LEARNING ITERATIVE UPDATES TO SAMPLING DISTRIBUTIONS FROM DENSITY-RATIO FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models enable inference from incomplete observations by sampling plausible underlying structures from learned distributions. However, a mismatch between the sampling distribution and the target posterior can leave regions with substantial posterior probability represented by only a few samples. Such mismatch can increase posterior-mean estimation error under a fixed number of target density evaluations. In this work, we introduce the Ratio-Guided Proposal Compiler (RGPC), a learned model that iteratively updates the sampling distribution to reduce distribution mismatch and posterior-mean estimation error. To guide these updates, RGPC compares the density of the target posterior with the current sampling density at trial samples, then combines the resulting density-ratio feedback with aggregate constraints to predict changes to the sampling distribution's parameters. Across 32 Public Use Microdata Areas (PUMAs), we evaluate population inference on development tasks defined by subsets of American Community Survey (ACS) aggregate statistics, with all methods approximating the same target posterior for each task. We measure distribution mismatch using squared maximum mean discrepancy (MMD^2), which compares distributions through their samples. For posterior-mean estimation error, we use total variation distance (TVD) between the estimated and reference posterior means. RGPC achieves the lowest MMD^2 and TVD among all nine evaluated methods, including conditional generators and conventional sampling baselines. When using the same number of target density evaluations, RGPC reduces MMD^2 by 42.65% and TVD by 26.50% relative to the best-performing baseline. Code is available at `https://anonymous.4open.science/r/RGPC-FD4B/`.

1 INTRODUCTION

Generative models support applications ranging from language generation (Dubey et al., 2024) and visual synthesis (Esser et al., 2024) to tabular data generation (Shi et al., 2025). In these applications, generating realistic samples requires capturing relationships among variables. The same requirement arises in population modeling, where travel-demand and public-health simulations use joint demographic information to represent differences among population groups (Rineer et al., 2025). When only aggregate statistics are available, population inference estimates the population distribution over combinations of demographic attributes. However, even when generated population distributions match the observed aggregates, the sampling distribution can still differ from the target posterior (Blumenthal et al., 2026). This distribution mismatch can increase posterior-mean estimation error under a fixed number of target density evaluations.

To reduce this mismatch under a fixed number of target density evaluations, we seek a generative model that can condition its sampling distribution on the observed aggregates and iteratively update that distribution toward the target posterior. Recent work has used Transformer architectures to construct such observation-conditioned sampling distributions. For example, Distribution Transformers (Whittle et al., 2026) use attention to combine information from a prior distribution and observations before predicting the parameters of the sampling distribution. Nevertheless, drawing more samples from an unchanged sampling distribution does not reduce its mismatch with the target posterior. To guide an update, we can use the ratio of the density of the target posterior to the current sampling density at trial samples, which underlies importance weighting (Blumenthal et al., 2026). This

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

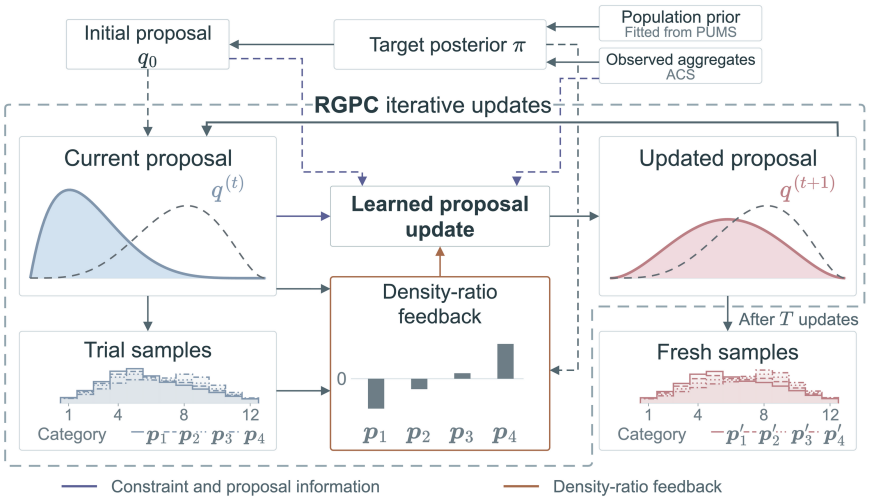


Figure 1: **Iterative updates with RGPC.** Observed aggregates and a population prior fitted from ACS Public Use Microdata Sample (PUMS) records define the target posterior π (gray dashed curves). Starting from sampling distribution (proposal) $q^{(0)} = q_0$, the learned proposal update combines trial samples $\boldsymbol{p}_i$ and density-ratio feedback with aggregate constraints and information from the initial and current proposals to produce $q^{(t+1)}$. The target posterior π and initial proposal q_0 remain unchanged. After T updates, fresh samples $\boldsymbol{p}'_i$ are drawn from the final proposal $q^{(T)}$. The proposal panels show densities along an illustrative coordinate; sample profiles show category probabilities in sampled population distributions. Dashed arrows mark external inputs. All plotted quantities are schematic.

density-ratio feedback indicates how sampling probability should be redistributed, with larger ratios favoring relatively more sampling probability. As the sampling distribution changes, recomputing the ratios provides feedback on the remaining mismatch. We therefore ask how to learn iterative updates that use this feedback to reduce mismatch.

Prior work has developed iterative methods for reducing distribution mismatch. Gaussian-mixture variational inference (Arenz et al., 2023) uses log-density ratios to optimize mixture parameters. Beyond mixture fitting, CRAFT (Matthews et al., 2022) learns transformations between intermediate distributions, while SCLD (Chen et al., 2025) combines learned diffusion dynamics with importance weighting and resampling. However, these methods fit their sampling distributions through target-specific optimization, which can not be reused in new task. In this sense, the model build with these methods must be performed again based on new target posteriors. To address the reusabilities, iterative amortized inference (Mittal et al., 2025) learns reusable updates across tasks, using samples from the target distribution to guide updates in the reported generative experiments. But, no such samples are provided for the target posterior in population inference, where the target posterior can be defined using observed aggregates and a population prior. Thus, we seek reusable iterative updates that use density-ratio feedback at trial samples to determine how sampling probability should be redistributed in population inference tasks. To reduce distribution mismatch in such tasks under a fixed number of target density evaluations, we propose RGPC, which learns iterative updates to the sampling distribution against a fixed target posterior, with decoded population distributions satisfying the observed aggregates (Figure 1).

To this end, we combine density-ratio feedback with information about changes to the population distribution permitted by the observed aggregates. We posit that a learned model can reduce mismatch by interpreting density-ratio feedback in terms of these permitted changes. We operationalize this idea in RGPC by directly conditioning each location and spread prediction on its associated density-ratio feedback and constraint information. As the sampling distribution changes, RGPC recomputes density-ratio feedback against the same target posterior. On each new task, RGPC performs these updates under a fixed number of target density evaluations without retraining.

For the evaluation of RGPC, we use 192 development tasks from 32 Public Use Microdata Areas (PUMAs). Each task is defined by a subset of American Community Survey (ACS) aggregate statistics

(United States Census Bureau, 2023a; 2026). On these tasks, we compare nine methods, including conditional generators and conventional samplers, all approximating the same target posterior for each task. RGPC achieves the lowest distribution mismatch (MMD^2) and posterior-mean estimation error (TVD). When using the same number of target density evaluations, RGPC reduces MMD^2 by 42.65% and TVD by 26.50% relative to the best-performing baseline, the population-adapted Distribution Transformer (PDT). With trained networks and sampling configurations frozen, we further evaluate 192 tasks from 32 additional PUMAs excluded from sampler training and configuration selection. RGPC retains lower mean MMD^2 and TVD than PDT and the selected conventional samplers with iterative updates. To test the proposed mechanism, we conduct controlled comparisons. Both metrics improve with current-task constraint information and density-ratio feedback at parameter prediction, and with correctly paired feedback and trial samples during later updates. Our contributions are summarized as follows:

- We introduce RGPC, which learns to use density-ratio feedback and information about the aggregate constraints to predict iterative updates to the sampling distribution.
- We show that RGPC's posterior approximation improves when location and spread predictions directly use current-task constraint information and density-ratio feedback, and when later updates preserve the correspondence between feedback and its trial samples.
- We demonstrate lower distribution mismatch and posterior-mean estimation error than eight learned and conventional sampling baselines on the population-inference development tasks when using the same number of target density evaluations.

2 Related Work

We provide a literature review of population inference from aggregate statistics below. Population synthesis uses aggregate statistics (de Mooij et al., 2024) or microdata (Sané et al., 2025). For Bayesian inference from aggregates, Frogner & Poggio (2019) compute a posterior mode. RGPC instead estimates the posterior mean.

To approximate these posteriors for different observed aggregates, neural samplers learn reusable conditional distributions. Distribution Transformers (Whittle et al., 2026) use a Gaussian-mixture prior-to-posterior mapping. However, additional draws from the same sampling distribution leave residual mismatch unchanged. RGPC instead uses density-ratio feedback to update the sampling distribution with its network parameters frozen.

These updates build on learned iterative inference. Marino et al. (2018) learn posterior updates from variational-objective gradients or model prediction errors. More recently, Mittal et al. (2025) learn iterative updates from observation batches, including generation conditioned on target-distribution samples. In our population setting, target posterior samples are available for training but not at inference time. RGPC instead computes density-ratio feedback at trial samples from its current sampling distribution. At parameter prediction, it combines this feedback with the corresponding constraint information for each location and spread, rather than relying only on a shared component representation as in Distribution Transformers (section 3). Appendix E extends this review.

3 Ratio-Guided Proposal Compiler

Our Ratio-Guided Proposal Compiler (RGPC) iteratively updates a sampling distribution to approximate a target posterior conditioned on observed aggregate statistics. Like mixture-based inference methods (Arenz et al., 2023), RGPC uses samples and target density evaluations to update its sampling distribution. Unlike target-specific optimization, RGPC predicts these updates with a network trained across tasks, combining density-ratio feedback with the corresponding constraint information at each location and spread prediction. With network weights frozen, each updated sampling distribution supplies trial samples for the next update. Figure 2 illustrates these iterative updates.

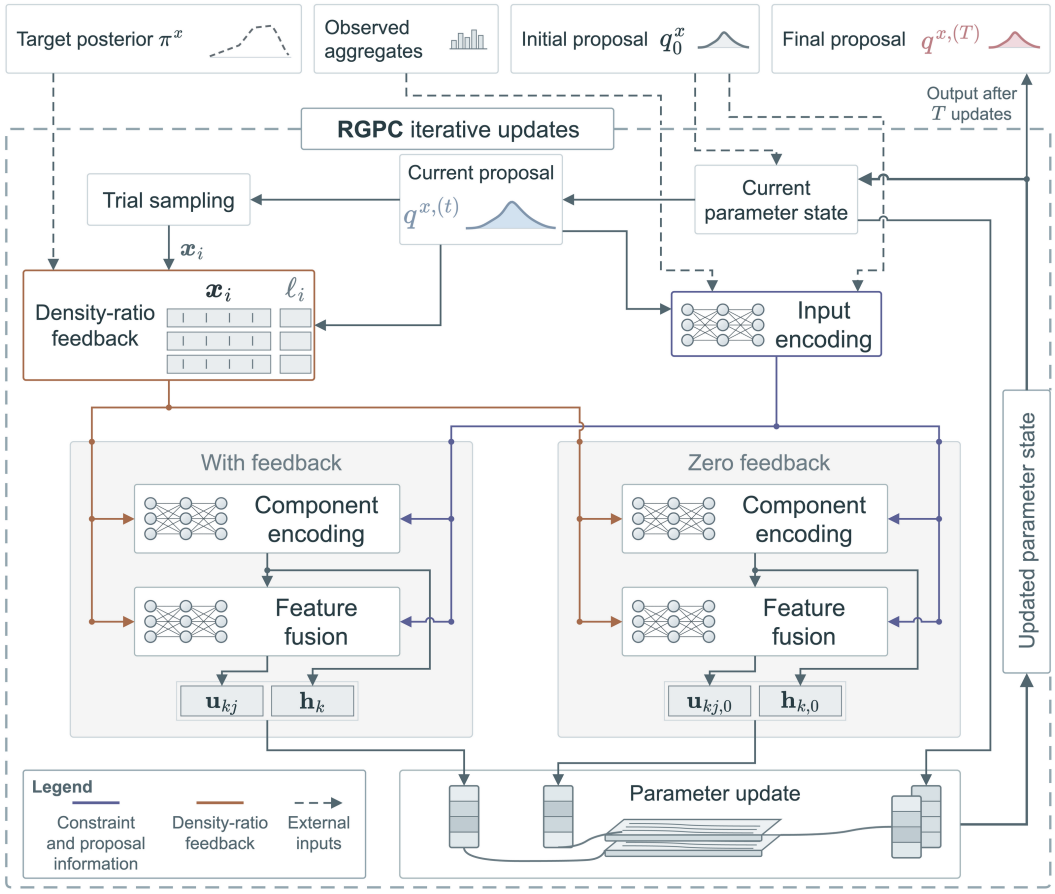


Figure 2: **Our RGPC architecture.** Working coordinates $\boldsymbol{x}$ decode to population distributions $\boldsymbol{p}$; q_0^x is represented by a zero parameter state. Each update obtains density-ratio feedback from trial samples. The two passes share trial samples and network parameters; the zero-feedback pass sets the feedback and its summaries to zero. Bias-free heads map representation differences to state increments; the updated state defines the next proposal. Dashed arrows mark fixed external inputs. Network parameters are frozen during inference. Figure 6 gives layer-level details. Sample counts and curves are schematic.

3.1 Population Inference under Aggregate Constraints

For region r, a candidate population distribution $\boldsymbol{p} \in \Delta^{K-1}$ gives proportions in K attribute combinations. Its entries are nonnegative and sum to one. All tasks use the same attribute combinations. An observation operator $\mathbf{A}_o$ sums selected entries to form the supplied aggregate statistics $\boldsymbol{y}_{r,o}$, defining a task (r, o). We treat these observed aggregates as exact constraints, without modeling survey uncertainty, and assume at least one population distribution satisfies them. Let $\mathcal{S}_{r,o}$ contain the cells not forced to zero by the observed aggregates or known support restrictions. The population fiber is the set of population distributions satisfying these conditions,

$$\mathcal{F}_{r,o} = \left\{ \boldsymbol{p} \in \Delta^{K-1} : \mathbf{A}_o \boldsymbol{p} = \boldsymbol{y}_{r,o},\ p_j = 0 \text{ for } j \notin \mathcal{S}_{r,o} \right\}. \tag{1}$$

The same observed aggregates can admit many population distributions in $\mathcal{F}_{r,o}$. To assign probabilities among these distributions, we use a population prior with density $\rho_r(\boldsymbol{p})$ learned from source regions. This population prior can already incorporate aggregate statistics common to all tasks. Restricting the population prior to the population fiber defines the target posterior,

$$\pi^p_{r,o}(\boldsymbol{p}) \propto \rho_r(\boldsymbol{p}), \qquad \boldsymbol{p} \in \mathcal{F}_{r,o}, \tag{2}$$

with respect to intrinsic volume on the fiber, assuming the restriction is normalizable. The superscript p denotes a density over population distributions. The population prior and observed aggregates

determine the target posterior before a sampling distribution is constructed. Appendix B.2 specifies the population prior.

