A Benchmark Suite for Online Learning of Non-Stationary User Preferences in Recommender Systems

Tonmoy Hasan
thasan1@charlotte.edu
UNC Charlotte
Charlotte, NC, USA

Wenyu Gao
wgao4@charlotte.edu
UNC Charlotte
Charlotte, NC, USA

Razvan Bunescu
rbunescu@charlotte.edu
UNC Charlotte
Charlotte, NC, USA

Abstract

A recommender system's performance depends crucially on accurate estimates of user preferences, which are often non-stationary. Correspondingly, the speed with which recommender systems adapt to changes in preferences becomes an important contributor to user satisfaction. However, evaluation benchmarks are dominated by performance measures that aggregate rating prediction accuracy across histories of recommended items. Due to the infrequent nature of user preference changes, such aggregate measures cannot characterize how quickly models can detect these changes. This paper introduces a benchmark suite for evaluating online learning models in terms of their capability to effectively track non-stationary user preferences while also maintaining a stable recommendation performance. The synthetic data portion contains a large set of user profiles, with time-varying preferences that reflect a diverse set of behaviors, such as preference shifting, preference broadening, or preference conservation. Time series of items are generated using topic distributions that differ in terms of how they match either ground truth or estimated user preferences. The real dataset portion of the benchmark is compiled from a large book recommendation dataset, where a set of heuristic procedures are used to determine time steps where user preferences are either stable or change substantially. We provide implementations and comparisons of 8 online learning algorithms, as well as metrics that quantify performance on rating prediction, preference estimation, and preference change detection. Experimental evaluations reveal an empirical dissociation between aggregate rating prediction accuracy and preference tracking capability: models that exhibit strong rating prediction do not necessarily provide effective preference tracking. The datasets, code, and appendix are available at https://github.com/Ton-moy/nonstationary-user-preferences.

1 Introduction and Motivation

A fundamental challenge in real-world recommender systems is non-stationarity: user preferences are not fixed but change over time due to evolving interests, life events, seasonal trends, or exposure to novel content [13]. Consequently, a growing body of work studies non-stationary user preferences from multiple perspectives, including studying the impact of aggregate taste shifts on recommendation outcomes [44], sequential modeling of evolving latent preferences to capture preference shift [7], inference-time model adaptation to mitigate the performance degradation caused by user interest shifts [42], and online collaborative filtering for non-stationary environments [16].

Benchmarking and evaluation metrics, however, have not kept pace with advances in modeling non-stationary user preference. Given that ground truth preference changes are unknown, most prior works evaluate non-stationary performance indirectly by calculating the impact of temporal components such as sliding windows, cumulative sum, or adaptive embeddings, on downstream recommendation metrics [6, 10, 16, 38]. This practice treats improved downstream performance as evidence that a model handles non-stationarity more effectively, without directly evaluating whether it is able to quickly detect preference shifts or adapt to them. However, prior work has shown that downstream gains in recommendation metrics may instead arise from noise suppression in implicit feedback [36], regularization effects, or recency-dominated modeling that overemphasizes recent interactions rather than capturing structural preference shifts [29], or from correcting evaluation bias under non-stationarity, rather than from explicit detection or modeling of non-stationary user preferences [20].

In this paper, we propose that recommender models be evaluated explicitly on tracking non-stationary user preferences. We draw inspiration from the need to balance *stability* and *plasticity* [40, 41], which highlights that a recommender system should remain stable enough to ignore noise in stationary phases while maintaining enough plasticity to react quickly to genuine user preference shifts. However, evaluating a recommender model for user preference tracking poses several fundamental challenges. First, user preferences are latent vectors that are never directly observable in real-world interaction logs, and therefore are not provided in recommendation datasets [3, 12]. In standard latent factor models, a rating r is typically modeled as the dot product between a user preference vector $\mathbf{p}$ and an item attribute vector $\boldsymbol{\theta}$, such that $r \approx \mathbf{p}^\top \boldsymbol{\theta}$. This formulation induces a many-to-one mapping from latent preferences to observed ratings: multiple, substantially different preference vectors can yield nearly identical ratings for the same item. As a result, a model may achieve high rating prediction accuracy while its internal preference representation remains misaligned with the user's true underlying preferences, particularly after a preference change. Second, preference shifts are typically infrequent relative to the total number of interactions. As a result, aggregate rating prediction metrics such as Mean Squared Error (MSE) are dominated by a large number of time steps in which preferences are stationary, rendering them insensitive to tracking lag. A model that lags behind a preference change by 10 time steps may achieve a comparable aggregate MSE to a model that adapts

Online and Adaptive Recommender Systems (OARS) Workshop @ ACM RecSys 2026, September 28, 2026, Minneapolis, Minnesota, USA.
https://tinagwy.netlify.app/ (W. Gao);
https://webpages.charlotte.edu/rbunescu (R. Bunescu)
0000-0002-2116-6822 (T. Hasan);
0000-0002-2128-9232 (W. Gao);
0000-0003-2919-3566 (R. Bunescu)
Workshop website: https://oars-workshop.github.io/

immediately but exhibits slightly higher variance. Yet, the former subjects the user to a critical window of misaligned recommendations immediately after the change, and such sustained mismatch can substantially degrade user satisfaction and trust in the system.

To address these challenges, we develop a benchmark suite that is specifically aimed at evaluating the preference tracking capabilities of online learning algorithms in non-stationary recommendation settings. Contributions are made along 3 dimensions: *data*, *metrics*, and *algorithms*. First, we introduce 2 types of datasets:

(i) **Synthetic datasets** covering six user behavior scenarios across three item recommendation settings; the datasets provide ground-truth preference vectors for all 36,000 time steps, with preference change events happening at 2,436 time steps (Section 4).
(ii) **Real-world datasets** constructed from Goodreads, where similarity based heuristics identify near-duplicate item *pairs* and rating discrepancies provide evidence of preference change for approximately 60,000 interaction pairs (Section 5).

Second, we propose a diverse set of **evaluation metrics** that explicitly measure preference change detection and preference tracking performance, complementing standard rating-accuracy metrics (Section 6). Third, we provide implementations of eight **online preference learning models** (Section 3). The data, metrics, and algorithms are then brought together in extensive experimental evaluations that demonstrate that high rating prediction accuracy can mask substantial tracking lag and poor preference-change detection, highlighting the necessity for explicit preference-tracking evaluation in recommender systems (Section 7).

Additional **Related Work** is discussed at length in Appendix A.

2 Recommendation Setting and Notation

We consider a recommendation setting with a collection of items $\mathcal{I}$ (e.g., books) and a fixed set of K latent topics. Each item $i \in \mathcal{I}$ is represented by a topic distribution:

$$\boldsymbol{\theta}_i = (\theta_{i,1}, \ldots, \theta_{i,K})^T, \qquad \sum_{k=1..K} \theta_{i,k} = 1$$

where $\theta_{i,k}$ denotes the relevance of topic k to item i. Users interact with items sequentially over time. For each user u, time-varying preferences are represented by the latent vector $\mathbf{p}_t(u) = (p_{t,1}, \ldots, p_{t,K})^\top \in \mathbb{R}^K$, where $p_{t,k}$ denotes the user's preference for topic k at time t. For notational simplicity, the user index u is omitted when clear from context. At time t, the user consumes an item $\boldsymbol{\theta}_t$ and provides a scalar reward, or rating, r_t. The observed reward follows a linear Gaussian observation model, $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t + \epsilon_t$, where $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ denotes additive white noise that captures random variability in user feedback not explained by the linear preference model.

Let $\mathbf{p}_t$ and $\hat{\mathbf{p}}_t$ denote the ground-truth and estimated preference vectors, respectively. Similarly, let r_t and $\hat{r}_t = \hat{\mathbf{p}}_t^T \boldsymbol{\theta}_t$ denote the ground truth rating and the estimated rating, respectively. The ground truth rating is assumed to be scaled, i.e., $r_t \in [-1, 1]$.

3 Online Preference Learning Algorithms

We consider eight online learning methods for updating user preference distributions. All methods perform sequential Bayesian updates after observing $(\boldsymbol{\theta}_{t+1}, r_{t+1})$. For Methods 1–7, we initialize the preference vector at time 0 as $\mathbf{p}_0 \sim \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$ and maintain a Gaussian posterior $\mathbf{p}_{t+1} \sim \mathcal{N}(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1})$ at each subsequent time step, either through conjugate updates or Gaussian approximations. These methods differ in how they handle non-stationarity, uncertainty control, and noise modeling. Method 8 additionally models uncertainty in the observation noise variance and therefore departs from a purely Gaussian posterior. Each method is applied independently per user: the model initializes a separate posterior and updates it sequentially using that user's interaction history alone.

To quantify preference change, we measure the KL divergence between consecutive posterior distributions. Because Methods 1–7 yield Gaussian posteriors, they share a common KL form:

$$D_{\mathrm{KL}}\big(\mathcal{N}(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}) \,\big\|\, \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)\big)$$

whereas Method 8 requires a different expression:

$$D_{\mathrm{KL}}\big(\mathcal{NIG}(\boldsymbol{\mu}_{t+1}, \Sigma_{t+1}, a_{t+1}, b_{t+1}) \,\big\|\, \mathcal{NIG}(\boldsymbol{\mu}_t, \Sigma_t, a_t, b_t)\big)$$

Method 1. Kalman Filter with Adaptive Forgetting (KF-AF). Kalman filtering with Adaptive Forgetting provides a principled baseline for modeling non-stationary user preferences by treating the preference vector as a latent state that evolves according to a linear Gaussian state-space model [34]. We model user preferences as $\mathbf{p}_{t+1} = \gamma_{t+1}\mathbf{p}_t + \mathbf{w}_{t+1}$, where $\mathbf{w}_{t+1} \sim \mathcal{N}(\mathbf{0},\ (1-\gamma_{t+1}^2)\sigma_{\mathrm{p}}^2 I)$ and $\gamma_{t+1} \in (0, 1]$ is a time-varying forgetting coefficient. Values of γ_{t+1} close to 1 preserve past preferences, whereas smaller values induce stronger forgetting and enable faster adaptation to preference changes. Given $\mathbf{p}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t)$, the prediction step is

$$\boldsymbol{\mu}_{t+1|t} = \gamma_{t+1}\,\boldsymbol{\mu}_t, \qquad \Sigma_{t+1|t} = \gamma_{t+1}^2\,\Sigma_t + (1-\gamma_{t+1}^2)\,\sigma_{\mathrm{p}}^2 I$$

After observing $(\boldsymbol{\theta}_{t+1}, r_{t+1})$, the update step becomes

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_{t+1|t} + \frac{\Sigma_{t+1|t}\boldsymbol{\theta}_{t+1}}{\sigma^2 + \boldsymbol{\theta}_{t+1}^T \Sigma_{t+1|t}\boldsymbol{\theta}_{t+1}}\big(r_{t+1} - \boldsymbol{\theta}_{t+1}^T \boldsymbol{\mu}_{t+1|t}\big)$$

$$\Sigma_{t+1} = \Sigma_{t+1|t} - \frac{\Sigma_{t+1|t}\boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T \Sigma_{t+1|t}}{\sigma^2 + \boldsymbol{\theta}_{t+1}^T \Sigma_{t+1|t}\boldsymbol{\theta}_{t+1}}$$

The parameter γ is re-parameterized as $\gamma_{t+1} = e^{-0.5\,\delta_{t+1}}, \delta_{t+1} \geq 0$, and updated online via stochastic gradient descent on the one-step predictive log-likelihood, with learning rate η:

$$\delta_{t+2} = \delta_{t+1} + \eta\,\nabla_{\delta_{t+1}} \log \mathcal{N}\big(r_{t+1} \big| \boldsymbol{\theta}_{t+1}^T \boldsymbol{\mu}_{t+1|t},\ \boldsymbol{\theta}_{t+1}^T \Sigma_{t+1|t}\boldsymbol{\theta}_{t+1} + \sigma^2\big)$$

Method 2. Adaptive Regularization of Weights (AROW). This method updates the posterior mean and covariance by minimizing a KL-regularized objective with squared error loss [8]. As a confidence-weighted second-order online regression method, AROW adapts its update magnitude based on posterior uncertainty, allowing parameters with higher uncertainty to change more rapidly over time. Here we use the notations introduced in [14], separating and tuning the hyperparameters $r_1 > 0$ and $r_2 > 0$ controlling the trade-off between prediction accuracy and confidence. Given $(\boldsymbol{\theta}_{t+1}, r_{t+1})$, the online update is:

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \frac{r_{t+1} - \boldsymbol{\mu}_t^T\boldsymbol{\theta}_{t+1}}{r_1 + \boldsymbol{\theta}_{t+1}^T\Sigma_t\boldsymbol{\theta}_{t+1}}\Sigma_t\boldsymbol{\theta}_{t+1} \;\Bigg|\; \Sigma_{t+1} = \Sigma_t - \frac{\Sigma_t\boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T\Sigma_t}{r_2 + \boldsymbol{\theta}_{t+1}^T\Sigma_t\boldsymbol{\theta}_{t+1}}$$

Method 3. Bayesian Linear Regression (BLR). BLR serves as a stationary conjugate baseline [4]. At time step $t+1$, given the

current estimates at time t and the new observation $(\boldsymbol{\theta}_{t+1}, r_{t+1})$, the parameters are updated as follows:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T \;\Big|\; \boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Big(\Sigma_t^{-1}\boldsymbol{\mu}_t + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}r_{t+1}\Big)$$

Method 4. Variance-Bounded BLR (BLR-VB). To prevent overconfidence under non-stationarity, BLR-VB enforces a minimum variance. Following [14], we eigendecompose $\Sigma_t = U_t\Lambda_t U_t^T$, clip eigenvalues to be at least τ_v via $\Lambda_v = \mathrm{diag}(\max(\lambda_1, \tau_v), \ldots, \max(\lambda_K, \tau_v))$, and form $S_t = U_t\Lambda_v U_t^T$. The update then uses S_t in place of Σ_t:

$$\Sigma_{t+1}^{-1} = S_t^{-1} + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T \;\Big|\; \boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Big(S_t^{-1}\boldsymbol{\mu}_t + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}r_{t+1}\Big)$$

Method 5. BLR with Forgetting Factor (BLR-FF). The forgetting factor introduces decay on past information, allowing the model to adapt smoothly to gradual preference drift without explicitly specifying a state evolution model. To discount older observations, we apply a decay factor $\rho \in (0, 1]$ to the prior precision:

$$\Sigma_{t+1}^{-1} = \rho\Sigma_t^{-1} + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T \;\Big|\; \boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Big(\rho\Sigma_t^{-1}\boldsymbol{\mu}_t + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}r_{t+1}\Big)$$

Method 6. BLR with Sliding Window (BLR-SW). Sliding window estimation provides a simple non-stationary baseline by restricting inference to the most recent observations, effectively discarding older data entirely. We refit BLR using only the most recent m observations, where $\boldsymbol{\Theta}_t^{(m)} = \big(\boldsymbol{\theta}_{t-m+1}^T, \ldots, \boldsymbol{\theta}_t^T\big) \in \mathbb{R}^{m\times K}$ and $\mathbf{r}_t^{(m)} \in \mathbb{R}^m$ denote the windowed topic matrix and rewards:

$$\Sigma_{t+1}^{-1} = \Sigma_0^{-1} + \frac{1}{\sigma^2}\boldsymbol{\Theta}_{t+1}^{(m),T}\boldsymbol{\Theta}_{t+1}^{(m)} \;\Big|\; \boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Big(\Sigma_0^{-1}\boldsymbol{\mu}_0 + \frac{1}{\sigma^2}\boldsymbol{\Theta}_{t+1}^{(m),T}\mathbf{r}_{t+1}^{(m)}\Big)$$

Method 7. BLR with Power Prior (BLR-PP). Power priors [17, 18] downweigh historical likelihood contributions using a factor $\alpha \in (0, 1]$. This provides a principled Bayesian mechanism for adapting to non-stationarity by controlling the effective sample size of past data, without introducing explicit temporal dynamics. Let $p_0(\mathbf{p}) = \mathcal{N}(\boldsymbol{\mu}_0, \Sigma_0)$. The power prior distribution is:

$$p(\mathbf{p} \mid \mathcal{D}_t, \alpha) \propto \left[\prod_{i=1}^{t} p(r_i \mid \boldsymbol{\theta}_i, \mathbf{p})\right]^{\alpha} p_0(\mathbf{p})$$

yielding the power prior parameters:

$$\Sigma_t^{-1} = \Sigma_0^{-1} + \frac{\alpha}{\sigma^2}\boldsymbol{\Theta}_t^T\boldsymbol{\Theta}_t \;\Big|\; \boldsymbol{\mu}_t = \Sigma_t\Big(\Sigma_0^{-1}\boldsymbol{\mu}_0 + \frac{\alpha}{\sigma^2}\boldsymbol{\Theta}_t^T\mathbf{r}_t\Big)$$

Correspondingly, the sequential update at time $t+1$ is:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T \;\Big|\; \boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Big(\Sigma_t^{-1}\boldsymbol{\mu}_t + \frac{1}{\sigma^2}\boldsymbol{\theta}_{t+1}r_{t+1}\Big)$$

Method 8. Normal–Inverse-Gamma (BLR-NIG). To jointly model uncertainty in user preferences and observation noise, we place a conjugate $\mathcal{NIG}(\boldsymbol{\mu}, \Sigma, a, b)$ prior on $(\mathbf{p}, \sigma^2)$. Marginalizing over σ^2 yields a multivariate Student's t distribution for $\mathbf{p}$, which is heavier-tailed than a Gaussian and therefore better suited to handling outliers and time-varying noise levels in non-stationary settings. This allows the KL signal to capture both preference shifts and changes in rating noise. The posterior updates are:

$$\Sigma_{t+1}^{-1} = \Sigma_t^{-1} + \boldsymbol{\theta}_{t+1}\boldsymbol{\theta}_{t+1}^T \;\Big|\; \boldsymbol{\mu}_{t+1} = \Sigma_{t+1}\Big(\Sigma_t^{-1}\boldsymbol{\mu}_t + \boldsymbol{\theta}_{t+1}r_{t+1}\Big)$$

$$a_{t+1} = a_t + \frac{1}{2} \;\Big|\; b_{t+1} = b_t + \frac{1}{2}\left(\boldsymbol{\mu}_t^T\Sigma_t^{-1}\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t+1}^T\Sigma_{t+1}^{-1}\boldsymbol{\mu}_{t+1} + r_{t+1}^2\right)$$

The KL divergence between consecutive NIG posteriors decomposes into a Gaussian term and an inverse-gamma term, yielding:

$$\begin{aligned} D_{\mathrm{KL}} =& \frac{1}{2}\frac{a_{t+1}}{b_{t+1}}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t+1})^T\Sigma_t^{-1}(\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t+1}) + \frac{1}{2}\mathrm{tr}(\Sigma_t^{-1}\Sigma_{t+1}) \\ &+ \frac{1}{2}\ln\frac{|\Sigma_t|}{|\Sigma_{t+1}|} - \frac{K}{2} + a_t\ln\frac{b_{t+1}}{b_t} - \ln\frac{\Gamma(a_{t+1})}{\Gamma(a_t)} \\ &+ (a_{t+1} - a_t)\psi(a_{t+1}) - (b_{t+1} - b_t)\frac{a_{t+1}}{b_{t+1}}, \end{aligned}$$

where $\psi(a_{t+1})$ is the digamma function at a_{t+1}.

