks. However, empirical evidence consistently suggests otherwise. Most existing spectral collaborative filtering methods rely on manually designed filters with carefully tuned hyperparameters, while learnable filters often exhibit unstable training behavior and inferior performance (He et al., 2025) (see Figure 1(a)). This discrepancy raises a fundamental question: *Why do learnable spectral filters fail when trained with standard recommendation objectives?* To better understand this phenomenon, we examine how the learned spectral response evolves during training. Figure 1(b) plots the filter curves at different training stages for a naive learnable spectral filter. We observe a clear and consistent phenomenon termed *Low-Frequency Explosion*: the magnitude assigned to low-frequency components grows continuously throughout training and never converges.

In this work, we show that this behavior is not accidental. We first prove that the frequency bias of common recommendation objectives breaks *Filter Stability* of learnable spectral modules. It over-allocates optimization emphasis to low-frequency components, and systematically under-optimizes high-frequency signals, even when such signals are informative for recommendation (Guo et al., 2023; Choi et al., 2023; Liu et al., 2025). Consequently, directly learning spectral filters under conventional supervision leads to spectral imbalance and the collapse of *Metric Stability*.

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.



(a) NDCG@20 Trajectory (b) Naive-Learnable Filter Curve (c) ASPIRE (Ours) Filter Curve

Figure 1: Comparison between ASPIRE and Naive-Learnable (a simple learnable polynomial filter) on the Baby dataset. In (a), ▲ marks the epoch achieving the best metric performance. In (b) and (c), we plot the learned filter $g_\theta(\lambda)$ in the spectral domain, whose effect is described by equation 1. Note that the eigenvalues λ_k of $\tilde{A}$ range from -1 (high-frequency) to 1 (low-frequency). Refer to Appendix D.1 for supplementary results.

Building on the above theoretical analysis and insights from prior spectral analysis methods (Guo et al., 2023; He et al., 2025), we propose ASPIRE, a bi-level optimization framework that decouples filter learning and directly optimizes for the fundamental goal of graph filters: predicting oracle interactions. Our novel optimization objective completely alters the spectral learning dynamics. As shown in Figure 1(c), the filter learned under ASPIRE maintains a balanced spectral profile and is able to preserve informative high-frequency components. In extensive experiments across multiple graph structures and evaluation settings, ASPIRE learns spectral filters whose performance matches carefully engineered, task-specific designs, while exhibiting markedly improved training stability. Apart from validations on diverse graph scenarios, we further conduct experiments on LLM-powered CF task with two distinct adapter designs, consistently achieving outstanding performance and strong stability.

Our contributions are summarized as follows:

- We identify an intrinsic low-frequency bias in conventional recommendation objectives and provide a theoretical proof that aligns with the empirical observations in the spectral domain, thereby explaining why naive learnable filters collapse during training.
- We present a adaptive graph filter learning framework ASPIRE that separates filter optimization from the traditional objective, avoiding low-frequency explosion.
- We empirically demonstrate that our approach achieves comparable performance with handcrafted filters while significantly improving filter and metric stability.

2 Preliminaries

2.1 Graph Collaborative Filtering

Collaborative filtering is a fundamental paradigm in recommender systems, which aims to infer user preferences from historical user-item interaction data and generate personalized recommendation lists. In the implicit feedback setting, the interaction data is typically represented as a sparse binary matrix $R \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{I}|}$, where $r_{ui} \in \{0, 1\}$ indicates whether user u has interacted with item i. The interaction data can also be viewed as a bipartite graph $\mathcal{G} = (\mathcal{E}, \mathcal{N})$ with $\mathcal{N} = \mathcal{U} \cup \mathcal{I}$, and its adjacency matrix A can be derived from R. In graph collaborative filtering, users and items are often associated with learnable embeddings $E \in \mathbb{R}^{|\mathcal{N}|\times d}$, where d denotes the embedding dimension. A common practice is to perform message propagation over the symmetrically normalized bipartite adjacency matrix $\tilde{A} = D^{-1/2}AD^{-1/2}$. For personalized ranking optimization, Bayesian Personalized Ranking (BPR) loss (Rendle et al., 2009) is widely adopted in mainstream CF methods.

2.2 Spectral Graph Collaborative Filtering

Generally, spectral collaborative filtering (Shen et al., 2021; Qin et al., 2025) methods use eigendecomposition to transform embeddings into the spectral domain, perform filtering on graph signals, and then transform them back to the original space. Specifically, given a normalized bipartite adjacency matrix $\tilde{A}$, spectral filtering can be written as

$$H = U g(\Lambda) U^T E = U \operatorname{diag}\big(g(\lambda_1), \ldots, g(\lambda_{|\mathcal{U}|+|\mathcal{I}|})\big)\, U^T E, \tag{1}$$

where U and Λ are eigenvectors and eigenvalues of $\tilde{A}$ and $g(\cdot)$ denotes a spectral filter applied to each frequency. As a classic graph convolution-based CF method, LightGCN (He et al., 2020) can also be interpreted from a spectral perspective as a simple uniform polynomial graph filter with $g(\lambda) = \frac{1}{L+1}\sum_{l=0}^{L} \lambda^l$:

$$H = \frac{1}{L+1}\sum_{l=0}^{L} \tilde{A}^l E = U\left(\frac{1}{L+1}\sum_{l=0}^{L}\Lambda^l\right) U^T E. \tag{2}$$

While some methods use the scoring matrix as the filter input (Liu et al., 2023; Qin et al., 2025) and rely on matrix decomposition (Peng et al., 2024; Kim et al., 2025), our work targets graph embedding- and graph convolution-based filtering mechanisms.

3 Analysis and Methodology

3.1 Understanding Low-Frequency Explosion from a Loss Perspective

To explain the phenomenon observed in Figure 1(b), we analyze the recommendation objective from a spectral perspective. Under the Perfect Uniformity condition defined by Wang et al. (2022), and assuming that the mean absolute gradient deviation δ defined in equation 7 is negligible, the following theorem characterizes how the BPR objective shapes individual spectral responses near convergence, providing insight into the observed training instability.

Theorem 1. *Consider a spectral GNN near an optimal solution of the BPR loss, with the underlying graph $\phi(A) = \frac{1}{2|\mathcal{E}|}A = U\Lambda U^\top$. Let $\rho \in (0,1)$ be the edge density of A, and define $\tilde{E} = U^\top E$, with $\tilde{E}_k$ denoting the k-th row of $\tilde{E}$. For any eigenvalue λ_k satisfying $|\lambda_k g(\lambda_k)|\,\|\tilde{E}_k\|^2 > 5(1-\rho)\delta$, the gradient of the BPR loss with respect to the spectral response $g(\lambda_k)$ satisfies*

$$\begin{cases} \dfrac{\partial \ell_{\mathrm{BPR}}}{\partial g(\lambda_k)} < -\dfrac{1}{5(1-\rho)}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 + \delta < 0, & \text{if } \lambda_k g(\lambda_k) > 0, \\ \dfrac{\partial \ell_{\mathrm{BPR}}}{\partial g(\lambda_k)} > -\dfrac{1}{5(1-\rho)}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 - \delta > 0, & \text{if } \lambda_k g(\lambda_k) < 0. \end{cases}$$

The proof of Theorem 1 is provided in Appendix B.1. The proof proceeds by showing that positive sample pairs consistently promote low-frequency components, while negative pairs, under the Perfect Uniformity condition, exhibit a similar tendency. Specifically, this theorem reveals that even when a model is close to an optimal solution, it still exerts a significant gradient on the filter $g_\theta(\Lambda)$, which undermines filter stability. Moreover, analyzing this bound helps identify the key factors driving the filter toward low-frequency explosion.

From equation 8 in the Appendix, when user and item embeddings are symmetrically filtered and predictions are computed via inner product, positive and negative $g(\lambda_k)$ are functionally equivalent, with only their absolute values impacting filtering. Without loss of generality, we consider the common case where $g(\lambda_k) > 0$. Under this condition, the optimization dynamics are directly modulated by the eigenvalue λ_k. When $\lambda_k > 0$, the gradient drives $g(\lambda_k)$ to increase, whereas when $\lambda_k < 0$, the gradient pushes $g(\lambda_k)$ toward zero. Moreover, the gradient magnitude contains a multiplicative factor λ_k. When $g(\lambda_k)$ and $\|\tilde{E}_k\|^2$ are of comparable scale across different k, spectral components associated with larger $|\lambda_k|$ tend to receive stronger updates. As optimization proceeds, this effect may further amplify $g(\lambda_k)$ in the low frequency area, leading to increasingly larger gradients and thus introducing a severe spectral bias during training.

Table 1: Comparison of recommendation performance and computational overhead for upper-level gradient approximations (memory in MB). 'Avg-P' refers to the baseline filter defined in equation 2. 'Ex' and 'Im' denote the explicit and implicit term, respectively. Refer to Appendix D.6 for supplementary results.

Method	Baby				Electronics			
	R@20	N@20	(Extra) Time	(Extra) Mem	R@20	N@20	(Extra) Time	(Extra) Mem
Avg-P	0.0851	0.0372	10m09s	488	0.0579	0.0272	120m16s	1970
Ex&Im	0.0870	0.0380	+ 5m11s	+ 148	0.0636	0.0306	+ 161m14s	+ 1082
Ex	0.0871	0.0380	+ 0m35s	+ 0	0.0635	0.0306	+ 19m22s	+ 66

Therefore, learnable graph filters suffer from an inherent strong low-pass bias and over-regularization, leading to severe high-frequency suppression and a prominent low-frequency explosion. This lopsided frequency preference precisely aligns with the trend in Figure 1(b). Notably, the same phenomenon also occurs under the cross-entropy (CE) loss, with theoretical analysis provided in Appendix B.2 and experimental details elaborated in Appendix D.2.

3.2 Adaptive Graph Filter Learning

As revealed in the previous section, conventional recommendation objectives are inherently misaligned with the purpose of graph filters. Building on insights from prior spectral analysis methods, we propose a bi-level optimization scheme to decouple filter learning.

Pre-filter Normalization. To prevent dominant high-norm embeddings from overwhelming the filter learning process, we first perform row-wise ℓ_2 normalization on the embedding matrix $\bar{E}_i = E_i / \|E_i\|_2$, where E_i denotes the i-th row of E. This operation removes scale variations across nodes before spectral filtering. It accelerates optimization convergence, boosts recommendation performance, and greatly enhances training stability, as detailed in Appendix D.5.

Adaptive Filtering. Given an arbitrary graph $\mathcal{G}$ with its graph matrix factorized as $U\Lambda U^\top$, the adaptive filtering procedure on the input embeddings $\bar{E}$ can be written as

$$H = \mathcal{F}\big(U g_\theta(\Lambda) U^\top \bar{E}\big), \quad g_\theta(\Lambda) = \sum_{l=0}^{L} \theta_l \Lambda^l, \tag{3}$$

where $g_\theta(\Lambda)$ denotes a learnable spectral response function with learnable coefficients $\theta = \{\theta_l\}_{l=0}^{L}$, and $\mathcal{F}$ represents a post-filter operator that maps the filtered embeddings to the final representation space (*e.g.*, multi-branch fusion). We initialize the filter as a full-pass filter with $\theta_0 = 1$ and $\theta_{1\dots L} = 0$, which corresponds to an identity transformation and provides a general starting point that allows the model to learn an appropriate graph filter from data.

Bi-level Optimization Formulation. Previous spectral analysis methods, either from a matrix transformation perspective (Guo et al., 2023) or a signal importance perspective (He et al., 2025), share the same intuition that an effective filter should construct oracle interactions from prior knowledge. Based on this insight, we reformulate the supervision signal and propose the following training objective:

$$\begin{aligned} \theta^* &= \arg\min_{\theta} \mathcal{L}_{\text{valid}}\left(\theta, E^*(\theta)\right), \\ \text{s.t.} \quad E^*(\theta) &= \arg\min_{E} \mathcal{L}_{\text{train}}(\theta, E). \end{aligned} \tag{4}$$

This formulation is well-motivated, as it optimizes the filter parameters θ by minimizing the validation loss, which serves as an oracle with respect to the remaining model parameters. In general CF tasks, these parameters correspond to the embedding E.

Bi-level Optimization Solution. Given that the filter $g_\theta(\Lambda)$ contains only a small number of parameters, we sample a subset of the validation data, which is considerably smaller than the training set, to construct an auxiliary set for upper-level optimization (see Appendix D.8). Inspired

by Yin et al. (2024), we employ a computationally efficient scheme that interleaves optimization across levels, executing a single upper-level update after T steps of lower-level updates. The lower-level optimization can be readily implemented via gradient descent with θ fixed. The upper-level optimization involves two distinct sources of gradients: an explicit gradient from the filter parameter θ, and an implicit gradient within E^* inherited from the lower-level optimization.

$$\nabla_\theta \mathcal{L}_{\text{valid}}\left(\theta, E^*(\theta)\right) = \underbrace{\nabla_\theta \mathcal{L}_{\text{valid}}}_{\text{Explicit Term}} \underbrace{-\nabla_E \mathcal{L}_{\text{valid}} \cdot \left(\nabla_E^2 \mathcal{L}_{\text{train}}\right)^{-1} \cdot \nabla_\theta \nabla_E \mathcal{L}_{\text{train}}}_{\text{Implicit Term}} \tag{5}$$

Our implicit differentiation follows the methods introduced by Lorraine et al. (2020); Navon et al. (2020). Inspired by prior first-order approximations in bi-level optimization and meta-learning (Finn et al., 2017; Liu et al., 2019; Kwon et al., 2023), we empirically examine the effect of the implicit term in Table 1. Results show that the implicit term contributes negligibly to this task, and we therefore omit it, leading to substantial reductions in both computational time and memory overhead. We provide a thorough comparison of different approximation techniques in Appendix D.6.

4 Empirical Evaluation

To assess whether the learned graph filter exhibits desirable behaviors, we evaluate each method from three complementary perspectives. We first briefly introduce experimental setups, with details in Appendix C. All results in this paper are averaged over five runs.

Scenarios. To evaluate our method across different graph structures and model architectures, we consider three settings with different graphs $\mathcal{G}$ and post-filter operators $\mathcal{F}(\cdot)$: *Homogeneous*, *Heterogeneous*, and *Dual-Branch*. In the Homogeneous setting, we use the common interaction graph for $\mathcal{G}$ and set $\mathcal{F}(X) = X$. In the Heterogeneous setting, we adopt the side information augmented graph A_+ proposed by He et al. (2025), while keeping $\mathcal{F}(X) = X$. In the Dual-Branch setting, inspired by Zhou & Shen (2023), we employ both the interaction graph $\tilde{A}$ and the side information graph S, apply filtering to each branch separately, and fuse the resulting representations via average pooling, where $\mathcal{F}(\cdot)$ denotes this average pooling operation.

Datasets. We consider two types of side information: multimodal relationships and social connections. Our preprocessing follows Zhou (2023), while graph construction follows He et al. (2025). For multimodal datasets, we use two widely used datasets, Amazon Baby and Electronics, with 5-core filtering. The original modality data of each item, including both textual and visual modalities, are encoded into dense vectors using pretrained encoders. These representations are further transformed into a sparse item-item similarity graph via k-nearest-neighbors, where k is empirically set to 10. For social datasets, we also use two real-world datasets, Ciao and LastFM, with 3-core filtering. The social connections are symmetrized to construct a user-user graph. For both scenarios, we split the data into training, validation, and test sets with a ratio of 8:1:1.

Baselines. We consider three conventional filters that are commonly adopted as components in the aforementioned scenarios: Layer-0 (L-0), Layer-n (L-n), and Average-Pooling (Avg-P). Avg-P is especially popular, as it incorporates multi-order information from each layer (He et al., 2020). We further implement two manually designed filters motivated by empirical observations in specific architectures: Jacobi-Basis (Jacobi) and Linear-Correction (Lin-C). Jacobi is initially designed under Homogenerous setting (Guo et al., 2023), while Lin-C is designed for side-information augmented Heterogeneous graph (He et al., 2025). Alongside the non-learnable but tunable filters above, we include Naive-Learnable (Naive-L) which simply adopts a learnable polynomial filter without extra designs. The corresponding spectral response function of each method is listed in Table 2. Appendix C.2 presents representative models built with the above filters.

4.1 Recommendation Performance

We report the standard top-N recommendation metrics Recall@N and NDCG@N, where $N \in \{10, 20\}$. **Since many baselines rely on carefully designed filters with extensively tuned hyperparameters, our goal is not necessarily to surpass these handcrafted filters in every case, but to achieve comparable performance while learning the filter adaptively.**

Table 2: Baseline filters functions.