Our estimation target is the posterior mean,

$$\boldsymbol{\mu}_{r,o} = \mathbb{E}_{\boldsymbol{p}\sim\pi^{p}_{r,o}}[\boldsymbol{p}]. \tag{3}$$

Each entry of $\boldsymbol{\mu}_{r,o}$ is the expected proportion of an attribute combination under the target posterior.

RGPC constructs a final sampling distribution, or proposal, $q^{p}_{\theta,r,o}$ to approximate this posterior. For N fresh samples $\boldsymbol{p}'_n \sim q^{p}_{\theta,r,o}$, the posterior mean is estimated by their ordinary average,

$$\widehat{\boldsymbol{\mu}}_{r,o} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{p}'_n. \tag{4}$$

With a fixed final proposal, increasing N reduces sampling variability but does not remove bias from residual distribution mismatch. We next construct this proposal from density-ratio feedback while ensuring that every decoded sample satisfies the observed aggregates.

3.2 Ratio-Guided Proposal Update

RGPC represents the $d_{r,o}$ degrees of freedom in $\mathcal{F}_{r,o}$ by centered and rescaled coordinates $\boldsymbol{x} \in \mathbb{R}^{d_{r,o}}$. The initial proposal $q^{x}_{0,r,o}$ combines a standardized local Gaussian approximation to the target posterior with a Student-t tail component. After reversing the rescaling, the decoder uses information projection onto $\mathcal{F}_{r,o}$ to enforce the observed aggregates. All densities below use these coordinates. Appendix A gives the decoder and density transformations.

To obtain feedback about distribution mismatch, at update t we draw m trial samples $\boldsymbol{x}^{(t)}_i \sim q^{x,(t)}$ from the current proposal and compare the target and sampling densities at these locations. Starting from $q^{x,(0)} = q^{x}_{0,r,o}$, RGPC reuses the same network for T updates. We center the log-density ratios to remove the target density's unknown normalizing constant from the feedback. With normalized base masses $\bar{b}^{(t)}_i$, equal to $1/m$ for independent draws, the centered log-density ratio is

$$\begin{aligned}\ell^{(t)}_i = {} & \log \pi^{x}_{r,o}(\boldsymbol{x}^{(t)}_i) - \log q^{x,(t)}(\boldsymbol{x}^{(t)}_i) \\ & - \sum_{j=1}^{m} \bar{b}^{(t)}_j \left[\log \pi^{x}_{r,o}(\boldsymbol{x}^{(t)}_j) - \log q^{x,(t)}(\boldsymbol{x}^{(t)}_j)\right].\end{aligned} \tag{5}$$

Larger $\ell^{(t)}_i$ indicate where target density is greater relative to the current sampling density.

RGPC encodes the trial samples and their feedback, then uses attention conditioned on observed aggregates and current proposal parameters to obtain a representation $\boldsymbol{h}^{(t)}_k$ for each Gaussian component. To predict its location and spread, the model also retains information specific to each coordinate j. A coordinate encoding $\boldsymbol{e}_j$ describes constraint-permitted directions of population change and their rescaling under the initial proposal. The feedback summary $\boldsymbol{\delta}^{(t)}_j$ contains first- and second-moment differences between density-ratio-weighted and base-weighted trial samples along that coordinate. A shared encoder maps the current component parameters at coordinate j to $\boldsymbol{c}^{(t)}_{kj}$, whose average over active coordinates also conditions the component's attention query. A shared neural rule combines these inputs into a coordinate representation,

$$\boldsymbol{u}^{(t)}_{kj} = G_\theta\left(\boldsymbol{h}^{(t)}_k + \boldsymbol{e}_j + S_\theta(\boldsymbol{\delta}^{(t)}_j) + \boldsymbol{c}^{(t)}_{kj}\right), \tag{6}$$

where S_θ encodes the moment differences and G_θ is shared across components and coordinates. These inputs let each location and spread respond to its coordinate's feedback summary on the current task.

To preserve the current proposal under zero density-ratio feedback, RGPC runs a zero-feedback pass with density-ratio feedback and its summaries set to zero. Both passes share network parameters and all other inputs. RGPC subtracts the zero-feedback representations from those computed with feedback. Bias-free heads map component-representation differences to mixture-logit increments and

coordinate-representation differences to location, log-scale, and covariance-factor increments. These increments are accumulated in the proposal's unconstrained parameter state before the output transformations yield logits $\boldsymbol{a}^{(t+1)}$, locations $\boldsymbol{\xi}_k^{(t+1)}$, and covariances $\mathbf{C}_k^{(t+1)}$. The updated proposal is

$$q^{x,(t+1)}(\boldsymbol{x}) = (1-\tau)\sum_{k=1}^{4}\mathrm{softmax}(\boldsymbol{a}^{(t+1)})_k\mathcal{N}(\boldsymbol{x};\boldsymbol{\xi}_k^{(t+1)},\mathbf{C}_k^{(t+1)}) + \tau\, t_\nu(\boldsymbol{x};\mathbf{0},\mathbf{S}_{\mathrm{tail}}). \tag{7}$$

The Student-t component retains its mass τ, degrees of freedom ν, and scale $\mathbf{S}_{\mathrm{tail}}$ from q_0^x across updates. Zero density-ratio feedback therefore gives $q^{x,(t+1)} = q^{x,(t)}$. The next update draws trial samples from $q^{x,(t+1)}$ and recomputes density-ratio feedback against the same target posterior using the explicit mixture density. Appendix A defines representation differences, heads, and covariances.

To learn this update, we fit the proposals in each sampled update sequence to posterior samples $\boldsymbol{x}^*$ from training tasks by minimizing the negative log likelihood averaged over the T updates,

$$\mathcal{L}(\theta) = -\mathbb{E}_{(r,o)}\mathbb{E}_{\boldsymbol{x}^*\sim\pi_{r,o}^x}\left[\frac{1}{T}\sum_{t=1}^{T}\log q^{x,(t)}(\boldsymbol{x}^*)\right]. \tag{8}$$

For each proposal, expected negative log likelihood equals $\mathrm{KL}(\pi_{r,o}^x \| q^{x,(t)})$ up to a term independent of θ, so the objective penalizes distribution mismatch across the update sequence. Training propagates gradients through the shared network and accumulated parameter state, treating trial samples and density evaluations as external inputs. At inference, θ is frozen, and T updates give the final proposal $q_{\theta,r,o}^x = q^{x,(T)}$. Fresh samples from this proposal, drawn separately from the trials, are decoded and averaged as in equation 4. With M target density evaluations for the updates, we set $m = M/T$; fresh sampling and averaging require no additional target density evaluations.

4 RGPC Reduces Distribution Mismatch and Posterior-Mean Estimation Error

To test whether RGPC reduces distribution mismatch and posterior-mean estimation error under a fixed number of target density evaluations, we compare RGPC with learned and conventional baselines using MMD^2 and TVD. Controlled comparisons examine current-task inputs and iterative updates. We then evaluate new tasks with frozen networks and sampling configurations.

Models. We compare RGPC with a population-adapted Distribution Transformer (PDT) (Whittle et al., 2026), conditional adaptations of CTGAN and TVAE (Xu et al., 2019), TabDDPM (Kotelnikov et al., 2023), and TabDiff (Shi et al., 2025). All six learned methods use the same source-posterior training samples and observed aggregates. The generator adaptations retain their native training objectives and sampling procedures. RGPC and PDT use covariance-factor rank 4. The conventional baselines are the initial proposal with importance sampling (IS), weighted expectation-maximization (WEM) with IS (Cappé et al., 2008), and adaptive-tempered sequential Monte Carlo (SMC) (Del Moral et al., 2006).

Tasks. We evaluate 192 development tasks from 32 PUMAs using 2023 ACS aggregate statistics (United States Census Bureau, 2023a; 2026). Each task represents the population distribution as a 96-cell table of age, sex, education, and employment. The evaluation PUMAs are excluded from learned-sampler training, with their states and the six observation patterns represented in training. We use these development tasks to select RGPC's update count by average TVD (Appendix B.3). All methods approximate the same target posterior for each task, as defined in section 3.1.

Sampling and evaluation settings. All methods use 1,024 target density evaluations per task, excluding source training and shared initial-proposal construction. RGPC uses 16 updates with 64 trial samples each; the other learned methods obtain feedback from the initial proposal in one pass. Learned methods return 4,096 samples; Table 1 gives all output counts and weights. We measure distribution mismatch on decoded population distributions by Gaussian-kernel squared maximum mean discrepancy (MMD^2) (Gretton et al., 2012) against 512 shared Hamiltonian Monte Carlo

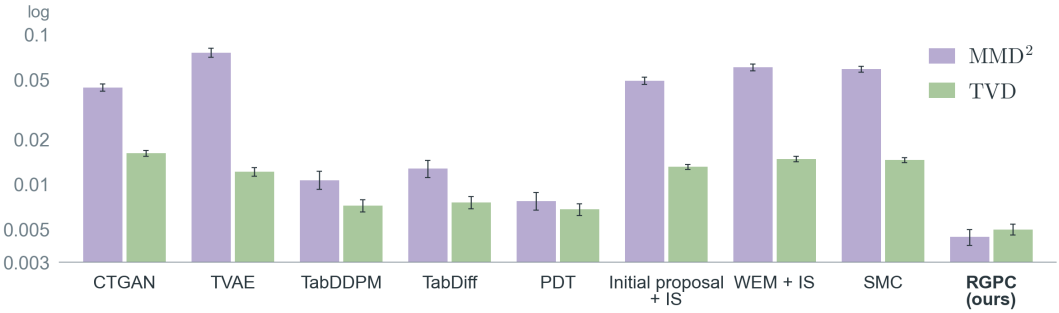


Figure 3: **Posterior approximation.** Each method uses 1,024 target density evaluations per task. Bars show mean MMD^2 and TVD across 32 development PUMAs, with pointwise 95% PUMA-bootstrap intervals. Output counts and weighting follow Table 1. The logarithmic axis retains the original metric values; lower values are better.

(HMC) reference samples. Posterior-mean estimation error is the TVD $\frac{1}{2}\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}^{\mathrm{ref}}\|_1$, where $\boldsymbol{\mu}^{\mathrm{ref}}$ is the common HMC reference mean. Both metrics retain the output weights. Reference diagnostics and metric details are in Appendices B.3 and B.9.

The learned methods use three training seeds in one source-training group, and the conventional methods use three sampling seeds. We average eight sampling repeats and the three seeds within each task, observation patterns within each PUMA, and then PUMAs equally. The 16 paired comparisons across eight baselines and both metrics use 10,000 PUMA-cluster bootstrap draws with Bonferroni-adjusted family-wise 95% intervals, conditional on the selected configuration, trained models, and shared references.

4.1 Posterior Approximation Accuracy

When using the same number of target density evaluations, RGPC achieves the lowest distribution mismatch and posterior-mean estimation error among all nine methods (Figure 3). Table 1 reports the means and output counts.

Table 1: Posterior approximation on 192 development tasks. Metric values are means with equal weight for each of the 32 PUMAs. All methods target the same posterior for each task and use 1,024 target density evaluations, excluding source training and shared initial-proposal construction. Lower metric values are better.

(a) Learned methods

Method	MMD^2	TVD
CTGAN (NeurIPS 2019)	0.0441	0.0160
TVAE (NeurIPS 2019)	0.0751	0.0121
TabDDPM (ICML 2023)	0.0106	0.0071
TabDiff (ICLR 2025)	0.0127	0.0075
PDT (ICML 2026)	0.0077	0.0068
RGPC (ours)	**0.0044**	**0.0050**

Each learned method returns 4,096 equally weighted samples per task.

(b) Conventional baselines

Method	Samples	MMD^2	TVD
Initial proposal + IS	1,024	0.0487	0.0130
WEM + IS	896	0.0602	0.0147
SMC	512	0.0585	0.0144
RGPC (ours)	4,096	**0.0044**	**0.0050**

IS outputs use normalized importance weights; SMC and RGPC outputs are equally weighted. Counts are per task.

Venues and years refer to the source methods; PDT adapts Distribution Transformers. RGPC is repeated in (b) for comparison. Metrics are rounded to four decimal places; reported relative changes use unrounded means.

Relative to PDT, the best-performing baseline, RGPC reduces MMD^2 by 42.65%. The paired MMD^2 difference is -0.003282, with family-wise 95% interval $[-0.004181, -0.002454]$. On the same outputs, posterior-mean estimation error is 26.50% lower than that of PDT, with paired difference -0.001790 and family-wise 95% interval $[-0.002220, -0.001377]$. Across individual tasks, RGPC has lower MMD^2 on 185 of 192 tasks and lower TVD on 186 of 192 tasks than PDT.

All 16 paired comparisons favor RGPC, with family-wise intervals below zero. Both metric means remain lower for RGPC when conventional samplers also use iterative updates (Appendix B.6).

4.2 Current-task information and iterative updates

To test how current-task information and density-ratio feedback support RGPC's updates (section 3.2), we intervene at parameter prediction and during later sampling updates. The controls use the main configuration, development tasks, three seeds, and eight sampling repeats.

First, we test the direct coordinate features and feedback summaries used to predict each component's location and spread. One control substitutes training-task averages for the coordinate features; the other zeros only the feedback summaries. Both retain the actual aggregate constraints, target posteriors, attention inputs, and current proposal parameters, with matched architecture, initialization, and retraining settings. Replacing coordinate features increases MMD^2 by 45.34% and TVD by 23.84%; zeroing feedback summaries increases them by 12.00% and 6.09%, respectively. Both direct inputs improve posterior approximation.

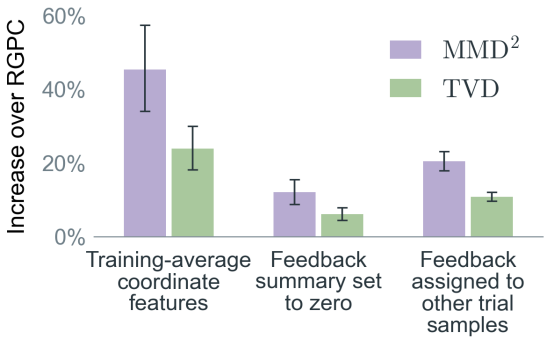


Figure 4: **Current-task inputs and feedback pairing.** Bars show percentage increases in MMD^2 and TVD relative to full RGPC. The first two controls are retrained after changing one direct input at parameter prediction. The third shuffles density-ratio feedback among trial locations at updates 2–16 without retraining. Whiskers show family-wise 95% paired PUMA-bootstrap intervals across all six comparisons.