4 Synthetic Datasets

To evaluate recommender models under diverse non-stationary conditions, a synthetic data generation framework is developed involving **6 user behavior scenarios** (Section 4.1) under **3 item recommendation settings** (Section 4.2). The data generation procedures control both how the user preferences $\mathbf{p}_t$ change over time (user behavior scenarios) and how the topic distributions $\boldsymbol{\theta}_t$ are generated at each time step (item recommendation settings). We say that a user is *exposed* to a topic k at time step t when the generated item has a high weight to that topic, i.e., $\theta_{t,k}$ is large.

4.1 User Behavior Scenarios

Synthetic timelines of user preferences are generated and organized into a sequence of stationary *cycles*, where each cycle contains 10 time steps. Within each cycle, user preferences remain stable; these within-cycle time steps are referred to as *p-Stable* time steps. Preferences may change when the timeline transitions to the next cycle; these cycle-transition time steps are referred to as *p-Change* time steps. Starting with the second cycle, at each *p-Change* time step $t \in \{11, 21, 31, \ldots\}$, the user preference generator triggers a candidate preference update, which can result in changing preferences for some topics, introducing high preference weight for a new topic, or introducing high preference weights for a new combination of topics, depending on the scenario. Six scenarios are defined for changes in user preferences, by controlling how a user's ground-truth preference vector $\mathbf{p}_t$ changes at *p-Change* time steps.

4.1.1 Preference Shifting (PS). At a *p-Change* time step, the user is exposed to a new topic with high topic weight. The preference weight for this topic increases, while the preference weights of the other previously explored topics decrease proportionally. Thus, the user develops a strong preference for the new topic while shifting away from previously explored topics. A topic is considered as *explored* at time t if it receives high topic weight in at least one earlier item shown to the user; the exact criterion for "high topic weight" depends on the item-generation setting (Section 4.2).

4.1.2 Preference Shifting & Conservation (PSC1T). At a *p-Change* time step, the newly introduced topic is associated with either a *PS* preference update or no update. In the latter case, the preference weight for the new topic remains near zero, indicating neutral preference for that topic, while the preference weights of the other topics remain approximately unchanged.

4.1.3 Preference Shifting & Conservation with Two Topics (PSC2T). This scenario extends *PSC1T* by allowing some *p-Change* time steps to present an item with high weight for *two* topics that the user has previously encountered separately with high topic weights. Although the user previously had negative or near-neutral preference for these topics individually, their joint occurrence induces a strong preference shift toward both topics. We apply the *PS* update to the preference weights corresponding to these two topics.

4.1.4 Preference Broadening (PB). At a *p-Change* time step, a newly introduced topic is associated with a strong positive preference update, while the preference weights of the other topics remain approximately unchanged.

4.1.5 Preference Broadening & Conservation (PBC1T). At a *p-Change* time step, the newly introduced topic is associated with either the *PB* update or no update. In the latter case, the preference weight for the new topic remains near zero, while the preference weights of the other topics remain approximately unchanged.

4.1.6 Preference Broadening & Conservation with Two Topics (PBC2T). This scenario extends *PBC1T* by allowing some *p-Change* time steps to present an item with high weight on two previously explored topics. Although the user previously had negative or near-zero preference for these topics individually, their joint occurrence induces a strong preference increase for both topics. We apply the *PB* update to the corresponding preference weights of these two topics.

4.2 Item Recommendation Settings

Three recommendation settings are considered, differing in how topic vectors are generated for each time step:

(1) θ-**Driven**: This is a **topic-driven** setting, where the topic vectors are generated algorithmically at each time step, starting with high weights placed on a subset of an initial set of *explored* topics, that is updated at every time step (Section 4.4).
(2) p-**Driven**: This is a setting **driven by ground-truth preferences** $\mathbf{p}_t$, where high weights are placed on topics in θ_{t+1} that are aligned with the user's current ground truth preferences (Section 4.5).
(3) $\hat{p}$-**Driven**: This is a setting **driven by estimated preferences** $\hat{\mathbf{p}}_t$, where the item to be recommended at the next time step is the one that receives the highest predicted rating when using the current preference estimates (Section 4.6).

All six user behavior scenarios are instantiated for each recommendation setting, yielding 3×6 synthetic datasets in total. The following sections describe in detail the data generation procedure for the 3 recommendation settings in the preference shifting user behavior scenario, whereas the data generation procedures for the other 5 scenarios are described in Appendix B. Table 1 summarizes the key statistics of the synthetic datasets.

4.3 User Preference Generation

The ground truth preferences $\mathbf{p}_t$ are generated using a common procedure for all 3 recommendation settings, independent of the model's predictions. Across the 6 user behavior scenarios, the updates at *p-Stable* time steps are scenario-independent, whereas the updates at *p-Change* time steps are scenario-specific. Below we describe the user preference generation procedure for the Preference Shifting scenario; the corresponding updates for the other scenarios are detailed in Appendix B.

Table 1: Summary statistics of the synthetic benchmark. *#p-Change* reports the total number of time steps labeled as preference change events across all scenarios for each setting.

Setting	#Users	#Items	#Interactions	*#p-Change*
θ-Driven	90	5,996	5,996	411
$\mathbf{p}$-Driven	90	5,996	5,996	411
$\hat{\mathbf{p}}$-Driven (Top-2%)	180	2,960	11,992	812
$\hat{\mathbf{p}}$-Driven (Mixed)	180	2,960	11,992	812
Total	540	17,912	35,976	2,436

Let $\mathcal{S}_t \subseteq \{1, \ldots, K\}$ denote the set of topics explored by the user through time step t, which is initialized as $\mathcal{S}_0 = \{1, \ldots, K_0\}$, and remains unchanged at *p-Stable* time steps.

We initialize the preference vector as $\mathbf{p}_0 = \mathbf{0}$ and define the preference-increase operator $\text{boost}(p, \eta) = p + \eta(1-p)$, with $\eta_{\text{big}} = 0.8$ and $\eta_{\text{small}} = 0.3$. At $t = 1$, the preferences for the two initially explored topics k_a and k_b receive large increases, while preferences for the remaining initially explored topics increase by a smaller amount:

$$p_{1,k} \leftarrow \text{boost}(p_{0,k}, \eta_{\text{big}}) \qquad \forall\, k \in \{k_a, k_b\},$$
$$p_{1,k} \leftarrow \text{boost}(p_{0,k}, \eta_{\text{small}}) \qquad \forall\, k \in \mathcal{S}_0 \setminus \{k_a, k_b\}.$$

The preference weights for all initially unexplored topics are initialized to small values, $p_{1,k} \sim \text{Unif}[-0.01, 0.01]$ for all $k \notin \mathcal{S}_0$. For all subsequent time steps $t \geq 2$, preferences are updated as follows:

- *p-Stable* steps: The preference vector is perturbed by Gaussian noise and clipping, i.e., $p_{t,k} \leftarrow \text{clip}\big(p_{t-1,k} + \varepsilon_{t,k}, -1, 1\big)$, where $\varepsilon_{t,k} \sim \mathcal{N}(0, \sigma^2)$; for unexplored topics k, the noise $\varepsilon_{t,k}$ is further clipped to $[-.01, .01]$. In Figure 3(a), which illustrates the θ-driven setting, this appears as small fluctuations in the red $\mathbf{p}_t$ rows across the unshaded time steps within each cycle.
- *p-Change* steps: In scenarios where a new topic $k^\star$ is introduced with high weight, the *preference shift* update increases the preference weight for $k^\star$ by A and decreases the preference weights of all previously explored topics $j \in \mathcal{S}_{t-1}$ (if $Z \geq A$):

$$A = \eta_{\text{big}}\big(1 - p_{t-1,k^\star}\big), \qquad p_{t,k^\star} \leftarrow p_{t-1,k^\star} + A,$$
$$Z = \sum_{j \in \mathcal{S}_{t-1}} \big(1 + p_{t-1,j}\big), \quad p_{t,j} \leftarrow p_{t-1,j} - \big(1 + p_{t-1,j}\big)\frac{A}{Z}$$

For all remaining topics $k \notin \mathcal{S}_t$, preferences evolve only through the same noise injection and clipping rule used in the *p-Stable* step. In Figure 3(a), this update is visible at the shaded rows: when a new topic is introduced at $t = 11, 21, 31$, its corresponding entry in $\mathbf{p}_t$ goes up, while the previously dominant explored topics decrease, producing a characteristic shift pattern.

4.4 θ-Driven with Preference Shifting

In this setting, the item generation process starts from the initial explored set $\mathcal{S}_0$ defined in Section 4.3. At each *p-Stable* time step t, a topic vector $\boldsymbol{\theta}_t$ is generated by randomly selecting two topics from

$\mathcal{S}_{t-1}$ and assigning them high weights. At *p-Change* time steps, the topic vector places high weight on a topic that the user has not explored previously. This creates a sequence in which the user is repeatedly exposed to familiar topics during stable periods and to a new topic at the beginning of each new cycle.

The data generation process consists of three major steps. At each time step t, an item topic vector $\boldsymbol{\theta}_t$ is generated (Step 1), the ground-truth preference vector $\mathbf{p}_t$ is updated (Step 2), and the rating and preference-change label are computed (Step 3). An example is shown in Figure 3(a).

4.4.1 Step 1: Topic vector generation. The topic vector $\boldsymbol{\theta}_t$ is sampled from a Dirichlet distribution, $\boldsymbol{\theta}_t \sim \text{Dirichlet}(\boldsymbol{\alpha}_t)$. The procedure begins from a default concentration pattern that distinguishes explored and unexplored topics: $\alpha_{t,k} = 2$ for all $k \in \mathcal{S}_{t-1}$ and $\alpha_{t,k} = 1$ for all $k \notin \mathcal{S}_{t-1}$. Selected topics are then assigned higher concentrations, as described below.

- *p-Stable* steps: Two distinct topics (k_a, k_b) are sampled uniformly from $\mathcal{S}_{t-1}$ and assigned high concentrations $\{15, 20\}$ in random order, i.e., $(\alpha_{t,k_a}, \alpha_{t,k_b}) \in \{(15, 20), (20, 15)\}$. The explored set remains unchanged, $\mathcal{S}_t = \mathcal{S}_{t-1}$. This ensures that $\boldsymbol{\theta}_t$ places high weight on two explored topics, yielding random exposure over $\mathcal{S}_{t-1}$. The unshaded rows in Figure 3(a) show two large entries in $\boldsymbol{\alpha}_t$, which the corresponding dominant entries in $\boldsymbol{\theta}_t$.
- *p-Change* steps: If $|\mathcal{S}_{t-1}| < K$, a new topic $k^\star = |\mathcal{S}_{t-1}| + 1$ is introduced by setting $\alpha_{t,k^\star} = 20$, while retaining $\alpha_{t,k} = 2$ for all $k \in \mathcal{S}_{t-1}$ and $\alpha_{t,k} = 1$ for all remaining unexplored topics. The explored set is then updated as $\mathcal{S}_t = \mathcal{S}_{t-1} \cup \{k^\star\}$. This ensures that the generated item places high topic weight on the newly introduced topic. In Figure 3(a), the shaded rows at $t = 11, 21, 31$ show the largest entry of $\boldsymbol{\alpha}_t$, and consequently the dominant mass of $\boldsymbol{\theta}_t$, moving to a newly introduced topic.

4.4.2 Step 2: Preference vector generation. After generating $\boldsymbol{\theta}_t$ and updating $\mathcal{S}_t$, the ground-truth preference vector $\mathbf{p}_t$ is updated according to the procedure in Section 4.3.

4.4.3 Step 3: Rating computation and preference-change label. After updating $\mathbf{p}_t$, the rating is computed as $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t$. A binary preference-change label is assigned to time step t whenever the magnitude of the ground-truth preference update exceeds the threshold $\lambda = 0.25$, i.e., $\|\mathbf{p}_t - \mathbf{p}_{t-1}\|_2 > \lambda$. Thus, a scheduled *p-Change* time step is treated as a candidate change point and receives a positive label only when the resulting preference update exceeds the threshold.

4.5 p-Driven with Preference Shifting

This setting follows the same three-step procedure as Section 4.4, with one change in Step 1. At each *p-Stable* time step, the two topics assigned high Dirichlet concentrations are selected based on the ground-truth preference vector $\mathbf{p}_{t-1}$ rather than sampled uniformly from the explored set. At $t = 1$, two distinct topics (k_a, k_b) are sampled uniformly from the initial explored set $\mathcal{S}_0$, because an informative preference vector is not yet available. For $t \geq 2$, the three topics with the largest preference weights in $\mathbf{p}_{t-1}$ are identified, and two of them, denoted (k_a, k_b), are selected uniformly at random and assigned concentrations $(\alpha_{t,k_a}, \alpha_{t,k_b}) \in \{(15, 20), (20, 15)\}$. The topic vector generation rule at *p-Change* time steps, the preference vector generation in Step 2 (Section 4.3), and the rating computation and preference-change labeling in Step 3 are the same as in Section 4.4. Figure 3(b) shows an example. Compared with Figure 3(a), the dominant entries of $\boldsymbol{\alpha}_t$ and $\boldsymbol{\theta}_t$ during *p-Stable* steps now follow the largest coordinates of the red preference vector $\mathbf{p}_{t-1}$ rather than being selected uniformly from the explored set.

4.6 $\hat{p}$-Driven with Preference Shifting

In this setting, the item shown at time step t is selected using the model's previous preference estimate $\hat{\mathbf{p}}_{t-1}$. Specifically, the model scores all items in a pre-constructed catalog $\mathcal{J}$, ranks them by predicted rating, and selects $\boldsymbol{\theta}_t$ according to the induced ranking. Hyperparameters are tuned on training users from the $\boldsymbol{\theta}$-driven and $\mathbf{p}$-driven settings and then reused here. Consequently, we generate a separate $\hat{\mathbf{p}}$-driven dataset for each model and evaluate each model only on the dataset induced by its own selection policy. Figure 3(c) shows that the selected items tend to follow $\hat{\mathbf{p}}_{t-1}$ rather than the ground-truth preference vector $\mathbf{p}_t$, which can delay exposure to a newly preferred topic after a preference change.

4.6.1 Step 1: Item catalog generation. An item catalog $\mathcal{J}$ is constructed to span diverse topic-mixture structures. Each item is sampled as $\boldsymbol{\theta} \sim \text{Dirichlet}(\boldsymbol{\alpha})$, where a selected subset of topic coordinates is assigned a high concentration and all remaining coordinates are assigned $\alpha_j = 2$. Three item families are created, ranging from sharply focused to more diffuse mixtures:

- 1-topic items: For each topic $k \in \{1, \ldots, 10\}$, set $\alpha_k = 20$ and $\alpha_j = 2$ for all $j \neq k$, and sample 50 items. This yields 500 items.
- 2-topic items: For each unordered pair $\{i, j\}$ with $1 \leq i < j \leq 10$, set $\alpha_i = \alpha_j = 20$ and $\alpha_m = 2$ for all $m \notin \{i, j\}$, and sample 50 items per pair. This yields $45 \times 50 = 2{,}250$ items.
- 4-topic items: For each unordered 4-tuple $\{i, j, k, l\}$ of distinct topics, set $\alpha_i = \alpha_j = \alpha_k = \alpha_l = 8$ and $\alpha_u = 2$ for all $u \notin \{i, j, k, l\}$, and sample one item per 4-tuple. This yields $\binom{10}{4} = 210$ items.

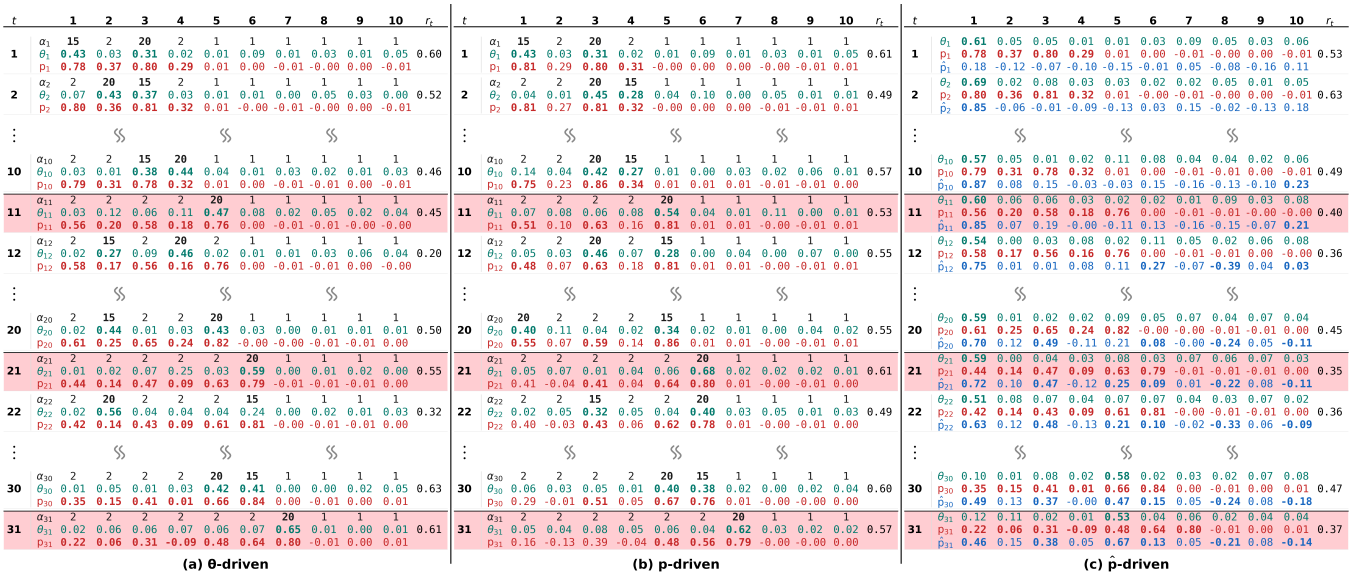
Altogether, $|\mathcal{J}| = 2{,}960$. In Figure 3(c), the teal vectors $\boldsymbol{\theta}_t$ are then drawn from this catalog.

4.6.2 Step 2: Item selection using $\hat{\mathbf{p}}_{t-1}$. Let $\mathcal{J}_t \subseteq \mathcal{J}$ denote the set of items remaining in the catalog at time step t. For each $\boldsymbol{\theta} \in \mathcal{J}_t$, the model computes predicted ratings $\hat{r}_t(\boldsymbol{\theta}) = \hat{\mathbf{p}}_{t-1}^\top \boldsymbol{\theta}$, ranks items in descending order of $\hat{r}_t(\boldsymbol{\theta})$, and partitions the ranked list into the top-2% set $\mathcal{T}_t$ and the rest $\mathcal{B}_t = \mathcal{J}_t \setminus \mathcal{T}_t$. Two selection policies are used to create two versions of the $\hat{\mathbf{p}}$-driven dataset for each model:

- *Top-2% selection:* Choose the item uniformly at random from $\mathcal{T}_t$.
- *Mixed selection:* Sample uniformly from $\mathcal{T}_t$ with probability 0.75, and from $\mathcal{B}_t$ with probability 0.25.