	$g(\lambda)$
L-0	1
L-n	λ^n
Avg-P	$\frac{1}{L+1}\sum_{l=0}^{L}\lambda^l$
Jacobi	$\sum_{l=0}^{L}\beta_l \text{Jacobi}_l(\lambda)$
Lin-C	$\frac{1}{L+1}\sum_{l=0}^{L}\text{linear}(\lambda)^l$
Naive-L	$\sum_{l=0}^{L}\theta_l\lambda^l$

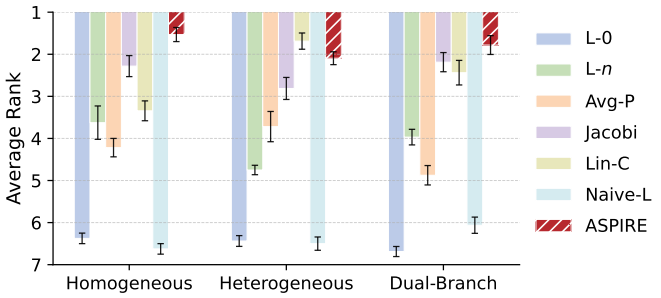


Figure 2: Average rank (mean ± s.e.m.) of each filter across scenarios, computed from the results in Table 3.

Table 3: Performance comparison between ASPIRE and six baseline filter settings. Δ Avg-P denotes the percentage improvement over Avg-P. The leading zero before decimal points is omitted.

	Method	Baby				Electronics				Ciao				LastFM			
		R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20
Homogeneous	L-0	.0417	.0638	.0227	.0284	.0372	.0557	.0208	.0256	.0450	.0737	.0269	.0353	.2357	.3220	.2165	.2513
	L-n	.0536	.0831	.0289	.0365	.0407	.0600	.0236	.0286	.0498	.0784	.0315	.0398	.2450	.3316	.2245	.2595
	Avg-P	.0543	.0851	.0293	.0372	.0393	.0579	.0224	.0272	.0473	.0770	.0295	.0381	.2527	.3430	.2326	.2690
	Jacobi	.0566	.0873	.0306	.0385	.0425	.0618	.0244	.0294	.0489	.0783	.0310	.0395	.2550	.3425	.2348	.2700
	Lin-C	.0542	.0850	.0292	.0372	.0394	.0585	.0225	.0274	.0484	.0786	.0303	.0390	.2558	.3454	.2352	.2712
	Naive-L	.0450	.0690	.0246	.0307	.0312	.0463	.0174	.0213	.0445	.0678	.0283	.0352	.2337	.3241	.2131	.2496
	ASPIRE	**.0550**	**.0871**	**.0297**	**.0380**	**.0440**	**.0635**	**.0255**	**.0306**	.0489	**.0796**	.0309	.0397	**.2565**	**.3475**	**.2366**	**.2733**
	Δ Avg-P	1.30%	2.38%	1.53%	2.19%	11.9%	9.72%	13.9%	12.4%	3.34%	3.44%	4.65%	4.25%	1.49%	1.30%	1.73%	1.58%
Heterogeneous	L-0	.0417	.0638	.0227	.0284	.0372	.0557	.0208	.0256	.0450	.0737	.0269	.0353	.2357	.3220	.2165	.2513
	L-n	.0576	.0922	.0312	.0401	.0432	.0638	.0242	.0295	.0517	.0802	.0326	.0409	.2435	.3315	.2238	.2592
	Avg-P	.0627	.0990	.0337	.0430	.0456	.0675	.0257	.0313	.0524	.0808	.0327	.0410	.2362	.3257	.2174	.2536
	Jacobi	.0654	.0996	.0358	.0446	.0436	.0639	.0248	.0300	.0554	.0849	.0347	.0433	.2541	.3440	.2346	.2710
	Lin-C	.0666	.1044	.0361	.0458	.0452	.0677	.0254	.0312	.0547	.0861	.0340	.0431	.2588	.3502	.2406	.2772
	Naive-L	.0538	.0833	.0293	.0369	.0330	.0486	.0187	.0228	.0442	.0708	.0282	.0360	.2332	.3268	.2128	.2506
	ASPIRE	**.0666**	**.1027**	**.0361**	**.0454**	.0454	**.0679**	.0255	.0313	.0540	.0844	.0338	.0426	**.2570**	**.3476**	**.2389**	**.2755**
	Δ Avg-P	6.22%	3.82%	7.17%	5.55%	-0.50%	0.60%	-0.79%	-0.13%	3.08%	4.57%	3.22%	3.99%	8.79%	6.75%	9.88%	8.61%
Dual-Branch	L-0	.0417	.0638	.0227	.0284	.0372	.0557	.0208	.0256	.0450	.0737	.0269	.0353	.2357	.3220	.2165	.2513
	L-n	.0640	.0985	.0345	.0433	.0447	.0665	.0251	.0307	.0529	.0827	.0328	.0414	.2423	.3286	.2226	.2573
	Avg-P	.0630	.0967	.0343	.0429	.0465	.0695	.0260	.0319	.0519	.0807	.0326	.0410	.2360	.3247	.2171	.2529
	Jacobi	.0670	.1029	.0358	.0450	.0473	.0698	.0267	.0325	.0533	.0826	.0330	.0416	.2566	.3471	.2361	.2726
	Lin-C	.0642	.0974	.0347	.0432	.0465	.0695	.0260	.0319	.0547	.0830	.0339	.0421	.2578	.3490	.2393	.2761
	Naive-L	.0557	.0866	.0305	.0384	.0343	.0508	.0189	.0231	.0453	.0702	.0292	.0365	.2369	.3258	.2177	.2536
	ASPIRE	**.0659**	**.1021**	**.0354**	**.0447**	**.0479**	**.0710**	**.0270**	**.0330**	.0525	**.0830**	.0328	**.0417**	**.2580**	**.3502**	**.2390**	**.2761**
	Δ Avg-P	4.64%	5.61%	3.29%	4.08%	2.92%	2.12%	3.89%	3.29%	1.07%	2.77%	0.47%	1.72%	9.30%	7.84%	10.1%	9.16%

Notations. Table 3 demonstrates the overall performance comparison. For a more detailed evaluation, we categorize the results into four levels. Level 0 indicates results that outperform the initialization filter (*i.e.*, L-0), which is satisfied by all reported results. Level 1 indicates results that outperform all commonly used filters, marked with an underline. Level 2 indicates results whose performance lies between the two manually designed filters, marked in **bold**. Level 3 indicates results that outperform all manually designed filters, marked with both **bold and underline**. Note that some results may satisfy a higher level without satisfying Level 1; in such cases, we mark them as Level 0.

Observations. Across different scenarios and datasets, the manually designed filters - Lin-C and Jacobi - outperform empirical ones in most cases, likely due to their spectral insights and careful hyperparameter tuning. For ASPIRE, 75% of the results reach at least Level 2, *i.e.*, comparable to the two manually designed filters Furthermore, 33.3% of the results surpass both of these two manually designed filters. To intuitively compare the overall performance of all methods across scenarios, we plot their average rankings in Figure 2. As shown in this figure, Jacobi and Lin-C achieve distinct advantages in their respective tailored scenarios (*i.e.*, Homogeneous and Heterogeneous, respectively). In contrast, ASPIRE delivers competitive and relatively stable performance across all settings, even outperforming both methods overall in the Homogeneous and Dual-Branch scenarios. This clearly validates that our proposed learnable filter possesses stronger generalization ability. By comparison, Naive-L performs significantly worse than ASPIRE due to the low-frequency explosion. Compared with Avg-P, which is commonly used as a default component in many works, ASPIRE

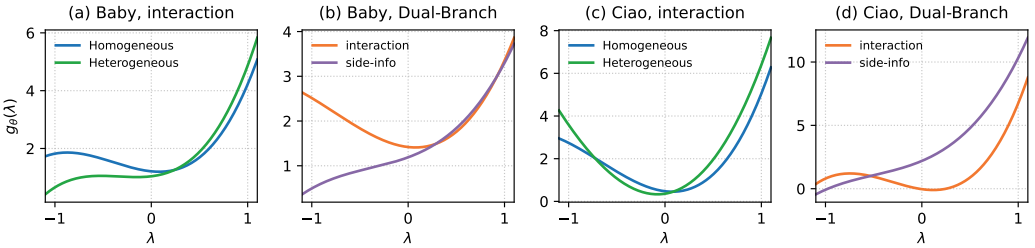


Figure 3: Learned filters across different graph settings. (a) and (c) present the filters for the interaction graph under the Homogeneous (standard) and Heterogeneous (augmented) settings, while (b) and (d) present the filters for the standard interaction graph and the side information graph in the Dual-Branch setting.

Table 4: Initialization Robustness. 'E@x%' indicates the initial epoch achieving x% of the final N@20 on Baby. Evaluation is conducted every 5 epochs. Refer to Appendix D.4 for supplementary results.

Initialization	$g_\theta^{init}(\lambda)$	Convergence Speed		Final Perf - Baby		Final Perf - Elec	
		E@80%	E@95%	R@20	N@20	R@20	N@20
full-pass	1	20	105	0.0871	0.0380	0.0635	0.0306
zero-crossing	λ	5	70	0.0865	0.0378	0.0635	0.0306
low-pass	$\frac{1}{4}\sum_{l=0}^{3}\lambda^l$	5	40	0.0884	0.0383	0.0634	0.0305
mid-pass	$1-\lambda^2$	100	145	0.0873	0.0381	0.0636	0.0307
high-pass	$1-2\lambda+\lambda^2$	105	175	0.0870	0.0375	0.0630	0.0304
Baseline: Average-Pooling		60	190	0.0851	0.0372	0.0579	0.0272

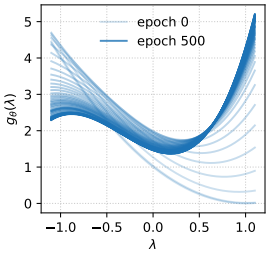


Figure 4: Filter evolution of high-pass initialization.

yields positive gains on the vast majority of metrics in various scenarios, with improvements up to 13.9%. This result shows that introducing ASPIRE as the graph filtering mechanism can generally enhance recommendation performance. Notably, unlike manually designed filters, ASPIRE obviates the need for prior spectral analysis of the graph while achieving superior generalizability.

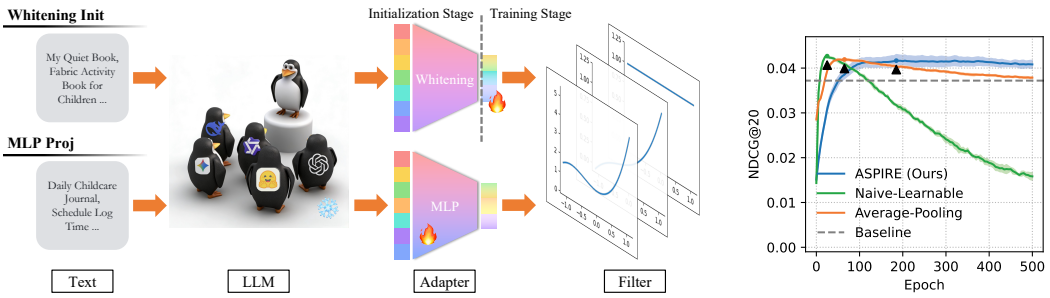
4.2 Spectral Adaptivity

A desirable filter should adapt to spectral variations across different graphs (Luo et al., 2024; Rabiah & McAuley, 2025). We evaluate this property by examining whether the learned filters differ across datasets with distinct graph structures, and whether these differences are consistent with observations reported in prior studies. The learned filters are visualized in Figure 3.

• First, the learned filters vary noticeably across different graph types. For example, unlike other graph types, the learned filter on the side-information graph (purple line) exhibits a clear low-pass tendency, which may explain why traditional low-pass filters perform well in similar scenarios (Wang et al., 2023a; Yu et al., 2025). • Second, prior studies suggest that effective recommendation models should properly emphasize both low- and high-frequency components of the interaction graph spectrum (Kim et al., 2025; Liu et al., 2025). In Figure 3, the blue and orange curves both exhibiting responses on low- and high-frequency regions. Nevertheless, subtle differences are still observed, further supporting our first observation. • Additionally, by inspecting the oracle spectrum reported by He et al. (2025), we observe that when the interaction graph is augmented with side information, the relative importance of low-frequency components increases compared to high-frequency ones. Comparing the low-frequency regions in (a) and (c), the learned filters exhibit a similar trend (noting that the effective high-frequency spectrum shifts rightward away from $\lambda = -1$).

4.3 Training Stability

Finally, we examine the Metric Stability and Filter Stability during training. **A reliable method should exhibit stable metric trajectories, smooth parameter evolution, and consistent convergence under different initializations.**



(a) Illustrative Overview (b) Case Study

Figure 5: Two benchmarked LLM-assisted architectures and metric stability analysis of MiniLM-L6 with Whitening Init adapter.

Table 5: Performance comparison of LLM-assisted CF on Baby. Best results are in **bold**. Refer to Appendix D.3 for details of the LMs and supplementary results.

Adapter	Filter	Metric Sta.	Filter Sta.	Filter Learn.	MiniLM-L6		Qwen2.5-7B		SFR-Emb	
					R@20	N@20	R@20	N@20	R@20	N@20
Whitening Init	Avg-P	✗	✓	✗	0.0928	0.0407	0.0931	0.0407	0.0961	0.0424
	Naive-L	✗	✗	✓	0.0927	0.0407	0.0929	0.0403	0.0968	**0.0430**
	ASPIRE	✓	✓	✓	**0.0949**	**0.0415**	**0.0952**	**0.0418**	**0.0980**	**0.0430**
MLP Proj	Avg-P	✓	✓	✗	0.0831	0.0360	0.0859	0.0374	0.0889	0.0390
	Naive-L	✗	✗	✓	0.0755	0.0325	0.0775	0.0333	0.0801	0.0350
	ASPIRE	✓	✓	✓	**0.0935**	**0.0409**	**0.0928**	**0.0404**	**0.0926**	**0.0404**

[1] Sta. = Stability, Learn. = Learnability.

As illustrated in Figure 1, the performance of ASPIRE maintains a steady upward trend throughout training, in contrast to the sharp performance drop observed in the naive learnable model. Meanwhile, the learned filter evolves smoothly and maintains a reasonable scale during the optimization process. More frequent snapshots of the spectral response function are provided in Appendix D.1.

We further investigate the performance under different filter initializations in Table 4. With the full-pass, zero-crossing, and low-pass initialization, the model converges faster and consistently outperforms the baseline Avg-P. For mid-pass and high-pass initializations, which initially suppress low-frequency signals, the model experiences a longer warm-up stage but subsequently catches up and converges faster than Avg-P. Although NDCG@20 of the high-pass initialization is not significantly better than Avg-P on Baby, its filter evolution (Figure 4) shows that the learned filter eventually converges to the same solution obtained from the full-pass initialization. In fact, across all the initialization settings above, the learned filters converge to essentially the same spectral shape (or its sign-flipped counterpart mirrored across the x-axis, which is functionally equivalent (Liu et al., 2025)), with only slight differences in the resulting metrics.

5 Extending ASPIRE to LLM-Powered Collaborative Filtering

Large Language Models (LLMs) have been widely adopted in recommender systems. Unlike Section 4, which adopts multimodal information to construct graph structures, this section investigates the effectiveness of ASPIRE when employing LLMs as encoders to deliver powerful semantic representations for CF. We consider two representative adapter designs, as shown in Figure 5(a). Table 5 summarizes the overall performance, while Figure 5(b) illustrates metric stability for a representative case.[1]

In **Whitening Initialization** setting, we reduce the dimensionality of LLM-encoded features to the embedding space via whitening (Su et al., 2021; Huang et al., 2021), which is proved to be

[1]The relative performance between Table 5 (test results) and Figure 5(b) (validation trajectories) may differ slightly, potentially due to overfitting, especially for methods exhibiting unstable training dynamics.

effective in recommendation (Zhang et al., 2024b;a; Shi et al., 2025). We then use the resulting representations as initialization for subsequent training (Zhang et al., 2025b). This design is inspired by STAIR (Xu et al., 2024), which introduces step-wise convolution (a tunable graph filter) to mitigate modality erasure across deep convolutional layers. In contrast, ASPIRE inherently alleviates this issue by enabling adaptive spectral approximation. Interestingly, Avg-P does not exhibit full metric stability in this setting. We observe that its performance gradually deteriorates after around 65 epochs, eventually degrading to the NDCG level achieved by randomly initialized Avg-P. We hypothesize that the degradation is caused by modality forgetting, as discussed in STAIR. Encouragingly, this phenomenon is significantly mitigated by ASPIRE. In addition, we observe that Naive-L achieves relatively strong performance in this scenario, in contrast to Table 3, although low-frequency explosion still persists. We attribute this to the high-quality initialization provided by LLM features, where amplifying low-frequency components in the early stages can already yield competitive predictions. Overall, ASPIRE consistently achieves both metric and filter stability.