To test whether later updates use the correspondence between density-ratio feedback and trial locations, we keep the trained networks unchanged and shuffle feedback values among trial locations at updates 2–16. The first update is unchanged, and each later permutation preserves the set of feedback values. Shuffling increases MMD^2 by 20.37% and TVD by 10.78%. Correctly pairing density-ratio feedback with its trial samples therefore improves iterative proposal updates.

All three controls increase distribution mismatch and posterior-mean estimation error in each of the three seeds. Figure 4 shows the relative increases, and Table 2 reports the metric means.

We quantify uncertainty in these comparisons using 10,000 paired PUMA-bootstrap draws after averaging seeds. We apply Bonferroni adjustment across the six comparisons to obtain family-wise 95% intervals. All six intervals lie above zero, conditional on the trained models and shared references.

Table 2: Posterior approximation for RGPC and the three interventions. Values are means across PUMAs under the evaluation settings in Table 1; lower values are better.

Setting	MMD^2 $\times 10^{-4}$	TVD $\times 10^{-4}$
Training-average coordinate features	64.14	61.47
Feedback summary set to zero	49.43	52.66
Feedback assigned to other trial samples	53.13	54.99
RGPC (ours)	**44.13**	**49.64**

Means: two decimals; changes use unrounded values.

To distinguish the two parts of the coordinate features, we separately replace constraint directions and initial-proposal rescaling. Replacing directions increases MMD^2 by 43.33% and TVD by 22.71%. In contrast, both intervals for the rescaling control span zero (Table 13 in Appendix B.11). These comparisons support using current constraint directions at parameter prediction. Earlier single-update controls are in Appendix C.

Iterative updates. To test whether iterative updates improve posterior approximation under a fixed number of target density evaluations, we compare separately trained single-update and 16-update models. Both use 1,024 target density evaluations and 4,096 fresh output samples. The single-update model uses 1,024 trial samples, whereas the 16-update model uses 64 from its current sampling distribution per update. Across the 192 development tasks, the 16-update model achieves 13.85%

lower mean MMD^2 and 7.22% lower mean TVD, with lower means for each of the three training seeds (Table 5).

For one preselected task, Figure 5 shows how the sampling distribution changes during a single run of the 16-update model. Between updates 1 and 16, MMD^2 decreases from 0.007273 to 0.004596, and TVD from 0.007751 to 0.006198, with small fluctuations in the later updates. This trajectory illustrates continued refinement after the first update.

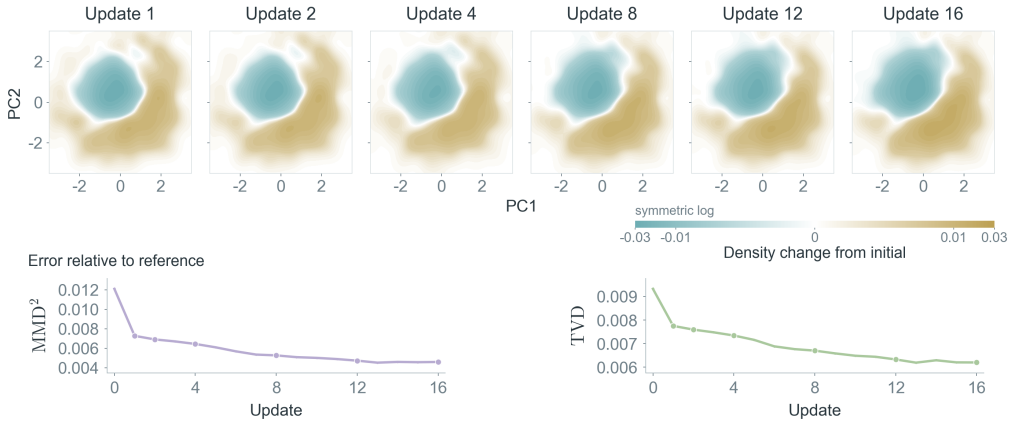


Figure 5: **Iterative proposal updates.** One preselected trajectory of the trained 16-update model is shown; each update uses 64 target density evaluations. Maps show projected density changes from the initial proposal: teal denotes a decrease and gold an increase, on a shared symmetric-log scale. Curves show MMD^2 and TVD on all 96 cell probabilities. The reference is used only for projection and evaluation; Appendix B.9 gives the projection and scoring settings.

4.3 Validation across tasks and evaluation settings

To test reuse without retraining, we freeze the networks and sampling configurations and evaluate 32 PUMAs excluded from sampler training and configuration selection. Their states and observation patterns are represented in training. At 1,024 target density evaluations, RGPC reduces mean MMD^2 by 49.53% and TVD by 30.76% relative to PDT. Both means are also lower than those of weighted EM and SMC after iterative updates, with their configurations selected on development tasks. All six paired comparisons have family-wise 95% intervals below zero (Appendix B.7).

On the development tasks, RGPC retains lower means than PDT and the conventional baselines in Table 1 at 512, 1,024, and 2,048 target density evaluations (Appendix B.8). At 1,024 evaluations, we also rescore unchanged RGPC and PDT outputs using two posterior references and three kernel bandwidths. Both means remain lower for RGPC in all six settings (Appendix B.10).

5 Conclusion

We introduced the Ratio-Guided Proposal Compiler (RGPC), a learned model for posterior inference under a fixed number of target density evaluations. By recomputing density-ratio feedback at each update, RGPC uses the same trained network to update its sampling distribution. On our population-inference benchmark, RGPC achieved the lowest distribution mismatch and posterior-mean estimation error among nine evaluated methods. Controlled comparisons further showed improved posterior approximation when RGPC retained current-task inputs at parameter prediction and correctly paired density-ratio feedback with trial samples in later updates. Together, these results support using current-task information and density-ratio feedback to guide iterative proposal updates for more accurate posterior inference. Future work could evaluate RGPC for higher-dimensional population distribution inference and explore training with previously unseen observation patterns.

AI USE STATEMENT

We used generative AI tools to assist with polishing, coding, and debugging. The authors take responsibility for the final content, including AI-assisted text and code.

ETHICS STATEMENT

This study uses publicly available ACS aggregate statistics and Public Use Microdata Sample (PUMS) records to infer population distributions. The outputs describe demographic proportions, not identified individual records. The method provides no formal privacy guarantee, and inferred proportions can reflect biases in the source data and population prior. Applications in planning or public health should therefore assess privacy risks, subgroup accuracy, and potential disparate impacts before deployment.

REPRODUCIBILITY STATEMENT

Appendix A specifies the representation, architecture, and configuration. Appendix B documents data processing, task splits, posterior references, baselines, metrics, and uncertainty estimation. The anonymous repository linked in the abstract provides the core implementation, focused tests, and released scores. It also includes one prepared task, a trained checkpoint, posterior reference samples, and a command that generates samples and computes both metrics. Full benchmark reproduction requires the remaining task assets, posterior caches, training runners, and checkpoints.

REFERENCES

Oleg Arenz, Philipp Dahlinger, Zihan Ye, Michael Volpp, and Gerhard Neumann. A unified perspective on natural gradient variational inference with Gaussian mixture models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=tLBjsX4tjs.

Emmy Blumenthal, Nikolas Claussen, Benjamin Eysenbach, Catherine Ji, Gautam Reddy, Colin Scheibner, and Benjamin Sorkin. Bridging control, inference, transport, and thermodynamics: From theory to applications in learning. *arXiv preprint arXiv:2609.15897*, 2026. URL https://arxiv.org/abs/2609.15897.

Olivier Cappé, Randal Douc, Arnaud Guillin, Jean-Michel Marin, and Christian P. Robert. Adaptive importance sampling in general mixture classes. *Statistics and Computing*, 18(4):447–459, 2008. doi: 10.1007/s11222-008-9059-x.

Junhua Chen, Lorenz Richter, Julius Berner, Denis Blessing, Gerhard Neumann, and Anima Anandkumar. Sequential controlled Langevin diffusions. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/a71c1931d3fb8ba564f7458d0657d0b1-Abstract-Conference.html.

Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3(1):146–158, 1975. doi: 10.1214/aop/1176996454.

Jan de Mooij, Tabea Sonnenschein, Marco Pellegrino, Mehdi Dastani, Dick Ettema, Brian Logan, and Judith A. Verstegen. GenSynthPop: Generating a spatially explicit synthetic population of individuals and households from aggregated data. *Autonomous Agents and Multi-Agent Systems*, 38:48, 2024. doi: 10.1007/s10458-024-09680-7.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006. doi: 10.1111/j.1467-9868.2006.00553.x.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783v1*, 2024. URL https://arxiv.org/abs/2407.21783v1.

Víctor Elvira, Luca Martino, David Luengo, and Mónica F. Bugallo. Generalized multiple importance sampling. *Statistical Science*, 34(1):129–155, 2019. doi: 10.1214/18-STS668. URL https://arxiv.org/abs/1511.03095.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Charlie Frogner and Tomaso Poggio. Fast and flexible inference of joint distributions from their marginals. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2002–2011. PMLR, 2019. URL https://proceedings.mlr.press/v97/frogner19a.html.

John Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57(6):1317–1339, 1989. doi: 10.2307/1913710.

Manuel Gloeckler, Michael Deistler, Christian Dietrich Weilbach, Frank Wood, and Jakob H. Macke. All-in-one simulation-based inference. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 15735–15766. PMLR, 2024. URL https://proceedings.mlr.press/v235/gloeckler24a.html.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL https://jmlr.org/papers/v13/gretton12a.html.

Tim Hesterberg. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37(2):185–194, 1995. doi: 10.1080/00401706.1995.10484303.

Akim Kotelnikov, Dmitry Baranchuk, Ivan Rubachev, and Artem Babenko. TabDDPM: Modelling tabular data with diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17564–17579. PMLR, 2023. URL https://proceedings.mlr.press/v202/kotelnikov23a.html.

Tuan Anh Le, Atilim Gunes Baydin, and Frank Wood. Inference compilation and universal probabilistic programming. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 1338–1348. PMLR, 2017. URL https://proceedings.mlr.press/v54/le17a.html.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, 2004. doi: 10.1016/S0047-259X(03)00096-4.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Joe Marino, Yisong Yue, and Stephan Mandt. Iterative amortized inference. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3403–3412. PMLR, 2018. URL https://proceedings.mlr.press/v80/marino18a.html.

Alex Matthews, Michael Arbel, Danilo Jimenez Rezende, and Arnaud Doucet. Continual repeated annealed flow transport Monte Carlo. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15196–15219. PMLR, 2022. URL https://proceedings.mlr.press/v162/matthews22a.html.

Sarthak Mittal, Divyat Mahajan, Guillaume Lajoie, and Mohammad Pezeshki. Iterative amortized inference: Unifying in-context learning and learned optimizers. *arXiv preprint arXiv:2510.11471*, 2025. URL https://arxiv.org/abs/2510.11471.

Zijing Ou, Chinmay Pani, and Yingzhen Li. Inference-time scaling of discrete diffusion models via importance weighting and optimal proposal design. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7wbrFQvfdH.

Ari Pakman and Liam Paninski. Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians. *Journal of Computational and Graphical Statistics*, 23(2):518–542, 2014. doi: 10.1080/10618600.2013.788448.

James Rineer, Nicholas Kruskamp, Caroline Kery, Kasey Jones, Rainer Hilscher, and Georgiy Bobashev. A national synthetic populations dataset for the United States. *Scientific Data*, 12:144, 2025. doi: 10.1038/s41597-025-04380-7.

Abdoul Razac Sané, Pierre-Olivier Vandanjon, Rachid Belaroussi, and Pierre Hankach. A comprehensive investigation of variational auto-encoders for population synthesis. *Journal of Computational Social Science*, 8:13, 2025. doi: 10.1007/s42001-024-00332-0.

Juntong Shi, Minkai Xu, Harper Hua, Hengrui Zhang, Stefano Ermon, and Jure Leskovec. TabDiff: A mixed-type diffusion model for tabular data generation. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=swvURjrt8z.

United States Census Bureau. 2023 ACS 5-year detailed tables: Geographies, 2023a. URL https://api.census.gov/data/2023/acs/acs5/geography.html. Accessed 2026-09-21.

United States Census Bureau. Public use microdata areas (PUMAs), 2023b. URL https://www.census.gov/programs-surveys/geography/guidance/geo-areas/pumas.html. Accessed 2026-09-21.

United States Census Bureau. Public use microdata sample (PUMS), 2026. URL https://www.census.gov/programs-surveys/acs/microdata.html. Accessed 2026-09-21.

George Whittle, Juliusz Ziomek, Jacob Rawling, and Maike A. Osborne. Distribution Transformers: Fast approximate Bayesian inference with on-the-fly prior adaptation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2502.02463. Spotlight.

Hao Wu, Heiko Zimmermann, Eli Sennesh, Tuan Anh Le, and Jan-Willem van de Meent. Amortized population Gibbs samplers with neural sufficient statistics. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10421–10431. PMLR, 2020. URL https://proceedings.mlr.press/v119/wu20h.html.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/254ed7d2de3b23ab10936522dd547b78-Abstract.html.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep Sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/f22e4747da1aa27e363d86d40ff442fe-Abstract.html.

A Representation and Architecture Details

This appendix gives the coordinate transformations and neural operations used in section 3.2. The main experimental protocol is specified in Appendix B, and the current-task input and feedback controls in Appendix B.11. Earlier single-update controls and supplementary comparisons are reported in Appendices C and D, respectively.

A.1 Feasible coordinates and posterior densities

After identifying cells forced to zero, we restrict $\mathbf{A}_o$ to the active cells $\mathcal{S}_{r,o}$. A singular-value decomposition gives the nullspace projector. Orthogonalizing projected cell axes in a prescribed order yields the canonical orthonormal basis $\mathbf{B}_{r,o}$, with $d_{r,o}$ columns. Row reordering, positive row scaling, redundant rows, and invertible row transformations preserve this projector and hence the coordinate convention. For $\boldsymbol{z} \in \mathbb{R}^{d_{r,o}}$, the decoder $\mathbf{D}_{r,o}$ exponentiates and normalizes $\mathbf{B}_{r,o}\boldsymbol{z}$ over active cells to form a positive seed $\boldsymbol{p}^{\text{seed}}$. It then minimizes $\mathrm{KL}(\boldsymbol{p}\|\boldsymbol{p}^{\text{seed}})$ over the population fiber. This information projection enforces the supplied aggregate constraints and returns the feasible population distribution $\mathbf{D}_{r,o}(\boldsymbol{z})$.

To express posterior training samples in these coordinates, we also need the inverse map. The projection's stationarity conditions make $\log \boldsymbol{p}_{\mathcal{S}_{r,o}} - \mathbf{B}_{r,o}\boldsymbol{z}$ a combination of constraint rows, including normalization. Multiplying by $\mathbf{B}_{r,o}^{\mathsf{T}}$ therefore gives

$$\boldsymbol{z} = \mathbf{B}_{r,o}^{\mathsf{T}} \log \boldsymbol{p}_{\mathcal{S}_{r,o}}, \qquad \boldsymbol{p} = \mathbf{D}_{r,o}(\boldsymbol{z}). \tag{9}$$

The logarithm is entrywise and uses only active cells. Strict convexity of the projection objective gives a unique decoded population distribution, so these maps are inverses on the relative interior of the fiber.