The selected item is denoted by $\boldsymbol{\theta}_t$ and removed from the catalog, i.e., $\mathcal{J}_{t+1} = \mathcal{J}_t \setminus \{\boldsymbol{\theta}_t\}$. As shown in Figure 3(c), the selected teal vector $\boldsymbol{\theta}_t$ tends to align with the strongest coordinates of the blue estimated preference vector $\hat{\mathbf{p}}_{t-1}$ rather than the red ground-truth preference vector $\mathbf{p}_{t-1}$, which can delay exposure to a newly preferred topic after a preference change.

4.6.3 Step 3: Preference vector generation, rating, label, and model update. After item selection, the ground-truth preference vector $\mathbf{p}_t$



(a) θ-driven (b) p-driven (c) p̂-driven

Figure 1: Sample user timelines under the preference shifting scenario across the 3 recommendation settings. Columns 1 through 10 correspond to the $K = 10$ latent topics. Each time step row shows the Dirichlet concentration vector α_t (black), the topic vector θ_t (teal), the ground-truth preferences $\mathbf{p}_t$ (red), and in the p̂-driven setting, the estimated preferences $\hat{\mathbf{p}}_t$ (blue). The column r_t reports the ground-truth rating. Red-shaded rows indicate preference-change time steps. Bold values highlight high Dirichlet concentrations ($\alpha_{t,k} \geq 10$), dominant topic weights ($\theta_{t,k} \geq 0.25$), and preference weights for explored topics.

is generated according to Section 4.3, independent of the model's estimate $\hat{\mathbf{p}}_{t-1}$. The ground-truth rating $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t$ and the preference-change label are then computed as in Step 3 of Section 4.4. After observing the interaction $(\boldsymbol{\theta}_t, r_t)$, the model updates its estimate to obtain $\hat{\mathbf{p}}_t$, which is used to select the item at the next time step.

5 Real Dataset

The real dataset is constructed by collecting reading histories for 26,374 users from Goodreads [35], where each user-item interaction has an associated review and a 1–5 star rating. Overall, the extracted dataset references 1,294,532 unique books. For each book, a text description is created by concatenating the original book description with the book's associated user reviews, excluding reviews written by the evaluated user to avoid target-user information leakage. A 768-dimensional contextual embedding is then created for each book's combined description using *jina-embeddings-v3* [32] and used only for identifying highly similar book pairs, whereas the online learning models uses the LDA topic vectors described in Appendix C.

To create examples for the preference change detection task, we assume that a large rating discrepancy between two highly similar items in a user's consumption history provides proxy evidence of preference change, while identical ratings suggest stable preferences. Note that these labels are imperfect proxies of preference change, as rating differences may also be caused by factors such as item quality, mood, context, or rating noise. Nevertheless, given that true latent user preferences are not directly observable in real-world datasets, we decided to leverage explicit ratings as an observable behavioral signal. Let j and k denote two time steps in a user's timeline with $j < k$. The pair (j, k) is treated as a *valid pair* if the corresponding books share the same author or belong to the same series, while their contextual embeddings have a high cosine similarity, i.e., $cos(\mathbf{e}_j, \mathbf{e}_k) > \tau$, where $\tau \in \{0.90, 0.95\}$. For each valid pair (j, k), the absolute rating difference $\Delta r = |r_j - r_k|$ is computed and pairs with $\Delta r \geq 2$ vs. $\Delta r = 0$ are considered as showing changed vs. stable preferences, respectively. To reduce the chance that preference change is induced by unobserved external factors, the two interactions are further required to occur close in time. Accordingly, three sets of valid pairs are constructed by upper-bounding the real-time gap $\Delta t = t_k - t_j$ as follows: all pairs, i.e., $\Delta t < \infty$; within 2 weeks, i.e., $\Delta t \leq 14$ days; and within a week, i.e., $\Delta t \leq 7$ days.

Finally, binary labels are assigned as follows: *stable* preferences for valid pairs with $\Delta r = 0$ vs. *changed* preferences for valid pairs with $\Delta r \geq 2$. Pairs with $\Delta r = 1$ are excluded as potentially ambiguous. Figure 2 illustrates one representative *stable*-preferences pair and one representative *changed*-preferences pair, while Table 2 provides useful statistics for the extracted dataset.

6 Evaluation Methodology

Evaluating recommender systems in non-stationary settings requires assessing two complementary capabilities: (i) **rating prediction**, namely how accurately a model predicts user ratings; and (ii) **preference tracking**, namely how well the model tracks changing user preferences. The online learning methods in Section 3 are trained to minimize squared error, either directly, e.g., AROW, or indirectly, e.g. BLR. Therefore, we use MSE to quantify overall rating prediction performance.

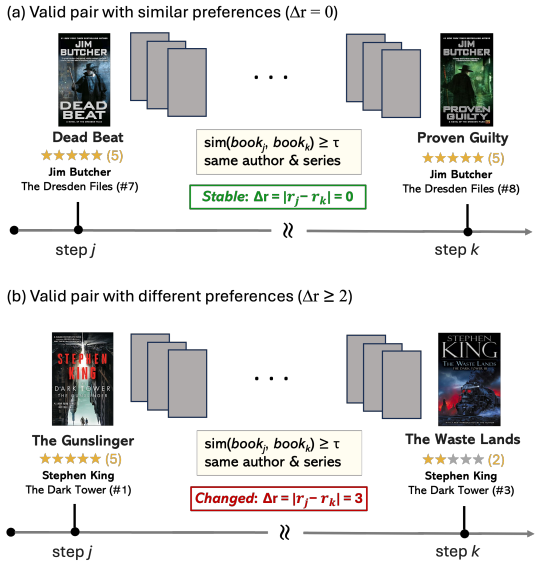


Figure 2: Illustration of preference change labels in the real dataset extracted from Goodreads.

Table 2: Real dataset statistics: distribution of valid book pairs across rating differences Δr and time-gap constraints, for similarity thresholds $\tau = 0.95$ and $\tau = 0.90$.

	$\tau = 0.95$			$\tau = 0.90$		
Δr	$\Delta t < \infty$	$\Delta t \leq 14$d	$\Delta t \leq 7$d	$\Delta t < \infty$	$\Delta t \leq 14$d	$\Delta t \leq 7$d
0	76,735	38,747	32,769	513,587	197,934	158,494
1	37,154	15,412	12,360	297,255	93,209	70,887
2	5,129	1,652	1,293	47,530	12,076	8,899
≥ 3	1,333	411	338	12,168	2,848	2,137

6.1 Evaluation Metrics for Synthetic Data

In the synthetic benchmark, the availability of ground-truth preference vectors and time step level preference-change labels enable direct evaluation of preference tracking, whereas ground-truth ratings enable evaluation of rating prediction. Rating prediction is reported as the MSE between predicted and observed ratings, averaged over all test users within each synthetic setting.

6.1.1 Preference Tracking. Preference tracking is evaluated along two complementary objectives: *preference change detection*, and *preference recovery*. To assess preference change detection performance, at each time step $t \geq 11$ a preference change score between consecutive time steps is computed as $s_t = D_{\mathrm{KL}}(\hat{\mathbf{p}}_t \,\|\, \hat{\mathbf{p}}_{t-1})$. Thresholded versions of this score are then evaluated against the ground-truth preference change labels using the area under the ROC curve (*ROC-AUC*) and the area under the precision-recall curve (*PR-AUC*).

Preference recovery measures how quickly a model's estimated preferences realign with the ground-truth preferences after a change. Let t_e denote the time step of a ground-truth preference change event e, and let t_{e+1} denote the time step of the next change event. The relative preference error at time step t is defined as $RelPE_t = \frac{1}{K}\sum_{k=1}^{K} \frac{|p_{t,k} - \hat{p}_{t,k}|}{\max(|p_{t,k}|, \epsilon)}$, with $\epsilon > 0$ for numerical stability. The tracking lag for event e is $L_e = \min\{t - t_e \,|\, t_e \leq t < t_{e+1} \wedge RelPE_t \leq 0.25\}$; if no such t exists, L_e is set to 10, matching the 10-step window between consecutive preference change candidates in the synthetic timelines. Three recovery metrics are introduced:

- **Mean Tracking Lag** $MTL = \frac{1}{N}\sum_{e=1}^{N} T_e$, where $T_e = \min(L_e, 10)$ and N is the total number of preference change events.
- **Tracking Ratio** TR as the fraction of events for which $RelPE_t \leq 0.25$ occurs for some t with $t_e \leq t < t_{e+1}$.
- **Lag** is the MTL computed only over the subset of true preference-change events for which $L_e < 10$, i.e., the model successfully "catches up" to the new true preferences before the next change.

Table 3: Performance in the $\hat{\mathbf{p}}$-driven setting under *mixed selection* of items. MSE↓ of rating prediction; ROC↑ and PR↑ areas for preference change detection (%); TR↑ is tracking ratio (%); Lag↓ is mean tracking lag over recovered events; MTL↓ is mean tracking lag over all events (Section 6.1).

	Preference Shifting					Preference Broadening				
Model	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**
KF-AF	**0.009**	**55.9**	19.9	**4.5/7.3**	**9.9**	**0.010**	53.5	19.2	**27.9/3.8**	**8.3**
AROW	0.016	53.3	19.1	1.5/8.0	10.0	0.024	**57.6**	21.8	0.8/8.0	10.0
BLR	0.015	53.6	19.3	0.0/∞	10.0	0.025	**57.6**	22.4	0.0/∞	10.0
BLR-VB	0.010	49.4	17.4	0.8/9.0	10.0	0.014	49.6	18.0	10.9/5.7	9.5
BLR-FF	0.014	55.7	**20.3**	0.0/∞	10.0	0.021	56.1	19.5	0.0/∞	10.0
BLR-SW	0.012	51.0	19.5	0.0/∞	10.0	0.023	51.5	18.7	0.0/∞	10.0
BLR-PP	0.014	54.1	**20.7**	0.0/∞	10.0	0.030	59.3	**22.9**	0.0/∞	10.0
BLR-NIG	0.018	53.1	17.1	0.8/8.0	10.0	0.017	52.1	21.2	2.3/**5.0**	9.9

6.2 Evaluation Metrics for Real Data

Since ground-truth preference vectors are unknown for real data, preference tracking is evaluated using the high-similarity interaction pairs constructed in Section 5. For each valid pair of time steps $j < k$, we first compute the ground-truth rating change $\Delta r = |r_j - r_k|$ and the preference change score $s_{j,k} = D_{\mathrm{KL}}(\hat{\mathbf{p}}_k \,\|\, \hat{\mathbf{p}}_j)$. Rating prediction performance is reported as the MSE between true and estimated ratings, averaged over all test users. We use two complementary measures:

- **SRC-Δ-KL**: The Spearman rank correlation (SRC) between the ground-truth rating change Δr and the model score $s_{j,k}$ is computed over valid pairs. Higher **SRC-Δ-KL** indicates stronger alignment of inferred preference change with true rating change.
- **ROC-Δ-KL** and **PR-Δ-KL**: The ROC-AUC and PR-AUC are computed over valid pairs (stable: $\Delta r = 0$; changed: $\Delta r \geq 2$), by thresholding over $s_{j,k}$ as the preference-change score.

7 Experimental Results

This section reports results on the synthetic and real datasets. A central finding emerges across both: models with similar rating prediction performance can exhibit substantially different capabilities in tracking non-stationary user preferences.

7.1 Results on Synthetic Data

Table 8 reports performance in the $\hat{\mathbf{p}}$-driven setting for the preference shifting (PS) and preference broadening (PB) scenarios. We

focus on the $\hat{\mathbf{p}}$-driven setting because it most closely reflects real recommendation settings, where item selection at each time step depends on the model's own estimate $\hat{\mathbf{p}}_{t-1}$. A complete set of results also showing the $\boldsymbol{\theta}$- and $\mathbf{p}$-driven settings and the PSC1T and PBC1T conservation-augmented scenarios are provided in Appendix E.1.

A central insight provided by these empirical results is the dissociation between rating prediction and preference tracking. In the PS scenario, the KF-AF and BLR-VB models attain nearly identical MSEs of 0.009 and 0.010, yet KF-AF recovers the new preference state on 4.5% of preference change events before the next change, while BLR-VB catches up on only 0.8% (p-value < 0.01 from one-tailed paired t-test). In the PB setting, KF-AF and BLR-VB again obtain the lowest MSEs of 0.010 and 0.014, however KF-AF catches up on 27.9% of events versus BLR-VB's 10.9% (p-values < 0.01), while every remaining method except BLR-NIG is stuck at 0.0% tracking performance. Overall, models with near-identical aggregate rating prediction error exhibit substantially different tracking performance, confirming that rating prediction MSE alone is insufficient for characterizing responsiveness to non-stationary preference behavior. Patterns consistent with this observation also hold in the $\boldsymbol{\theta}$- and $\mathbf{p}$-driven settings reported in Appendix E.1.

Another insight is that preference change detection is not necessarily correlated with tracking responsiveness. For example, BLR-PP achieves the highest detection PR in both scenarios (20.7% and 22.9%) and the highest detection ROC in PB (59.3%), but its tracking ratio is 0.0% in both: the posterior shifts enough for KL-based detection yet never catches up to the new ground-truth preference state before the next change. BLR-FF and BLR-SW show a similar pattern. This motivates separate measures of change detection and tracking, distinct from aggregate rating prediction.

Underlying this dissociation is the interplay between *plasticity* and *stability*. KF-AF's dynamical-state formulation injects process noise and adapts a time-varying forgetting factor γ_t online via SGD on the one-step predictive log-likelihood, preventing the posterior from over-concentrating during stable periods while enabling rapid re-alignment when observations contradict the current belief. AROW's KL-regularized objective penalizes deviations from the previous posterior, prioritizing stability over plasticity. BLR-VB's variance floor is a passive safeguard rather than an active dynamical mechanism: it preserves capacity to adapt when the preference update concentrates weight on a single new topic (PB), but not when previously explored topics must also be downweighed (PS).

The $\hat{\mathbf{p}}$-driven setting amplifies this dissociation through a self-reinforcing feedback loop: outdated preferences drive item selection, which generates ratings consistent with the outdated belief, which reinforces that belief. MSE remains small because the predicted rating is computed using the posterior estimate $\hat{\mathbf{p}}_t$; the Bayesian update moves the posterior mean along $\boldsymbol{\theta}_t$ by an amount that reduces the residual $\hat{\mathbf{p}}_t^\top \boldsymbol{\theta}_t - r_t$, so the predicted rating closely tracks the ground-truth $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t$ on the just-observed item regardless of whether $\hat{\mathbf{p}}_t$ has caught up to $\mathbf{p}_t$. Stochastic exploration in the mixed selection protocol (0.25 probability of sampling from the bottom 98%) preserves partial topic diversity, which is why KF-AF, BLR-VB, and BLR-NIG achieve nonzero tracking in PB, but this exploration is insufficient to prevent filter-bubble collapse for most methods. The tracking measures in this setting therefore isolate a failure mode that is invisible to aggregate rating prediction accuracy.

Table 4: Performance on real data. MSE↓ of rating prediction; SRC↑ is Spearman rank correlation between rating discrepancy Δr and the model's KL-based preference change score; PR↑ is area (%) for preference change detection (Section 6.2).

		sim ≥ 0.95				sim ≥ 0.90			
		$\Delta t < \infty$		$\Delta t \leq 7$d		$\Delta t < \infty$		$\Delta t \leq 7$d	
Model	**MSE**	**SRC**	**PR**	**SRC**	**PR**	**SRC**	**PR**	**SRC**	**PR**
KF-AF	0.78	**0.19**	**16.6**	**0.26**	**16.5**	**0.17**	**20.1**	**0.25**	**19.9**
AROW	**0.76**	0.18	14.4	0.22	15.1	0.16	17.3	0.21	17.8
BLR	0.77	0.17	14.1	0.20	13.6	0.15	17.0	0.18	16.1
BLR-VB	**0.76**	0.18	14.7	0.23	15.3	0.16	17.9	0.21	17.7
BLR-FF	0.81	0.17	12.8	0.25	15.2	0.14	15.5	0.24	18.3
BLR-SW	0.80	0.15	14.5	0.14	8.9	0.14	18.9	0.15	11.6
BLR-PP	0.77	0.13	13.3	0.07	7.6	0.12	16.4	0.08	9.9
BLR-NIG	0.78	0.16	12.6	0.19	13.8	0.13	14.9	0.18	16.1

7.2 Results on Real Data

Table 4 reports performance on the real-data portion of the benchmark at $\Delta t < \infty$ and $\Delta t \leq 7$ days under two similarity thresholds $\tau \in \{0.95, 0.90\}$, which control how near-duplicate two items in a valid pair must be. A full table also showing ROC-Δ-KL numbers and results for $\Delta t \leq 14$ days is provided in Appendix E.2.

The real-data exhibits the same rating prediction vs. tracking dissociation observed on synthetic data. AROW and BLR-VB achieve the best rating prediction in terms of MSE, yet on the more reliable within-one-week setting they fall below KF-AF on every tracking metric at both similarity thresholds ((p-value < 0.01). Overall, KF-AF achieves the strongest tracking performance (SRC and PR) while remaining competitive in rating prediction. Comparing the two similarity thresholds reveals complementary trends across metrics. The stricter threshold $\tau \geq 0.95$ yields higher SRC for most models because near-identical item pairs provide a cleaner preference-change signal. The lower threshold $\tau \geq 0.90$ yields higher PR because it admits substantially more valid pairs and thereby more positive-class pairs ($\Delta r \geq 2$), which reduces class imbalance.

8 Conclusion

We introduced a benchmark suite for evaluating online learning models on their ability to track non-stationary user preferences. The suite comprises synthetic datasets spanning six behavior scenarios and three item-recommendation settings, and a real dataset constructed from Goodreads. Eight online learning methods are evaluated under novel metrics that quantify preference-change detection and preference tracking. Experimental evaluations reveal an empirical dissociation between aggregate rating prediction and responsiveness to preference changes: models with near-identical rating prediction accuracy exhibit substantially different user preference tracking performance. Model-driven item selection further amplifies this dissociation through self-reinforcing feedback loops that suppress informative signal about preference changes even when aggregate rating prediction is good. A Kalman filter-based approach demonstrates the strongest tracking performance while being competitive in rating prediction, though even it does not fully track user preferences within a cycle. These results underscore the need for explicit user preference tracking metrics to rigorously evaluate and improve recommender systems under non-stationarity.

References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 2623–2631. doi:10.1145/3292500.3330701

[2] Vito Walter Anelli, Alejandro Bellogin, Antonio Ferrara, Daniele Malitesta, Felice Antonio Merra, Claudio Pomo, Francesco Maria Donini, and Tommaso Di Noia. 2021. Elliot: A Comprehensive and Rigorous Framework for Reproducible Recommender Systems Evaluation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 2405–2414. doi:10.1145/3404835.3463245

[3] James Bennett and Stan Lanning. 2007. The Netflix Prize.