In **MLP Projection** setting, we freeze the LLM encoder and project the extracted features into the embedding space using an MLP (Shi et al., 2025). User representations are constructed by applying average pooling over item features, followed by MLP projection. This design is inspired by AlphaRec (Sheng et al., 2025), which adopts a fixed Avg-P filter during the filtering stage. Despite the substantial change in embedding strategy, we observe that the stability patterns across different language models remain consistent with previous observations in this study. Without strong initialization, Naive-L degrades into poor performance. These results highlight the importance of stable and adaptive filter learning in LLM-assisted CF, with ASPIRE offering a robust and generalizable solution.

6 Related Work

Spectral Graph Collaborative Filtering. Spectral methods are widely used in CF due to their interpretability and powerful graph signal processing techniques. LightGCN (He et al., 2020) simplifies NGCF (Wang et al., 2019), and can be interpreted as a low-pass graph filter. More recently, Guo et al. (2023) proposed a spectral analysis paradigm for CF and adopted Jacobi-basis for filtering; Liu et al. (2025) measured signal contributions by comparing model performance under different filter settings. These methods lack generalizability, as they are either restricted to specific architectures or burdened by high computational costs, thus yielding only heuristic filters that require cumbersome hyperparameter tuning. Moreover, as emphasized by He et al. (2025), directly deploying advanced learnable spectral GNN architectures (Wang & Zhang, 2022; Hordan et al., 2025) to CF tasks remains infeasible. Therefore, there is an urgent demand to deeply investigate the inherent failure causes and develop a graph filter adaptive to diverse graph structures and signal distributions.

Collaborative Filtering with Side Information. Existing side-information-based CF methods typically employ empirical graph filters, such as L-n (Zhang et al., 2021; Ong & Khong, 2025) and Avg-P (Lin et al., 2024; Zhou et al., 2025), while recent approaches explore more flexible spectral designs (Xu et al., 2024; He et al., 2025). However, these methods still rely heavily on manually designed filter forms and task-specific hyperparameter tuning, whereas ASPIRE learns adaptive filters under a unified configuration. Detailed discussions are provided in Appendix A.

Bi-level Optimization in Recommendation. Bi-level optimization (BLO) has been introduced into recommendation to model nested dependencies between primary objectives and auxiliary optimization processes (Colson et al., 2007; Wu et al., 2026). Existing studies mainly employ BLO for specific downstream purposes, including feedback denoising (Wang et al., 2023b), tokenizer-recommender alignment (Bai et al., 2026), etc. Different from these applications, we leverage BLO to mitigate the bias inherent in conventional optimization objectives. A broader review is provided in Appendix A.

7 Conclusion

We propose ASPIRE, an adaptive spectral graph collaborative filtering framework that effectively addresses the low-frequency explosion issue and does not rely on prior knowledge of the underlying graph spectrum. It achieves performance comparable to heuristic-based and heavily hyperparameter-dependent methods, while exhibiting stable training behavior.

REFERENCES

Fabio Aiolli. Efficient top-n recommendation for very large scale binary rated datasets. In *ACM Conference on Recommender Systems (RecSys)*, pp. 273–280, 2013.

Yimeng Bai, Chang Liu, Yang Zhang, Dingxian Wang, Frank Yang, Andrew Rabinovich, Wenge Rong, and Fuli Feng. Bi-level optimization for generative recommendation: Bridging tokenization and generation. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, 2026.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *ACM Conference on Recommender Systems (RecSys)*, RecSys 2011, 2011.

Hong Chen, Xin Wang, Ruobing Xie, Yuwei Zhou, and Wenwu Zhu. Cross-domain recommendation with behavioral importance perception. In *ACM Web Conference (WWW)*, pp. 1294–1304, 2023.

Jeongwhan Choi, Seoyoung Hong, Noseong Park, and Sung-Bae Cho. Blurring-sharpening process models for collaborative filtering. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 1096–1106, 2023.

Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of Operations Sesearch*, 153(1):235–256, 2007.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, pp. 1126–1135. PMLR, 2017.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International Conference on Machine Learning (ICML)*, pp. 1263–1272. PMLR, 2017.

Jiayan Guo, Lun Du, Xu Chen, Xiaojun Ma, Qiang Fu, Shi Han, Dongmei Zhang, and Yan Zhang. On manipulating signals of user-item graph: A jacobi polynomial-based graph collaborative filtering. In *ACM International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 602–613, 2023.

Zhiqiang Guo, Jianjun Li, Guohui Li, Chaoyang Wang, Si Shi, and Bin Ruan. Lgmrec: Local and global graph learning for multimodal recommendation. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, pp. 8454–8462, 2024.

Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *ACM Web Conference (WWW)*, pp. 507–517, 2016.

Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 639–648, 2020.

Yunhang He, Cong Xu, Jun Wang, and Wei Zhang. Collaborative filtering meets spectrum shift: Connecting user-item interaction with graph-structured side information. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 920–931, 2025.

Snir Hordan, Maya Bechler-Speicher, Gur Lifshitz, and Nadav Dym. Spectral graph neural networks are incomplete on graphs with a simple spectrum. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Junjie Huang, Duyu Tang, Wanjun Zhong, Shuai Lu, Linjun Shou, Ming Gong, Daxin Jiang, and Nan Duan. Whiteningbert: An easy unsupervised sentence embedding approach. In *Findings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 238–244, 2021.

Wei Jiang, Xinyi Gao, Guandong Xu, Tong Chen, and Hongzhi Yin. Challenging low homophily in social recommendation. In *ACM Web Conference (WWW)*, pp. 3476–3484, 2024.

Clark M Ju, William Shiao, Zhichun Guo, Yanfang Ye, Yozen Liu, Neil Shah, and Tong Zhao. How does message passing improve collaborative filtering? In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 8760–8784, 2024.

Chanwoo Kim, Jinkyu Sung, Yebonn Han, and Joonseok Lee. Graph spectral filtering with chebyshev interpolation for recommendation. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 1964–1974, 2025.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2016.

Jeongyeol Kwon, Dohyun Kwon, Stephen Wright, and Robert D Nowak. A fully first-order method for stochastic bilevel optimization. In *International Conference on Machine Learning (ICML)*, pp. 18083–18113. PMLR, 2023.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

Guojiao Lin, Meng Zhen, Dongjie Wang, Qingqing Long, Yuanchun Zhou, and Meng Xiao. Gume: Graphs and user modalities enhancement for long-tail multimodal recommendation. In *ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1400–1409, 2024.

Fan Liu, Zhiyong Cheng, Lei Zhu, Zan Gao, and Liqiang Nie. Interest-aware message-passing gcn for recommendation. In *ACM Web Conference (WWW)*, pp. 1296–1305, 2021.

Feng Liu, Hao Cang, Huanhuan Yuan, Jiaqing Fan, Yongjing Hao, Fuzhen Zhuang, Guanfeng Liu, and Pengpeng Zhao. How do graph signals affect recommendation: Unveiling the mystery of low and high-frequency graph signals. *arXiv preprint arXiv:2512.15744*, 2025.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations (ICLR)*, 2019.

Jiahao Liu, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, Li Shang, and Ning Gu. Personalized graph signal processing for collaborative filtering. In *ACM Web Conference (WWW)*, pp. 1264–1272, 2023.

Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1540–1552. PMLR, 2020.

Haitong Luo, Xuying Meng, Suhang Wang, Hanyun Cao, Weiyao Zhang, Yequan Wang, and Yujun Zhang. Spectral-based graph neural networks for complementary item recommendation. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, pp. 8868–8876, 2024.

Rui Meng, Ye Liu, Shafiq Rayhan Joty, Caiming Xiong, Yingbo Zhou, and Semih Yavuz. Sfr-embedding-mistral: Enhance text retrieval with transfer learning. Salesforce AI Research Blog, 2024. URL `https://www.salesforce.com/blog/sfr-embedding/`.

Aviv Navon, Idan Achituve, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary learning by implicit differentiation. In *International Conference on Learning Representations (ICLR)*, 2020.

Rongqing Kenneth Ong and Andy WH Khong. Spectrum-based modality representation fusion graph convolutional network for multimodal recommendation. In *ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 773–781, 2025.

Shaowen Peng, Kazunari Sugiyama, and Tsunenori Mine. Svd-gcn: A simplified graph convolution paradigm for recommendation. In *ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1625–1634, 2022.

Shaowen Peng, Xin Liu, Kazunari Sugiyama, and Tsunenori Mine. How powerful is graph filtering for recommendation. In *ACM International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 2388–2399, 2024.

Yifang Qin, Wei Ju, Yiyang Gu, Ziyue Qiao, Zhiping Xiao, and Ming Zhang. Polycf: Towards optimal spectral graph filters for collaborative filtering. *ACM Transactions on Information Systems*, 43(4):1–28, 2025.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL `https://qwenlm.github.io/blog/qwen2.5/`.

Ahmad Bin Rabiah and Julian McAuley. Gsprec: Temporal-aware graph spectral filtering for recommendation. *arXiv preprint arXiv:2505.11552*, 2025.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Uncertainty in Artificial Intelligence (UAI)*, 2009.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pp. 400–407, 1951.

Yifei Shen, Yongji Wu, Yao Zhang, Caihua Shan, Jun Zhang, B Khaled Letaief, and Dongsheng Li. How powerful is graph convolution for recommendation? In *ACM International Conference on Information and Knowledge Management (CIKM)*, pp. 1619–1629, 2021.

Leheng Sheng, An Zhang, Yi Zhang, Yuxin Chen, Xiang Wang, and Tat-Seng Chua. Language representations can be what recommenders need: Findings and potentials. In *International Conference on Learning Representations (ICLR)*, 2025.

Kainan Shi, Peilin Zhou, Ge Wang, Han Ding, and Fei Wang. What matters in llm-based feature extractor for recommender? a systematic analysis of prompts, models, and adaptation. *arXiv preprint arXiv:2509.14979*, 2025.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*, 2021.

J. Tang, H. Gao, and H. Liu. mTrust: Discerning multi-faceted trust in a connected world. In *ACM International Conference on Web Search and Data Mining (WSDM)*, pp. 93–102. ACM, 2012.

Chenyang Wang, Yuanqing Yu, Weizhi Ma, Min Zhang, Chong Chen, Yiqun Liu, and Shaoping Ma. Towards representation alignment and uniformity in collaborative filtering. In *ACM International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 1816–1825, 2022.

Tianle Wang, Lianghao Xia, and Chao Huang. Denoised self-augmented learning for social recommendation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2023a.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 5776–5788, 2020.

Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, SIGIR '19, 2019.

Xiyuan Wang and Muhan Zhang. How powerful are spectral graph neural networks. In *International Conference on Machine Learning (ICML)*, pp. 23341–23362. PMLR, 2022.

Zongwei Wang, Min Gao, Wentao Li, Junliang Yu, Linxin Guo, and Hongzhi Yin. Efficient bi-level optimization for recommendation denoising. In *ACM International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 2502–2511, 2023b.

Longfeng Wu, Yao Zhou, Tong Zeng, Zhimin Peng, Bhanu Pratap Singh Rawat, Lecheng Zheng, Giovanni Seni, and Dawei Zhou. Bi-nas: Towards effective and personalized explanation for recommender systems via bi-level neural architecture search. In *IEEE International Conference on Data Mining (ICDM)*, pp. 1642–1651. IEEE, 2025.

Wei Wu, Chao Wang, Dazhong Shen, Chuan Qin, Liyi Chen, and Hui Xiong. Afdgcf: Adaptive feature de-correlation graph collaborative filtering for recommendations. In *ACM International Conference on Research and Development in Information Retrieval (SIGIR)*, pp. 1242–1252, 2024.

Yuman Wu, Xiaochuan Gong, Jie Hao, and Mingrui Liu. Bilevel optimization with lower-level uniform convexity: Theory and algorithm. In *International Conference on Learning Representations (ICLR)*, 2026.

Cong Xu, Jun Wang, and Wei Zhang. Stablegcn: Decoupling and reconciling information propagation for collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 36 (6):2659–2670, 2023.

Cong Xu, Yunhang He, Jun Wang, and Wei Zhang. Stair: Manipulating collaborative and multimodal information for e-commerce recommendation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

Mingjia Yin, Hao Wang, Wei Guo, Yong Liu, Suojuan Zhang, Sirui Zhao, Defu Lian, and Enhong Chen. Dataset regeneration for sequential recommendation. In *ACM International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 3954–3965, 2024.

Penghang Yu, Zhiyi Tan, Guanming Lu, and Bing-Kun Bao. Mind individual information! principal graph learning for multimedia recommendation. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, pp. 13096–13105, 2025.

Wendi Yu, Zhichao Hou, and Xiaorui Liu. Automated polynomial filter learning for graph neural networks. In *IEEE International Conference on Big Data (BigData)*, pp. 680–689. IEEE, 2024.

Jiaming Zhang, Yuyuan Li, Yiqun Xu, Li Zhang, Xiaohua Feng, Zhifei Ren, and Chaochao Chen. Bifair: A fairness-aware training framework for llm-enhanced recommender systems via bi-level optimization. *arXiv preprint arXiv:2507.04294*, 2025a.

Jinghao Zhang, Yanqiao Zhu, Qiang Liu, Shu Wu, Shuhui Wang, and Liang Wang. Mining latent structures for multimedia recommendation. In *ACM International Conference on Multimedia (MM)*, pp. 3872–3880, 2021.

Lingzi Zhang, Xin Zhou, Zhiwei Zeng, and Zhiqi Shen. Dual-view whitening on pre-trained text embeddings for sequential recommendation. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, pp. 9332–9340, 2024a.

Lingzi Zhang, Xin Zhou, Zhiwei Zeng, and Zhiqi Shen. Are id embeddings necessary? whitening pre-trained text embeddings for effective sequential recommendation. In *International Conference on Data Engineering (ICDE)*, pp. 530–543. IEEE, 2024b.

Weizhi Zhang, Liangwei Yang, Wooseong Yang, Henry Peng Zou, Yuqing Liu, Ke Xu, Sourav Medya, and Philip S Yu. Llminit: A free lunch from large language models for selective initialization of recommendation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP), Industry Track*, pp. 2016–2024, 2025b.

Xin Zhou. Mmrec: Simplifying multimodal recommendation. *arXiv preprint arXiv:2302.03497*, 2023.

Xin Zhou and Zhiqi Shen. A tale of two graphs: Freezing and denoising graph structures for multimodal recommendation. In *ACM International Conference on Multimedia (MM)*, pp. 935–943, 2023.

Xin Zhou, Hongyu Zhou, Yong Liu, Zhiwei Zeng, Chunyan Miao, Pengwei Wang, Yuan You, and Feijun Jiang. Bootstrap latent representations for multi-modal recommendation. In *ACM Web Conference (WWW)*, 2023.

Xin Zhou, Yongjie Wang, and Zhiqi Shen. Cm^3: Calibrating multimodal recommendation. *arXiv preprint arXiv:2508.01226*, 2025.

Zhangchi Zhu and Wei Zhang. Exploring feature-based knowledge distillation for recommender system: A frequency perspective. In *ACM International Conference on Knowledge Discovery & Data Mining (KDD)*, 2024.