Density transformation. Superscripts z and x denote densities in the original and rescaled free coordinates. To preserve the posterior defined with respect to intrinsic volume on the fiber, its density in free coordinates includes the decoder's volume Jacobian:

$$\log \pi_{r,o}^{z}(\boldsymbol{z}) = \log \rho_r(\mathbf{D}_{r,o}(\boldsymbol{z})) + \log \left|J_{\mathbf{D}_{r,o}}(\boldsymbol{z})\right| + C_{r,o}, \tag{10}$$

where $C_{r,o}$ is an unknown normalizing constant and

$$|J_{\mathbf{D}}(\boldsymbol{z})| = \sqrt{\det[(\partial \mathbf{D}/\partial \boldsymbol{z})^{\mathsf{T}}(\partial \mathbf{D}/\partial \boldsymbol{z})]}.$$

This factor makes the log-seed representation and the affine representation used for HMC refer to the same target posterior.

Initial proposal and rescaling. The task completion is a deterministic feasible population distribution computed from the observed aggregates, as specified in Appendix B.2. Starting from the encoded task completion, BFGS minimizes the negative coordinate log density to locate the Gaussian center $\boldsymbol{m}_{r,o}$. The maximum iteration count is 1,000 and the gradient-norm tolerance is 10^{-7}. If needed, a least-squares solve of the gradient equations refines the center. A relative eigenvalue floor of 10^{-4} regularizes the BFGS inverse-Hessian estimate to give covariance $\mathbf{L}_{r,o}\mathbf{L}_{r,o}^{\mathsf{T}}$. The initial proposal combines this Gaussian with a Student-t component using the same location and coordinate rescaling. The transformation

$$\boldsymbol{x} = \mathbf{L}_{r,o}^{-1}(\boldsymbol{z} - \boldsymbol{m}_{r,o}) \tag{11}$$

makes the Gaussian standard normal. A draw decodes as $\boldsymbol{p} = \mathbf{D}_{r,o}(\boldsymbol{m}_{r,o} + \mathbf{L}_{r,o}\boldsymbol{x})$. Conversely, a posterior sample $\boldsymbol{p}^*$ from a training task becomes

$$\boldsymbol{x}^* = \mathbf{L}_{r,o}^{-1}\left(\mathbf{B}_{r,o}^{\mathsf{T}} \log \boldsymbol{p}_{\mathcal{S}_{r,o}}^{*} - \boldsymbol{m}_{r,o}\right), \tag{12}$$

which supplies the training sample in equation 8.

Reversing the rescaling gives

$$q_{\theta,r,o}^{z}(\boldsymbol{z}) = \frac{q_{\theta,r,o}^{x}\left(\mathbf{L}_{r,o}^{-1}(\boldsymbol{z} - \boldsymbol{m}_{r,o})\right)}{|\det \mathbf{L}_{r,o}|}. \tag{13}$$

The same transformation applies to q_0. The target density satisfies $\pi_{r,o}^{x}(\boldsymbol{x}) = \pi_{r,o}^{z}(\boldsymbol{m}_{r,o} + \mathbf{L}_{r,o}\boldsymbol{x})|\det \mathbf{L}_{r,o}|$, so the density comparisons in equation 5 use the same coordinates.

Numerical decoding. To avoid numerical underflow, we decode only finite log-seed vectors whose range is below 650. If this check fails, or decoding does not return positive probabilities on all active cells, we substitute the task completion without resampling. In the main development comparison, this rule replaced one of 18,874,368 RGPC output samples.

A.2 Inputs, attention, and parameter prediction

Task and trial-sample inputs. Global features contain the task completion and the decoded initial-proposal mode, represented by 96-scaled cell probabilities and active-cell log probabilities. They also include the active-cell mask, normalized fiber dimension, active-cell fraction, and log condition number of the initial covariance. For each coordinate, features contain its canonical basis direction, the corresponding column of $\mathbf{B}_{r,o}\mathbf{L}_{r,o}$, the log norm of that column, the corresponding diagonal entry of $\mathbf{L}_{r,o}$, the normalized coordinate index, and the normalized fiber dimension. Trial-sample features combine $\boldsymbol{x}_i$, the log-seed displacement $\mathbf{B}_{r,o}\mathbf{L}_{r,o}\boldsymbol{x}_i$, and the centered log-density ratio ℓ_i. A dimension mask identifies active coordinates; inactive cell and unused coordinate entries are zero-padded. The numerical inputs are transformed by asinh before their respective encoders. At each update, a component–coordinate pair contributes seven current-proposal features: one centered logit, one location, one log scale, and four covariance-factor entries. A bias-free Linear–GELU–Linear encoder of widths $7 \to 64 \to 128$ maps these features to $\boldsymbol{c}_{kj}^{(t)}$. After masking inactive coordinates, the mean of these encodings over active coordinates conditions the attention query, while each $\boldsymbol{c}_{kj}^{(t)}$ enters its corresponding coordinate representation in equation 6. Table 3 specifies the main RGPC configuration, and Figure 6 shows the module connections and tensor shapes. The earlier single-update covariance-capacity comparison is reported separately in Appendix D.1.

Table 3: RGPC configuration in the main benchmark. Encoder and decoder entries give input, hidden, and output widths for Linear–GELU–Linear modules. Training settings refer to the recurrent stage initialized from the corresponding single-update checkpoint.

Quantity	Value
Population cells	96
Maximum fiber dimension	51
Updates	16
Trial target density evaluations per update	64
Total target density evaluations per task	1,024
Final output samples per task	4,096
Central mixture components	4
Model dimension	128
Attention heads	8 (16 dimensions each)
Trial-sample encoder	$148 \to 192 \to 128$
Global encoder	$483 \to 192 \to 128$
Coordinate encoder	$196 \to 192 \to 128$
Moment encoder	$2 \to 32 \to 128$
Current-proposal encoder	$7 \to 64 \to 128$
Component decoder	$256 \to 256 \to 128$
Pair decoder	$128 \to 256 \to 128$
Covariance-factor rank	4
Retained tail mass	0.5
Student-t degrees of freedom	3
Optimizer	AdamW
Learning rate	3×10^{-4}
Weight decay	0
Gradient clipping norm	10
Training batch size (tasks)	8
Training microbatch size (tasks)	2
Recurrent training epochs	20
Source-training groups	1
Training seeds	3

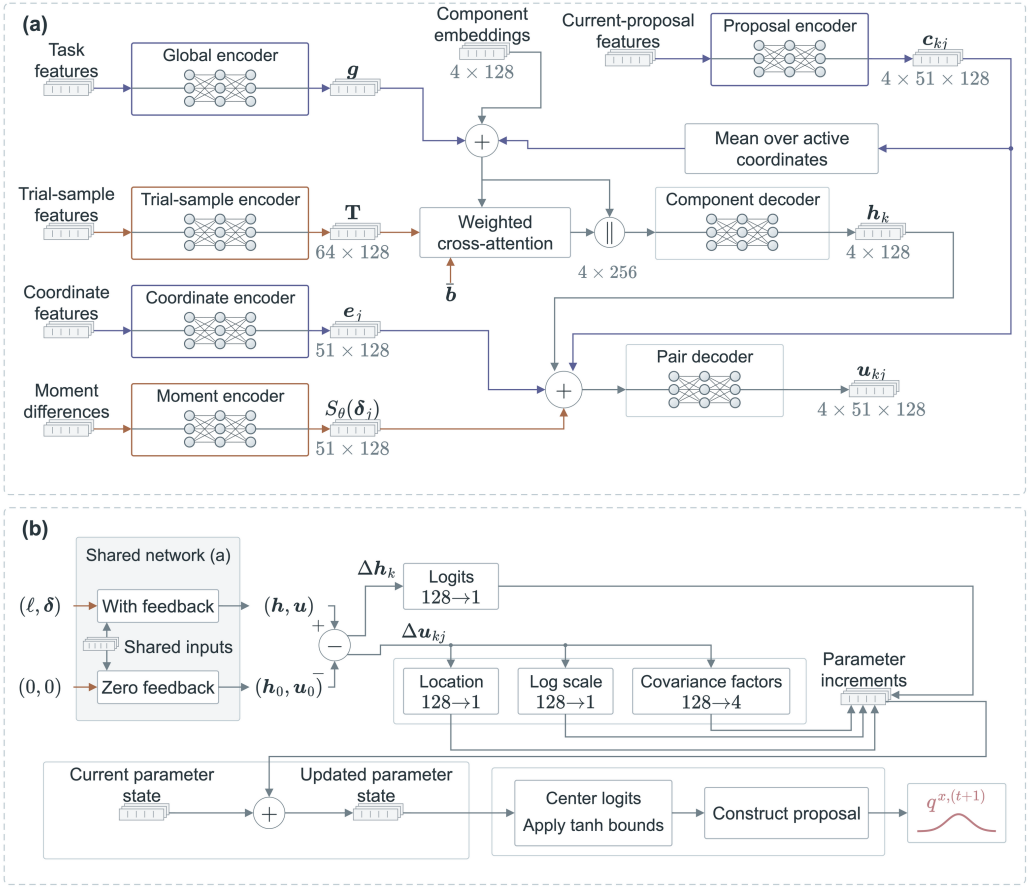


Figure 6: **One RGPC update at layer level.** (a) combines encoded task and sampling inputs into component representations $\boldsymbol{h}_k$ and component–coordinate representations $\boldsymbol{u}_{kj}$. Colors follow Figure 2. Weighted cross-attention uses base masses and bounded scores as defined in equations 14 and 15. (b) evaluates network (a) with feedback and with $\ell, \boldsymbol{\delta}$ set to zero, retaining all other inputs and network weights. Bias-free heads map $\Delta\boldsymbol{h}_k$ to logit increments and $\Delta\boldsymbol{u}_{kj}$ to location, log-scale, and covariance-factor increments. After state accumulation, parameter transforms and proposal construction give $q^{x,(t+1)}$, as defined in equation 7. Figure 2 shows the iterative updates with frozen network parameters. Table 3 gives the layer widths. Tensor shapes omit the batch axis and use four Gaussian components, 64 trial samples, 51 padded coordinates, and factor rank 4. The proposal curve is schematic.

Attention and moment differences. For component k and attention head n, let $\boldsymbol{q}_{kn}$ and $\boldsymbol{k}_{in}$ denote the component query and trial-sample key of dimension d_h. Before projection into attention heads, each query combines a learned component embedding with global task context and the mean current-proposal encoding over active coordinates. To bound the learned part of the attention score, we use

$$s_{kin} = 4 \tanh\left(\frac{\langle \boldsymbol{q}_{kn}, \boldsymbol{k}_{in} \rangle}{4\sqrt{d_h}}\right). \tag{14}$$

Each sample has base mass b_i, with $\bar{b}_i = b_i / \sum_j b_j$; ordinary independent trial draws have uniform masses. The attention weight is

$$\alpha_{kin} = \frac{\bar{b}_i \exp(s_{kin})}{\sum_j \bar{b}_j \exp(s_{kjn})}. \tag{15}$$

Concatenating the attention summary and query and passing them through the component decoder gives the component representation $\boldsymbol{h}_k$.

The moment differences describe how density-ratio weighting changes each coordinate's first and second moments. Let $c_i = \mathrm{clip}(\ell_i, -60, 60)$ and $\omega_i = \bar{b}_i \exp(c_i) / \sum_m \bar{b}_m \exp(c_m)$. The two entries of $\boldsymbol{\delta}_j$ are $\sum_i (\omega_i - \bar{b}_i) x_{ij}$ and $\sum_i (\omega_i - \bar{b}_i) x_{ij}^2$. Their encoding $S_\theta(\boldsymbol{\delta}_j)$ enters the coordinate representation in equation 6. These feature weights are distinct from the learned attention weights α_{kin} and the optional final importance weights in equation 18.

Zero-feedback pass and output heads. To preserve the current proposal when the density-ratio feedback is zero, the zero-feedback pass sets $\ell_i = 0$ and $\boldsymbol{\delta}_j = \mathbf{0}$, retaining trial locations, base masses, task and coordinate features, current-proposal encodings, and neural parameters. Within one update, denote the zero-feedback representations by $\boldsymbol{h}_{k,0}$ and $\boldsymbol{u}_{kj,0}$, where the subscript 0 identifies the zero-feedback pass. We form

$$\Delta \boldsymbol{h}_k = \boldsymbol{h}_k - \boldsymbol{h}_{k,0}, \qquad \Delta \boldsymbol{u}_{kj} = \boldsymbol{u}_{kj} - \boldsymbol{u}_{kj,0}. \tag{16}$$

Bias-free heads predict logit increments from $\Delta \boldsymbol{h}_k$ and location, log-scale, and covariance-factor increments from $\Delta \boldsymbol{u}_{kj}$. These increments are added to the proposal's unconstrained parameter state retained from the previous update. Only after this addition are logits centered across components and scaled $\tanh$ transforms applied to locations $\boldsymbol{\xi}_k$, log scales $\boldsymbol{s}_k$, and covariance factors $\mathbf{F}_k$. Coordinate masks retain active entries. The component covariance is

$$\mathbf{C}_k = \mathrm{diag}(\exp(2\boldsymbol{s}_k)) + \mathbf{F}_k \mathbf{F}_k^{\mathsf{T}}. \tag{17}$$

The positive diagonal term makes $\mathbf{C}_k$ positive definite. The factor rank is the number of columns of $\mathbf{F}_k$, not the rank of the full covariance or the number of trial samples. We use rank 4 in the main benchmark and mechanism controls.

The retained Student-t component uses $\tau = 0.5$, $\nu = 3$, and $\mathbf{S}_{\mathrm{tail}} = ((\nu - 2)/\nu)\mathbf{I}$, giving unit covariance after rescaling. Under zero density-ratio feedback, both representation differences and all parameter increments vanish, so $q^{x,(t+1)} = q^{x,(t)}$. At the start of each task, the parameter state is zero, all Gaussian components are standard normal, and $q^{x,(0)} = q_0^x$. The original single-update core initializes all output heads except the covariance-factor head at zero. Its covariance-factor head uses bound-scaled initialization to avoid the zero derivative of $\mathbf{F}\mathbf{F}^{\mathsf{T}}$ at $\mathbf{F} = \mathbf{0}$. Recurrent training starts from the trained single-update core, as specified in Table 3. Scaled $\tanh$ functions bound mean shifts, log scales, and factor entries by 12, 4, and 6, respectively. Centered log-density ratios are clipped to $[-60, 60]$ for neural input and moment computation. These settings are shared across tasks.