[4] Christopher M Bishop. 2006. Pattern Recognition and Machine Learning. (2006).

[5] Pedro G. Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24, 1 (Feb. 2014), 67–119. doi:10.1007/s11257-012-9136-x

[6] Luciano Caroprese, Francesco Sergio Pisani, Bruno Miguel Veloso, Matthias Konig, Giuseppe Manco, Holger Hoos, and Joao Gama. 2025. Modelling Concept Drift in Dynamic Data Streams for Recommender Systems. *ACM Trans. Recomm. Syst.* 3, 2, Article 24 (March 2025), 28 pages. doi:10.1145/3707693

[7] Chao Chen, Dongsheng Li, Junchi Yan, and Xiaokang Yang. 2022. Modeling Dynamic User Preference via Dictionary Learning for Sequential Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 34, 11 (2022), 5446–5458. doi:10.1109/TKDE.2021.3050407

[8] Koby Crammer, Alex Kulesza, and Mark Dredze. 2009. Adaptive Regularization of Weight Vectors. In *Advances in Neural Information Processing Systems*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (Eds.), Vol. 22. Curran Associates, Inc., Vancouver, British Columbia, Canada. https://proceedings.neurips.cc/paper_files/paper/2009/file/8ebda540cbcc4d7336496819a46a1b68-Paper.pdf

[9] Romain Deffayet, Thibaut Thonet, Dongyoon Hwang, Vassilissa Lehoux, Jean-Michel Renders, and Maarten de Rijke. 2024. SARDINE: Simulator for Automated Recommendation in Dynamic and Interactive Environments. *ACM Trans. Recomm. Syst.* 2, 3, Article 25 (June 2024), 34 pages. doi:10.1145/3656481

[10] Farzad Eskandanian and Bamshad Mobasher. 2019. Modeling the Dynamics of User Preferences for Sequence-Aware Recommendation Using Hidden Markov Models. In *FLAIRS*. AAAI Press, Sarasota, Florida, USA, 425–430.

[11] João Gama, Indrundefined Žliobaitundefined, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM Comput. Surv.* 46, 4, Article 44 (March 2014), 37 pages. doi:10.1145/2523813

[12] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[13] Hartatik, Lukman Heryawan, and Reza Pulungan. 2025. Trust Decay-Based Temporal Learning for Dynamic Recommender Systems With Concept Drift Adaptation. *IEEE Access* 13 (2025), 110955–110971. doi:10.1109/ACCESS.2025.3583186

[14] Tonmoy Hasan and Razvan Bunescu. 2023. Topic-Level Bayesian Surprise and Serendipity for Recommender Systems. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 933–939. doi:10.1145/3604915.3608851

[15] Hongtao Huang, Chengkai Huang, Tong Yu, Xiaojun Chang, Wen Hu, Julian McAuley, and Lina Yao. 2026. Dual Conditional Diffusion for Sequential Recommendation. In *Proceedings of the Nineteenth ACM International Conference on Web Search and Data Mining* (USA) *(WSDM '26)*. Association for Computing Machinery, New York, NY, USA, 206–216. doi:10.1145/3773966.3777926

[16] Wasim Huleihel, Soumyabrata Pal, and Ofer Shayevitz. 2021. Learning User Preferences in Non-Stationary Environments. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 130)*, Arindam Banerjee and Kenji Fukumizu (Eds.). PMLR, Virtual, 1432–1440.

[17] Joseph G. Ibrahim and Ming-Hui Chen. 2000. Power Prior Distributions for Regression Models. *Statist. Sci.* 15, 1 (2000), 46–60. http://www.jstor.org/stable/2676676

[18] Joseph G Ibrahim, Ming-Hui Chen, Yeongjin Gwon, and Fang Chen. 2015. The power prior: theory and applications. *Statistics in medicine* 34, 28 (2015), 3724–3749.

[19] Eugene Ie, Chih-wei Hsu, Martin Mladenov, Vihan Jain, Sanmit Narvekar, Jing Wang, Rui Wu, and Craig Boutilier. 2019. Recsim: A configurable simulation platform for recommender systems.

[20] Rolf Jagerman, Ilya Markov, and Maarten de Rijke. 2019. When People Change their Mind: Off-Policy Evaluation in Non-stationary Recommendation Environments. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* (Melbourne VIC, Australia) *(WSDM '19)*. Association for Computing Machinery, New York, NY, USA, 447–455. doi:10.1145/3289600.3290958

[21] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. 2023. A Critical Study on Data Leakage in Recommender System Offline Evaluation. *ACM Trans. Inf. Syst.* 41, 3, Article 75 (Feb. 2023), 27 pages. doi:10.1145/3569930

[22] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, Singapore, 197–206. doi:10.1109/ICDM.2018.00035

[23] Bart P. Knijnenburg, Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. 2012. Explaining the user experience of recommender systems. *User Modeling and User-Adapted Interaction* 22, 4 (Oct. 2012), 441–504. doi:10.1007/s11257-011-9118-4

[24] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Paris, France) *(KDD '09)*. Association for Computing Machinery, New York, NY, USA, 447–456. doi:10.1145/1557019.1557072

[25] Qidong Liu, Xian Wu, Yejing Wang, Zijian Zhang, Feng Tian, Yefeng Zheng, and Xiangyu Zhao. 2024. LLM-ESR: Large Language Models Enhancement for Long-tailed Sequential Recommendation. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., Vancouver, Canada, 26701–26727. doi:10.52202/079017-0839

[26] Haokai Ma, Ruobing Xie, Lei Meng, Xin Chen, Xu Zhang, Leyu Lin, and Zhanhui Kang. 2024. Plug-In Diffusion Model for Sequential Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. AAAI Press, Vancouver, Canada, 8886–8894. doi:10.1609/aaai.v38i8.28736

[27] Zaiqiao Meng, Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2020. Exploring Data Splitting Strategies for the Evaluation of Recommendation Models. In *Proceedings of the 14th ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 681–686. doi:10.1145/3383313.3418479

[28] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 188–197. doi:10.18653/v1/D19-1018

[29] Jeonglyul Oh and Sungzoon Cho. 2024. Measuring Recency Bias In Sequential Recommendation Systems. arXiv:2409.09722 [cs.IR] Accepted at the CONSEQUENCES '24 workshop, co-located with ACM RecSys '24.

[30] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. 2018. Recogym: A reinforcement learning environment for the problem of product recommendation in online advertising.

[31] Seongho Son, William Bankes, Sayak Ray Chowdhury, Brooks Paige, and Ilija Bogunovic. 2025. Right Now, Wrong Then: Non-Stationary Direct Preference Optimization under Preference Drift. In *Proceedings of the 42nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 267)*. PMLR, 56173–56203.

[32] Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Nan Wang, et al. 2024. jina-embeddings-v3: Multilingual embeddings with task lora.

[33] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1441–1450. doi:10.1145/3357384.3357895

[34] Michalis K. Titsias, Alexandre Galashov, Amal Rannen-Triki, Razvan Pascanu, Yee Whye Teh, and Jorg Bornschein. 2024. Kalman Filter for Online Classification of Non-Stationary Data. In *The Twelfth International Conference on Learning Representations (ICLR)*. OpenReview.net, Vienna, Austria. https://openreview.net/forum?id=ZzmKEpze8e

[35] Mengting Wan and Julian McAuley. 2018. Item Recommendation on Monotonic Behavior Chains. In *Proceedings of the 12th ACM Conference on Recommender Systems* (Vancouver, British Columbia, Canada) *(RecSys '18)*. Association for Computing Machinery, New York, NY, USA, 86–94. doi:10.1145/3240323.3240369

[36] Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2021. Denoising Implicit Feedback for Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. Association for Computing Machinery, New York, NY, USA, 373–381. doi:10.1145/3437963.3441800

[37] Shuhei Watanabe. 2025. Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance. arXiv:2304.11127 [cs.LG] https://arxiv.org/abs/2304.11127

[38] Xiao Xu, Fang Dong, Yanghua Li, Shaojian He, and Xin Li. 2020. Contextual-Bandit Based Personalized Recommendation with Time-Varying User Interests.

Proceedings of the AAAI Conference on Artificial Intelligence 34, 04 (Apr. 2020), 6518–6525. doi:10.1609/aaai.v34i04.6125

[39] Yishi Xu, Yingxue Zhang, Wei Guo, Huifeng Guo, Ruiming Tang, and Mark Coates. 2020. GraphSAIL: Graph Structure Aware Incremental Learning for Recommender Systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 2861–2868. doi:10.1145/3340531.3412754

[40] Hyunsik Yoo, SeongKu Kang, Ruizhong Qiu, Charlie Xu, Fei Wang, and Hanghang Tong. 2025. Embracing Plasticity: Balancing Stability and Plasticity in Continual Recommender Systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Padua, Italy) *(SIGIR '25)*. Association for Computing Machinery, New York, NY, USA, 2092–2101. doi:10.1145/3726302.3729964

[41] Hyunsik Yoo, Seongku Kang, and Hanghang Tong. 2025. Continual Recommender Systems. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management* (Seoul, Republic of Korea) *(CIKM '25)*. Association for Computing Machinery, New York, NY, USA, 6857–6860. doi:10.1145/3746252.3761452

[42] Changshuo Zhang, Xiao Zhang, Teng Shi, Jun Xu, and Ji-Rong Wen. 2025. Test-Time Alignment with State Space Model for Tracking User Interest Shifts in Sequential Recommendation. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems (RecSys '25)*. Association for Computing Machinery, New York, NY, USA, 461–471. doi:10.1145/3705328.3748060

[43] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, Yingqian Min, Zhichao Feng, Xinyan Fan, Xu Chen, Pengfei Wang, Wendi Ji, Yaliang Li, Xiaoling Wang, and Ji-Rong Wen. 2021. RecBole: Towards a Unified, Comprehensive and Efficient Framework for Recommendation Algorithms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 4653–4664. doi:10.1145/3459637.3482016

[44] Xing Zhao, Ziwei Zhu, and James Caverlee. 2021. Rabbit Holes and Taste Distortion: Distribution-Aware Recommendation with Evolving Interests. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 888–899. doi:10.1145/3442381.3450099

A Related Work

The related work is organized into three strands: non-stationary preference modeling in recommender systems, benchmarks and evaluation under temporal dynamics, and the stability–plasticity perspective on adaptation and user satisfaction. Together, these lines of work motivate the need for benchmarks that evaluate not only aggregate recommendation accuracy, but also preference change detection and post-change recovery.

Non-stationary preference modeling in recommender systems. Temporal dynamics have long been incorporated into collaborative filtering. Early work such as timeSVD++ models time-varying user and item biases and allows user latent factors to evolve over time, improving rating prediction under evolving preferences [24]. More broadly, the concept drift literature decomposes learning under non-stationarity into design choices about memory and forgetting (e.g., sliding windows or fading schemes), change detection (e.g., monitoring prediction error with sequential tests or comparing distributions across reference and recent windows), and adaptation strategies ranging from blind continual updating (without explicit change detection) to informed (detector-triggered) adaptation [11]. This perspective aligns naturally with preference-shift settings in recommender systems, where models must both recognize shifts and update user representations rapidly after a change. Sequential recommenders can condition on recent interaction histories to reflect evolving behavior, but they are typically evaluated via next-item accuracy rather than explicit preference-change detection and recovery [15, 22, 25, 26, 33]. More explicit approaches use Hidden Markov Models to detect preference change points in user interaction sequences and generate recommendations aligned with users' inferred current tastes [10].

Related empirical work shows that attempts to mitigate the rabbit-hole effect via diversity-focused and distribution-aware recommendation can introduce a taste distortion problem, where the taste distribution reflected in recommendation outcomes diverges from a user's true, evolving tastes; TecRec addresses this by predicting a user's future taste distribution from recent history and applying post-ranking calibration to match recommendations to that predicted distribution [44]. Other recent directions include test-time adaptation for sequential recommendation, where a state-space-model backbone is updated during inference using self-supervised alignment objectives to better track user interest shifts without full retraining [42]. On the theoretical side, non-stationary collaborative filtering has been formalized by modeling users as latent preference types that may change over time and analyzing a batched online algorithm that detects preference variations and adapts recommendations accordingly, with regret guarantees that quantify a "price of non-stationarity" due to preference changes [16]. While these works model evolving preferences or mitigate preference shifts, they are typically evaluated through downstream accuracy metrics (e.g., MSE or top-N ranking), leaving explicit benchmarking of preference *tracking*, including change detection and post-change recovery, underdeveloped.

Benchmarks and evaluation under temporal dynamics. Standard datasets such as MovieLens and the Netflix Prize provide temporal interaction logs that support time-aware evaluation, but they do not provide ground-truth annotations of when user preferences change [3, 12]. Large-scale review datasets such as Amazon Reviews and Goodreads similarly provide rich histories and text signals but lack labeled preference-change events [28, 35]. Prior work has shown that temporal evaluation choices can substantially alter algorithm rankings and that offline evaluation can suffer from leakage when the global timeline is not respected [5, 21]. Systematic reviews have also identified inconsistencies in data splitting and experimental protocols [27].

Tooling efforts such as RecBole and Elliot standardize experimental pipelines and temporal splits [2, 43], but provide evaluation infrastructure rather than controlled preference-change benchmarks. RecSim and RecoGym support sequential interaction and feedback-loop analysis [19, 30]. SARDINE updates the simulated user state from the interaction history: clicked items move the user embedding toward their mean embedding, and repeated clicks on the same topic temporarily zero either the corresponding topic component or the entire user embedding [9]. It evaluates recommendation policies using cumulative clicks and the number of bored time steps, rather than measuring whether a model detects a known preference change or how quickly its estimated preferences recover afterward.

Closely related work creates synthetic datasets with controlled preference drift. Caroprese et al. modify matrix-factorization embeddings over time and use drift boundaries to divide the interaction stream into regimes [6]. Son et al. assign preference examples to a 100-step timeline and use two reward models to create either a sudden switch at a change point or a gradual transition through linear reward interpolation [31]. NS-DPO discounts older examples to optimize an LLM policy for the preference distribution at the final time step. In contrast to these lines of work, our benchmark suite

complements these resources by providing (i) synthetic datasets with ground-truth latent preferences and time step-level preference-change labels, and (ii) a real dataset that uses near-duplicate items and rating discrepancies as signals for preference change, enabling direct measurement of change detection and post-change recovery/lag rather than evaluating drift mainly through regime segmentation and downstream recommendation accuracy. We address this gap by separating (1) the procedure for generating the true preferences of the simulated users, from (2) the procedure for selecting items the user consumes at each time step. Unlike these approaches, our benchmark suite separates (1) the procedure for generating the ground-truth preferences of simulated users from (2) the procedure for selecting the item consumed at each time step. Consequently, preference changes occur independently of the item-selection history and can be labeled at individual time steps. Together with a real dataset that uses near-duplicate items and rating discrepancies as signals of preference change, this design enables direct measurement of preference-change detection and post-change recovery lag, rather than evaluating non-stationarity through cumulative interaction rewards or downstream recommendation accuracy.

Stability–plasticity and user satisfaction. The stability–plasticity dilemma is widely used to frame continual recommendation, capturing the tension between retaining previously learned user representations and rapidly adapting to preference shifts [41]. A recent continual recommendation method operationalizes this trade-off by first quantifying preference shift as changes in a user's distances to item clusters across successive time stages, and then updating user representations by aligning them to prior-stage versus current-stage knowledge using an InfoNCE-style mutual-information objective to encourage stability for stable users and plasticity for dynamic users [40]. Complementarily, stability-oriented incremental-learning approaches for graph-based recommenders mitigate forgetting during frequent updates via structure-aware distillation, including self-embedding regularization and preservation of local and global graph structure [39]. User-centric evaluation work argues that prediction accuracy captures only part of recommender-system user experience, and emphasizes subjective perceptions and outcomes such as perceived recommendation quality, effort, and choice satisfaction [23]. Tracking lag matters precisely at this intersection: even if long-run MSE remains strong, delayed adaptation creates an immediate window of mismatched recommendations that can reduce item relevance and undermine trust. Motivated by these perspectives, this benchmark evaluates not only aggregate accuracy but also tracking lag, capturing the user-critical post preference-change interval where delayed adaptation produces misaligned recommendations.

B Detailed Procedures for Synthetic Dataset Generation

This appendix provides the complete data generation procedures for all six user behavior scenarios under the three recommendation settings. The descriptions are self-contained and sufficient for exact reproduction of the synthetic benchmark, and correspond to the compressed descriptions in Sections 4.3–4.5 of the main text. All six scenarios use the same timeline construction, initialization, notation, rating computation, and preference-change labeling, which are presented in full for the Preference Shifting (PS) scenario in Appendix B.1. Unless stated otherwise, the initialization of $\mathbf{p}_0$, the boost operator, and all *p-Stable* step behavior in the other five scenarios are identical to those in the corresponding PS generator.

Compared with PS, the remaining five scenarios differ in only two things at *p-Change* time steps: the rule used to update the preference vector and, in the two-topic scenarios, the rule used to generate the item topic vector at some of those time steps. In scenarios with preference conservation, a candidate *p-Change* time step either produces a strong or a near-neutral preference weight update for the corresponding topics. In scenarios with two-topic events, some candidate *p-Change* time steps present an item with high weight on two topics rather than one newly introduced topic. Within each scenario, we first describe the $\boldsymbol{\theta}$-driven setting and then state how the $\mathbf{p}$-driven and $\hat{\mathbf{p}}$-driven settings differ. Figures 4–8 show representative user timelines for the five scenarios beyond PS, using the same visualization format as Figure 3 in the main paper.

B.1 Preference Shifting (PS)

This subsection provides the full data generation procedures for the Preference Shifting scenario under the $\boldsymbol{\theta}$-driven, $\mathbf{p}$-driven, and $\hat{\mathbf{p}}$-driven settings. The compressed descriptions in Sections 4.3–4.5 of the main text convey the three settings at a conceptual level; the content below is sufficient for exact reproduction.

B.1.1 $\boldsymbol{\theta}$-Driven Setting. In this setting, the item generation process starts by assuming a small set of topics $\mathcal{S}$ that the user has *explored* already. At each *p-Stable* time step t, a topic vector $\boldsymbol{\theta}_t$ is generated by randomly selecting two topics from $\mathcal{S}$ and assigning them high weights. Over these stable time steps, the ground-truth preference vector remains unchanged or changes only slightly due to noise. At *p-Change* time steps, topic vectors are generated with high topic weight on a topic the user has not explored yet. This creates a sequence in which the user is repeatedly exposed to familiar topics during stable periods, and then to a new topic at the start of each new cycle.

The data generation process proceeds according to the three major steps detailed below. A timeline of length T is considered with $K = 10$ topics and the explored topic set is initialized as $\mathcal{S} = \{1, \ldots, K_0\}$ with $K_0 = 4$. At each time step t, an item topic vector $\boldsymbol{\theta}_t$ is generated and then the ground-truth preference vector $\mathbf{p}_t$ is updated, followed by the rating computation. An example is shown in Figure 3(a).