CONTENTS

1 Introduction **1**

2 Preliminaries **2**

2.1 Graph Collaborative Filtering 2

2.2 Spectral Graph Collaborative Filtering 3

3 Analysis and Methodology **3**

3.1 Understanding Low-Frequency Explosion from a Loss Perspective 3

3.2 Adaptive Graph Filter Learning 4

4 Empirical Evaluation **5**

4.1 Recommendation Performance 5

4.2 Spectral Adaptivity 7

4.3 Training Stability 7

5 Extending ASPIRE to LLM-Powered Collaborative Filtering **8**

6 Related Work **9**

7 Conclusion **9**

A Additional Related Work **16**

B Proofs **16**

B.1 Proof of Theorem 1 16

B.2 Extension to Cross-Entropy Loss 18

C Experimental Setups **19**

C.1 Datasets 19

C.2 Baselines 19

C.3 Implementation Details 19

D Additional Experimental Results **20**

D.1 Extended Stability Evaluation for Figure 1 20

D.2 Stability Evaluation for CE Loss 20

D.3 Stability Evaluation for Table 5 26

D.4 Extended Sensitivity Analysis for Table 4 26

D.5 Effect of Pre-filter Normalization 29

D.6 Time and Memory Efficiency 29

D.7 Ablation Study 34

D.8 Hyperparameter Sensitivity Analysis 35

E Broader Impact and Limitations **37**

F LLM Usage **37**

A Additional Related Work

Collaborative Filtering with Side Information. In CF, Side information is usually modeled as graph structures and processed by graph encoders. Existing methods typically rely on empirical filters, such as L-n (Zhang et al., 2021; Guo et al., 2024; Ong & Khong, 2025) and Avg-P (Jiang et al., 2024; Lin et al., 2024; Zhou et al., 2025). FREEDOM (Zhou & Shen, 2023) introduces a concise dual-branch framework but overlooks the inherent spectral characteristics of this scenario. STAIR (Xu et al., 2024) focuses on the modality erasure issue and expands the search space of graph filter via step-wise convolution. SSC (He et al., 2025) extends spectral filter design to heterogeneous graphs and devises a correction module to alleviate spectrum shifts. Despite improved spectral-domain flexibility, STAIR and SSC remains experiment-driven with non-unified hyperparameter settings, leading to substantial tuning overhead and poor generalization. In contrast, the adaptive filter proposed in this paper achieves unified hyperparameter configuration, superior spectral adaptivity and stability.

Bi-level Optimization in Recommendation. Bi-level optimization (BLO) (Colson et al., 2007; Wu et al., 2026) can address nested interdependencies between core recommendation objectives and auxiliary tasks through nested optimization. For instance, to denoise implicit feedback, BOD (Wang et al., 2023b) proposes a gradient-matching method for the upper-level optimization to dynamically adjust sample weights. For generative recommendation, BLOGER (Bai et al., 2026) reformulates the task into a BLO framework to bridge the misalignment between the tokenizer and the recommender. Other application scopes of BLO include promoting fairness (Zhang et al., 2025a), providing explanations (Wu et al., 2025), facilitating cross-domain capability (Chen et al., 2023), etc (Yin et al., 2024). In this work, we leverage BLO to avoid the bias inherent in conventional objectives.

B Proofs

B.1 Proof of Theorem 1

Under the Perfect Uniformity assumption defined by Wang et al. (2022), for a nearly optimized model, the representation h_u and h_i uniformly distribute on the surface of the d-dimensional unit sphere. Consider dot product as similarity function of two nodes $s(a,b) = h_a^\top h_b \in [-1,1]$. Thus we have

$$\begin{aligned} 0 &= |\mathcal{N}|^2 \mathbb{E}_{n_1,n_2\sim P_{\mathcal{N}}}\left[s(n_1,n_2)\right] \\ &= \rho|\mathcal{U}||\mathcal{I}|\mathbb{E}_{(u,i^+)\sim P_{\text{pos}}}\left[s(u,i^+)\right] + (1-\rho)|\mathcal{U}||\mathcal{I}|\mathbb{E}_{(u,i^-)\sim P_{\text{neg}}}\left[s(u,i^-)\right] \\ &\quad + |\mathcal{U}|^2\mathbb{E}_{u_1,u_2\sim P_{\mathcal{U}}}\left[s(u_1,u_2)\right] + |\mathcal{I}|^2\mathbb{E}_{i_1,i_2\sim P_{\mathcal{I}}}\left[s(i_1,i_2)\right] \\ &= \rho|\mathcal{U}||\mathcal{I}|\mathbb{E}_{(u,i^+)\sim P_{\text{pos}}}\left[s(u,i^+)\right] + (1-\rho)|\mathcal{U}||\mathcal{I}|\mathbb{E}_{(u,i^-)\sim P_{\text{neg}}}\left[s(u,i^-)\right], \end{aligned} \tag{6}$$

where $P_{\mathcal{N}}$, $P_{\mathcal{U}}$, and $P_{\mathcal{I}}$ are the uniform distributions over $\mathcal{N}$, $\mathcal{U}$, and $\mathcal{I}$, respectively, while P_{pos} and P_{neg} denote the uniform distributions over the positive and negative user-item pairs, respectively. Moreover, $\rho = \frac{|\mathcal{E}|}{|\mathcal{U}||\mathcal{I}|} \in (0,1)$ denotes the edge density of the dataset.

We denote

$$s^+ = s(u,i^+), \quad s^- = s(u,i^-), \quad t = 1-\sigma(s^+-s^-) \in \left[\frac{1}{e^2+1}, \frac{e^2}{e^2+1}\right],$$

$$\nabla^+ = \frac{\partial s^+}{\partial g(\lambda_k)}, \quad \nabla^- = \frac{\partial s^-}{\partial g(\lambda_k)}, \quad \nabla = \nabla^+ - \nabla^-.$$

Taking the partial derivative of both sides of equation 6 with respect to $g(\lambda_k)$, we obtain

$$\rho\mathbb{E}\left[\nabla^+\right] + (1-\rho)\mathbb{E}\left[\nabla^-\right] = \rho\frac{\partial\mathbb{E}\left[s^+\right]}{\partial g(\lambda_k)} + (1-\rho)\frac{\partial\mathbb{E}\left[s^-\right]}{\partial g(\lambda_k)} = 0$$

So we have

$$\mathbb{E}\left[\nabla\right] = \mathbb{E}\left[\nabla^+\right] - \mathbb{E}\left[\nabla^-\right] = \frac{1}{1-\rho}\mathbb{E}\left[\nabla^+\right].$$

The BPR loss is defined as

$$\ell_{\mathrm{BPR}} = -\mathbb{E}_{(u,i^+)\sim P_{\mathrm{pos}}} \mathbb{E}_{(u,i^-)\sim P_{\mathrm{neg}}} \log\left(\sigma(s^+ - s^-)\right).$$

The derivative of the spectral response is

$$\begin{aligned}
\frac{\partial \ell_{\mathrm{BPR}}}{\partial g(\lambda_k)} &= -\mathbb{E}\left[(1-\sigma(s^+ - s^-))\left(\frac{\partial s^+}{\partial g(\lambda_k)} - \frac{\partial s^-}{\partial g(\lambda_k)}\right)\right] \\
&= -\mathbb{E}[t\nabla] \\
&= -\mathbb{E}[t]\mathbb{E}\left[\nabla\right] - \mathbb{E}[t(\nabla - \mathbb{E}\left[\nabla\right])] \\
&= -\frac{1}{1-\rho}\mathbb{E}[t]\mathbb{E}\left[\nabla^+\right] + \epsilon
\end{aligned}$$

where

$$|\epsilon| < \delta = \mathbb{E}\left[|\nabla - \mathbb{E}\left[\nabla\right]|\right] \approx 0. \tag{7}$$

Next we compute $\mathbb{E}\left[\nabla^+\right]$.

Let $w_{ui} = P_{\mathrm{pos}}(u,i)$ be the sampling probability of the positive pair (u,i) from A, and $H \in \mathbb{R}^{|\mathcal{N}|\times d}$ be the matrix stacking all node embeddings. Then

$$\mathbb{E}_{(u,i)\sim P_{\mathrm{pos}}}[s(u,i)] = \sum_{u,i} w_{ui}\, h_u^\top h_i = \mathrm{Tr}(H^\top W H), \quad W = [w_{ui}], \quad \sum_{u,i} w_{ui} = 1.$$

Under uniform edge sampling, we have $W = \frac{1}{2|\mathcal{E}|}A = \phi(A)$. Further substitute the spectral filtering form $H = Ug(\Lambda)U^\top E$, we obtain

$$\begin{aligned}
\mathbb{E}[s^+] &= \mathrm{Tr}\big(E^\top U g(\Lambda) U^\top \phi(A)\, U g(\Lambda) U^\top E\big) \\
&= \mathrm{Tr}\big(E^\top U g(\Lambda) U^\top U \Lambda U^\top\, U g(\Lambda) U^\top E\big) \\
&= \mathrm{Tr}\big(E^\top U g(\Lambda) \Lambda g(\Lambda) U^\top E\big) \\
&= \mathrm{Tr}\big(g(\Lambda)\Lambda g(\Lambda)\, U^\top E E^\top U\big)
\end{aligned}$$

Let $\tilde{E} = U^\top E$, then

$$\begin{aligned}
\mathbb{E}[s^+] &= \mathrm{Tr}\Big(g(\Lambda)\Lambda g(\Lambda)\, \tilde{E}\tilde{E}^\top\Big) \\
&= \sum_k \lambda_k g(\lambda_k)^2 \left(\tilde{E}\tilde{E}^\top\right)_{kk} \\
&= \sum_k \lambda_k g(\lambda_k)^2 \|\tilde{E}_k\|^2.
\end{aligned} \tag{8}$$

where $\tilde{E}_k$ is the k-th row of $\tilde{E}$.

Taking the partial derivative with respect to $g(\lambda_k)$ yields

$$\mathbb{E}\left[\nabla^+\right] = \frac{\partial \mathbb{E}\left[s^+\right]}{\partial g(\lambda_k)} = 2\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2,$$

Thus for $\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 > 5(1-\rho)\delta$,

$$\begin{aligned}
\frac{\partial \ell_{\mathrm{BPR}}}{\partial g(\lambda_k)} &< -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 \mathbb{E}[t] + \delta \\
&\le -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 \frac{1}{e^2+1} + \delta \\
&< -\frac{1}{5(1-\rho)}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 + \delta < 0
\end{aligned}$$

And for $\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 < -5(1-\rho)\delta$,

$$\begin{aligned}\frac{\partial \ell_{\mathrm{BPR}}}{\partial g(\lambda_k)} &> -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2\mathbb{E}[t] - \delta \\ &\geq -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2\frac{1}{e^2+1} - \delta \\ &> -\frac{1}{5(1-\rho)}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 - \delta > 0\end{aligned}$$

This completes the proof.

B.2 Extension to Cross-Entropy Loss

An analogous result holds for the CE loss.

With $M = |\mathcal{I}| - 1$, the CE loss is defined as

$$\ell_{\mathrm{CE}} = -\mathbb{E}_{(u,i^+)\sim P_{\mathrm{pos}}} \log\left(\frac{\exp(s^+)}{\exp(s^+) + \sum_{m=1}^{M}\exp(s_m^-)}\right).$$

We denote

$$p^+ = \frac{\exp(s^+)}{\exp(s^+) + \sum_{m=1}^{M}\exp(s_m^-)},\ \ p_m^- = \frac{\exp(s_m^-)}{\exp(s^+) + \sum_{m=1}^{M}\exp(s_m^-)} \in \left[\frac{1}{Me^2+1}, \frac{e^2}{e^2+M}\right].$$

The derivative of the spectral response is

$$\begin{aligned}\frac{\partial \ell_{\mathrm{CE}}}{\partial g(\lambda_k)} &= -\mathbb{E}\left[(1-p^+)\nabla^+ - \sum_{m=1}^{M} p_m^-\nabla_m^-\right] \\ &= -\mathbb{E}\left[\sum_{m=1}^{M} p_m^-\nabla^+ - \sum_{m=1}^{M} p_m^-\nabla_m^-\right] \\ &= -\sum_{m=1}^{M}\mathbb{E}_{(u,i^+)\sim P_{\mathrm{pos}}}\left[p_m^-\nabla_m\right] \\ &= -M\mathbb{E}_{(u,i^+)\sim P_{\mathrm{pos}},\, m\sim\mathrm{Uniform}\{1,M\}}\left[p_m^-\nabla_m\right] \\ &\approx -M\mathbb{E}_{(u,i^+)\sim P_{\mathrm{pos}},\,(u,i^-)\sim P_{\mathrm{neg}}}\left[p^-\nabla\right] \\ &= -M\mathbb{E}\left[p^-\right]\mathbb{E}\left[\nabla\right] - M\mathbb{E}\left[p^-(\nabla - \mathbb{E}\left[\nabla\right])\right] \\ &= -\frac{M}{1-\rho}\mathbb{E}[p^-]\mathbb{E}\left[\nabla^+\right] + \epsilon'\end{aligned}$$

where $|\epsilon'| \leq \delta' = \frac{Me^2}{e^2+M}\mathbb{E}\left[|\nabla - \mathbb{E}\left[\nabla\right]|\right]$.

$\mathbb{E}\left[\nabla^+\right]$ is consistent with the previous section.

Thus for $\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 > 5(1-\rho)\delta'$,

$$\begin{aligned}\frac{\partial \ell_{\mathrm{CE}}}{\partial g(\lambda_k)} &< -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 M\mathbb{E}[t] + \delta' \\ &\leq -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2\frac{M}{Me^2+1} + \delta' \\ &< -\frac{1}{5(1-\rho)}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 + \delta' \\ &< 0 \quad \text{when } \delta' \text{ is sufficiently small}\end{aligned}$$

Table 6: Dataset statistics. '#U-I': the number of interactions. '#I-I' the number of item-item edges. '#U-U': the number of user-user edges.

	#Users	#Items	#U-I	#I-I	#U-U	Sparsity
Baby	19,445	7,050	160,792	141,000	-	99.88%
Electronics	192,403	63,001	1,689,188	1,260,020	-	99.99%
Ciao	6,804	17,660	139,112	-	106,433	99.88%
LastFM	1,875	4,613	77,465	-	25,182	99.10%

Table 7: Typical models using baseline filters.

Filter	Tunable	Learnable	Models
L-0	✗	✗	MF (Rendle et al., 2009)
L-n	✓	✗	FREEDOM (Zhou & Shen, 2023), LGMRec (Guo et al., 2024), SMORE (Ong & Khong, 2025)
Avg-P	✓	✗	LightGCN (He et al., 2020), BM3 (Zhou et al., 2023), SHaRe (Jiang et al., 2024)
Jacobi	✓	✗	JGCF (Guo et al., 2023), SimGCF (Liu et al., 2025) (a Jacobi-based variant)
Lin-C	✓	✗	SSC (He et al., 2025)
Naive-L	✓	✓	-

And for $\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 < -5(1-\rho)\delta'$,

$$\begin{aligned}\frac{\partial \ell_{\mathrm{CE}}}{\partial g(\lambda_k)} &> -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 M\mathbb{E}[t] - \delta' \\ &\geq -\frac{2}{1-\rho}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 \frac{M}{Me^2+1} - \delta' \\ &> -\frac{1}{5(1-\rho)}\lambda_k g(\lambda_k)\|\tilde{E}_k\|^2 - \delta' \\ &> 0 \quad \text{when } \delta' \text{ is sufficiently small}\end{aligned}$$

However, since $\delta' = \frac{Me^2}{e^2+M}, \delta$, the coefficient $\frac{Me^2}{e^2+M}$ is monotonically increasing in M and converges to e^2 as $M \to \infty$. Therefore, the sufficient condition here becomes slightly more restrictive as M grows.

C Experimental Setups

C.1 Datasets

The two multimodal datasets, Amazon Baby and Electronics[1], were originally published by He & McAuley (2016). Item title, brand, categories and description are encoded via all-MiniLM-L6-v2 (Wang et al., 2020) as textual features. Item thumbnails are encoded with CNN as visual features. Among the two social datasets, Ciao[2] was released by Tang et al. (2012), while LastFM[3] was provided by Cantador et al. (2011). The dataset statistics are presented in Table 6.

C.2 Baselines

We present typical models for each baseline filter in Table 7.