A.3 Optional importance correction

The main RGPC estimate is the ordinary mean in equation 4. When additional target density evaluations are available, the same proposal also permits self-normalized importance weighting. For fresh draws $\boldsymbol{x}'_n \sim q^x_{\theta,r,o}$, let $\boldsymbol{p}'_n = \mathbf{D}_{r,o}(\boldsymbol{m}_{r,o} + \mathbf{L}_{r,o}\boldsymbol{x}'_n)$. The corrected estimate is

$$\widetilde{w}_n = \frac{\pi^x_{r,o}(\boldsymbol{x}'_n)}{q^x_{\theta,r,o}(\boldsymbol{x}'_n)}, \qquad w_n = \frac{\widetilde{w}_n}{\sum_m \widetilde{w}_m}, \qquad \widehat{\boldsymbol{\mu}}^{\mathrm{IS}}_{r,o} = \sum_{n=1}^{N} w_n \boldsymbol{p}'_n. \tag{18}$$

The target density's normalizing constant cancels. The weights use the target and proposal densities, without the feedback clipping used inside RGPC. This correction adds N target density evaluations beyond the M trial-sampling evaluations, for a total of $M + N$. For the main 16-update configuration, applying this correction to all 4,096 fresh samples would require $1{,}024 + 4{,}096 = 5{,}120$ evaluations in total. The earlier single-update comparisons in Appendix D.1 use 128 trial evaluations and 4,096 fresh samples, totaling 4,224 target density evaluations.

B Experimental Protocol

This protocol describes the nine-method benchmark in section 4. Its learned estimators use ordinary sample means; importance-corrected and crossed-jurisdiction comparisons have separate protocols in Appendix D.

B.1 Population data and observation patterns

Public 2023 ACS tables supply aggregate constraints, and survey-weighted Public Use Microdata Sample (PUMS) records yield source-region population distributions (United States Census Bureau, 2023a;b; 2026). The 96 cells combine three age groups (25–34, 35–44, and 45–64), two sex groups, four education groups, and four employment groups. We use these attributes because compatible ACS margins constrain their joint distribution while leaving some associations to be inferred. ACS B15001 and age-resolved B23001 define the common age–sex–education and age–sex–employment margins. The common operator contains 48 rows and has rank 42. Each additional observation pattern supplies four education–employment statistics from B23006. Their sum is determined by a common margin, so they increase the operator rank by three to 45. The nominal fiber dimension is therefore 51 before forced-zero cells are removed; active dimensions range from 31 to 51 in the source assets.

Four patterns select education rows of the B23006 table, and four select employment columns (Table 4). Each supplies one block and leaves twelve B23006 statistics unobserved. A pattern excluded from training thus changes which statistics are supplied, not just their numerical values. The main benchmark uses six patterns represented in training; the separate crossed study evaluates patterns excluded from training.

Table 4: Additional observed blocks used with the common ACS margins. The main benchmark evaluates the six blocks other than less than high school and employed.

Orientation	Published B23006 block
Education	Less than high school
Education	High school or GED
Education	Some college or associate degree
Education	Bachelor's degree or higher
Employment	Employed
Employment	Unemployed
Employment	Armed Forces
Employment	Not in labor force

B.2 Gaussian population prior

Let $\mathbf{B}_0$ be an orthonormal basis for the kernel of the common observation operator. For region r, the common completion $\boldsymbol{p}_{0,r}$ matches the public common margins by making education and employment conditionally independent within each age–sex group. Projecting this completion onto the additional observed block gives the task completion used by the model. Both are deterministic feasible constructions.

To model population interactions, each population distribution from source PUMS records is centered on the independent distribution formed from its own PUMS margins and projected onto $\mathbf{B}_0$. A Gaussian is fitted to these interaction coordinates. At inference, the public common completion centers the interaction model under the supplied margins:

$$\boldsymbol{\eta}_r(\boldsymbol{p}) = \mathbf{B}_0^{\mathsf{T}}(\boldsymbol{p} - \boldsymbol{p}_{0,r}), \qquad \rho_r(\boldsymbol{p}) \propto \mathcal{N}(\boldsymbol{\eta}_r(\boldsymbol{p}); \boldsymbol{m}_f, \boldsymbol{\Sigma}_f)\,. \tag{19}$$

This density is restricted to population distributions satisfying the common margins. The index f denotes an outer geographic fold: $(\boldsymbol{m}_f, \boldsymbol{\Sigma}_f)$ are fitted from PUMS-derived population distributions in jurisdictions outside that fold using Ledoit–Wolf covariance estimation (Ledoit & Wolf, 2004). The main benchmark uses the first fold's 32 source states. Its evaluation PUMAs contribute to this earlier population-prior fit but are excluded from learned-sampler training. The benchmark therefore evaluates sampling under a given population prior. Restricting the population prior to each task's additional observed aggregates gives equation 2, a truncated Gaussian in affine probability coordinates.

B.3 Training tasks and posterior references

Within each source state and outer fold, two distinct PUMAs are selected using their public active-dimension profiles and identifiers. Their training and development roles alternate deterministically

across state–fold pairs. The main evaluation uses one development PUMA in each of 32 source states and six observation patterns, giving 192 tasks.

The earlier source assets define two training groups, omitting the less-than-high-school education block and the employed-status block, respectively. The main benchmark uses only the first group. It contains 224 tasks: 32 training PUMAs crossed with seven included patterns. All six evaluation patterns are represented in this group. Each learned method uses three training seeds, and each trained model has eight sampling repeats per evaluation task. These development PUMAs were also used in earlier development comparisons under the two omitted patterns. They remain excluded from learned-sampler training, while the unified benchmark and the mechanism study reuse the same 192-task inventory.

Source-posterior supervision. For each training task, four HMC chains retain 4,096 draws each after 2,048 burn-in transitions. Only the first two chains supply learning targets, giving 8,192 posterior samples per task; the remaining chains support source-reference diagnostics. RGPC and PDT first train for 20 epochs using 128 trial target density evaluations per task. From these single-update checkpoints, they train for 20 further epochs using 1,024 evaluations per task. Both stages cycle through eight disjoint blocks of 1,024 posterior samples per task and use AdamW with learning rate 3×10^{-4}, zero weight decay, task batch size 8, and gradient clipping norm 10. The four generator adaptations use the same 8,192 samples with their own objectives and optimization settings.

Computational cost and reuse. RGPC is intended for repeated inference over related target posteriors under the population prior. Source-posterior generation and network training are performed offline, and the trained network is reused as the observed aggregates change. For each new task, the compared methods share the initial-proposal construction described in Appendix A. RGPC then uses 1,024 target density evaluations across 16 updates with 64 trial samples each, keeping its network weights frozen. In the development benchmark, the first update reuses 64 cached trial samples from q_0 and their target density evaluations. These 64 evaluations count toward the total of 1,024; the remaining 15 updates require 960 new evaluations. Finally, it draws and decodes 4,096 fresh samples and uses their ordinary mean to estimate the posterior mean. These final operations require computation but no additional target density evaluations. Our comparison equalizes target density evaluations, with offline supervision and training, initial-proposal construction, and final sampling and decoding treated as separate costs. Posterior reference samples for a new task are used only for evaluation.

Evaluation references. The common reference comes from earlier cached references for these development PUMAs, not from the larger training-target sample sets. Each evaluation task has two HMC chains with 512 burn-in transitions and 512 retained draws per chain, without thinning (Pakman & Paninski, 2014). We take the second half of each retained chain, decode the resulting 512 samples to population distributions, and use their ordinary mean as $\boldsymbol{\mu}^{\mathrm{ref}}$. The same decoded array is used by every method for both metrics.

The cached-asset checks require complete task coverage, disjoint training and development PUMAs within each state–fold pair, finite values and diagnostics, and normalization and constraint errors at most 10^{-9}. To assess chain mixing, the stored rank-normalized split $\widehat{R}$ and bulk effective sample size (ESS) summarize the affine coordinates and transformed log density on both complete retained chains, totaling 1,024 draws per task. Across the 192 tasks, the maximum $\widehat{R}$ per task has median 1.005 and maximum 1.013; the minimum bulk ESS per task has median 871.6 and minimum 570.7. For the 512 decoded samples used in scoring, we recompute these diagnostics on nonconstant cell probabilities. The corresponding per-task maxima of $\widehat{R}$ have median 1.012 and maximum 1.026, and the per-task minima of bulk ESS have median 337.6 and minimum 102.0. The unified scoring checks also verify common reference hashes and kernel bandwidths across methods. Both metrics and their intervals remain conditional on these finite, correlated reference arrays.

To assess reference-sampling variability independently, we run longer HMC chains for the same 192 development tasks. Each task uses four chains, with 2,048 burn-in transitions and 4,096 retained draws per chain, without thinning. The registered checks require a maximum rank-normalized split $\widehat{R}$ of 1.01 and minimum bulk ESS of 400 on the affine coordinates and transformed log density, together

with normalization and constraint errors at most 10^{-9}. All 192 tasks pass these checks. Across the complete retained chains, the largest $\widehat{R}$ is 1.00085 and the smallest bulk ESS is 10,244.1.

To quantify variation in posterior means across chains, we pool the first two chains and the last two chains separately, giving 8,192 draws per group. The TVD between their decoded means has median 6.05×10^{-4} and maximum 11.10×10^{-4} across tasks. This diagnostic measures agreement between independent chain groups. For the sensitivity checks in Appendix B.10, we take the last 1,024 draws from each chain to form a 4,096-sample reference, whose maximum normalization and constraint error is 1.74×10^{-12}. The main benchmark retains its original 512-sample reference.

Configuration selection. We use 1,024 target density evaluations and 4,096 fresh outputs per learned method as the common comparison settings. Within this setting, we compare $T \in \{1, 2, 4, 8, 16, 32, 64\}$, allocating $m = 1{,}024/T$ trial samples to each update during both training and inference. Each configuration is retrained for 20 epochs from the same seed-matched initialization with the same source-posterior samples and optimization settings. For each of three training seeds, we average eight sampling repeats, six observation patterns per PUMA, and the 32 PUMAs equally. We select the update count with the lowest unrounded average TVD across seeds; numerical ties within 10^{-12} favor fewer updates. Table 5 reports the complete comparison. This rule selects $T = 16$, so each update uses 64 trial samples.

Table 5: Update-count selection under 1,024 target density evaluations. Each configuration returns 4,096 equally weighted samples per task and repeat. Values are means $\pm$ standard deviations across three training seeds after averaging repeats and tasks as in the main comparison. Selection uses unrounded average TVD.

Updates T	Trials per update m	MMD^2 $\times 10^{-4}$	TVD $\times 10^{-4}$
1	1,024	51.23 ± 1.84	53.50 ± 1.14
2	512	49.38 ± 1.81	52.26 ± 0.90
4	256	48.91 ± 3.50	52.15 ± 2.47
8	128	44.31 ± 2.23	49.66 ± 1.26
16 (selected)	64	44.13 ± 1.88	49.64 ± 1.23
32	32	45.23 ± 4.51	50.24 ± 2.92
64	16	44.56 ± 5.40	49.96 ± 3.47

B.4 Learned baselines

Population-adapted Distribution Transformer. PDT uses six Transformer decoder layers with model width 64, eight attention heads, feedforward width 2,048, and no dropout (Whittle et al., 2026). Its four component tokens interact through self-attention and cross-attend to global, coordinate, and trial-sample tokens. Linear heads map each component representation to the complete location, scale, and covariance-factor vectors. Inputs, source-posterior supervision, Gaussian component count, factor rank 4, and retained Student-t tail match RGPC. The architectures have different parameter counts; the capacity controls in Appendix C address parameter prediction separately. RGPC and PDT use their final-epoch checkpoints.

Conditional generator inputs. CTGAN, TVAE, TabDDPM, and TabDiff generate the 51 padded, rescaled coordinates, with only active coordinates passed to the common population-distribution decoder. Their condition contains the global features, coordinate features, dimension mask, trial coordinates, trial log-seed displacements, log-density ratios, and base masses available to RGPC. Each evaluation context contains 1,024 trial samples from the initial proposal: 128 cached trials and 896 additional draws, with all evaluations counted for each method. During training, CTGAN, TVAE, and TabDiff use one 1,024-trial batch per source task. TabDDPM uses four independently drawn batches per task, choosing one batch for each task represented in an optimizer update. RGPC and PDT acquire new feedback for each training trajectory. The arrays are flattened in a prescribed order, with log-density ratios clipped to $[-60, 60]$, followed by asinh and source-training-only feature standardization. Standardized context entries are clipped to $[-20, 20]$. A context encoder with

hidden width 256 outputs 128 features, followed by layer normalization without learned scale or shift parameters. Identifiers are not network inputs.

CTGAN and TVAE use the official mode-specific continuous-data transformation, fitted within each source-training group. TabDDPM and TabDiff use a source-fitted quantile transformation with normal output and 1,000 quantiles. The transformations are fitted on 32,768 source-posterior rows per group, while all 8,192 samples per task remain available for learning. Training and evaluation use the same source-fitted transformations.

Objectives and sampling. CTGAN concatenates context to its generator and discriminator inputs and retains the Wasserstein adversarial objective, gradient penalty, and native output activations (Xu et al., 2019). TVAE concatenates context to its encoder and decoder inputs and retains reconstruction plus latent KL loss. TVAE sampling includes the decoder observation noise through the native inverse transformation. TabDDPM and TabDiff add a context projection to the native time-conditioning embedding. TabDDPM uses its native Gaussian diffusion loss, linear noise schedule, and 1,000-step reverse sampler (Kotelnikov et al., 2023). TabDiff retains its continuous diffusion objective, learned per-column noise schedules, and stochastic second-order sampler with 50 steps (Shi et al., 2025). Each returns 4,096 samples with uniform output weights; its ordinary mean approximates the same task posterior mean as RGPC. These adaptations preserve the methods' native objectives and samplers while changing their task to conditional posterior sampling.

Training and checkpoint selection. All four generators train for 20,000 optimizer updates, saving candidate checkpoints at 2,000, 5,000, 10,000, and 20,000 updates. For TabDDPM and TabDiff, candidates include both current parameters and exponential-moving-average (EMA) parameters, with decay 0.999 and 0.997, respectively. For each model and seed, we select the numerically eligible candidate with the lowest mean MMD^2 on seven prespecified source tasks, one per source observation pattern. Selection uses independently drawn 1,024-trial feedback batches, 256 generated samples per task, and 512 reference samples from the source-posterior cache.

To qualify for selection, generated coordinates must be finite, and normalization and constraint errors after decoding must be at most 10^{-9}. Before inverse transformation, the root-mean-square magnitude of generated coordinates must be at most three times that of the transformed source samples. For TabDDPM and TabDiff, at most 0.1% of transformed coordinates may exceed six in absolute value. We then check the selected models on 14 prespecified source tasks with both training and independently drawn feedback, using 1,024 outputs and two repeats per context. All 12 selected models pass these checks before main-task generation. Table 6 reports the optimizer settings and selected checkpoints.

Table 6: Conditional-generator architecture and optimizer settings. Listed learning rates are initial values. CTGAN and TVAE use Adam; TabDDPM and TabDiff use AdamW.