Step 1: Topic vector generation. The topic vector $\boldsymbol{\theta}_t$ is sampled from a Dirichlet distribution, $\boldsymbol{\theta}_t \sim \text{Dirichlet}(\boldsymbol{\alpha}_t)$. The procedure begins from a default concentration pattern that distinguishes explored and unexplored topics: $\alpha_{t,k} = 2$ for all $k \in \mathcal{S}$, and $\alpha_{t,k} = 1$ for all $k \notin \mathcal{S}$. Selected topics are then promoted by overwriting their default concentration with higher values, as described below.

- *p-Stable* steps: Two distinct topics (k_a, k_b) are sampled uniformly from $\mathcal{S}$ and assigned high concentrations $\{15, 20\}$ (random order), i.e., $(\alpha_{t,k_a}, \alpha_{t,k_b}) \in \{(15, 20), (20, 15)\}$. This ensures $\boldsymbol{\theta}_t$ places high topic weight on two explored topics, yielding random exposure over $\mathcal{S}$. In Figure 3(a), this appears in the unshaded rows as two large entries in $\boldsymbol{\alpha}_t$, which in turn produce two dominant entries in $\boldsymbol{\theta}_t$ concentrated on already explored topics.

- *p-Change* steps: If $|\mathcal{S}| < K$, a new topic $k^\star = |\mathcal{S}|+1$ is introduced by updating $\mathcal{S} \leftarrow \mathcal{S} \cup \{k^\star\}$ and setting $\alpha_{t,k^\star} = 20$, while keeping $\alpha_{t,k} = 2$ for the explored topics. This guarantees that the generated item has a high topic weight on a newly introduced topic. In Figure 3(a), the shaded rows at $t = 11, 21, 31$ show this pattern clearly: the largest entry of $\boldsymbol{\alpha}_t$ moves to a newly introduced topic, and the sampled $\boldsymbol{\theta}_t$ places its dominant mass on that topic.

Step 2: Preference vector generation. We initialize the preference vector as $\mathbf{p}_0 = \mathbf{0}$ and define the preference-change operator $\text{boost}(p, \eta) = p + \eta(1-p)$ with $\eta_{\text{big}} = 0.8$ and $\eta_{\text{small}} = 0.3$. At $t = 1$, after generating the item with a topic vector $\boldsymbol{\theta}_1$ that has high weight on topics k_a and k_b, the preference weights are increased for these topics by a large amount, while for the remaining initially explored topics they are increased by a smaller amount:

$$p_{1,k} \leftarrow \text{boost}(p_{0,k}, \eta_{\text{big}}) \qquad \forall\, k \in \{k_a, k_b\},$$
$$p_{1,k} \leftarrow \text{boost}(p_{0,k}, \eta_{\text{small}}) \qquad \forall\, k \in \mathcal{S} \setminus \{k_a, k_b\}.$$

The preference weights for all unexplored topics $k \notin \mathcal{S}$ are initialized to small values, $p_{1,k} \sim \text{Unif}[-0.01, 0.01]$. For all subsequent time steps $t \geq 2$, the preferences are generated as follows:

- *p-Stable* steps: The preference vector is perturbed by Gaussian noise and clipping, i.e., $p_{t,k} \leftarrow \text{clip}(p_{t-1,k} + \varepsilon_{t,k}, -1, 1)$, where $\varepsilon_{t,k} \sim \mathcal{N}(0, \sigma^2)$ if $k \in \mathcal{S}$ and otherwise $\varepsilon_{t,k} \sim \mathcal{N}(0, \sigma^2)$ with $\varepsilon_{t,k}$ clipped to $[-0.01, 0.01]$. In Figure 3(a), this appears as small fluctuations in the red $\mathbf{p}_t$ rows across the unshaded time steps within each cycle.
- *p-Change* steps: When a new topic $k^\star$ is introduced with high weight, the *preference shift* update increases the preference weight for $k^\star$ by A and decreases the preference weights of all other previously explored topics $j \in \mathcal{S} \setminus \{k^\star\}$ proportionally, whenever $Z \geq A$:

$$A = \eta_{\text{big}}(1 - p_{t-1,k^\star}), \qquad p_{t,k^\star} \leftarrow p_{t-1,k^\star} + A,$$
$$Z = \sum_{j \in \mathcal{S} \setminus \{k^\star\}} (1 + p_{t-1,j}), \qquad p_{t,j} \leftarrow p_{t-1,j} - (1 + p_{t-1,j})\frac{A}{Z}$$

  For all remaining topics $k \notin \mathcal{S}_t$, preferences evolve only through the same noise injection and clipping rule used in the *p-Stable* step. In Figure 3(a), this update is visible at the shaded rows: when a new topic is introduced at $t = 11, 21, 31$, its corresponding entry in $\mathbf{p}_t$ goes up, while the previously dominant explored topics decrease. This produces the characteristic shift pattern that defines the PS scenario.

Step 3: Rating computation and preference-change label. After the preference-shift update at the *p-Change* time step, the rating $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t$ is computed and a binary preference-change label is assigned to time step t whenever the change in the preference vector exceeds a predefined threshold $\lambda = 0.25$, i.e., $\|\mathbf{p}_t - \mathbf{p}_{t-1}\|_2 > \lambda$. In Figure 3(a), the shaded rows correspond to *p-Change* time steps, and in the PS scenario these rows produce visible jumps in $\mathbf{p}_t$, consistent with positive preference-change events.

B.1.2 **p**-*Driven Setting.* In this setting, before any item is generated, the system is also assumed to have knowledge of a small set of topics $\mathcal{S}$ that the user has explored already. At each *p-Stable* time step t, the three topics with the largest preference weights in $\mathbf{p}_{t-1}$ are identified, two of these three topics are randomly selected, and an item topic vector $\boldsymbol{\theta}_t$ is generated that places high topic weights on the selected topics. At *p-Change* time steps, the topic vector is generated identically to the $\boldsymbol{\theta}$-driven setting.

The data generation process for the **p**-driven setting under the preference shifting scenario is described below. The same initialization of the timeline length T, topic count $K = 10$, and initial explored set $\mathcal{S}$ is used as in Appendix B.1.1. An example is shown in Figure 3(b). Compared with Figure 3(a), the main visible difference is that during unshaded *p-Stable* time steps, the dominant entries of $\boldsymbol{\alpha}_t$ and $\boldsymbol{\theta}_t$ tend to follow the strongest coordinates of the red preference vector $\mathbf{p}_{t-1}$, rather than being chosen uniformly from the explored topics.

Step 1: Topic vector generation. $\boldsymbol{\theta}_t \sim \text{Dirichlet}(\boldsymbol{\alpha}_t)$ is sampled using the same baseline concentrations as in Appendix B.1.1 ($\alpha_{t,k} = 2$ for $k \in \mathcal{S}$ and $\alpha_{t,k} = 1$ otherwise), but during *p-Stable* steps the topics that receive high Dirichlet concentrations are chosen based on $\mathbf{p}_{t-1}$.

- *p-Stable* steps: At $t = 1$, two distinct topics (k_a, k_b) are sampled uniformly at random from the initial explored set $\mathcal{S}$ as no preference signal is yet available. For $t \geq 2$, the three topics with the largest preference weights in $\mathbf{p}_{t-1}$ are identified, two of them are randomly selected, and they are assigned high concentrations $\{15, 20\}$ (random order), i.e., $(\alpha_{t,k_a}, \alpha_{t,k_b}) \in \{(15, 20), (20, 15)\}$. In Figure 3(b), this is visible in the unshaded rows: after the first cycle, the large entries in $\boldsymbol{\alpha}_t$ and the dominant entries in $\boldsymbol{\theta}_t$ tend to align with the largest coordinates of the previous time step preference vector $\mathbf{p}_{t-1}$ shown in red.
- *p-Change* steps: This is identical to the *p-Change* step for topic-vector generation in the $\boldsymbol{\theta}$-driven setting (Appendix B.1.1). Thus, as in Figure 3(a), the shaded rows in Figure 3(b) correspond to time steps where a new topic becomes dominant in $\boldsymbol{\alpha}_t$ and $\boldsymbol{\theta}_t$.

Step 2: Preference vector generation. This step is identical to the preference vector generation procedure in Appendix B.1.1. In particular, it uses the same initialization of $\mathbf{p}_0$, the same boost updates at $t = 1$, and the same *preference shift* update at *p-Change* and *p-Stable* steps (including the subsequent noise injection and clipping). In Figure 3(b), this appears as repeated concentration on the strongest preference dimensions within each cycle, followed by a visible shift in $\mathbf{p}_t$ at the shaded *p-Change* rows when a new topic is introduced.

Step 3: Rating computation and preference-change label. Rating computation and preference-change labeling follow the same procedure described in Step 3 of Appendix B.1.1.

B.1.3 $\hat{\mathbf{p}}$-*Driven Setting.* In this setting, at each time step t, the model uses its previous preference estimate $\hat{\mathbf{p}}_{t-1}$ to score all items in a pre-constructed catalog $\mathcal{J}$, ranks items by the predicted rating, and selects the item $\boldsymbol{\theta}_t$ presented at time step t according to this ranking. Each model's hyperparameters are tuned on training users created for the $\boldsymbol{\theta}$-driven and **p**-driven settings and reused here. Consequently, we generate a separate $\hat{\mathbf{p}}$-driven dataset for each model (eight in total) and evaluate each model only on the dataset induced by its own selection policy.

The four-step data generation procedure for the $\hat{\mathbf{p}}$-driven setting under the preference shifting scenario is described below. An example is shown in Figure 3(c). Compared with settings (a) and (b),

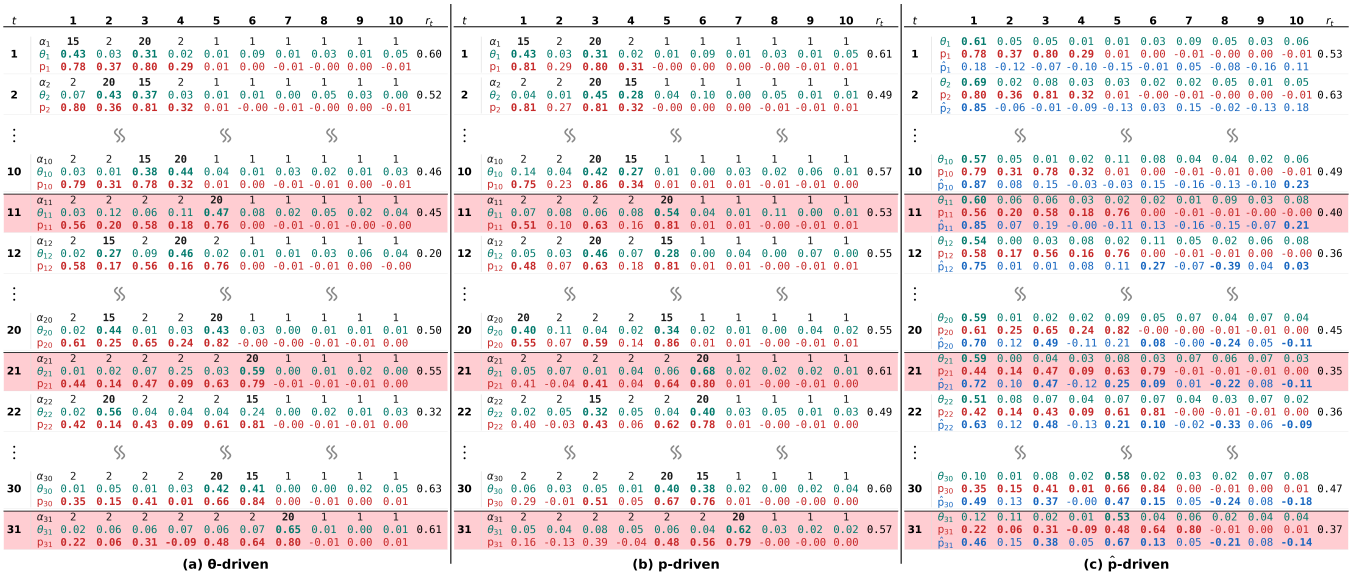


Figure 3: Example user timelines under the preference shifting scenario across three recommendation settings: (a) θ-driven, (b) p-driven, and (c) $\hat{\mathrm{p}}$-driven. Columns 1 through 10 correspond to the $K = 10$ latent topics. Within each time step, rows show the Dirichlet concentration vector $\boldsymbol{\alpha}_t$ (black), the item topic vector $\boldsymbol{\theta}_t$ (teal), the ground-truth preference p_t (red), and in the $\hat{\mathrm{p}}$-driven setting, the estimated preference $\hat{\mathrm{p}}_t$ (blue). The column r_t reports the ground-truth rating. Red-shaded rows indicate preference-change time steps. Bold values highlight high Dirichlet concentrations ($\alpha_{t,k} \geq 10$), dominant topic weights ($\theta_{t,k} \geq 0.25$), and preference weights for explored topics.

this includes an additional blue row for the estimated preference vector $\hat{\mathbf{p}}_t$. The key visual pattern is that the selected item topic vectors tend to follow the model's blue estimate rather than the ground-truth red preference vector. As a result, after a preference change, the model may continue recommending items aligned with its previous estimate, delaying exposure to the newly preferred topic.

Step 1: Item catalog generation. An item catalog $\mathcal{J}$ is constructed that spans diverse topic-mixture structures. Each item is a K-dimensional topic distribution $\boldsymbol{\theta}$ (i.e., $\theta_k \geq 0$ and $\sum_{k=1}^{K} \theta_k = 1$), sampled as $\boldsymbol{\theta} \sim \mathrm{Dirichlet}(\boldsymbol{\alpha})$. Three item types are considered here, distinguished by how many topics receive high concentration. These families range from sharply focused to more diffuse mixtures: 1-topic items reflect preferences for a single topic, 2-topic items reflect the two-topic structure used in user behavior scenarios, and 4-topic items test preference estimation when item weight is spread across multiple topics. 3-topic items are omitted by default because they are intermediate between 2- and 4-topic mixtures; adding them does not change the benchmark protocol.

- 1-topic items: For each topic $k \in \{1, \ldots, 10\}$, set $\alpha_k \leftarrow 20$ and $\alpha_j \leftarrow 2$ for all $j \neq k$, then sample $\boldsymbol{\theta} \sim \mathrm{Dirichlet}(\boldsymbol{\alpha})$. This yields 50 items per k, for a total of 500 items.
- 2-topic items: For each unordered pair $\{i, j\}$ with $1 \leq i < j \leq 10$, set $\alpha_i, \alpha_j \leftarrow 20$ and $\alpha_m \leftarrow 2$ for all $m \notin \{i, j\}$, then sample $\boldsymbol{\theta} \sim \mathrm{Dirichlet}(\boldsymbol{\alpha})$. This yields 50 items per pair, for a total of $45 \times 50 = 2{,}250$ items.
- 4-topic items: For each unordered 4-tuple $\{i, j, k, l\}$ of distinct topics (there are $\binom{10}{4} = 210$ such tuples), set $\alpha_i, \alpha_j, \alpha_k, \alpha_l \leftarrow 8$ and $\alpha_u \leftarrow 2$ for all $u \notin \{i, j, k, l\}$, then sample $\boldsymbol{\theta} \sim \mathrm{Dirichlet}(\boldsymbol{\alpha})$. This yields one item per 4-tuple, for a total of 210 items.

Altogether, the catalog contains $|\mathcal{J}| = 2{,}960$ items. In Figure 3(c), the teal vectors $\boldsymbol{\theta}_t$ shown at each time step are selected from this catalog rather than generated directly from a scenario-specific Dirichlet concentration vector.

Step 2: Item selection using $\hat{\mathbf{p}}_{t-1}$. Let $\mathcal{J}_t \subseteq \mathcal{J}$ denote the current item catalog at time step t. At each time step, the model computes predicted ratings $\hat{r}_t(\boldsymbol{\theta}) = \hat{\mathbf{p}}_{t-1}^{\top} \boldsymbol{\theta}$ for all $\boldsymbol{\theta} \in \mathcal{J}_t$, ranks items by $\hat{r}_t(\boldsymbol{\theta})$ in descending order, and partitions the ranked list into the top-2% set $\mathcal{T}_t$ and the remaining set $\mathcal{B}_t = \mathcal{J}_t \setminus \mathcal{T}_t$. We consider two item selection policies to create two versions of the $\hat{\mathbf{p}}$-driven dataset for each model:

- *Top-2% selection:* Choose the item uniformly at random from $\mathcal{T}_t$.
- *Mixed selection:* Sample uniformly from $\mathcal{T}_t$ with probability 0.75, and uniformly from $\mathcal{B}_t$ with probability 0.25.

In both cases, the selected item is denoted by $\boldsymbol{\theta}_t$ and removed from the catalog, i.e., $\mathcal{J}_{t+1} = \mathcal{J}_t \setminus \{\boldsymbol{\theta}_t\}$. In Figure 3(c), this is visible in the way the selected teal vectors $\boldsymbol{\theta}_t$ tend to align with the strongest coordinates of the blue estimated preference vector $\hat{\mathbf{p}}_{t-1}$. Early in the timeline, when the estimate is still inaccurate, the selected items can differ noticeably from the user's true preference vector. After a preference change, the model may continue choosing items aligned with its previous estimate, which reduces exposure to the newly preferred topic.

Step 3: Preference vector generation. This step is identical to the preference vector generation procedure in Appendix B.1.1. In Figure 3(c), the shaded rows at $t = 11, 21, 31$ again correspond to candidate *p-Change* time steps, where the ground-truth red preference vector $\mathbf{p}_t$ shifts toward a newly introduced topic. However, because item selection is based on the blue estimate $\hat{\mathbf{p}}_{t-1}$, the selected items may lag behind this change.

Step 4: Rating computation, preference-change label, and model update. After selecting $\boldsymbol{\theta}_t$, the ground-truth rating $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t$ is computed and the preference-change label is assigned following the same procedure described in Step 3 of Appendix B.1.1. The model then updates its preference estimate using the interaction $(\boldsymbol{\theta}_t, r_t)$ to obtain $\hat{\mathbf{p}}_t$.

B.2 Preference Shifting & Conservation (PSC1T)

PSC1T extends PS by allowing some *p-Change* time steps to produce only a near-neutral update to the preference weight. At such timesteps, the preference weight for the newly introduced topic moves only toward a near-neutral value (0), while the preference weights for previously explored topics remain approximately unchanged. Figure 4 shows representative timelines for this scenario across the three recommendation settings.

B.2.1 $\boldsymbol{\theta}$*-Driven Setting.* In the $\boldsymbol{\theta}$-driven setting, PSC1T differs from PS only at *p-Change* time steps, where some newly introduced topics trigger near-neutral updates instead of full preference shifts. Visually, this appears in Figure 4**(a)** as red-shaded rows in which the newly introduced topic still receives high topic weight in $\boldsymbol{\theta}_t$, but the corresponding change in the red preference vector $\mathbf{p}_t$ is much smaller than in the usual PS-style shift rows.