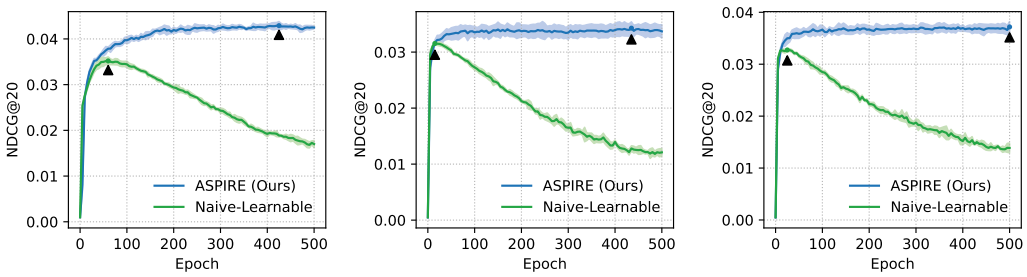
C.3 Implementation Details

We fix the embedding dimension to 64 and the batch size to 2048. For embedding optimization, we adopt the Adam optimizer (Kingma, 2014), while filter parameters are optimized using SGD (Robbins

[1] https://github.com/enoche/MMRec

[2] https://www.cse.msu.edu/~tangjili/datasetcode/truststudy.htm

[3] https://github.com/Sherry-XLL/Social-Datasets/tree/main/lastfm



(a) Baby, Heterogeneous (b) Ciao, Homogeneous (c) Ciao, Heterogeneous

Figure 6: NDCG@20 Trajectory of ASPIRE and Naive-L.

& Monro, 1951). The learning rate of Adam is tuned within $\{1\text{e}{-4}, 5\text{e}{-4}, 1\text{e}{-3}, 5\text{e}{-3}\}$, and that of SGD within $\{1\text{e}{-2}, 5\text{e}{-2}, 1\text{e}{-1}, 5\text{e}{-1}\}$. Weight decay is carefully tuned for all baselines, while it is fixed to 0 for our method. For graph filtering, we set the filter order $L = 3$ by default, except for Layer-n where L is searched over $\{1, 2, 3\}$. Other model-specific hyperparameters are tuned according to the settings recommended in the original papers.

For the lower-level objective $\mathcal{L}_{\text{train}}$ of ASPIRE, we employ the standard BPR loss to ensure fair comparison with existing methods. For the upper-level objective $\mathcal{L}_{\text{valid}}$, we consider two loss variants: BPR and CE. When using the CE loss, we introduce a temperature parameter τ and tune it over $\{0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$. Following our decoupled training strategy, we randomly split 50% of the validation data as an auxiliary set for upper-level optimization, while the remaining 50% is used for hyperparameter tuning. We fix the upper-level update interval T to 5 in our bi-level optimization scheme.

D Additional Experimental Results

D.1 Extended Stability Evaluation for Figure 1

Metric Stability. In Figure 6, we extend Figure 1 (a) to the Heterogeneous scenario and the Ciao dataset, demonstrating that the flaws of Naive-L in metric stability are universally prevalent.

Filter Stability. This section aims to clearly demonstrate how the filters evolve throughout the training process, serving as a more detailed version of Figure 1 (b) and (c). In Figure 7 and Figure 8, we plot one curve every 10 epochs with gradually increasing opacity to illustrate the evolution of the spectral response function. To make the evolution of ASPIRE easier to observe, we further sample 35 curves across training and plot them separately, resulting in Figure 9 and Figure 10.

D.2 Stability Evaluation for CE Loss

Metric Stability. As shown in Figure 11, Naive-L also suffers from a sudden performance drop when cross-entropy (CE) loss is adopted for filter learning. Different from BPR loss, the model converges rapidly to an inferior performance plateau, which further demonstrates the inherent gap in supervision signals.

Filter Stability. As shown in Figure 12, for Naive-L, although high-frequency components are not suppressed as strongly as under the BPR loss, the low-frequency explosion issue still persists, evidenced by the consistently large and increasing $g_\theta(\lambda_{max})$. Since the convergence trend is not straightforward to observe from the above figure, we further visualize the evolution of $|g_\theta(\lambda_{max})|$ over the entire training process in Figure 13. For Naive-L, the magnitude of $|g_\theta(\lambda_{\text{max}})|$ spikes sharply within the initial roughly 20 epochs. It continues to rise rapidly under BPR optimization, while exhibiting a slower growth rate with the CE loss, and both approaches maintain an approximately linear upward trend. Even with the milder growth brought by CE loss, its persistent upward trend remains substantial in contrast to the near-stagnant convergence behavior of ASPIRE, thereby proving

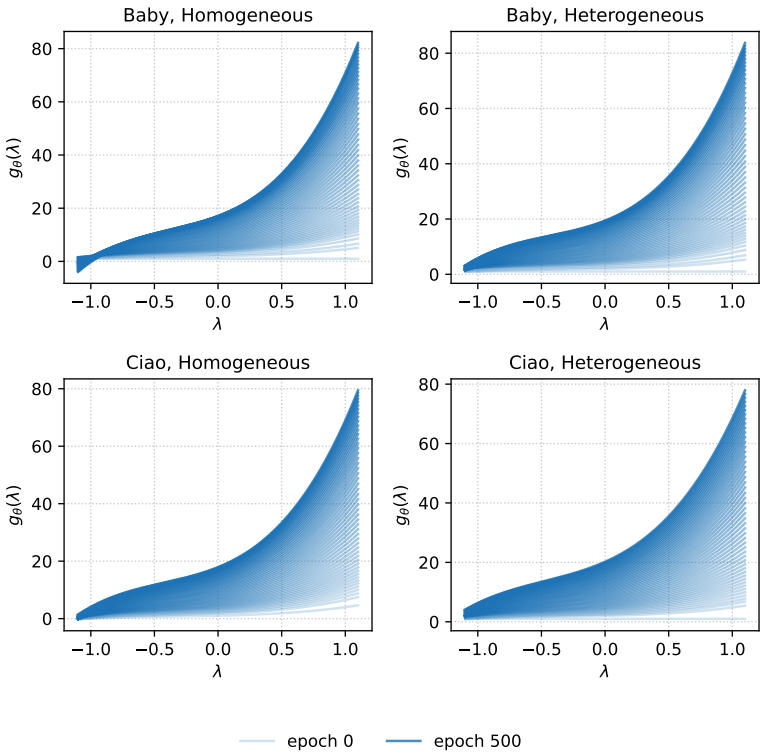


Figure 7: Filter evolution of Naive-L.

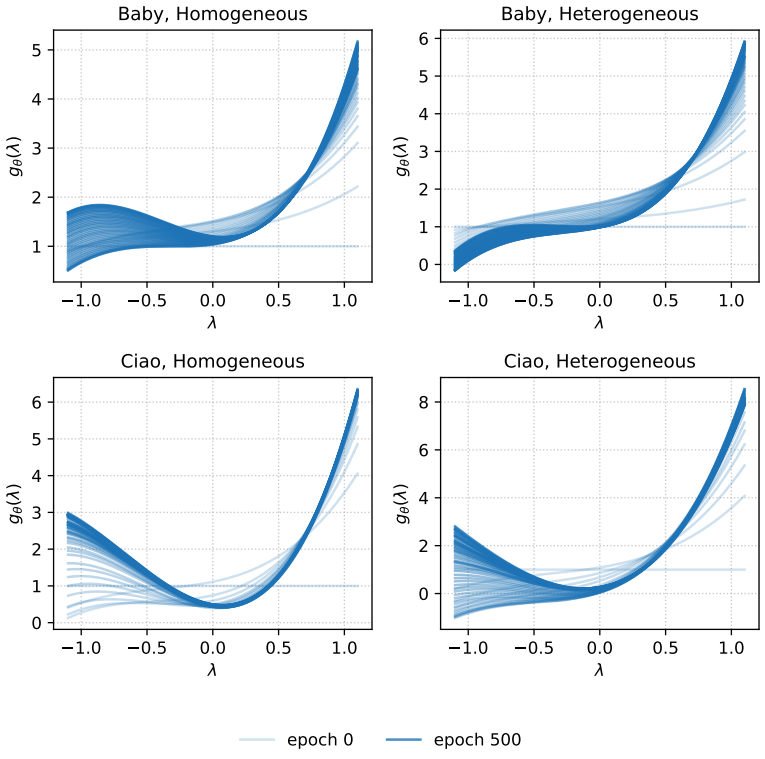


Figure 8: Filter evolution of ASPIRE.

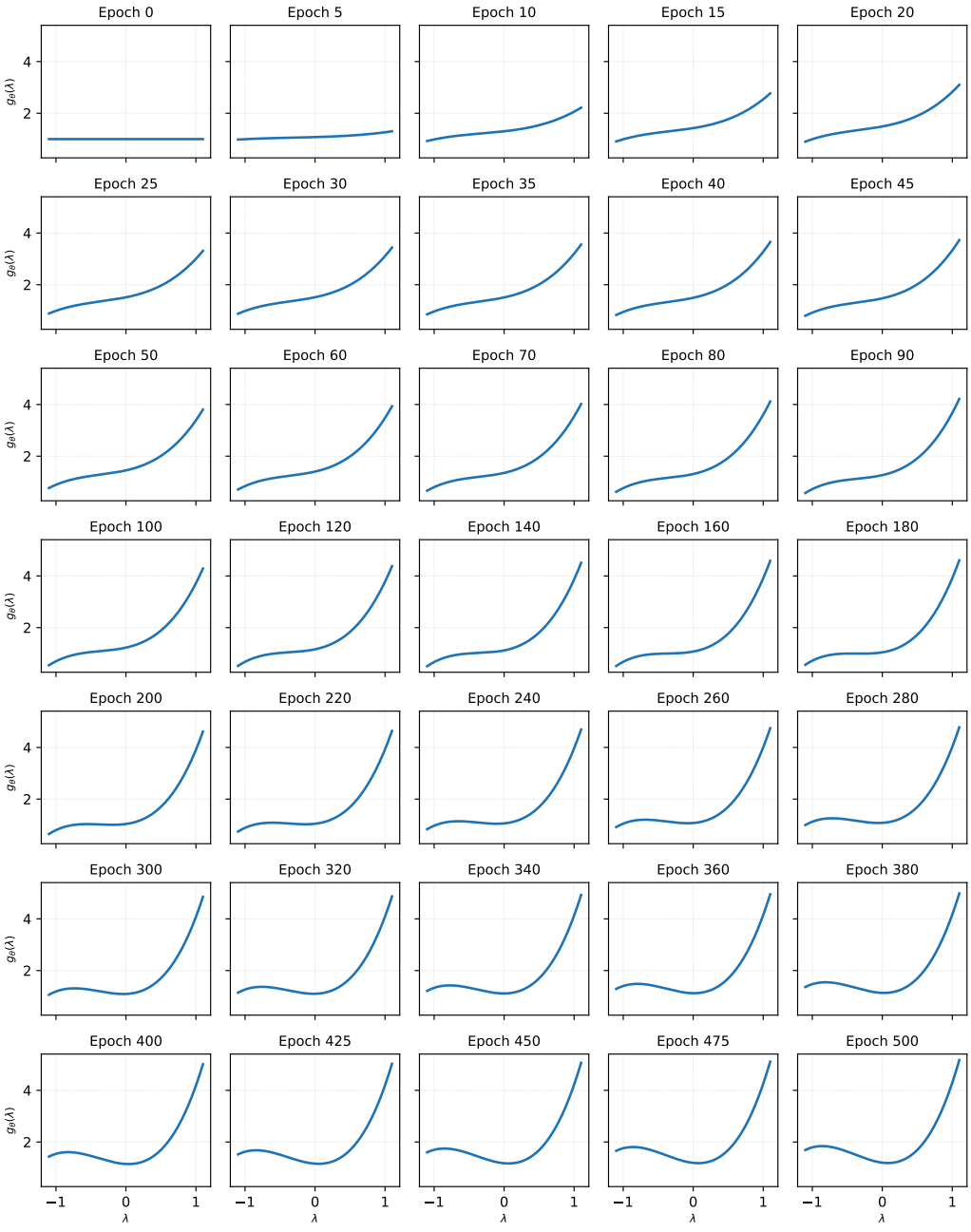


Figure 9: Filter evolution of ASPIRE under Homogeneous scenario on Baby.

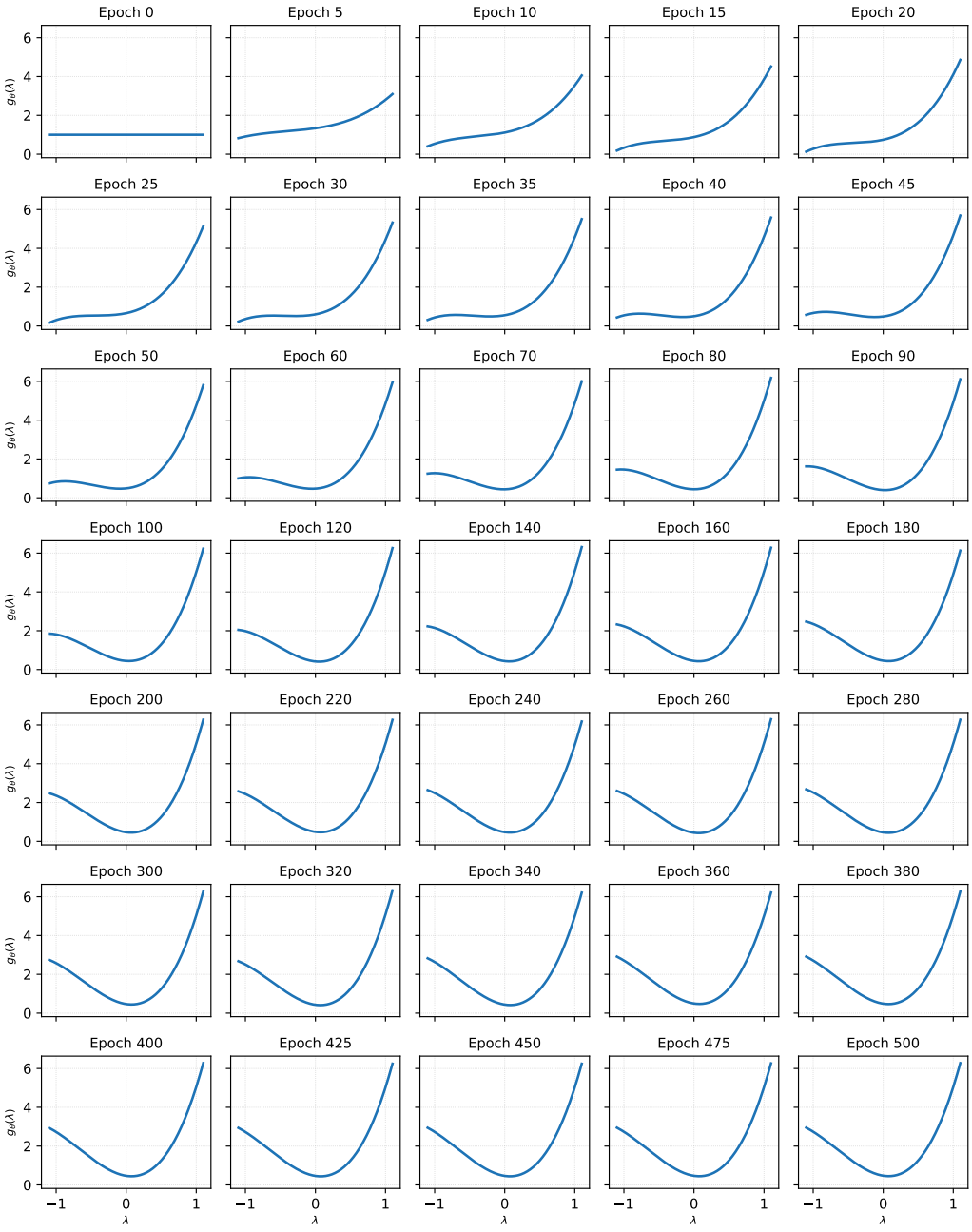
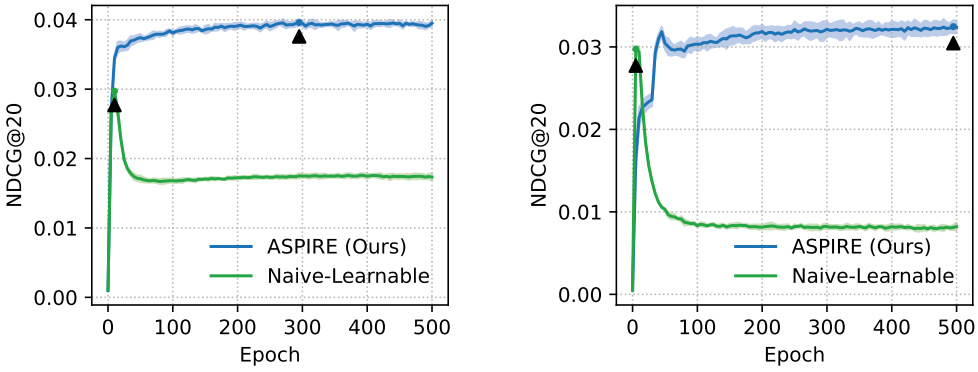


Figure 10: Filter evolution of ASPIRE under Homogeneous scenario on Ciao.