Method (year)	Batch	Learning rate	Weight decay	Core network settings
CTGAN (2019)	1,000	2×10^{-4}	10^{-6}	Two 256-wide hidden layers; latent width 128; packing size 10
TVAE (2019)	2,048	10^{-3}	10^{-5}	Two 128-wide hidden layers; latent width 128; reconstruction factor 2
TabDDPM (2023)	2,048	10^{-3}	10^{-5}	Two 256-wide hidden layers; time embedding width 128
TabDiff (2025)	1,024	10^{-3}	0	Two layers; token width 4; one attention head; feedforward factor 32; time embedding width 1,024

CTGAN and TVAE use constant learning rates; TabDDPM and TabDiff linearly decay them to zero over 20,000 updates. Selected update counts for seeds 0, 1, and 2 are CTGAN: (20,000, 20,000, 20,000); TVAE: (2,000, 20,000, 5,000); TabDDPM: (20,000, 20,000, 20,000); and TabDiff: (2,000, 20,000, 2,000). Both diffusion models select EMA parameters for seed 1 and unaveraged parameters for seeds 0 and 2.

B.5 Conventional sampling baselines

All three conventional baselines use 1,024 target density evaluations per task. The initial-proposal IS baseline draws 1,024 samples from the initial Gaussian–Student-t mixture and returns their normalized importance-weighted mean.

WEM uses 128 trial weights to refit the Gaussian component of q_0 (Cappé et al., 2008). It combines the fitted and original Gaussians with equal central weights while retaining the Student-t tail. The resulting central mixture has two components. The fitting settings are at most 80 iterations, convergence tolerance 10^{-7}, relative covariance floor 10^{-4}, and component mass floor 10^{-3}. A further 896 target density evaluations importance-weight the fresh draws.

SMC starts with 512 particles from q_0 and uses adaptive tempering with conditional ESS fraction 0.5 (Del Moral et al., 2006). Temperatures are found by bisection with tolerance 10^{-6} and at most 80 iterations. Each tempering step uses systematic resampling and q_0-independence Metropolis–Hastings moves. The 1,024 evaluations comprise 512 initial evaluations and one full 512-particle mutation round. If necessary, the last full tempering step sets the inverse temperature to one; remaining moves then target the full posterior. The output is the ordinary mean of the final 512 particles, including repeated particles. Forced terminal increments, acceptance rates, and particle lineages are recorded. This SMC configuration differs from the earlier crossed-study implementation.

Each conventional method has 24 sampling repeats per task, formed from three random streams and eight repeats. There are no artificial training-group duplicates for these nonlearned methods. The six learned methods use 1,024 target density evaluations and 4,096 outputs per task, as reported in Table 1. All per-task counts exclude source-posterior generation, model training, and shared initial-proposal construction; they are not total-runtime comparisons.

B.6 Iterative updates in conventional samplers

To give conventional samplers additional adaptation opportunities, we vary the number of mixture refits and the allocation of SMC evaluations between particles and mutation rounds. All settings retain the main comparison's 192 development tasks, three random streams, eight repeats, target posteriors, initial proposals, decoder, and reference samples. Each run uses 1,024 target density evaluations. We select configurations by mean development TVD and report both metrics for the full scan, using the same equal-PUMA aggregation as Table 1.

For iterative mixture fitting, we retain the Student-t component of q_0 and refit four Gaussian components, matching RGPC's component count, by importance-weighted expectation–maximization (EM). We test 1, 2, 4, 8, and 16 refits with either full or diagonal covariances in coordinates whitened by q_0. Each refit uses a new batch from the current sampling distribution and an equal share of the 1,024 evaluations. The first refit initializes centers by weighted k-means++; later refits start from the current mixture. The Student-t component participates in mixture responsibilities, but its parameters and mass remain unchanged. Each refit allows up to 80 EM iterations with convergence tolerance 10^{-7}, a covariance floor of 10^{-4} times the weighted mean marginal variance (at least 10^{-12}), and a component mass floor of 10^{-3}. These EM iterations reuse the current batch's density ratios and require no additional target density evaluations.

We evaluate two outputs from this fitting procedure. For importance sampling, we pool all 1,024 trial samples and normalize their importance weights jointly, using the sampling distribution that generated each sample in its weight denominator. To assess the final fitted distribution under RGPC's output convention, we separately draw 4,096 fresh samples and use their ordinary mean, without further target density evaluations. The one-refit IS output equals the initial-proposal IS baseline because all its trials come from q_0; the refit affects the fresh-sample output. Table 7 reports both outputs separately.

Table 7: Iterative mixture fitting under 1,024 target density evaluations. Means use the same development tasks and aggregation as Table 1. The output groups distinguish pooled importance-weighted trials from fresh, equally weighted samples of the final fitted distribution. RGPC is repeated in the latter group for comparison; lower values are better.

			1,024 weighted trials		4,096 fresh samples	
Covariance	Refits	Trials/refit	MMD^2 $\times 10^{-4}$	TVD $\times 10^{-4}$	MMD^2 $\times 10^{-4}$	TVD $\times 10^{-4}$
Full	1	1,024	486.66	130.07	505.41	144.63
Full	2	512	635.62	152.19	271.40	107.87
Full	4	256	723.89	163.62	281.47	109.70
Full	8	128	770.98	170.28	292.17	111.49
Full	16	64	795.61	173.16	301.01	112.95
Diagonal	1	1,024	486.66	130.07	590.64	157.08
Diagonal	2	512	635.62	152.19	297.98	112.38
Diagonal	4	256	723.89	163.62	301.23	112.78
Diagonal	8	128	770.98	170.28	306.68	113.65
Diagonal	16	64	795.61	173.16	310.85	114.34
RGPC (ours), 16 updates				–	**44.13**	**49.64**

For SMC, we test all powers of two from 2 to 512 particles. After evaluating the initial particles, all remaining evaluations fund complete particle-mutation rounds. The adaptive-tempering, resampling, and q_0-independence Metropolis–Hastings rules remain unchanged from Appendix B.5. Each setting returns its final particles with uniform weights. Table 8 reports the allocations, metric means, acceptance rates, and the fraction of runs requiring a forced final temperature increment.

Table 8: SMC particle–round allocation under 1,024 target density evaluations. Metric means use the same development tasks and aggregation as Table 1. Acceptance is averaged across runs; forced increments report the percentage of runs requiring the final temperature to be set to one. Each SMC setting returns the stated number of particles; RGPC returns 4,096 fresh samples.

Particles	Mutation rounds	MMD^2 $\times 10^{-4}$	TVD $\times 10^{-4}$	Acceptance (%)	Forced increments (%)
2	511	1985.63	292.27	8.51	0.00
4	255	999.83	207.16	9.57	0.00
8	127	529.70	149.32	10.77	0.00
16	63	368.60	118.98	11.70	0.00
32	31	422.19	118.66	12.13	0.00
64	15	560.92	135.90	12.97	0.00
128	7	641.84	148.38	16.10	0.00
256	3	573.66	141.48	25.79	26.76
512	1	584.74	144.18	7.45	100.00
RGPC (ours)		**44.13**	**49.64**	–	–

Among the tested mixture settings, full covariance with two refits gives the lowest TVD for the ordinary-mean output: $\mathrm{MMD}^2 = 0.0271$ and $\mathrm{TVD} = 0.0108$. Mean TVD selects 32 particles and 31 mutation rounds for SMC, yielding $\mathrm{MMD}^2 = 0.0422$ and $\mathrm{TVD} = 0.0119$; this allocation reaches the target temperature without a forced final increment in every run. SMC's lowest MMD^2 instead occurs at 16 particles, so the two metrics do not select the same allocation. RGPC's means of 0.0044 and 0.0050, respectively, are lower than every scanned setting for both mixture outputs and SMC. Thus, iterative updates in conventional samplers do not close the approximation gap on these development tasks. The full grids describe sensitivity to iteration allocation; the selected configurations and their reported means use the same development tasks.

B.7 VALIDATION ON NEW PUMAS

To test the selected configurations on new regions, we evaluate 192 tasks from 32 PUMAs excluded from sampler training and configuration selection. The tasks combine PUMAs from 21 source states with the six observation patterns used in development. We first exclude PUMAs appearing in archived training, development, and evaluation inventories. From the remaining candidates, we select one PUMA per eligible source state by a deterministic hash of its identifier, then fill the remaining slots in hash order with at most two PUMAs per state. The task inventory is frozen before inference, and the population prior is reused.

RGPC and PDT use the same trained checkpoints as in the development comparison, with 16 updates for RGPC and one prediction for PDT. The conventional configurations are selected by development TVD as described in Appendix B.6. Weighted EM uses four full-covariance Gaussian components and two refits with 512 trial evaluations each; SMC uses 32 particles and 31 mutation rounds. All four methods use 1,024 target density evaluations per task. RGPC, PDT, and weighted EM each return 4,096 fresh samples, and SMC returns its final 32 particles; all outputs are equally weighted.

For each task, we use four HMC chains with the longer-chain settings in Appendix B.3, taking the last 1,024 draws from each chain to form a shared 4,096-sample reference. All 192 references pass the registered checks: the largest split $\widehat{R}$ is 1.00086 and the smallest bulk ESS is 12,147.3. The Gaussian-kernel bandwidth is determined from the reference samples and shared by all methods. We average eight sampling repeats and three seeds within each task, six observation patterns within each PUMA, and then PUMAs equally.

Table 9: Posterior approximation on 32 new PUMAs. All methods use 1,024 target density evaluations per task and equally weighted outputs. The upper block gives metric means; the lower block gives RGPC-minus-comparator differences with family-wise 95% intervals across all six comparisons. Metrics, differences, and interval bounds are in units of 10^{-4}.

Method	Samples	MMD^2	TVD
PDT	4,096	67.90	62.51
Weighted EM (2 refits)	4,096	263.88	102.40
SMC (32 particles, 31 rounds)	32	413.12	111.83
RGPC (ours)	4,096	**34.27**	**43.28**

Comparator	$\Delta\mathrm{MMD}^2$ [interval]	$\Delta\mathrm{TVD}$ [interval]
PDT	-33.63 $[-42.50, -24.97]$	-19.23 $[-23.99, -14.39]$
Weighted EM (2 refits)	-229.61 $[-235.64, -223.55]$	-59.12 $[-63.12, -54.73]$
SMC (32 particles, 31 rounds)	-378.85 $[-412.05, -346.34]$	-68.55 $[-74.35, -62.58]$

Table 9 reports the means and paired differences. Relative to PDT, RGPC reduces mean MMD^2 by 49.53% and TVD by 30.76%; both means are also lower than those of the selected conventional configurations. For uncertainty, we use 10,000 paired PUMA-bootstrap draws with Bonferroni adjustment across the three comparators and two metrics, conditional on the trained models and common references. All six family-wise 95% intervals lie below zero. The approximation advantage therefore extends to new PUMAs after configuration selection.

B.8 SENSITIVITY TO TARGET DENSITY EVALUATIONS

To assess sensitivity to target density evaluations, we compare RGPC, PDT, initial-proposal IS, WEM with IS, and SMC at 512, 1,024, and 2,048 evaluations. We retain the same 192 development tasks, initial proposals, three seeds, eight sampling repeats, and 512-sample posterior references as in the main comparison. The kernel bandwidths and equal-PUMA aggregation also remain unchanged. The four conditional generators use fixed-length contexts containing 1,024 trial samples and are evaluated only at that count in the main comparison.

RGPC and PDT reuse their models trained with 1,024 target density evaluations, without retraining or checkpoint selection at the other counts. RGPC retains 16 updates, allocating 32, 64, or 128 trial

samples to each update. PDT receives all trial samples from the initial proposal in one pass. Both methods return 4,096 equally weighted samples at every setting.

For the conventional baselines, we preserve the allocation rules in Appendix B.5. Initial-proposal IS uses all evaluations for its weighted output. WEM uses 128 evaluations for one mixture fit and the remaining evaluations for its importance-weighted output. SMC uses half the evaluations for initial particles and half for one complete mutation round, returning the final particles with uniform weights. Thus, the conventional output counts vary with the available evaluations. These rules are separate from the allocations for iterative updates examined in Appendix B.6.

Table 10: Posterior approximation at three target density evaluation counts. Values are descriptive means on the same development tasks as Table 1, with both metrics reported in units of 10^{-4}. Learned weights remain unchanged across counts; conventional methods retain their original allocation rules. Lower values are better.

	Target density evaluations					
	512		1,024		2,048	
Method	MMD^2	TVD	MMD^2	TVD	MMD^2	TVD
PDT	76.96	67.53	76.96	67.53	76.95	67.53
Initial proposal + IS	592.96	145.70	486.66	130.07	419.89	118.89
WEM + IS	764.65	168.28	602.14	146.86	479.16	128.37
SMC	699.56	159.74	584.74	144.18	482.60	129.24
RGPC (ours)	**45.53**	**50.58**	**44.13**	**49.64**	**42.87**	**48.82**

Table 10 reports descriptive means for all five methods at each count. RGPC has lower mean MMD^2 and TVD than each baseline at all three counts. Increasing evaluations from 512 to 2,048 reduces RGPC's mean MMD^2 by 5.85% and TVD by 3.49%, with the learned weights and output sample count unchanged.

B.9 Metrics, aggregation, and uncertainty

To compare each method's actual inference output, let $\boldsymbol{p}_i$ be its N returned population distributions and w_i^{out} their normalized weights. Weights are uniform for RGPC, PDT, the four generators, and final SMC particles. For q_0 + IS and WEM with IS, they are the original normalized importance weights. Thus $\widehat{\boldsymbol{\mu}} = \sum_i w_i^{\mathrm{out}} \boldsymbol{p}_i$, and posterior-mean estimation error (TVD) is $\frac{1}{2}\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}^{\mathrm{ref}}\|_1$. The reference is the specified posterior mean approximated by HMC, not an empirical population distribution estimated from PUMS.

For distribution mismatch, use all 96 decoded cell probabilities without method-specific scaling. Let N_{ref} denote the number of reference samples $\boldsymbol{p}_a^{\mathrm{ref}}$, and define

$$k(\boldsymbol{p}, \boldsymbol{p}') = \exp\left(-\frac{\|\boldsymbol{p} - \boldsymbol{p}'\|_2^2}{2h^2}\right).$$

We use $N_{\mathrm{ref}} = 512$ for the main comparison and $N_{\mathrm{ref}} = 4{,}096$ for the new-PUMA validation (Appendix B.7). For these comparisons, h^2 is the median squared distance over distinct pairs of each task's reference samples and is shared by all methods. The weighted empirical MMD^2 is

$$\begin{aligned}\widehat{\mathrm{MMD}}^2 = &\sum_{i,j=1}^{N} w_i^{\mathrm{out}} w_j^{\mathrm{out}} k(\boldsymbol{p}_i, \boldsymbol{p}_j) \\ &+ \frac{1}{N_{\mathrm{ref}}^2} \sum_{a,b=1}^{N_{\mathrm{ref}}} k(\boldsymbol{p}_a^{\mathrm{ref}}, \boldsymbol{p}_b^{\mathrm{ref}}) \\ &- \frac{2}{N_{\mathrm{ref}}} \sum_{i=1}^{N} \sum_{a=1}^{N_{\mathrm{ref}}} w_i^{\mathrm{out}} k(\boldsymbol{p}_i, \boldsymbol{p}_a^{\mathrm{ref}}).\end{aligned} \tag{20}$$

All diagonal terms are included (Gretton et al., 2012). This is the distance between the finite weighted output and reference measures, so sampling error and weight concentration contribute to the

score. The kernel calculation uses every output sample, including repeated SMC particles, without subsampling or resampling.