- Step 1: Topic vector generation. This step is identical to PS. At each *p-Change* time step, a new topic $k^\star$ is introduced into $\mathcal{S}$ with high Dirichlet concentration, and at each *p-Stable* time step two explored topics are assigned high concentrations as in Section B.1.1.
- Step 2: Preference vector generation. At each *p-Change* time step where a new topic $k^\star$ is introduced, the time step produces either a high weight or a near-neutral preference update for the corresponding new topic. In the former case, we apply the same preference-shift update as in PS: the preference for $k^\star$ increases, while the preferences for the other explored topics decrease proportionally. In the latter case, the preference weight of the newly introduced topic is updated toward a near-neutral value. The following stochastic perturbation rule, identical to that used at *p-Stable* time steps, is then applied: the newly introduced topic and the previously explored topics receive only Gaussian noise, while all other unexplored topics receive Gaussian noise followed by clipping. Because this near-neutral update produces only a small change in $\mathbf{p}_t$, such time steps typically fall below the labeling threshold δ and are therefore not labeled as preference-change events. In Figure 4**(a)**, this is illustrated at time step 21, where the new topic is clearly visible in $\boldsymbol{\theta}_{21}$, but the preference vector $\mathbf{p}_{21}$ changes only slightly relative to the previous time step, $\mathbf{p}_{20}$.

B.2.2 $\mathbf{p}$*-Driven Setting.* The PSC1T preference-update rule at *p-Change* time steps is identical to that used in the $\boldsymbol{\theta}$-driven setting described in Section B.2.1. The only difference is that *p-Stable* time steps use the top-3 preference-guided topic selection rule from Section B.1.2. Consequently, in Figure 4**(b)**, the unshaded rows tend to follow the user's currently strongest preference dimensions, while the red-shaded rows alternate between large PS-style shifts and near-neutral updates, the latter visible at time steps 21 and 31 where the preference vector changes only slightly despite the introduction of a new topic.

B.2.3 $\hat{\mathbf{p}}$*-Driven Setting.* Following Section B.1.3, we use the same item catalog generated in Step 1 and the same model-specific item selection procedure described in Step 2. The PSC1T preference-update rule for generating the ground-truth preference vector at *p-Change* and *p-Stable* time steps is identical to that used in the $\boldsymbol{\theta}$-driven setting described in Section B.2.1. In Figure 4**(c)**, the selected items follow the model's previous estimate $\hat{\mathbf{p}}_{t-1}$ rather than the ground-truth preference vector. Time steps 11 and 21 illustrate PS-style preference changes, whereas time step 31 illustrates a near-neutral update, where the change in the preference vector remains small, similar to *p-Stable* behavior.

B.3 Preference Shifting & Conservation with Two Topics (PSC2T)

PSC2T extends PSC1T by allowing some *p-Change* time steps to present an item with high weight on two previously explored topics. Although each of these topics has appeared earlier with high topic weight, the user currently has negative or near-neutral preference for one or both of them. When the two topics occur together, however, they can induce a strong preference shift toward both topics. Figure 5 shows representative timelines for this scenario across the three recommendation settings. Compared with PSC1T, the key additional visual pattern in PSC2T is that some red-shaded rows emphasize two previously explored topics rather than one newly introduced topic, and these rows can produce large changes in the corresponding entries of the preference vector.

B.3.1 $\boldsymbol{\theta}$*-Driven Setting.* In the $\boldsymbol{\theta}$-driven setting, PSC2T extends PSC1T by allowing some *p-Change* time steps to become two-topic combination events. Visually, this appears in Figure 5**(a)** as red-shaded rows in which two previously explored topics receive high topic weight in $\boldsymbol{\theta}_t$, followed by a large coordinated change in the corresponding entries of the red preference vector $\mathbf{p}_t$.

Step 1: Topic vector generation. At each *p-Change* time step, whether the time step follows the usual one-topic PSC1T rule or becomes a two-topic combination event is first determined. Let $\mathcal{S}_t^- = \{k \in \mathcal{S} : p_{t-1,k} \le 0\}$ denote the set of explored topics whose preference values are negative or near-neutral at time $t-1$. If $|\mathcal{S}_t^-| \ge 2$, then at the current *p-Change* time step one of two event types is chosen: with probability $\pi_{2\mathrm{T}}$, a two-topic combination event is instantiated, and with probability $1 - \pi_{2\mathrm{T}}$, the standard one-topic PSC1T event is used.

For a two-topic combination event, two distinct topics (k_1, k_2) are sampled uniformly from $\mathcal{S}_t^-$ and $\alpha_{t,k_1} = \alpha_{t,k_2} = 20$ is set, while all remaining concentrations stay at their baseline values. No new topic is added to $\mathcal{S}$. For a one-topic event, topic generation is identical to PSC1T. Thus, relative to PSC1T, some red-shaded rows now

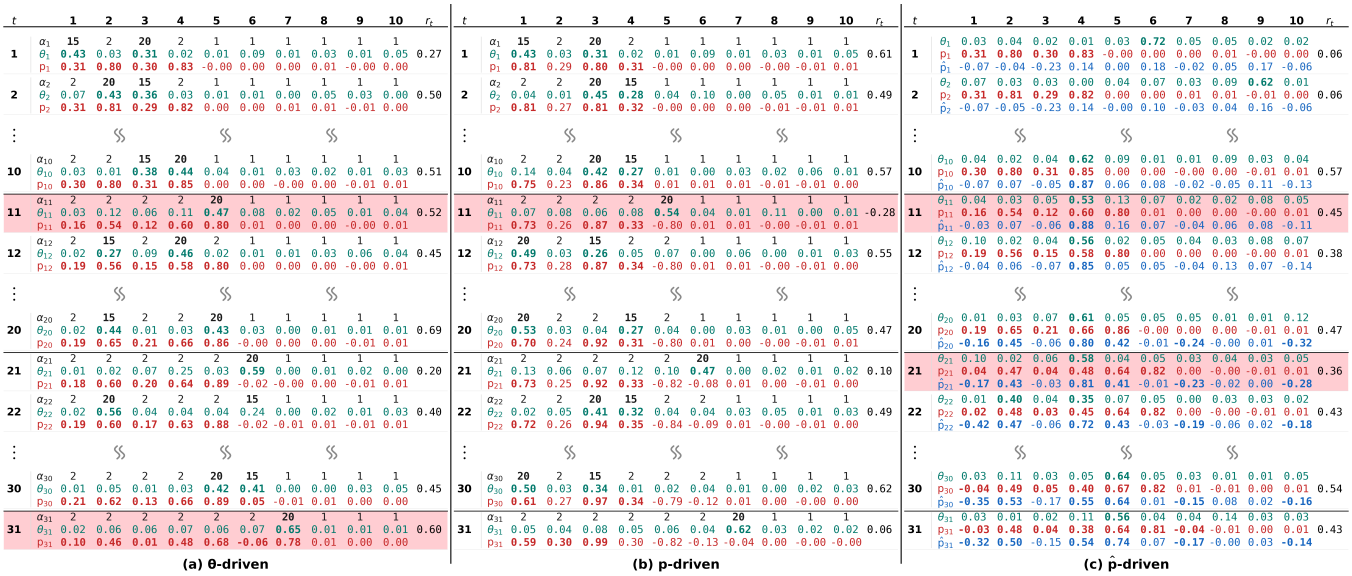


Figure 4: Example user timelines under the PSC1T scenario across three recommendation settings: (a) θ-driven, (b) p-driven, and (c) $\hat{p}$-driven. The visualization format is the same as in Figure 3. Red-shaded rows indicate preference-change time steps. In PSC1T, some time steps produce PS-style updates (e.g., time steps 11, 31 in (a), time step 11 in (b), and time steps 11, 21 in (c)), whereas others produce only near-zero updates to the preference weight of the newly introduced topic (e.g., time step 21 in (a), time steps 21, 31 in (b), and time step 31 in (c)).

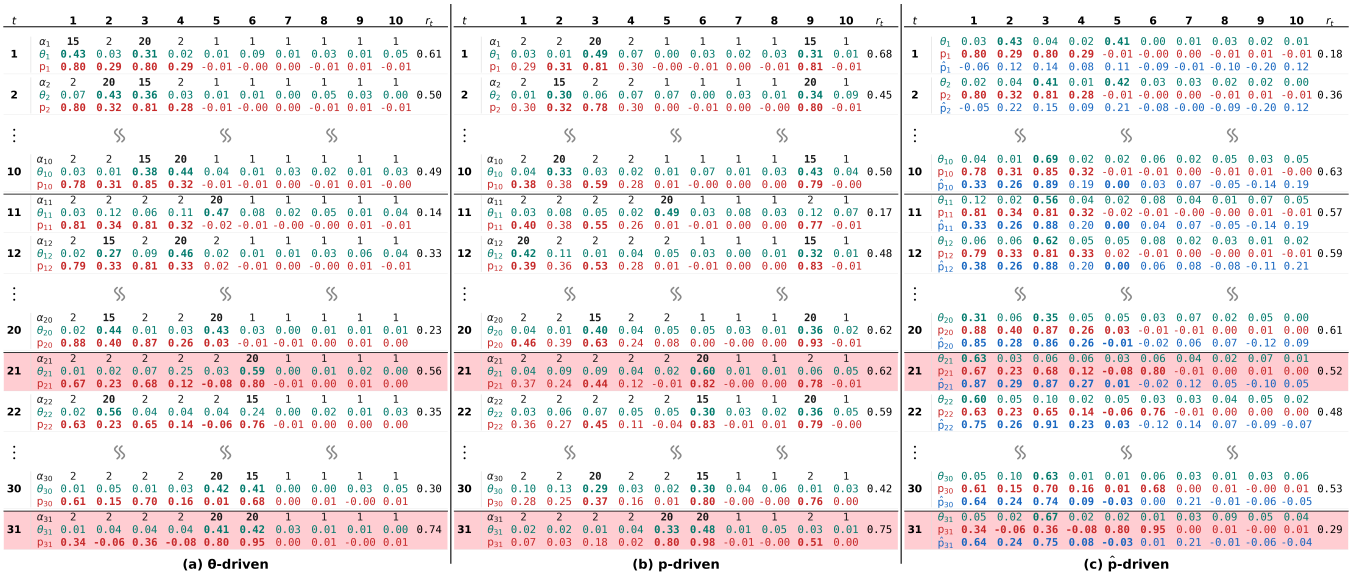


Figure 5: Example user timelines under the PSC2T scenario across three recommendation settings: (a) θ-driven, (b) p-driven, and (c) $\hat{p}$-driven. The visualization format is the same as in Figure 3. Red-shaded rows indicate preference-change time steps. In PSC2T, some time steps follow the PSC1T update rule (e.g., time steps 11, 21 in (a) and time steps 11, 21 in (b)), while others become two-topic combination events (e.g., time step 31 in (a) and time step 31 in (b)).

emphasize two previously explored topics instead of one newly introduced topic.

Step 2: Preference vector generation. If the time step is selected for a two-topic combination event, then the user develops a strong preference shift toward both topics. We therefore apply a two-topic

version of the PS update:

$$A_1 = \eta_{\text{big}}(1 - p_{t-1,k_1}), \qquad p_{t,k_1} \leftarrow p_{t-1,k_1} + A_1,$$
$$A_2 = \eta_{\text{big}}(1 - p_{t-1,k_2}), \qquad p_{t,k_2} \leftarrow p_{t-1,k_2} + A_2.$$

Let $A = A_1 + A_2$ and $Z = \sum_{j \in \mathcal{S} \setminus \{k_1,k_2\}} (p_{t-1,j} + 1)$. If $Z \geq A$, we decrease the remaining explored-topic preferences proportionally:

$$p_{t,j} \leftarrow p_{t-1,j} - (p_{t-1,j} + 1)\frac{A}{Z}, \qquad \forall j \in \mathcal{S} \setminus \{k_1, k_2\}.$$

If the time step is not selected for a two-topic event, then it follows the one-topic PSC1T update described in Section B.2.1. In Figure 5**(a)**, time step 31 illustrate this two-topic behavior: two previously explored topics receive high weight, and the corresponding entries of $\mathbf{p}_{31}$ increase sharply relative to the previous time step.

B.3.2 $\mathbf{p}$*-Driven Setting.* The PSC2T event logic and preference-update rule are identical to those in the $\boldsymbol{\theta}$-driven setting described in Section B.3.1. The only difference is that *p-Stable* time steps use the top-3 preference-guided topic selection rule from Section B.1.2. Consequently, in Figure 5**(b)**, *p-Stable* time steps tend to follow the user's currently highest preference weights, while the red-shaded rows alternate between one-topic PSC1T events and two-topic combination events. The latter are visible, for example, at time steps 11 and 21, where two previously explored topics are emphasized in $\boldsymbol{\theta}_t$ and the preference vector exhibits a joint shift.

B.3.3 $\hat{\mathbf{p}}$*-Driven Setting.* Following Section B.1.3, we use the same item catalog generated in Step 1 and the same model-specific item selection procedure described in Step 2. The PSC2T preference-update rule for generating the ground-truth preference vector at *p-Change* and *p-Stable* time steps is identical to that used in the $\boldsymbol{\theta}$-driven setting described in Section B.3.1.

B.4 Preference Broadening (PB)

PB differs from PS only in the preference update at *p-Change* time steps. When a new topic is introduced, the preference weight for that topic increases, but the preference weights for previously explored topics remain approximately same. This models a user who accumulates new interests without replacing old ones. Figure 6 shows representative timelines for PB.

B.4.1 $\boldsymbol{\theta}$*-Driven.*

Step 1: Topic vector generation. This step is identical to PS.

Step 2: Preference vector generation. At each *p-Change* time step where a new topic $k^\star$ is introduced, we increase the preference for the new topic:

$$p_{t,k^\star} \leftarrow p_{t-1,k^\star} + \eta_{\text{big}}(1 - p_{t-1,k^\star}).$$

For all other explored topics $j \in \mathcal{S} \setminus \{k^\star\}$, preferences evolve only through the same noise-injection-and-clipping rule used at *p-Stable* time steps in Step 2 of Section B.1.2. In contrast to PS, no proportional decrease is applied to these explored topics.

B.4.2 $\mathbf{p}$*-Driven.* The PB preference-generation rule is identical to that in the $\boldsymbol{\theta}$-driven PB setting. The only difference is that *p-Stable* timesteps use the top-3 preference-guided topic selection rule from Section B.1.2.

B.4.3 $\hat{\mathbf{p}}$*-Driven.* Following Section B.1.3, we use the same item catalog generated in Step 1 and the same model-specific item selection procedure described in Step 2. The preference-generation rule for the ground-truth preference vector at *p-Change* and *p-Stable* time steps is identical to that used in the $\boldsymbol{\theta}$-driven PB setting.

B.5 Preference Broadening & Conservation (PBC1T)

PBC1T extends PB by allowing some *p-Change* time steps to produce only a near-neutral update rather than a full broadening update. At such time steps, the preference weight for the newly introduced topic stays near zero which means neutral preference, while the preference weights for previously explored topics remain approximately unchanged. Figure 7 shows representative timelines for this scenario across the three recommendation settings.

B.5.1 $\boldsymbol{\theta}$*-Driven.* In the $\boldsymbol{\theta}$-driven setting, PBC1T differs from PB only at *p-Change* time steps, where some newly introduced topics trigger near-neutral updates instead of full broadening updates.

Step 1: Topic vector generation. This step is identical to PS and PB.

Step 2: Preference vector generation. At each *p-Change* time step where a new topic $k^\star$ is introduced, the time step produces either a preference-broadening update or a near-neutral update. In the former case, we apply the PB update:

$$p_{t,k^\star} \leftarrow p_{t-1,k^\star} + \eta_{\text{big}}(1 - p_{t-1,k^\star}).$$

In the latter case, the preference weight of the newly introduced topic is kept close to zero, and all remaining topic preferences evolve, by applying the same noise-injection-and-clipping rule used at *p-Stable* time steps. Because the near-neutral case produces only a small change in $\mathbf{p}_t$, such time steps typically fall below the labeling threshold δ and therefore receive a stable-preference label.

B.5.2 $\mathbf{p}$*-Driven.* The PBC1T preference-generation process at *p-Change* time steps is identical to that used in the $\boldsymbol{\theta}$-driven PBC1T setting. The only difference is that *p-Stable* time steps use the top-3 preference-guided topic selection rule from Section B.1.2.

B.5.3 $\hat{\mathbf{p}}$*-Driven.* We use the same item catalog generated in Step 1 of Section B.1.3 and the same model-specific item selection procedure described in Step 2 of Section B.1.3. The rule for generating the ground-truth preference vector at both *p-Change* and *p-Stable* time steps is identical to that used in the $\boldsymbol{\theta}$-driven PBC1T setting.

B.6 Preference Broadening & Conservation with Two Topics (PBC2T)

PBC2T extends PBC1T by allowing some *p-Change* time steps to present an item with high topic weight on two previously explored topics. As in PSC2T, these two topics have negative or near zero preference weights for one or both of them.

B.6.1 $\boldsymbol{\theta}$*-Driven Setting.*

Step 1: Topic vector generation. This step is identical to PSC2T. At each *p-Change* timestep, if $|\mathcal{S}_t^-| \geq 2$, then with probability $\pi_{2\text{T}}$ the timestep becomes a two-topic combination event; otherwise it follows the one-topic PBC1T rule.

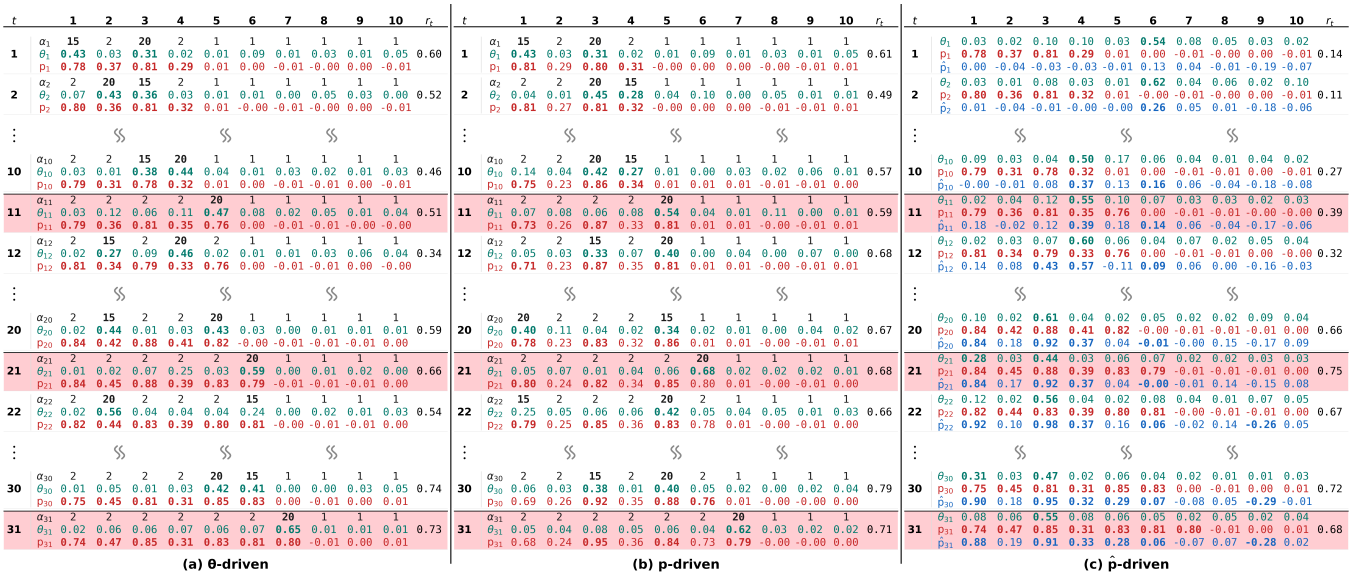


Figure 6: Example user timelines under the PB scenario across three recommendation settings: (a) θ-driven, (b) p-driven, and (c) $\hat{p}$-driven. The visualization format is the same as in Figure 3. Red-shaded rows indicate preference-change time steps. In PB, each newly introduced topic receives a high update to its preference weight, while the preference weights of previously explored topics remain approximately unchanged.