(a) Baby (b) Ciao

Figure 11: NDCG@20 Trajectory of ASPIRE and Naive-L under CE loss.

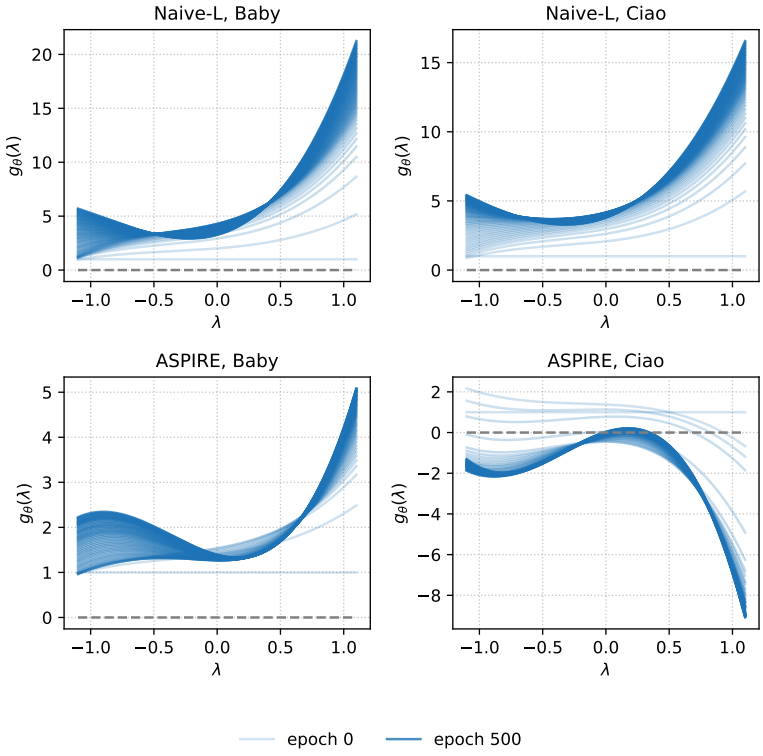


Figure 12: Filter evolution of Naive-L and ASPIRE under CE.

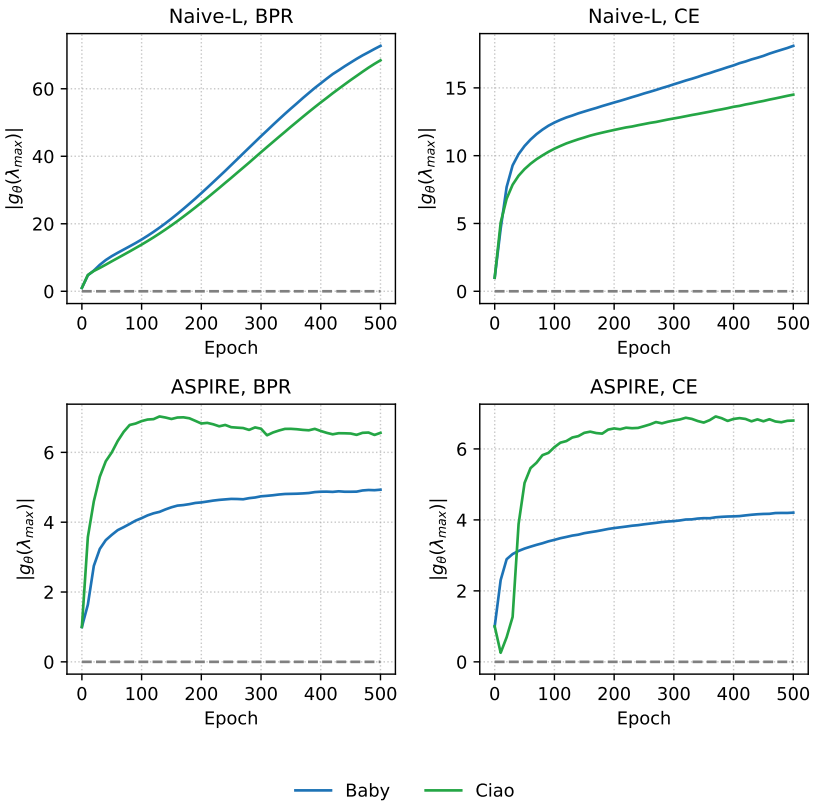


Figure 13: Low-frequency explosion from a quantitative perspective, derived from Figures 7, 8 and 12. In Homogeneous setting, $\lambda_{max} = 1$.

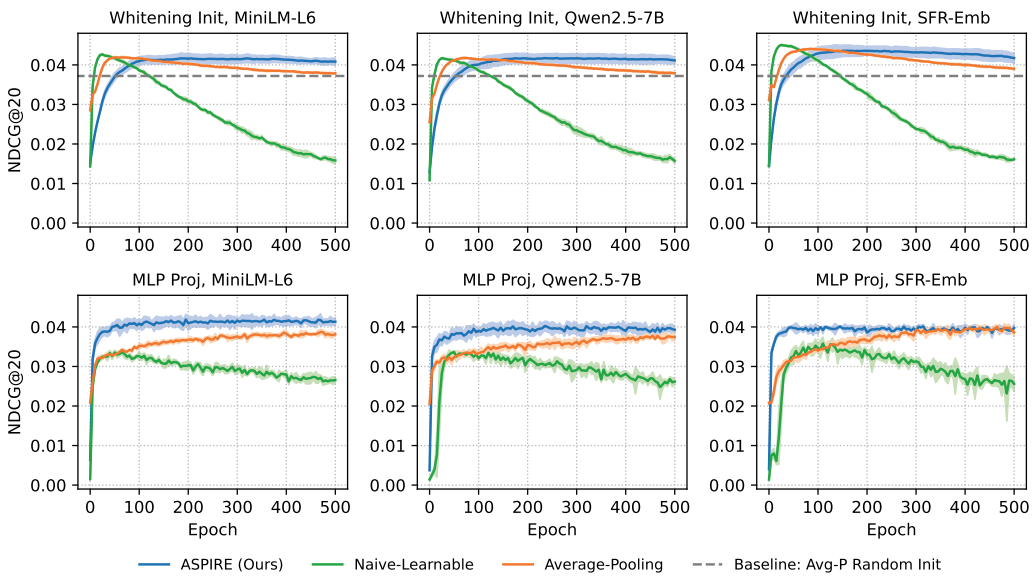


Figure 14: Metric Stability in LLM-powered CF.

that low-frequency explosion still prevails. A possible reason for the discrepancy in their growth slopes is explained in Appendix B.2.

D.3 Stability Evaluation for Table 5

Metric Stability. As shown in Figure 14, ASPIRE maintains consistent stability throughout training, while Naive-I exhibits obvious instability due to an early performance decline in the training process. Avg-P behaves unstably under the Whitening Init setting, with its performance gradually degrading toward the baseline, as elaborated in the main text. In contrast, it remains stable under the MLP Proj setting. It is also worth noting that Avg-P achieves better initial performance than the other two methods, and the Whitening Init setup outperforms the MLP Proj setup at initialization. This phenomenon reflects the differences in initialization quality: low-pass filter-based initialization presents clear advantages over full-pass initialization, and direct adoption of semantic embeddings delivers better results than a randomly initialized MLP.

Filter Stability. For Naive-L (Figure 15) equipped with the SFR-Emb encoder and MLP Proj adapter, the evolutionary pattern of low-frequency components exhibits slight discrepancies. Nevertheless, the prevalent low-frequency explosion issue remains prominent. Beyond this minor difference, both Naive-L and ASPIRE (Figure 16) yield consistent overall behaviors with the conventional CF settings analyzed before.

Results on Electronics. We also report results on Electronics in Table 8 for reference.

D.4 Extended Sensitivity Analysis for Table 4

We further conduct identical experiments on Ciao and present the corresponding results in Table 9. The convergence trends are highly consistent with the analysis elaborated in the main text. In Figure 17, we normalize all NDCG@20 curves to the range [0,1] to intuitively compare the convergence speed of different initialization strategies. The gray horizontal lines mark the 80% and 95% convergence thresholds reported in our quantitative tables. From the visualized results, we conclude that prior-knowledge-aligned initializations (*e.g.*, low-pass initialization) converge rapidly and possess prominent advantages over the Avg-P baseline. In contrast, improper initializations that contradict inherent prior knowledge (*e.g.*, high-pass initialization) degrade model performance in the early training phase, but such methods can quickly recover, and their overall convergence speeds

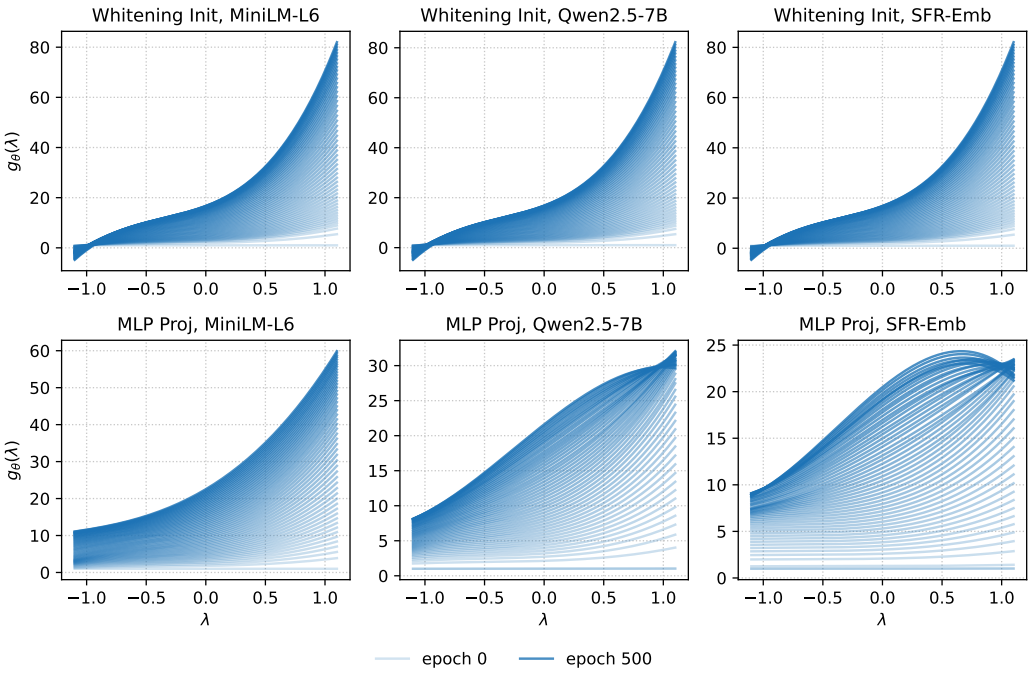


Figure 15: Filter Stability of Naive-L in LLM-powered CF.

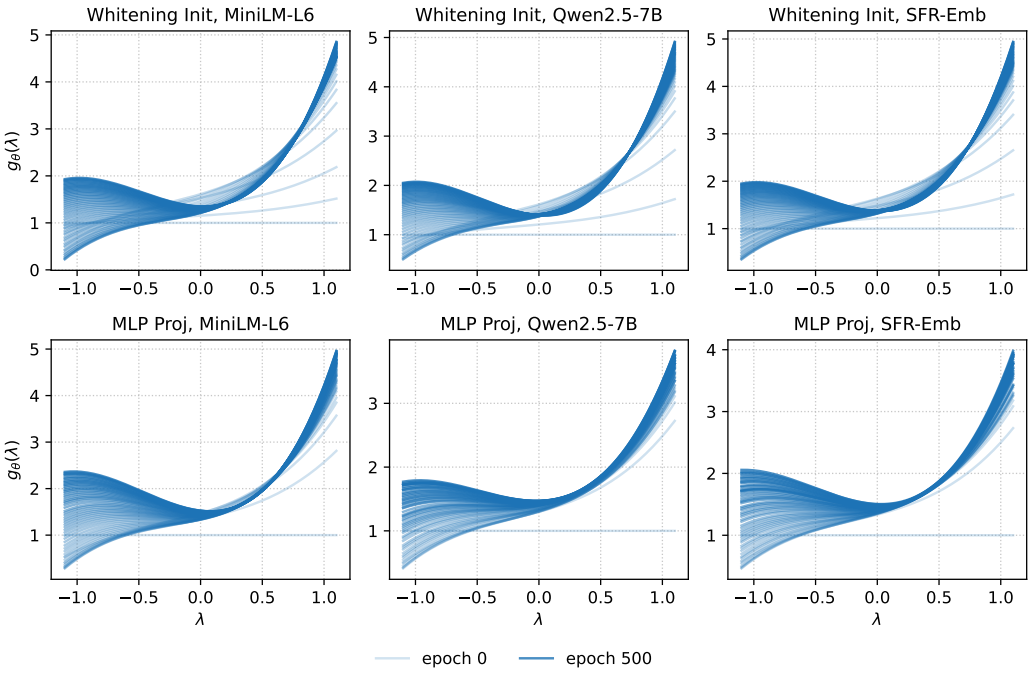
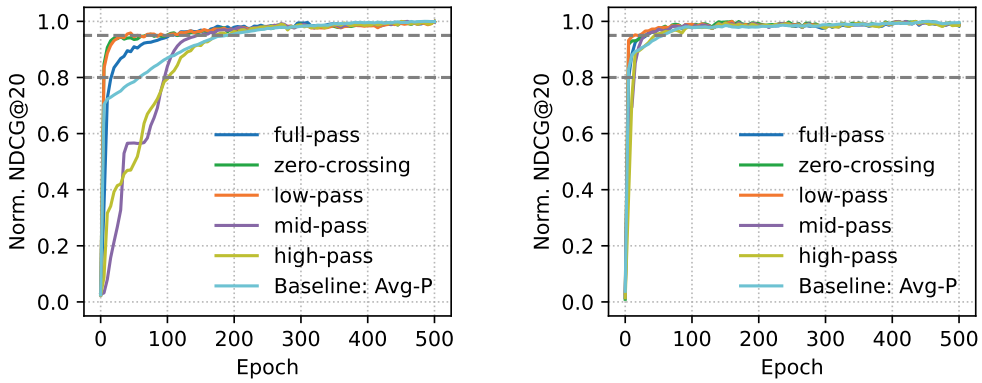


Figure 16: Filter Stability of ASPIRE in LLM-powered CF.



(a) Baby (b) Ciao

Figure 17: Min-Max Normalized NDCG@20 Trajectory of ASPIRE under different initialization.

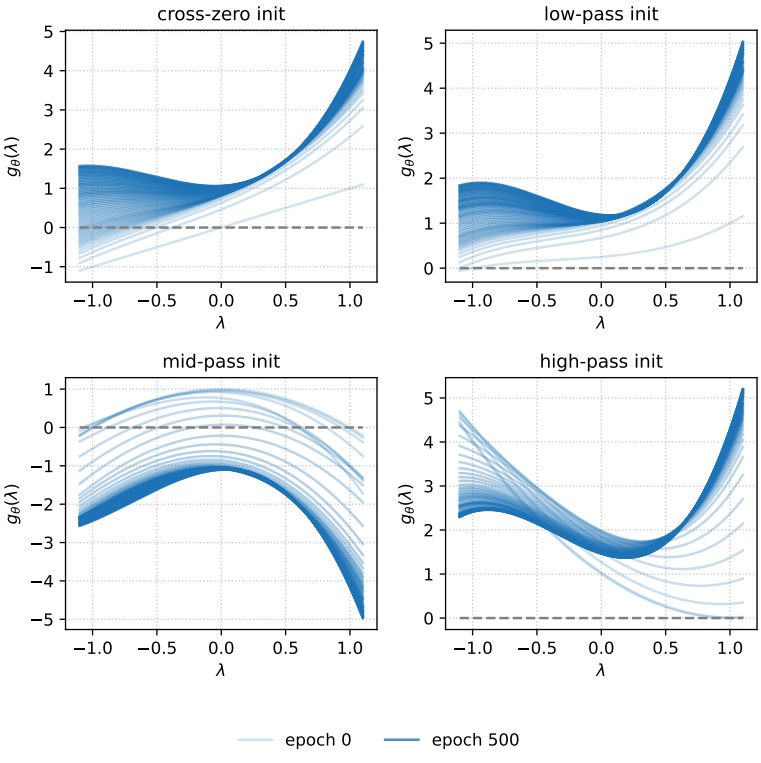


Figure 18: Filter evolution of ASPIRE under different initialization on Baby.