Both metrics are computed separately for each sampling repeat. We average repeats and seeds within each task, six observation patterns within each PUMA, and then the 32 PUMAs equally. Each method contributes 4,608 records from 192 tasks, three seeds, and eight repeats, totaling 41,472 records across the nine methods. The largest support error across scored outputs is 2.28×10^{-11}.

The 16 RGPC-minus-baseline comparisons across both metrics use 10,000 paired PUMA-bootstrap draws, conditional on the selected configuration, trained models, and common reference arrays. Bonferroni adjustment gives family-wise 95% intervals from 99.6875% marginal intervals. Table 11 reports the eight MMD^2 comparisons from this jointly adjusted family. The posterior-mean intervals in the supplementary rank and IS study and the intervals for retrained mechanism controls use their own protocols, stated with those experiments.

Update-trajectory visualization. Figure 5 uses one preselected task, training seed, and sampling repeat of the trained 16-update model. Each update uses 64 target density evaluations. Density maps use the first two principal components of the 96-cell reference samples, accounting for 30.25% of reference variance; axes are measured in the corresponding reference standard deviations. The maps share a density-estimation bandwidth and one symmetric-log color scale, linear within $[-0.003, 0.003]$. Both metrics use all 96 cell probabilities and 4,096 samples at each state. The reference is used only for projection and evaluation.

Table 11: Eight paired distribution-mismatch comparisons on the main 192 development tasks. Differences are RGPC minus comparator in MMD^2. Intervals use 10,000 paired PUMA-bootstrap draws with Bonferroni family-wise 95% coverage across all 16 comparisons over both metrics, conditional on the selected configuration, trained models, and shared references. Differences and bounds are rounded to four decimal places.

Comparator	$\Delta\mathrm{MMD}^2$	Family-wise 95% interval
CTGAN	-0.0397	$[-0.0434, -0.0364]$
TVAE	-0.0706	$[-0.0788, -0.0636]$
TabDDPM	-0.0062	$[-0.0078, -0.0048]$
TabDiff	-0.0083	$[-0.0103, -0.0066]$
PDT	-0.0033	$[-0.0042, -0.0025]$
Initial proposal + IS	-0.0443	$[-0.0488, -0.0401]$
WEM + IS	-0.0558	$[-0.0606, -0.0502]$
SMC	-0.0541	$[-0.0584, -0.0499]$

B.10 Sensitivity to reference samples and kernel bandwidth

To test whether RGPC's advantage over PDT depends on the posterior reference or kernel bandwidth, we reevaluate both methods on the same 192 development tasks. Each evaluation setting uses the same 4,096 output samples per task, seed, and sampling repeat, with three training seeds and eight repeats per seed. The two reference sets are the original 512 samples and the 4,096 samples from the longer HMC chains described in Appendix B.3. Using h^2 from the original reference, we set the squared kernel bandwidth to $0.5h^2$, h^2, and $2h^2$ under each reference set.

Table 12 reports descriptive means using the same equal-PUMA aggregation as the main comparison. RGPC has lower mean MMD^2 in all six settings, with relative reductions of 37.16–47.72% compared with PDT. The mean TVD is 26.50% lower with the original reference and 29.02% lower with the longer-chain reference.

Table 12: Sensitivity to reference samples and kernel bandwidth. Values are means on the 192 development tasks, in units of 10^{-4}. The MMD^2 columns use multiples of the original squared bandwidth h^2. TVD is reported once per reference because it does not depend on the kernel.

		MMD^2			
Reference (samples)	Method	$0.5h^2$	h^2	$2h^2$	TVD
Original (512)	PDT	97.72	76.96	51.76	67.53
	RGPC (ours)	**61.40**	**44.13**	**28.88**	**49.64**
Longer chains (4,096)	PDT	86.99	70.32	48.02	65.24
	RGPC (ours)	**50.65**	**37.47**	**25.10**	**46.30**

B.11 Recurrent mechanism controls

The single-input controls in section 4.2 intervene only on the direct inputs used at location and spread prediction. For the coordinate-feature control, averages use only the 224 training tasks. Each coordinate's features are averaged over tasks where it is active, before encoding. These features include both constraint directions and initial-proposal rescaling (Appendix A). The density-ratio feedback and moment differences remain task-specific. For the feedback-summary control, only the moment differences supplied to the moment encoder are set to zero; the coordinate features remain task-specific. Both retain the original task and trial-sample inputs to attention, current-proposal features, aggregate constraints, and target posterior. The architecture, parameter count, initialization, source-posterior supervision, minibatch order, and optimization settings match full RGPC. Each control is retrained for three seeds with 16 updates, 64 trial evaluations per update, and 4,096 fresh output samples.

The feedback-pairing control instead uses full RGPC's trained networks without retraining. It independently permutes density-ratio feedback among each task's trial locations at updates 2–16, preserving the first update and the set of feedback values at each later update. This retains the available feedback values while changing which trial locations they describe.

We also retain the earlier joint control, which applies both the training-average coordinate features and zero moment differences at parameter prediction. Its mean MMD^2 is 0.0084 and TVD is 0.0071, increases of 89.33% and 43.70% relative to full RGPC. The single-input controls separate the two changes combined in this control.

For the three controls in Figure 4, we average repeats and observation patterns within each PUMA, then average seeds before drawing 10,000 paired PUMA-bootstrap samples. The same resampled PUMAs are used for each control and full RGPC. We compute percentage increases as $100(\bar{e}_{\mathrm{control}}/\bar{e}_{\mathrm{RGPC}} - 1)$, with $\bar{e}$ the mean of either metric across PUMAs. Bonferroni adjustment across the three controls and two metrics uses quantiles $0.05/(2 \times 6)$ and $1 - 0.05/(2 \times 6)$ for family-wise 95% intervals. These six-comparison intervals are recomputed jointly from the verified results, conditional on the trained models and shared references. The joint-removal result above is reported descriptively, outside this comparison family.

Constraint directions and proposal rescaling. To distinguish the two sources of coordinate information, we separately replace the constraint-direction matrix $\mathbf{B}_{r,o}$ and initial-proposal Cholesky factor $\mathbf{L}_{r,o}$ in the direct input to the coordinate encoder. The direction control replaces $\mathbf{B}_{r,o}$ by a training-task average while retaining the current $\mathbf{L}_{r,o}$. The rescaling control replaces $\mathbf{L}_{r,o}$ while retaining the current $\mathbf{B}_{r,o}$. Each replacement recomputes $\mathbf{B}_{r,o}\mathbf{L}_{r,o}$, its column norms, and any affected diagonal-scale features. Normalized coordinate indices, fiber dimensions, and inactive padding are unchanged. Averages use only the 224 training tasks where the relevant coordinate or coordinate pair is active.

Only this direct encoder input changes. The actual initial proposal, coordinate transformations, decoder, aggregate constraints, target posterior, and other input routes are retained. Both controls use the same three seed-matched initializations and 20-epoch training protocol as full RGPC, with 16 updates, 1,024 target density evaluations, and 4,096 output samples. Their later proposals and trial samples can differ as a consequence of the interventions.

Table 13 reports both controls. Replacing constraint directions increases both metrics in every seed, with family-wise 95% intervals above zero. Replacing initial-proposal rescaling changes mean MMD^2 by -0.67% and TVD by $+0.20\%$, with both intervals spanning zero. Thus, the comparisons support direct access to current constraint directions but do not resolve an additional benefit from task-specific rescaling at this input. They do not remove the initial proposal from sampling or the model's other inputs. We use 10,000 paired PUMA-bootstrap draws after averaging seeds, with Bonferroni adjustment across these four factor–metric comparisons. This is a separate comparison family from the six comparisons in Figure 4; all intervals remain conditional on the trained models and shared references.

Table 13: Separating constraint directions from initial-proposal rescaling at parameter prediction. The upper block gives metric means in units of 10^{-4}. The lower block gives percentage changes relative to full RGPC, with family-wise 95% intervals across the four comparisons. Positive changes indicate higher error. All settings use 1,024 target density evaluations and 4,096 equally weighted outputs.

Setting	MMD^2	TVD
Training-average constraint directions	63.26	60.91
Training-average initial-proposal rescaling	43.84	49.74
RGPC (ours)	**44.13**	**49.64**

Replaced factor	$\Delta\mathrm{MMD}^2$ (%) [interval]	$\Delta\mathrm{TVD}$ (%) [interval]
Constraint directions	$43.33\ [32.76, 54.41]$	$22.71\ [17.25, 28.41]$
Initial-proposal rescaling	$-0.67\ [-2.75, 1.31]$	$0.20\ [-0.80, 1.19]$

B.12 Information boundary and reproducibility

The learned inputs contain neither PUMA/state identifiers nor individual microdata records. At each update, RGPC compares the task's specified target density with the current proposal density at samples drawn from that proposal to obtain density-ratio feedback. Evaluation PUMAs are excluded from sampler training and preprocessing; their contribution to the earlier population-prior fit is specified in Appendix B.2. Evaluation reference arrays are used for scoring and update-count selection (Appendix B.3). They are not used to fit task-specific updates or select training epochs.

Task inventories, training settings, checkpoint identities, random seeds, and prediction artifacts are recorded before their corresponding scores are produced. For each method, both metrics are computed from the same decoded outputs and shared posterior references. Numerical outputs are reproduced with the original random streams and density implementation. Reference hashes, output weights, sample counts, and task, seed, and repeat coverage are checked before aggregation. The input controls use retrained models; feedback shuffling reuses full RGPC. Their results are reported separately from the main benchmark.

C Supplementary Single-Update Parameter-Prediction Controls

These earlier single-update controls complement the recurrent comparisons in section 4.2 by testing how current-task information enters distribution-parameter prediction. They use the training groups, 192 evaluation tasks, and cached references in Appendix B. All seven settings use factor rank 4, four Gaussian components, the same retained tail, 128 trial evaluations, and ordinary means of 4,096 fresh samples. Each setting is retrained under two training groups and three seeds, giving 42 models, with eight sampling repeats per task.

C.1 Controlled changes to parameter prediction

RGPC component-summary control. To remove coordinate-specific access at the final prediction step, we retain the coordinate encoders, moment differences, shared pair decoder, and representation subtraction. We average the coordinate-representation differences $\Delta \boldsymbol{u}_{kj}$ over active coordinates before vector heads predict all location, log-scale, and covariance-factor entries for each component.

Mixture-logit prediction is unchanged. Thus the control still processes the current-task features and density-ratio feedback, but individual parameters receive a component summary instead of their corresponding coordinate-representation differences.

RGPC capacity control. The component-summary control has 508,256 parameters, compared with 469,856 for the single-update RGPC reference. To compare the two prediction rules at similar capacity, we widen only the coordinate-conditioned pair decoder's hidden layer from 256 to 405. This width follows the parameter-count difference without a width search. The resulting model has 508,149 parameters, 107 fewer than the component-summary control, and retains coordinate-specific prediction.

PDT prediction rules. The linear-head control retains PDT's component-level vector prediction. The nonlinear-head control adds a 32-wide GELU hidden layer before the location, log-scale, and covariance-factor vector heads, while retaining the mixture-logit path. For coordinate-conditioned PDT, we retain the original Transformer decoder and its context inputs. Each component representation is combined with the current coordinate encoding and the encoded moment differences, then passed through a shared 64–128–64 pair decoder. A 2–32–64 network encodes the moment differences. The location and spread heads act on each component–coordinate pair. This variant uses a single pass with feedback and retains PDT's mixture-logit prediction; RGPC's representation subtraction is not added.

Source-average input control. To test whether the coordinate-conditioned PDT uses the current task's inputs at prediction, we replace only its prediction-stage coordinate features by source-training averages and its moment differences by zero. Each coordinate's average uses only training tasks in which that coordinate is active and is formed before encoding. The Transformer decoder continues to receive the actual task features and density-ratio feedback. This control has the same topology and parameter count as coordinate-conditioned PDT.

C.2 Training, pairing, and statistical comparisons

Within each family, unchanged layers share their initial tensors, minibatch order, posterior-sample blocks, and computing device. Changed heads use the prescribed initialization for their shapes. Coordinate-conditioned PDT and its source-average input control start from identical parameters. All settings use 20 epochs of AdamW with learning rate 3×10^{-4}, zero weight decay, task batch size 8, and gradient clipping norm 10. Training cycles through the same eight 1,024-sample posterior blocks per task. Final-epoch models and task predictions are saved before reference arrays are opened for scoring.

For uncertainty, we average sampling repeats and training groups within each task and seed, then observation patterns within each state. The mean gives equal weight to states and seeds. We draw 10,000 paired crossed state/seed bootstrap samples. The five planned comparisons in Table 14 use Bonferroni-adjusted family-wise 95% intervals, corresponding to 99% marginal intervals. Each difference subtracts control error from error with current-coordinate inputs. All five upper endpoints are below zero, and the differences favor current-coordinate inputs in each seed mean and each training-group mean.

Table 14: Five planned comparisons of supplementary single-update parameter-prediction controls. Each row subtracts control error from error with current-coordinate inputs. Posterior means are estimated by ordinary sample averaging. The family-wise 95% intervals use 10,000 paired crossed state/seed bootstrap draws and 99% marginal intervals.

Comparison	ΔTVD $\times 10^{-4}$	Family-wise 95% interval $\times 10^{-4}$
PDT: coordinate inputs minus linear component-summary head	-12.43	$[-16.73, -8.49]$
PDT: coordinate inputs minus nonlinear component-summary head	-12.94	$[-16.90, -9.21]$
PDT: current-coordinate inputs minus source-average inputs	-14.30	$[-18.49, -10.36]$
RGPC: coordinate inputs minus component summary	-13.46	$[-19.22, -8.88]$
RGPC: capacity-matched coordinate inputs minus component summary	-14.17	$[-18.47, -9.98]$

Table 15: Posterior-mean estimation error under supplementary single-update parameter-prediction controls. These models use 128 trial evaluations and are retrained separately from the original single-update benchmark. Coordinate inputs contain the current-task coordinate features and moment differences defined in section 3.2.

Model	Parameter prediction	Trainable parameters	Posterior-mean error (TVD) $\times 10^{-4}$
PDT	Component summary, linear head	2,004,531	69.03
PDT	Component summary, nonlinear head	1,996,819	69.54
PDT	Coordinate features and moment differences	2,003,815	56.60
PDT	Source-average features, zero moment differences	2,003,815	70.90
RGPC	Component summary only	508,256	70.23
RGPC	Coordinate inputs, matched capacity	508,149	56.07
RGPC (ours)	Coordinate features and moment differences	469,856	56.77

The complete model means and parameter counts are reported in Table 15. These comparisons evaluate coordinate features, moment differences, and a shared coordinate prediction rule together. The retrained RGPC reference has error 56.77×10^{-4}; the corresponding single-update benchmark uses its original 57.81×10^{-4} result and sampling streams.