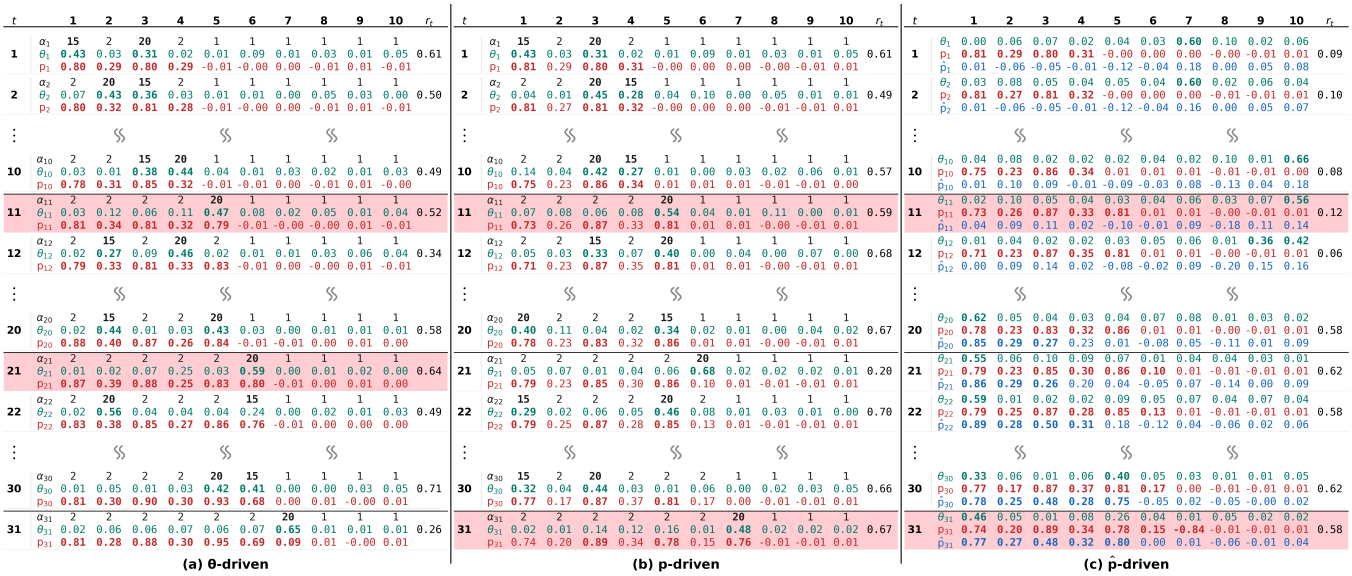
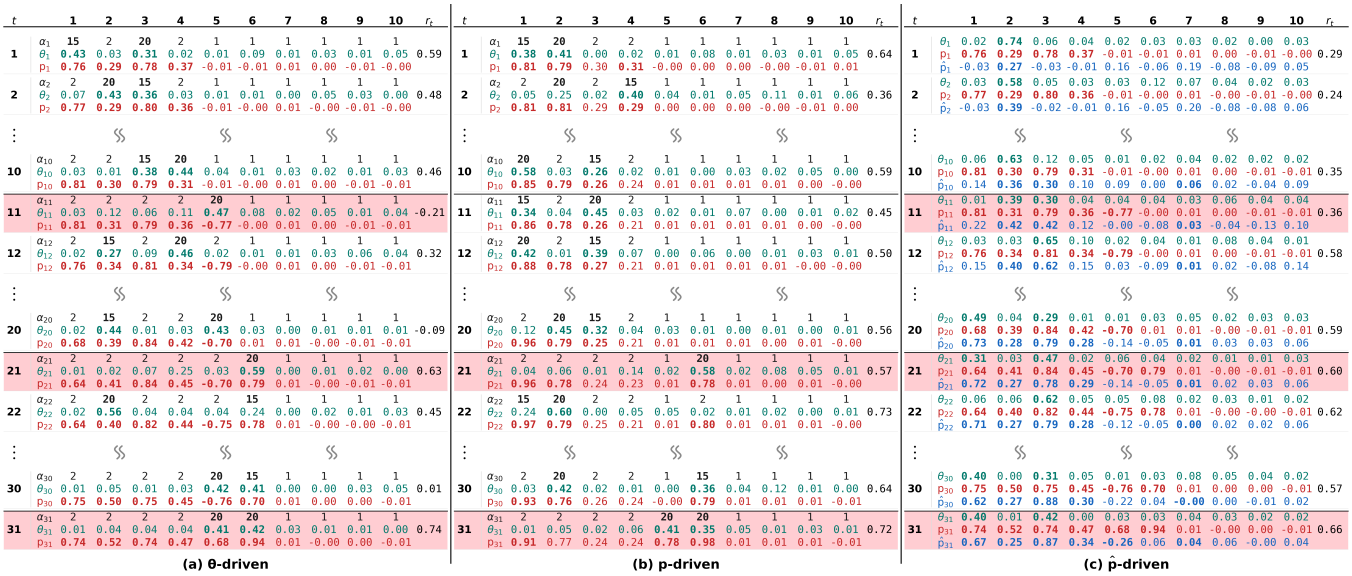


Figure 7: Example user timelines under the PBC1T scenario across three recommendation settings: (a) θ-driven, (b) p-driven, and (c) $\hat{p}$-driven. The visualization format is the same as in Figure 3. Red-shaded rows indicate preference-change time steps. In PBC1T, some time steps produce PB-style updates (e.g., time steps 11, 21 in (a) and time steps 11, 31 in (b)), whereas others produce only near-zero updates to the preference weight of the newly introduced topic (e.g., time step 31 in (a) and time step 21 in (b)).

Step 2: Preference vector generation. If the timestep is realized as a two-topic combination event, then the preferences for the two triggering topics are increased:

$$p_{t,k_1} \leftarrow p_{t-1,k_1} + \eta_{\text{big}}(1 - p_{t-1,k_1}),$$
$$p_{t,k_2} \leftarrow p_{t-1,k_2} + \eta_{\text{big}}(1 - p_{t-1,k_2}).$$



(a) θ-driven (b) p-driven (c) p̂-driven

Figure 8: Example user timelines under the PBC2T scenario across three recommendation settings: (a) θ-driven, (b) p-driven, and (c) $\hat{\mathbf{p}}$-driven. The visualization format is the same as in Figure 3. The visualization format is the same as in Figure 3. Red-shaded rows indicate preference-change time steps. In PBC2T, some such time steps follow the PBC1T update rule (e.g., time steps 11, 21 in (a) and time steps 21 in (b)), while others become two-topic based events(e.g., time step 31 in (a) and time step 31 in (b)).

All remaining topic preferences evolve only through the same noise injection and clipping used at *p-Stable* timesteps. In contrast to PSC2T, no proportional decrease is applied to the remaining explored topics. If the timestep is not realized as a two-topic event, then it follows the one-topic PBC1T update described above.

B.6.2 **p**-*Driven Setting.* The PBC2T event logic and preference-update rule are identical to those in the $\boldsymbol{\theta}$-driven setting. The only difference is that *p-Stable* timesteps use the top-3 preference-guided topic selection rule from Section B.1.2.

B.6.3 $\hat{\mathbf{p}}$-*Driven Setting.* Item selection follows Section B.1.3. At a candidate *p-Change* timestep, if the selected item places high weight on two previously explored topics with negative preference values, then the timestep follows the PBC2T two-topic update. If instead the selected item places high weight on one newly introduced topic, then the timestep follows the one-topic PBC1T update. Otherwise the timestep follows the standard *p-Stable* update.

C Topic Distribution Construction for the Real Dataset

Since the Goodreads dataset does not include full book texts, book-level topic distributions are estimated from the aggregated text of user reviews associated with each book. For every book in the catalog, English-language reviews are concatenated and truncated at 10,000 tokens to produce a *book content field* that serves as a textual proxy for the book's substance. Books with fewer than 50 tokens in their concatenated review text are excluded from the item set because their topic distributions cannot be estimated reliably.

Topic distributions are obtained by training a Latent Dirichlet Allocation (LDA) model with $K = 100$ topics. The LDA training corpus is restricted to books whose book content field contains at least 1,000 tokens, in order to ensure that training documents carry sufficient lexical signal for stable topic inference. This yields a training corpus of roughly 3×10^5 books. After training, the LDA model is applied to every book in the item set to produce a 100-dimensional topic vector $\boldsymbol{\theta}_i$, where $\theta_{i,k} \geq 0$ and $\sum_{k=1}^{100} \theta_{i,k} = 1$, as required by the benchmark's topic representation.

The set of users is restricted to those who rated at least 100 distinct books in the item set, which yields approximately 2.6×10^4 users. This threshold reduces the incidence of users with systematically missing ratings, under the assumption that users with long rating histories are more likely to record every book they have read. The combined book-rating set over these users contains slightly over 10^6 unique books. The resulting book-level topic distributions $\boldsymbol{\theta}_t$ serve as the per-item feature vectors consumed by the online learning models at each time step, whereas similarity-based pair identification for deriving the real-data preference-change labels is performed using the Jina embeddings described in Section 5.

D Hyperparameter Tuning and Reproducibility

All model hyperparameters are tuned using Optuna [1] on the training users only. For each model, Optuna samples hyperparameter configurations from predefined search ranges using the Tree-structured Parzen Estimator (TPE) sampler [37] and evaluates them under the online update protocol. A single-objective optimization setup is used, in which the objective is the average post-update MSE over the training users after the warm-up period. Concretely, for each interaction, the model is first updated using the observed item–rating pair, and MSE is then computed between the observed rating and the rating predicted from the updated posterior for the same

item. Optuna uses these MSE values to guide the search toward more promising regions of the hyperparameter space. 50,000 Optuna trials per model are run on the synthetic dataset and 30,000 trials per model on the real dataset. Table 5 reports the search ranges and the final selected hyperparameter values for both datasets. Table 5 reports the search ranges and the final selected hyperparameter values for both datasets.

After optimization, the hyperparameter configuration with the lowest training-set objective value is selected for each model and applied unchanged in the final evaluation. Preference-change detection and preference-recovery metrics are not used during hyperparameter selection; they are computed only after the best hyperparameters have been selected. On the test users, MSE is computed in a pre-update, next-step prediction manner: at time step t, the model predicts the rating for the current item using the posterior from the previous time step, and only then updates the posterior using the observed interaction at time step t. This pre-update protocol corresponds to next-item recommendation accuracy. The post-update tuning objective and the pre-update evaluation MSE are intentionally computed at different stages because they measure different aspects of the online learning process: how well the model incorporates a newly observed item and its next-item predictive accuracy, respectively. All models are evaluated under the same data splits, warm-up period, and metrics described in the *Evaluation Methodology* section of the main text.

Table 5: Hyperparameter search ranges and optimized values for each model on the synthetic and real datasets.

Model	Hyperparameter	Range	Synthetic	Real
KF-AF	σ_p^2	$[10^{-2}, 10^{2}]$	2.74805	3.52167
	σ^2	$[10^{-2}, 10^{2}]$	0.01199	0.57455
	δ_{init}	$[10^{-4}, 10^{0}]$	0.01454	0.01497
	η	$[10^{-7}, 10^{-1}]$	0.00001	0.00006
AROW	r_1	$(0, 10^{2}]$	59.47278	1.62345
	r_2	$(0, 10^{2}]$	1.28900	0.56232
BLR	σ^2	$[10^{-2}, 10^{2}]$	3.97625	0.82913
BLR-VB	σ^2	$[10^{-2}, 10^{2}]$	1.83389	1.29658
	τ_v	$[10^{-2}, 10^{0}]$	0.45313	0.28582
BLR-FF	σ^2	$[10^{-2}, 10^{2}]$	1.39941	1.16874
	ρ	$[0.8, 1.0]$	0.98253	0.99192
BLR-SW	m	$[2, 100]$	21.00000	50.00000
	σ^2	$[10^{-2}, 10^{2}]$	1.37627	0.39872
BLR-PP	α	$[10^{-2}, 10^{0}]$	0.87015	0.85000
	σ^2	$[10^{-2}, 10^{2}]$	2.04119	8.61530
BLR-NIG	σ^2	$[10^{-2}, 10^{2}]$	37.38376	27.79795
	a_0	$[10^{-2}, 10^{2}]$	1.24829	4.37219
	b_0	$[10^{-2}, 10^{2}]$	2.23896	0.07659

E Detailed Experimental Results

This appendix reports the full experimental results on the synthetic and real datasets, corresponding to the discussion in the "Results on Synthetic Data" and "Results on Real Data" sections of the main text. For the synthetic benchmark, we present the complete metric suite (MSE, ROC, PR, TR, Lag, and MTL) across all three recommendation settings and the full per-scenario breakdown under the $\hat{\mathbf{p}}$-driven setting. For the real dataset, we present results across the full cross-product of similarity thresholds $\tau \in \{0.95, 0.90\}$ and time-gap constraints $\Delta t \in \{\infty, 14, 7\}$ days. Across all experiments, a central finding emerges: predictive accuracy alone is insufficient, as models with similar rating error can exhibit substantially different capabilities in tracking non-stationary user preferences.

Table 6: Tuning and test user splits for the synthetic and real datasets.

Dataset	Setting	Tuning	Test	Total
Synthetic	$\boldsymbol{\theta}$-driven	30	60	90
	$\mathbf{p}$-driven	30	60	90
	$\hat{\mathbf{p}}$-driven (Top-2%)	60	120	180
	$\hat{\mathbf{p}}$-driven (Mixed)	60	120	180
Real	Similarity $\geq \tau = 0.90$	9,736	14,605	24,341
	Similarity $\geq \tau = 0.95$	7,046	10,570	17,616

E.1 Synthetic Data: Per-Scenario and Per-Setting Breakdown

Tables 7 and 8 report the full results on the synthetic benchmark, where ground-truth preference vectors enable direct evaluation of preference tracking via tracking ratio, tracking lag, and MTL. A first observation across the $\boldsymbol{\theta}$-driven and $\mathbf{p}$-driven settings is that preference-change detection is substantially easier than preference recovery: all models achieve ROC above 97%, indicating that posterior shifts reliably separate preference-change time steps from stable ones, yet the tracking metrics reveal large differences in how effectively models act on these detected changes to realign their preference estimate with the new ground-truth state.

The KF-AF demonstrates the strongest preference tracking in Table 7, recovering 78.3% and 70.4% of preference change events in the $\boldsymbol{\theta}$-driven and $\mathbf{p}$-driven settings respectively, with a mean lag of 2.1 and 1.5 for the recovered events and MTL of 3.8 and 4.0 for all preference change events, while also achieving the highest ROC and PR in both settings. At the same time, all models achieve low MSE in both settings, indicating strong aggregate rating prediction. However, preference tracking performance varies substantially, especially in PR and in the recovery metrics TR/Lag and MTL, showing that accurate rating prediction does not necessarily imply fast or reliable adaptation after preference changes.

Although AROW and BLR-VB achieve the lowest rating prediction error (MSE 0.019) alongside KF-AF in the $\boldsymbol{\theta}$-driven setting, and AROW alone achieves the lowest MSE in the $\mathbf{p}$-driven setting, both exhibit substantially lower preference estimation performance (TR 54.4% and 66.9% in the $\boldsymbol{\theta}$-driven setting). This pattern reflects a difference in how these models handle non-stationarity. The KF-AF's dynamical-state formulation explicitly models preference evolution: it injects uncertainty through the process-noise term and uses a time-varying forgetting factor γ_t, which is updated after each time step via an SGD step on the one-step predictive log-likelihood. This adaptive forgetting prevents the posterior from concentrating too tightly around outdated beliefs during stable periods while

Table 7: Test-set performance across the θ-driven, p-driven, and $\hat{p}$-driven settings. For rating prediction, MSE↓ measures mean squared error. For preference-change detection, ROC↑ and PR↑ report the corresponding AUC (%). For tracking performance, TR↑ is the tracking ratio (%), Lag↓ is the mean tracking lag over recovered events, and MTL↓ reports the mean tracking lag over all events. For all settings, tracking uses RelPE ≤ 0.25.

	θ-driven					p-driven					$\hat{p}$-driven				
Model	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**
KF-AF	**0.019**	**99.6**	**95.1**	**78.3**/2.1	**3.8**	0.022	**99.6**	**96.0**	**70.4**/1.5	**4.0**	**0.003**	**61.8**	**20.8**	**0.7/7.5**	10.0
AROW	**0.019**	98.6	89.5	54.4/2.0	5.7	**0.019**	99.5	95.6	59.9/1.2	4.7	0.013	58.1	17.6	0.0/∞	10.0
BLR	0.034	99.3	91.5	27.2/3.3	8.2	0.025	98.6	85.8	37.9/1.8	6.8	0.015	55.7	16.5	0.0/∞	10.0
BLR-VB	**0.019**	99.2	92.5	66.9/3.1	5.4	0.020	99.3	91.0	63.2/2.1	5.0	0.005	56.4	17.4	0.0/∞	10.0
BLR-FF	0.030	99.3	91.6	34.6/2.1	7.3	0.023	99.0	87.6	44.4/1.8	6.3	0.012	57.6	16.5	0.0/∞	10.0
BLR-SW	0.027	98.4	82.3	26.5/4.2	8.5	0.023	97.3	71.8	33.9/3.4	7.8	0.011	52.4	14.7	0.0/∞	10.0
BLR-PP	0.035	99.5	94.4	21.3/3.8	8.7	0.027	98.8	88.1	35.4/2.3	7.3	0.017	58.0	17.3	0.0/∞	10.0
BLR-NIG	0.033	99.3	91.7	39.0/**1.1**	6.5	0.025	99.1	89.9	50.9/**0.6**	5.2	0.013	57.1	17.9	0.0/∞	10.0

Table 8: Performance on test-set users in the $\hat{p}$-driven setting, where items are sampled from the model's ranked list: with probability 0.75 from the top 2% and with probability 0.25 from the remaining 98%. For preference-change detection, ROC↑ and PR↑ report the corresponding AUC (%). For tracking performance, TR↑ is the tracking ratio (%), Lag↓ is the mean tracking lag over recovered events, and MTL↓ is the mean tracking lag over all events. In this setting, tracking uses RelPE ≤ 0.25.