Table 8: Performance comparison of LLM-assisted CF on Electronics. Best results are in **bold**.

Adapter	Filter	Metric Sta.	Filter Sta.	Filter Learn.	MiniLM-L6		Qwen2.5-7B		SFR-Emb	
					R@20	N@20	R@20	N@20	R@20	N@20
Whitening Init	Avg-P	✗	✓	✗	0.0649	0.0300	0.0635	0.0292	0.0655	0.0301
	Naive-L	✗	✗	✓	0.0634	0.0288	0.0612	0.0277	0.0643	0.0293
	ASPIRE	✓	✓	✓	**0.0664**	**0.0301**	**0.0653**	**0.0297**	**0.0667**	**0.0305**
MLP Proj	Avg-P	✓	✓	✗	0.0540	0.0241	0.0588	0.0262	0.0619	0.0278
	Naive-L	✗	✗	✓	0.0497	0.0221	0.0520	0.0230	0.0561	0.0252
	ASPIRE	✓	✓	✓	**0.0582**	**0.0258**	**0.0616**	**0.0278**	**0.0621**	**0.0280**

[1] Sta. = Stability, Learn. = Learnability.
[2] The MiniLM-L6 column corresponds to all-MiniLM-L6-v2 (Wang et al., 2020), as adopted in Section 4. Though strictly speaking this model is not an LLM, it is included for comparison. Qwen2.5-7B is officially released by Qwen Team (2024), and SFR-Emb denotes SFR-Embedding-Mistral (Meng et al., 2024).

Table 9: Initialization Robustness on Ciao.

Initialization	$g_\theta^{init}(\lambda)$	Convergence Speed		Final Performance			
		E@80%	E@95%	R@10	R@20	N@10	N@20
full-pass	1	10	35	0.0489	0.0796	0.0309	0.0397
zero-crossing	λ	5	30	0.0488	0.0795	0.0309	0.0397
low-pass	$\frac{1}{4}\sum_{l=0}^{3}\lambda^l$	5	15	0.0490	0.0795	0.0309	0.0397
mid-pass	$1-\lambda^2$	15	30	0.0482	0.0790	0.0308	0.0396
high-pass	$1-2\lambda+\lambda^2$	15	45	0.0491	0.0797	0.0311	0.0398
Baseline: Average-Pooling		5	60	0.0473	0.0770	0.0295	0.0381

are no worse than that of Avg-P. As a neutral scheme, full-pass initialization delivers considerably faster convergence compared with Avg-P, though it still falls slightly behind the optimal prior-aware initializations. In terms of the final performance, the five investigated initialization settings produce nearly indistinguishable results. Additionally, the filters learned under different initializations share great similarity, as illustrated in Figure 18 and Figure 19.

D.5 Effect of Pre-filter Normalization

As illustrated in Figure 20, the ASPIRE variant without pre-filter normalization yields inferior recommendation accuracy and exhibits considerably slower convergence. Besides, this model variant cannot guarantee stable training metrics, as validated by the filter evolution in Figure 21 and Figure 22. More critically, it is highly sensitive to hyperparameter configurations, where small deviations can lead to progressive filter degradation and eventually trigger a sudden loss divergence, causing training to collapse. We select a representative bad case and visualize its filter evolution dynamics in Figure 23.

D.6 Time and Memory Efficiency

Two types of approximations are commonly employed to solve bilevel optimization problems. The first is the Neumann approximation, which is adopted by Chen et al. (2023); Yin et al. (2024). Specifically, the inverse Hessian term in equation 5 is approximated using the truncated Neumann series as

$$\left(\nabla_E^2 \mathcal{L}_{\text{train}}\right)^{-1} \approx \sum_{m=0}^{M} \left(I - \nabla_E^2 \mathcal{L}_{\text{train}}\right)^m. \tag{9}$$

This approximation avoids the explicit computation of the Hessian inverse and enables efficient estimation via iterative Hessian-vector products. The second is the DARTS approximation (Liu et al.,

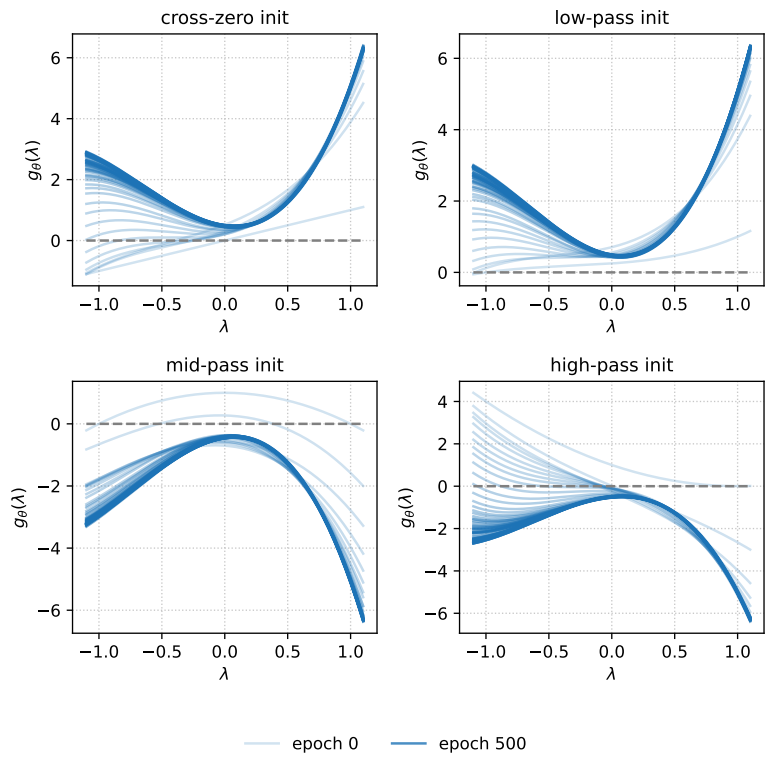
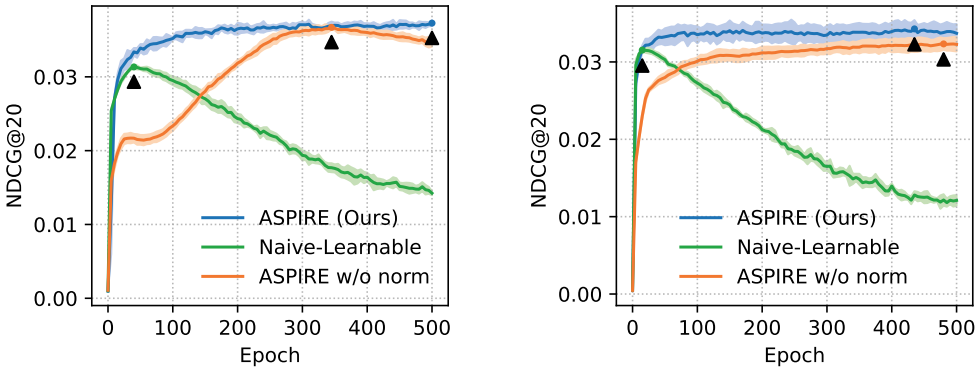


Figure 19: Filter evolution of ASPIRE under different initialization on Ciao.



(a) Baby (b) Ciao

Figure 20: NDCG@20 Trajectory comparison.

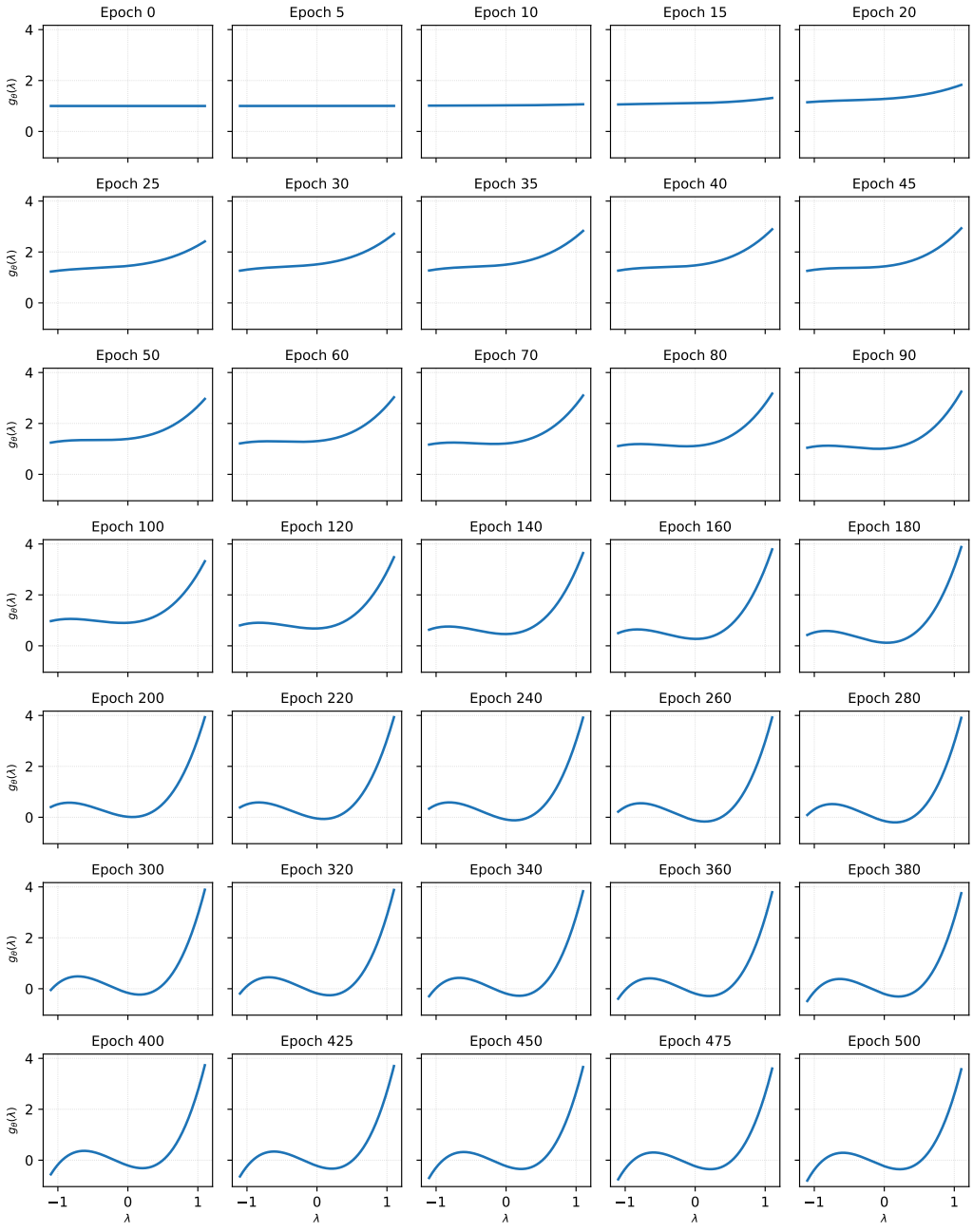


Figure 21: Filter evolution of ASPIRE without prefilter normalization on Baby.

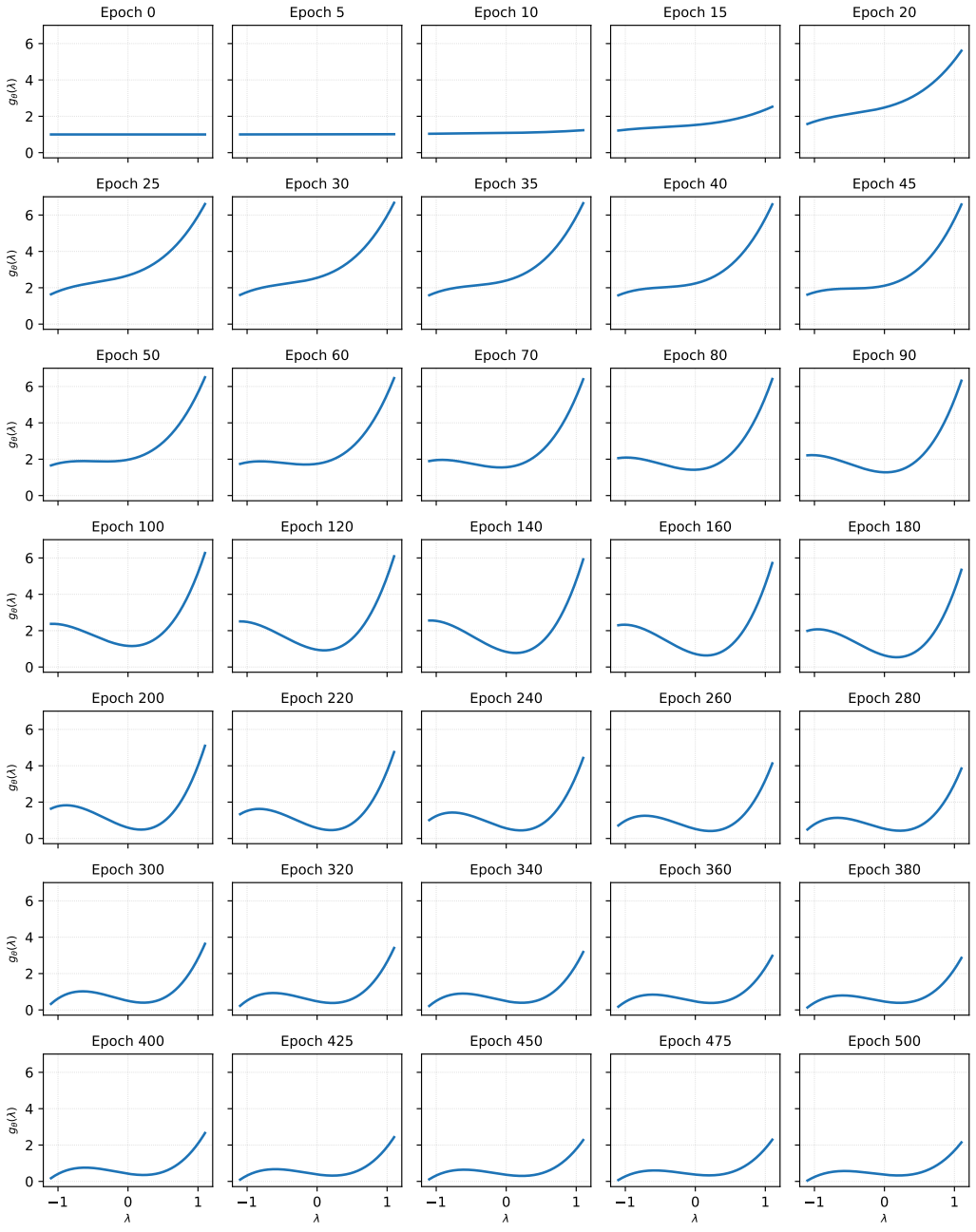


Figure 22: Filter evolution of ASPIRE without prefilter normalization on Ciao.