C.3 Separate representation-subtraction control

To test the subtraction in equation 16, a separate paired experiment removes subtraction of the zero-feedback representations while retaining the pass with feedback, centered ratios, moment differences, output bounds, covariance family, masks, and tail. Both versions start from identical network parameters and share the source minibatches and posterior blocks. They use the same training and evaluation task inventory, two groups, three seeds, and eight sampling repeats. For ordinary sample means, we compare posterior-mean estimation error (TVD) using 5,000 paired crossed state/seed bootstrap draws and a 95% interval.

With subtraction, mean TVD is 0.14×10^{-4} higher than without subtraction (95% interval: $[-2.56, 3.09] \times 10^{-4}$). The interval spans zero. The zero-feedback identity follows from the construction in Appendix A, independently of this accuracy comparison.

D SUPPLEMENTARY PROPOSAL COMPARISONS

These comparisons examine optional importance correction, covariance capacity, combinations of proposals, and a separate crossed-jurisdiction setting. Their models, samples, and reference protocols are identified below so that each result can be interpreted within its own comparison.

D.1 ORDINARY MEANS, IMPORTANCE CORRECTION, AND COVARIANCE CAPACITY

The ordinary mean is the main estimator defined in equation 4. Table 16 compares ranks 4 and 51 using earlier single-update RGPC/PDT checkpoints, separate from the main 16-update models. Both ranks use the same 192 source-development tasks, two training groups, three seeds, and eight sampling repeats. Each ordinary estimate uses 128 trial target density evaluations and 4,096 fresh samples. The intervals come from the original 5,000 paired crossed state/seed bootstrap draws.

Table 16: Supplementary single-update covariance-capacity comparison on 192 source-development tasks. These are earlier checkpoints, separate from the 16-update RGPC in Table 1. Each estimate uses 128 target density evaluations and 4,096 fresh samples, with the common HMC mean as reference. Intervals use the original 5,000 paired state/seed bootstrap draws; Δ is RGPC minus PDT.

Factor rank	PDT TVD $\times 10^{-4}$	**RGPC (ours)** TVD $\times 10^{-4}$	Reduction	95% interval for ΔTVD $\times 10^{-4}$
4	69.06	57.81	16.29%	$[-16.62, -6.04]$
51	69.67	55.31	20.62%	$[-17.73, -11.09]$

RGPC reduces posterior-mean estimation error relative to PDT by 16.29% at rank 4 and 20.62% at rank 51. Both paired intervals lie below zero, with favorable differences in every seed, training group, and evaluation pattern. Within RGPC, rank 51 has 4.33% lower error than rank 4, but its paired difference interval, $[-6.63, 0.55] \times 10^{-4}$, crosses zero. Rank 4 remains the main configuration.

To examine optional importance correction, we evaluate the target density at the same 4,096 fresh samples and apply equation 18. This adds 4,096 target density evaluations, for a total of 4,224 per task. Table 17 reports the resulting IS estimates with the original paired state/seed intervals. The earlier study designated TVD from ordinary sample means as its primary endpoint and these same-draw IS estimates as a control. Both methods have higher mean TVD after weighting in the recorded samples; that within-method comparison is descriptive.

Table 17: Supplementary single-update source-domain comparison with importance correction on 192 tasks from 32 PUMAs excluded from sampler training within source states and six observation patterns represented in training. Each method uses 4,224 target density evaluations, including 128 trial evaluations, and 4,096 fresh draws. TVD compares the IS mean with the common HMC reference mean. Rows compare matched covariance-factor ranks; intervals use the original 5,000 paired state/seed bootstrap draws. Δ is RGPC minus PDT.

Factor rank	PDT + IS TVD $\times 10^{-4}$	**RGPC + IS (ours)** TVD $\times 10^{-4}$	Reduction	95% interval for ΔTVD $\times 10^{-4}$
4	90.98	87.40	3.94%	$[-6.25, -1.11]$
51	91.19	88.00	3.50%	$[-5.33, -1.03]$

D.2 CROSSED-JURISDICTION AND OBSERVATION-PATTERN TRANSFER

This separate study assigns eight distinct PUMAs to each of eight observation patterns in each of three geographic folds, giving 192 PUMAs across 32 states and D.C. Each task's jurisdiction is excluded from population-prior fitting and sampler training, and its observation pattern is excluded from sampler training. The earlier source assets contain 1,504 PUMA–pattern tasks over 47 source jurisdictions and three outer folds, split into 752 training and 752 validation tasks with disjoint

PUMAs within each state–fold pair. The models use 1,024 posterior samples per training task from two HMC chains, and factor rank 4. These checkpoints and evaluation references differ from the main benchmark's.

References and estimators. Each evaluation reference uses four HMC chains, 2,048 burn-in transitions, and 4,096 retained draws per chain. Qualification requires split $\widehat{R} \leq 1.01$, bulk ESS at least 400, constraint error at most 10^{-10}, and no probability below -10^{-12}. All 192 references pass: the maximum split $\widehat{R}$ is 1.00117, the minimum bulk ESS is 9,220.6, and the maximum constraint error is 1.56×10^{-12}. Active fiber dimensions range from 33 to 51. The second half of each retained chain supplies the reference mean and posterior summaries. The exact-HMC dynamics produce a finite-sample numerical reference.

The IS comparisons use 4,224 target density evaluations. The earlier adaptive-tempered SMC comparison uses a variable number of evaluations, with median 4,096, as listed in Table 18. It is separate from the SMC configuration using 1,024 target density evaluations in the main benchmark. For the twelve B23006 statistics not supplied as constraints, mean and 90% interval-endpoint errors compare the estimated posterior summaries with their HMC counterparts.

Aggregation and findings. For each task and IS method, ESS is $1/\sum_n w_n^2$, averaged over sampling repeats and the three model seeds where applicable. ESS ratios are formed from these task averages, summarized by the median within each jurisdiction, and then by the median across jurisdictions. Their bootstrap resamples jurisdiction summaries. Absolute errors and paired differences instead use means within and across jurisdictions; the paired posterior-error intervals use 5,000 jurisdiction-bootstrap draws. Thus mean ESS and median ESS ratios are different summaries.

Table 18: Supplementary crossed study on 192 distinct PUMAs, with their jurisdictions and observation patterns excluded from sampler training. This study uses earlier checkpoints and source supervision, not the main comparison's models. ESS and errors give equal weight to each state and D.C. Target density evaluation counts include adaptation and weighting but exclude shared initial-proposal optimization. HMC supplies the reference means. Bold marks the best point estimates among the IS methods, not statistical significance.

Method	Target density evaluations	ESS	Posterior-mean error (TVD) $\times 10^{-4}$	Withheld means MAE to HMC $\times 10^{-4}$
Initial proposal + IS	4,224	34.58	106.64	6.81
WEM + IS	4,224	28.72	106.85	6.63
PDT + IS	4,224	39.84	90.78	5.63
SMC	4,096 median	–	111.58	6.66
RGPC + IS (ours)	4,224	**42.30**	**89.88**	**5.60**

RGPC has the highest mean ESS among these IS methods. Its median jurisdiction-level ESS ratio is 1.159 relative to PDT (95% interval $[1.034, 1.236]$) and 1.739 relative to WEM (interval $[1.540, 1.826]$). RGPC has higher ESS than WEM on 160 of 192 tasks. The gain over WEM is positive across all three seeds, three folds, two pattern orientations, and eight patterns. The equal-jurisdiction mean ESS difference relative to PDT is 2.46, with paired interval $[0.85, 4.14]$.

Posterior-mean estimation error is lower than for q_0+IS and WEM with IS, with paired intervals below zero. The RGPC-minus-WEM difference is -16.96×10^{-4}, with interval $[-23.66, -9.09] \times 10^{-4}$. Against PDT, the difference is -0.90×10^{-4}, with interval $[-4.13, 2.42] \times 10^{-4}$. This crossed setting supports reduced weight concentration but does not establish a posterior-mean accuracy advantage over PDT. The withheld-statistic errors have the same comparison pattern (Table 19).

An earlier evaluation of these crossed-study checkpoints on source-validation PUMAs under the omitted patterns gave a median jurisdiction-level ESS ratio of 1.452 relative to WEM, with interval $[1.382, 1.651]$. Every pattern had a median ratio above one, and the forward log-score gain over q_0 was positive. This source-validation result and the crossed evaluation use different PUMA inventories.

Table 19: Supplementary crossed-study posterior errors at 4,224 target density evaluations. Entries are equal-jurisdiction mean differences between RGPC and comparator, followed by 95% intervals from 5,000 jurisdiction-bootstrap draws. Negative values favor RGPC. All three errors use HMC references. These are not comparisons on the main source-domain PUMA inventory.

Comparator	Posterior-mean TVD Δ [95% interval] $\times 10^{-4}$	Withheld mean MAE Δ [95% interval] $\times 10^{-4}$	Interval-endpoint MAE Δ [95% interval] $\times 10^{-4}$
Initial proposal + IS	-16.76 $[-22.32, -11.43]$	-1.21 $[-1.70, -0.75]$	-1.35 $[-1.95, -0.64]$
WEM + IS	-16.96 $[-23.66, -9.09]$	-1.03 $[-1.57, -0.46]$	-1.78 $[-2.51, -0.85]$
PDT + IS	-0.90 $[-4.13, 2.42]$	-0.03 $[-0.28, 0.24]$	-0.22 $[-0.49, 0.13]$

D.3 Optional mixtures of complete proposals

An evaluable proposal can be combined with others for the same target posterior through deterministic-mixture importance sampling (Elvira et al., 2019). Using earlier single-update checkpoints, we examine an equal-weight mixture of RGPC and PDT at factor ranks 4 and 51. Each of the four complete proposals supplies 1,024 fresh samples, and every sample is evaluated under all four densities. Their arithmetic mean is the importance-weight denominator. All members share 128 trial evaluations, giving 4,224 target density evaluations in total, excluding initial-proposal construction. The mixture weights are set before evaluation; there is no jointly trained combination network.

This comparison uses 64 development tasks from 32 source states and two observation patterns, each omitted from the corresponding models' training. It includes three training seeds and eight sampling repeats. Controls comprise each individual RGPC/PDT proposal and the equal mixture of the two rank-4 proposals. Intervals use 5,000 bootstrap draws over jurisdictions and training seeds while preserving paired comparisons.

Table 20: All posterior-mean comparisons for the prespecified four-proposal mixture on 64 development tasks under two observation patterns excluded from training. The mixture combines earlier single-update RGPC and PDT proposals at covariance-factor ranks 4 and 51 and has IS posterior-mean TVD of 81.53×10^{-4} against HMC. All methods use 4,224 target density evaluations. Intervals use 5,000 bootstrap draws over jurisdictions and model seeds while preserving paired comparisons. Here, Δ is four-proposal mixture minus comparator; reductions are relative to the comparator.

Comparator	Factor rank	Comparator TVD $\times 10^{-4}$	Reduction	95% interval for ΔTVD $\times 10^{-4}$
PDT	4	87.10	6.39%	$[-10.78, -0.20]$
PDT	51	86.26	5.49%	$[-8.89, -0.22]$
RGPC–PDT equal mixture	4	82.60	1.29%	$[-6.63, 4.27]$
RGPC (ours)	4	83.91	2.84%	$[-8.94, 3.96]$
RGPC (ours)	51	83.61	2.49%	$[-7.02, 2.58]$

The four-member mixture lowers posterior-mean estimation error by 6.39% and 5.49% relative to the two PDT configurations (Table 20). Both comparisons favor the mixture across seeds and patterns, with paired intervals below zero. Comparisons with either RGPC alone and with the two-member mixture have intervals crossing zero. The study supports this combined estimator's improvement over the listed PDT controls; it does not isolate RGPC-specific complementarity because a same-size PDT-only ensemble is absent.

D.4 Observation-representation checks

To check sensitivity to the observed aggregates separately from sampling accuracy, we examined deterministic task representations. Across 14,024 public PUMA–pattern instances, reversing, scaling,

invertibly mixing, or duplicating constraint rows preserved the canonical basis and projector at the reported precision. Adding an observed block gave a median TVD of 0.0320 between the common and task completions. The resulting task completion remained a median TVD of 0.0447 from the completion using all B23006 cells. These distances compare deterministic feasible population distributions; posterior-mean estimation error is measured against HMC as specified in Appendix B.9.

E Extended Related Work

GenSynthPop (de Mooij et al., 2024) constructs synthetic populations from spatial and demographic aggregates. When microdata are available, Sané et al. (2025) learn joint demographic distributions using variational autoencoders. For Bayesian inference from aggregates, Frogner & Poggio (2019) combine a prior distribution with observed marginals to compute a posterior mode. RGPC instead uses a learned sampler for posterior-mean estimation under a population prior conditioned on observed aggregates.

Conditional posterior samplers learn reusable distributions for new observations. Inference compilation (Le et al., 2017) learns execution-trace proposals for importance sampling in probabilistic programs. Simformer (Gloeckler et al., 2024) learns a diffusion model of the joint parameter–data distribution, supporting different conditioning subsets with one trained network. This simulation-based approach does not require likelihood evaluations. RGPC uses an evaluable unnormalized target density to obtain density-ratio feedback. For the broader generator comparison, CTGAN and TVAE model tabular data distributions (Xu et al., 2019), and TabDDPM and TabDiff use diffusion models for tabular generation (Kotelnikov et al., 2023; Shi et al., 2025). Our conditional adaptations place these generators in the posterior-sampling task defined in section 4, with source-posterior training samples and the same task and trial-sampling inputs. Their original tabular-generation tasks and this posterior-sampling comparison therefore have different targets.

Learned iterative samplers provide another point of comparison. Amortized population Gibbs samplers (Wu et al., 2020) learn conditional proposals for groups of variables and reuse them across observations. For pretrained discrete diffusion models, learned proposals (Ou et al., 2026) modify denoising transitions within sequential Monte Carlo. RGPC instead updates the parameters of an explicit sampling distribution from density-ratio feedback, with the same target posterior at every update.

Constraint preservation and tail coverage use established sampling constructions. Information projection finds a distribution satisfying supplied linear constraints by minimizing relative entropy from a reference distribution (Csiszár, 1975). Defensive mixtures retain a reference component alongside an adapted proposal (Hesterberg, 1995). We use these constructions to support feasible sampling and retain tail coverage. An evaluable proposal density also permits optional self-normalized importance weighting for posterior expectations (Geweke, 1989), as described in Appendix A.

Finally, set encoders provide ways to combine unordered inputs. Deep Sets characterizes permutation-invariant aggregation (Zaheer et al., 2017), and Set Transformer uses attention to process sets (Lee et al., 2019). These works provide the set-processing background for RGPC's trial-sample aggregation. Our comparison with PDT focuses on the subsequent use of current-coordinate information at parameter prediction, with the construction and its evaluation given in section 3.2 and Appendix C, respectively.