	PS					**PSC1T**					**PB**					**PBC1T**				
Model	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**
KF-AF	0.009	55.9	19.9	4.5/7.3	9.9	0.009	63.3	18.8	8.3/6.4	9.7	0.010	53.5	19.2	27.9/3.8	8.3	0.009	56.3	16.6	27.1/4.4	8.5
AROW	0.016	53.3	19.1	1.5/8.0	10.0	0.021	56.4	14.0	0.4/9.0	10.0	0.024	57.6	21.8	0.8/8.0	10.0	0.022	54.5	16.3	0.7/9.0	10.0
BLR	0.015	53.6	19.3	0.0/∞	10.0	0.028	57.8	14.2	0.0/∞	10.0	0.025	57.6	22.4	0.0/∞	10.0	0.020	53.3	13.7	0.7/7.5	10.0
BLR-VB	0.010	49.4	17.4	0.8/9.0	10.0	0.012	53.7	12.6	0.4/7.0	10.0	0.014	49.6	18.0	10.9/5.7	9.5	0.011	50.4	15.9	6.2/5.2	9.7
BLR-FF	0.014	55.7	20.3	0.0/∞	10.0	0.022	54.3	14.5	0.0/∞	10.0	0.021	56.1	19.5	0.0/∞	10.0	0.018	54.1	14.6	0.7/4.0	10.0
BLR-SW	0.012	51.0	19.5	0.0/∞	10.0	0.016	47.9	12.8	0.0/∞	10.0	0.023	51.5	18.7	0.0/∞	10.0	0.018	48.1	12.6	0.0/∞	10.0
BLR-PP	0.014	54.1	20.7	0.0/∞	10.0	0.024	55.6	15.3	0.0/∞	10.0	0.030	59.3	22.9	0.0/∞	10.0	0.023	55.3	15.7	0.0/∞	10.0
BLR-NIG	0.018	53.1	17.1	0.8/8.0	10.0	0.027	60.4	17.3	0.7/5.5	10.0	0.017	52.1	21.2	2.3/5.0	9.9	0.016	56.1	14.1	2.9/7.5	9.9

Table 9: Extension of Table 8, reporting performance on test-set users in the $\hat{p}$-driven setting for the two additional scenarios PSC2T and PBC2T. Items are sampled from the model's ranked list: with probability 0.75 from the top 2% and with probability 0.25 from the remaining 98%. For preference-change detection, ROC↑ and PR↑ report the corresponding AUC (%). For tracking performance, TR↑ is the tracking ratio (%), Lag↓ is the mean tracking lag over recovered events, and MTL↓ reports the mean tracking lag over all events. In this setting, tracking uses RelPE ≤ 0.25.

	PSC2T					**PBC2T**				
Model	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**	**MSE**	**ROC**	**PR**	**TR/Lag**	**MTL**
KF-AF	0.012	65.5	19.0	6.3/6.9	9.8	0.010	55.4	13.2	27.7/4.3	8.4
AROW	0.027	56.5	12.8	0.0/∞	10.0	0.028	51.6	12.2	0.6/8.0	10.0
BLR	0.039	58.6	12.6	0.0/∞	10.0	0.025	48.9	11.9	1.1/7.5	10.0
BLR-VB	0.015	57.0	12.4	0.6/7.0	10.0	0.013	52.2	14.5	7.3/5.3	9.7
BLR-FF	0.030	54.7	11.9	0.0/∞	10.0	0.022	49.6	11.8	0.6/6.0	10.0
BLR-SW	0.020	48.6	13.1	0.0/∞	10.0	0.020	46.0	12.0	0.0/∞	10.0
BLR-PP	0.033	55.7	13.9	0.0/∞	10.0	0.027	52.9	14.1	0.0/∞	10.0
BLR-NIG	0.039	60.4	13.4	0.6/5.0	10.0	0.023	52.2	11.3	1.1/8.0	10.0

enabling rapid re-alignment when new observations become inconsistent with the current state. In contrast, AROW's KL-regularized objective explicitly penalizes large deviations from the previous posterior, prioritizing stability over the plasticity required to capture preference changes. By enforcing a lower bound on the posterior covariance, BLR-VB avoids over-confidence and preserves greater capacity to adapt to preference changes than standard BLR. However, this variance floor is a passive safeguard rather than an active dynamical mechanism, so the resulting updates can still be insufficiently aggressive when genuine preference changes occur, limiting how quickly the posterior mean moves toward the new preference state.

Table 10: Performance on test-set users on the real dataset portion of this benchmark. For predictive accuracy, MSE↓ measures mean squared error. For preference tracking, SRC↑ denotes Spearman rank correlation, while ROC↑ and PR↑ report the corresponding AUC (%). Results are reported under two similarity thresholds (Similarity $\geq \tau$, $\tau \in 0.95, 0.90$) and three pairing protocols based on valid time step pairs defined by the real time gap ($\Delta t < \infty$, $\Delta t \leq 14$d, $\Delta t \leq 7$d).

		Similarity $\geq \tau = 0.95$									Similarity $\geq \tau = 0.90$								
		$\Delta t < \infty$			$\Delta t \leq 14$ days			$\Delta t \leq 7$ days			$\Delta t < \infty$			$\Delta t \leq 14$ days			$\Delta t \leq 7$ days		
Model	**MSE**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**
KF-AF	0.78	**0.19**	**69.7**	**16.6**	**0.25**	**79.0**	**15.8**	**0.26**	**80.4**	**16.5**	**0.17**	**67.0**	**20.1**	**0.24**	**77.6**	**19.0**	**0.25**	**79.5**	**19.9**
AROW	**0.76**	0.18	68.3	14.4	0.22	76.4	15.2	0.22	77.2	15.1	0.16	65.3	17.3	0.21	75.1	17.6	0.21	76.3	17.8
BLR	0.77	0.17	67.5	14.1	0.20	74.4	13.8	0.20	75.2	13.6	0.15	64.7	17.0	0.18	73.1	16.1	0.18	74.2	16.1
BLR-VB	**0.76**	0.18	68.5	14.7	0.22	76.6	15.1	0.23	77.7	15.3	0.16	65.5	17.9	0.21	75.2	17.4	0.21	76.6	17.7
BLR-FF	0.81	0.17	66.8	12.8	0.24	77.3	14.1	0.25	78.7	15.2	0.14	63.5	15.5	0.23	75.9	17.1	0.24	77.8	18.3
BLR-SW	0.80	0.15	65.8	14.5	0.14	68.9	9.4	0.14	69.4	8.9	0.14	64.1	18.9	0.14	67.5	12.0	0.15	68.8	11.6
BLR-PP	0.77	0.13	64.0	13.3	0.08	63.0	8.2	0.07	63.4	7.6	0.12	62.3	16.4	0.08	62.6	10.7	0.08	63.1	9.9
BLR-NIG	0.78	0.16	65.6	12.6	0.19	72.5	13.4	0.19	73.3	13.8	0.13	62.3	14.9	0.17	71.1	15.7	0.18	72.2	16.1

The remaining models exhibit a similar dissociation between preference detection and recovery: BLR-PP achieves high detection scores (ROC 99.5%, PR 94.4%) yet recovers only 21.3% of events, and BLR-FF and BLR-SW show comparable patterns. This comparison underscores a key distinction: preference change detection measures whether the model notices a change, whereas recovery measures whether it adapts to it. A model can produce detectable posterior shifts at preference change time steps without moving its preference estimate close enough to the new ground-truth state to yield a high tracking ratio. Standard aggregate metrics like MSE, which are substantially low for all models, conflate these distinct capabilities.

A notable exception to the general detection–recovery pattern is BLR-NIG, which achieves the lowest lag among all models (1.1 and 0.6 in the $\boldsymbol{\theta}$-driven and $\mathbf{p}$-driven settings) despite a moderate tracking ratio (39.0% and 50.9%). This behavior follows from BLR-NIG's Normal–Inverse-Gamma posterior, which jointly maintains uncertainty over the preference vector and the observation noise variance. After a preference change, when the resulting residuals are too large to be explained by the previously inferred noise level, the posterior update can re-center the preference mean rapidly (often within 1–2 interactions), yielding very small Lag among recovered events. In contrast, when post-change residuals can be accommodated by increasing the inferred noise variance, the inverse-gamma component absorbs much of the discrepancy as higher σ^2 rather than forcing a sufficiently large shift in the posterior mean. As a result, BLR-NIG recovers quickly when it recovers, but fails to recover for a substantial fraction of events, leading to a lower overall tracking ratio.

The $\hat{\mathbf{p}}$-driven setting highlights a severe degradation in both preference-change detection and tracking for all models. ROC drops to the 52%–62% range and PR falls below 21% across all models. The tracking ratio is effectively zero for all models except KF-AF, which achieves marginal tracking performance. This degradation has a clear explanation. When item selection at time step t is driven by the model's previous preference estimate $\hat{\mathbf{p}}_{t-1}$, the user interaction log becomes highly endogenous: models tend to select items that confirm their existing beliefs. After a preference change, the model continues selecting items aligned with previous preferences, producing ratings that remain moderately consistent with its outdated belief. The model therefore receives little informative signal about the new preference state because the selected items do not probe the changed dimensions. This creates a self-reinforcing feedback loop: outdated preferences drive item selection, which generates observations that reinforce those same outdated preferences. Notably, MSE remains small in this setting because the predicted rating is computed using the post-update estimate $\hat{\mathbf{p}}_t$: the Bayesian update moves the posterior mean along $\boldsymbol{\theta}_t$ by an amount that reduces the residual $\hat{\mathbf{p}}_t^\top \boldsymbol{\theta}_t - r_t$, so the predicted rating closely tracks the ground-truth $r_t = \mathbf{p}_t^\top \boldsymbol{\theta}_t$ on the just-observed item regardless of whether $\hat{\mathbf{p}}_t$ has caught up to $\mathbf{p}_t$. The feedback loop compounds this effect: the items the model selects have high topic weight on old-preferred topics, where the estimate is already accurate, while the latent misalignment is concentrated on the newly preferred topic, which these selected items rarely weight.

Table 8 breaks down performance in the $\hat{\mathbf{p}}$-driven setting by scenario. The KF-AF is the primary model achieving meaningful preference tracking performance across scenarios while AROW, BLR-VB, and BLR-NIG also exhibit measurable tracking performance. This behavior is attributable to the stochastic exploration introduced in the mixed item selection protocol, where items are sampled from the remaining 98% of the ranked list with probability 0.25. This injected randomness ensures that the user's interaction history retains a degree of topic diversity, mitigating complete convergence into a filter bubble for some models. Nevertheless, most methods still fail to recover within the post-change cycle, and MTL remains near 10, underscoring that closed-loop recommendation can hinder preference tracking even when rating prediction remains strong. These results reinforce the central motivation of this benchmark: preference-change detection and post-change recovery must be evaluated under different exposure regimes to understand model behavior under non-stationarity. Table 9 represents this scenario-level analysis to the PSC2T and PBC2T scenarios. The results show a similar pattern, with KF-AF achieving the strongest preference recovery in both scenarios, while most other models recover few or no preference-change events and MTL remains close to 10.

E.2 Real Data: Results Across Similarity Thresholds and Time Gaps

Table 10 reports the full performance on the real dataset portion of the benchmark. Because Goodreads does not provide ground-truth preference vectors, preference tracking is evaluated on the valid high-similarity interaction pairs, as described in the *Real Datasets* section of the main text. Spearman rank correlation (SRC) between the rating discrepancy Δr and the model's KL-based shift score $s_{j,k}$ is reported, along with ROC-AUC/PR-AUC on valid pairs (stable: Δr=0; change: $\Delta r \geq 2$). Results are shown across the full cross-product of two similarity thresholds $\tau \in \{0.95, 0.90\}$ and three time-gap constraints $\Delta t < \infty$ and $\Delta t \leq \{14, 7\}$ days.

Across both similarity thresholds, most models improve on SRC and ROC-AUC as the time-gap constraint becomes stricter. When $\Delta t < \infty$, valid pairs may be separated by long periods, so rating differences can reflect a mixture of short-horizon preference changes and longer-horizon drift; this weakens the alignment between Δr and the model's inferred preference change score. Restricting to $\Delta t \leq 14$ or $\Delta t \leq 7$ days focuses the evaluation on pairs of near-duplicate books that the user rated very differently within a short time window, which provides a cleaner signal for preference change. This trend is consistent across $\tau = 0.95$ and $\tau = 0.90$.

Comparing $\tau = 0.95$ and $\tau = 0.90$ reveals complementary trends across metrics. The stricter threshold ($\tau = 0.95$) generally yields higher ROC and SRC values across most models, because near-identical item pairs provide a cleaner signal: when two items are almost indistinguishable in content, a large rating discrepancy is more confidently attributable to a preference change rather than content differences. However, PR values tend to be higher at $\tau = 0.90$ for most models. This is because the looser threshold admits substantially more valid pairs (see the *Real Datasets* section of the main text), which increases the number of positive-class pairs ($\Delta r \geq 2$) and reduces class imbalance. Since PR-AUC is sensitive to the ratio of positive to negative examples, this less imbalanced setting leads to higher precision at comparable recall levels.

The model-level patterns on real data are consistent with the synthetic dataset findings reported in Section E.1. KF-AF achieves the strongest preference tracking across both similarity thresholds and all pairing protocols, leading in SRC and typically attaining the best ROC/PR, while remaining competitive in rating prediction (MSE = 0.78 versus the best 0.76). AROW and BLR-VB achieve the lowest MSE (0.76) but fall below KF-AF on the preference tracking metrics. Relative to standard BLR, BLR-VB improves tracking by preventing posterior covariance collapse, while BLR-FF sacrifices predictive accuracy (MSE = 0.81) and shows larger gains in preference tracking metrics as the time-gap decreases. As discussed in Section E.1, these differences follow the same mechanistic distinction: KF-AF's dynamical-state formulation with adaptive forgetting enables larger posterior re-alignment when observations become inconsistent with the current belief, whereas AROW's KL-regularized objective and BLR-VB's variance floor bias updates toward stability, yielding weaker KL-based preference change signals and slower adaptation. The consistency of these relative orderings across synthetic and real evaluations supports the usefulness of the synthetic dataset design and the associated preference tracking metrics.

Among the rest of the baselines, the same pattern persists across evaluation settings: methods with similar rating-prediction accuracy can exhibit meaningfully different preference-tracking behavior. Several models remain close in MSE yet differ substantially in SRC and in ROC/PR, indicating that aggregate accuracy is not reliable for preference tracking. Importantly, this relative ordering is consistent with the results on synthetic dataset and reflects the stability–plasticity trade-off: stability-oriented updates can preserve strong rating prediction accuracy while producing weaker preference change signals and slower adaptation, whereas more plastic updates yield larger posterior shifts that improve preference tracking. Together, these results support the central motivation this benchmark: tracking-oriented metrics are needed to characterize model behavior under non-stationarity beyond downstream accuracy alone.

F KL Divergence vs. Wasserstein Distance

KL divergence and Wasserstein distance are compared as alternative preference-change scores for the pairwise preference change detection task on the real dataset. For each valid pair of time steps (j, k) with $j < k$, both $s^{\mathrm{KL}}_{j,k} = D_{\mathrm{KL}}(\hat{\mathbf{p}}_k \,\|\, \hat{\mathbf{p}}_j)$ and the 2-Wasserstein distance $s^{\mathrm{WD}}_{j,k} = W_2(\hat{\mathbf{p}}_k, \hat{\mathbf{p}}_j)$ are computed, where $\hat{\mathbf{p}}_j$ and $\hat{\mathbf{p}}_k$ denote the posterior preference distributions at time steps j and k, respectively. Each score is then evaluated against the preference-change labels defined by $\Delta r = 0$ versus $\Delta r \geq 2$.

Table 11 reports the results for four representative models (KF-AF, AROW, BLR, and BLR-VB) on the training users, under similarity thresholds $\tau = 0.95$ and $\tau = 0.90$. Overall, KL divergence yields higher ROC-AUC and PR-AUC than Wasserstein distance for most models, while the two measures produce broadly similar Spearman rank correlations. The advantage of KL divergence is especially noticeable under tighter temporal constraints ($\Delta t \leq 7$ days and $\Delta t \leq 14$ days), where the evaluation emphasizes preference changes under small time gap and the labels provide a cleaner signal. Based on these results, KL divergence is used as the preference-change score in all main experiments.

Table 11: Comparison of KL divergence and Wasserstein distance (WD) as preference change scores on the training users of the real dataset. Results are shown for similarity thresholds $\tau = 0.95$ and $\tau = 0.90$. SRC↑ denotes Spearman rank correlation, and ROC↑ and PR↑ report the corresponding AUC (%). Higher values within each KL/WD pair are bolded.

Model		$\tau = 0.95$									$\tau = 0.90$								
		$\Delta t < \infty$			$\Delta t \leq 14$d			$\Delta t \leq 7$d			$\Delta t < \infty$			$\Delta t \leq 14$d			$\Delta t \leq 7$d		
		SRC	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**	**SRC**	**ROC**	**PR**
KF-AF	KL	**0.19**	**70.83**	**16.85**	**0.25**	**78.41**	**13.55**	**0.26**	**80.89**	**16.56**	**0.17**	**66.71**	**20.17**	**0.24**	**77.19**	**18.00**	**0.25**	**79.59**	**19.60**
	WD	0.19	70.09	15.96	0.24	77.61	12.46	0.25	80.04	14.67	0.16	66.00	18.90	0.23	76.67	16.88	0.25	79.12	18.31
AROW	KL	**0.17**	**69.31**	14.40	**0.20**	**75.50**	**12.33**	**0.21**	**77.30**	**14.33**	0.15	64.88	16.72	**0.20**	**74.75**	**16.63**	**0.21**	**76.48**	**17.50**
	WD	0.17	69.26	**14.48**	0.20	75.02	11.81	0.20	76.69	13.46	**0.15**	**64.94**	**16.96**	0.20	74.33	16.14	0.20	76.00	16.89
BLR	KL	**0.16**	68.49	14.17	**0.17**	**73.13**	**11.12**	**0.17**	**74.79**	**12.57**	0.14	64.23	16.42	**0.18**	**72.69**	**15.21**	**0.18**	**74.36**	**15.88**
	WD	0.16	**68.61**	**14.38**	0.17	72.92	10.73	0.17	74.38	11.89	**0.14**	**64.50**	**16.97**	0.17	72.25	14.82	0.18	73.84	15.39
BLR-VB	KL	**0.18**	**69.49**	**14.71**	**0.21**	**75.66**	**12.45**	**0.21**	**77.61**	**14.56**	**0.15**	**65.12**	**17.26**	**0.20**	**74.80**	**16.54**	**0.21**	**76.72**	**17.54**
	WD	0.17	69.13	14.52	0.20	74.99	11.46	0.21	76.86	13.07	0.15	64.92	17.14	0.20	74.32	15.65	0.21	76.24	16.56