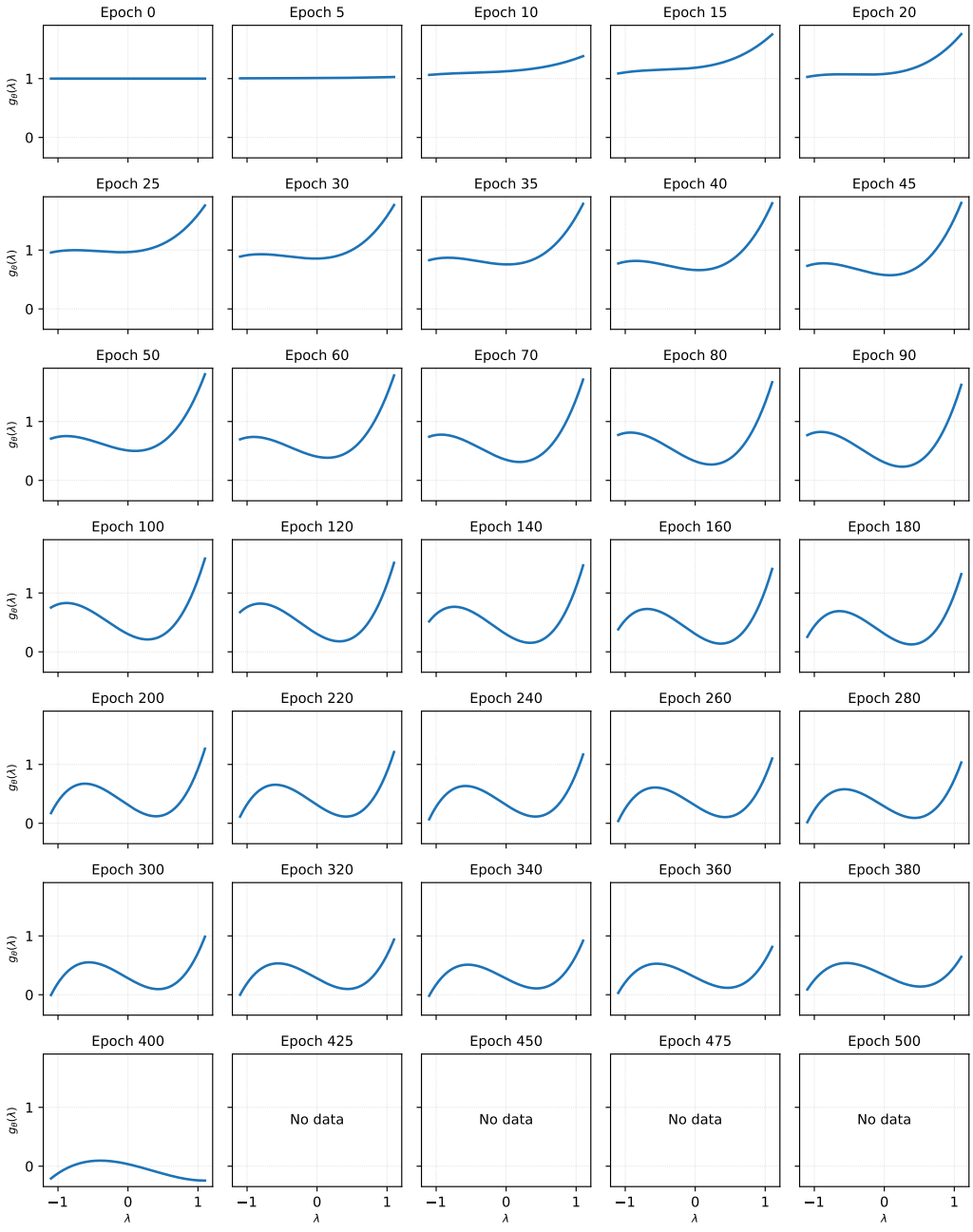


Figure 23: A bad case on Baby.

Table 10: Comparison of time and memory costs for different upper-level gradient approximations (memory in MB). N=Neumann, D=DARTS, Ex=Explicit Term, Im=Implicit Term. N-Ex is employed in ASPIRE. Results are obtained on an Intel Xeon E5-2680 v4 CPU and an NVIDIA RTX 3090 GPU.

	Method	Baby				Electronics				Ciao			
		R@20	N@20	Time	Mem	R@20	N@20	Time	Mem	R@20	N@20	Time	Mem
Homogeneous	Avg-P	0.0851	0.0372	10m09s	488	0.0579	0.0272	120m16s	1970	0.0770	0.0381	9m53s	640
	Naive-L	0.0690	0.0307	10m10s	512	0.0463	0.0213	120m11s	1970	0.0678	0.0352	9m52s	640
	N-Ex&Im	0.0870	0.0380	15m20s	636	0.0636	0.0306	281m30s	3052	0.0796	0.0397	14m23s	644
	N-Ex	0.0871	0.0380	10m44s	488	0.0635	0.0306	139m38s	2036	0.0796	0.0397	10m37s	640
	D-Ex&Im	0.0873	0.0381	13m40s	514	0.0644	0.0310	194m27s	2038	0.0798	0.0398	13m25s	640
	D-Ex	0.0877	0.0381	11m36s	488	0.0638	0.0306	159m21s	2036	0.0798	0.0397	11m34s	640
Heterogeneous	Avg-P	0.0990	0.0430	10m08s	506	0.0675	0.0313	159m47s	2000	0.0808	0.0410	9m30s	658
	Naive-L	0.0833	0.0369	10m14s	506	0.0486	0.0228	159m36s	2000	0.0708	0.0360	9m37s	658
	N-Ex&Im	0.1029	0.0454	15m31s	650	0.0681	0.0314	365m17s	3332	0.0848	0.0429	14m40s	664
	N-Ex	0.1027	0.0454	11m05s	506	0.0679	0.0313	181m50s	2066	0.0844	0.0426	10m26s	658
	D-Ex&Im	0.1031	0.0454	14m08s	508	0.0680	0.0314	251m09s	2068	0.0842	0.0427	12m47s	658
	D-Ex	0.1035	0.0456	12m41s	506	0.0679	0.0314	209m15s	2066	0.0844	0.0426	11m10s	658

Table 11: Ablation study. Best results are in **bold**. PN, BO, and OS denote pre-filter normalization, bi-level objective, and oracle supervision, respectively. The effect of removing pre-filter normalization is dataset-dependent: it causes clear metric instability on Baby but remains relatively stable on Ciao.

Variant	Metric Sta.	Filter Sta.	Baby				Ciao			
			R@10	R@20	N@10	N@20	R@10	R@20	N@10	N@20
Naive-L	✗	✗	0.0450	0.0690	0.0246	0.0307	0.0445	0.0678	0.0283	0.0352
ASPIRE	✓	✓	**0.0550**	**0.0871**	**0.0297**	**0.0380**	**0.0489**	**0.0796**	**0.0309**	**0.0397**
w/o PN	✓/✗	✗	0.0526	0.0824	0.0286	0.0362	0.0470	0.0740	0.0298	0.0377
w/o BO	✗	✗	0.0519	0.0815	0.0280	0.0356	0.0476	0.0731	0.0300	0.0375
w/o OS	✗	✗	0.0363	0.0570	0.0200	0.0254	0.0279	0.0458	0.0189	0.0226

2019), used by Yu et al. (2024); Zhang et al. (2025a):

$$\nabla_\theta \mathcal{L}_{\text{valid}}\left(\theta, E^*(\theta)\right) \approx \underbrace{\nabla_\theta \mathcal{L}_{\text{valid}}\left(\theta, E'\right)}_{\text{Explicit Term}} \underbrace{-\frac{\xi}{2\epsilon}\left[\nabla_\theta \mathcal{L}_{\text{train}}\left(\theta, E^+\right) - \nabla_\theta \mathcal{L}_{\text{train}}\left(\theta, E^-\right)\right]}_{\text{Implicit Term}} \tag{10}$$
$$E' = E - \xi \nabla_E \mathcal{L}_{\text{train}}\left(\theta, E\right), \quad E^\pm = E \pm \epsilon \nabla_E \mathcal{L}_{\text{valid}}\left(\theta, E'\right),$$

where ξ denotes the learning rate for the lower-level optimization, and ϵ is a small scalar controlling the finite-difference approximation. We compare these approximations in Table 10. ASPIRE and its three variants yield nearly identical recommendation performance. However, the Neumann approximation without the implicit term is the most time- and memory-efficient, particularly on large-scale datasets (Electronics). This comparison is conducted under a unified setting of 500 epochs, and when taking into account the accelerated convergence of ASPIRE, its actual convergence time is expected to be more advantageous than that of traditional methods.

D.7 Ablation Study

Appendix D.5 already studies the effect of pre-filter normalization, and we now demonstrate additional ablations that separately evaluate normalization, the bi-level objective, and oracle supervision in Table 11. From the experiment, we have the following observations:

- Removing pre-filter normalization significantly reduces both performance and stability, consistent with Appendix D.5.
- Normalization alone can partially stabilize training, but low-frequency explosion and non-convergence still persist, and the resulting performance remains below most baselines, especially Avg-P.

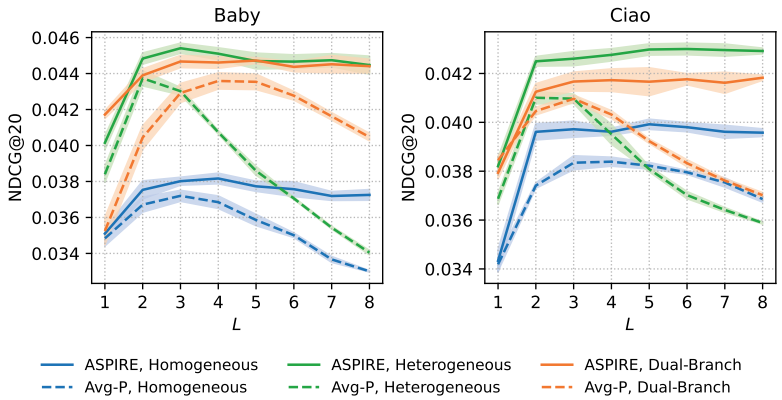


Figure 24: Performance comparison between ASPIRE and Avg-P with different layers.

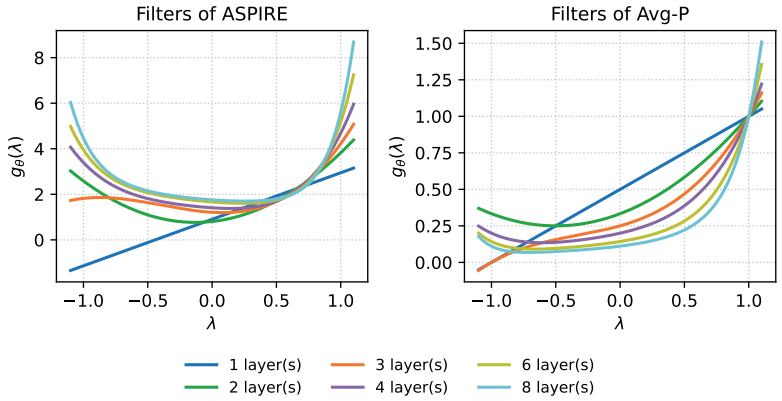


Figure 25: Filter comparison between ASPIRE and Avg-P with different layers.

- Replacing the validation-set oracle supervision with training-set supervision, while retaining the other designs, makes the instability more severe and substantially degrades recommendation performance.

D.8 Hyperparameter Sensitivity Analysis

Filter Order / Convolution Layers L. In this experiment, we vary the number of propagation layers and compare ASPIRE with Avg-P, as shown in Figure 24. For Avg-P, the best performance typically occurs around 2-4 layers and degrades noticeably as the depth increases, a phenomenon commonly attributed to over-smoothing (Liu et al., 2021; Wu et al., 2024; Li et al., 2018). In contrast, ASPIRE exhibits significantly more stable performance once the number of layers exceeds three. To better understand this difference, we visualize the learned spectral filters in Figure 25. As the number of layers increases, the filters learned by ASPIRE gradually converge to a consistent shape, indicating that higher-order representations are not strictly necessary but are also not detrimental. By comparison, Avg-P increasingly biases the spectral response toward low-frequency components, progressively suppressing higher-frequency signals as the depth grows. This demonstrates that ASPIRE not only effectively mitigates over-smoothing, but also holds strong promise as a practical tool for graph spectral analysis.

Temperature τ. According to Appendix C.3, we evaluate both BPR and CE as $\mathcal{L}_{\text{valid}}$. Specifically, CE performs better on the Baby dataset, whereas BPR achieves better results on Ciao (Figure 26).

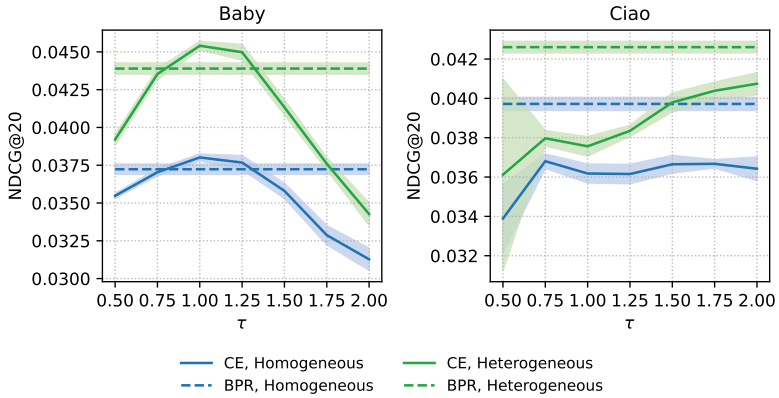


Figure 26: Performance of ASPIRE with different temperature settings in $\mathcal{L}_{\text{valid}}$.

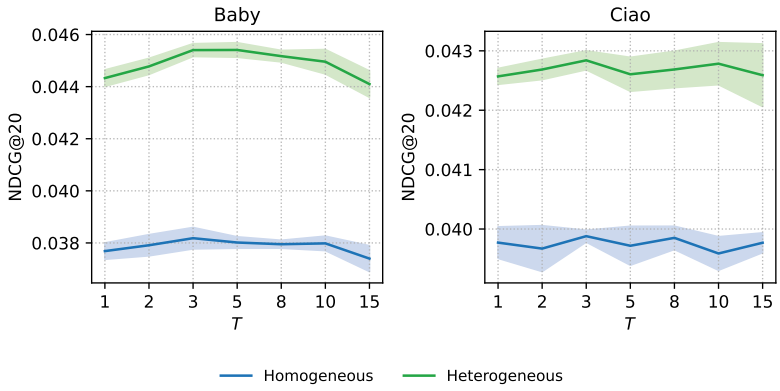


Figure 27: Performance of ASPIRE with different upper-level update intervals.

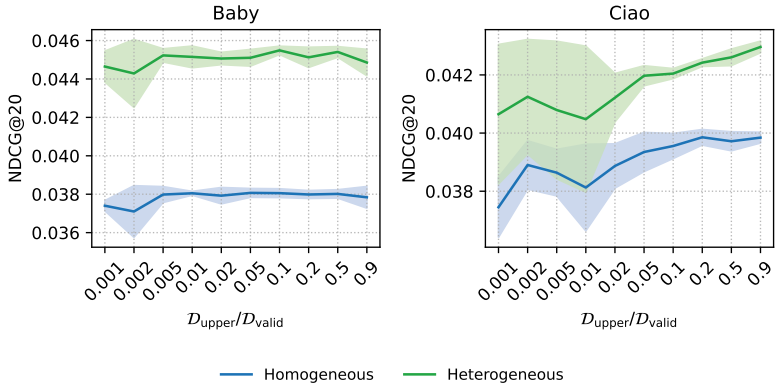


Figure 28: Performance of ASPIRE with different dataset sizes for upper-level updates.

Upper-level Optimization Configurations. The upper-level update interval T and the dataset size are not critical to the performance of ASPIRE. As illustrated in Figure 27, a moderate choice of T yields slightly better results, while both overly frequent and overly infrequent updates lead to only marginal differences. Figure 28 shows that on Baby, as little as 0.5% of the validation set is sufficient for effective upper-level optimization. On Ciao, a more noticeable performance drop, along with increased variance across random seeds, is observed when the ratio falls below 5%.

E Broader Impact and Limitations

Given the advantages of ASPIRE demonstrated in Section 4 and Section D.8, we believe it has strong potential as a practical tool for graph spectral analysis. Existing spectral analyses (Guo et al., 2023; He et al., 2025) for a given graph are typically restricted to specific graph classes and mainly provide qualitative characterizations of frequency distributions. They rarely offer an accurate estimate of the effective spectral filter that a learning objective induces. In contrast, ASPIRE directly learns a quantitative filter function from data, providing an effective and precise approximation of the spectral response preferred by the interaction graph.

Another long-standing problem in collaborative filtering is the parameterization of propagation layers, *e.g.*, introducing linear transformations between layers to enhance expressivity (Wang et al., 2019). Such parameterizations also suffer from abrupt performance degradation, consistent with the phenomenon illustrated in Figure 1(a). Xu et al. (2023) attempted to address this issue, but their approach mainly increases the complexity of pre-filter encoding while still relying on a fixed graph filter as in LightGCN. The stability of ASPIRE, as discussed in Section 4.3, further highlights its potential to support the development of more expressive and stable graph neural networks for collaborative filtering.

However, this work still has several limitations. From a spectral perspective on collaborative filtering, our experiments do not cover more complex GNN architectures, such as those incorporating learnable linear transformations. Also, our exploration of LLM-powered collaborative filtering is limited to using LLMs as encoder. It remains to be investigated whether the low frequency explosion phenomenon manifests across a wider range of tasks, such as sequential recommendation, and whether ASPIRE remains effective in those settings.

F LLM Usage

Large language models were used solely for grammar checking and language polishing of the manuscript.