CONTINUAL MEMORY: SEPARATING SELECTION AND REPRESENTATION IN CONTINUAL REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Rehearsal reduces forgetting in continual reinforcement learning by storing a small memory of past states and constraining the policy on them. Yet this memory is typically formed by uniform sampling, leaving open which states should be retained and how they should be compared. We introduce Continual Memory, a framework for rehearsal memory construction that separates two decisions: a representation that defines similarity between states and a selection rule that determines which states are stored. On Continual World, we vary these decisions separately while holding the policy, training budget, and rehearsal loss fixed. The selection rule changes retention. Coverage based selection increases forgetting in every representation we test, whereas density based selection shows no detectable increase in forgetting and forgets 0.029 less than coverage on the same 20 seeds. With the selection rule fixed, the representation changes forward transfer. A merge and reduce representation of policy gradients reaches 0.47 forward transfer, compared with 0.42 for our ClonEx-SAC reproduction and 0.44 reported previously, although this improvement is supported by the unpaired analysis and comes with increased forgetting. Continual Memory with density selection reaches 0.44 forward transfer while preserving ClonEx-SAC's performance and retention. These results separate two roles of rehearsal memory construction: selection determines retention in the fixed representation we study, while representation determines forward transfer under coverage selection.

1 INTRODUCTION

Continual reinforcement learning studies agents that must learn a sequence of tasks without returning to previous tasks for retraining (Khetarpal et al., 2022). Such agents should retain skills acquired on earlier tasks while using prior experience to learn later tasks more efficiently (Wołczyk et al., 2021; Yu et al., 2020). This remains difficult even on controlled benchmarks. On Continual World (Wołczyk et al., 2021), agents trained sequentially on ten manipulation tasks can learn each task successfully and still lose substantial performance on tasks encountered earlier.

Methods for continual learning address this problem through regularisation (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018), parameter isolation (Mallya & Lazebnik, 2018; Rusu et al., 2016b; Zheng et al., 2025), and rehearsal (Rolnick et al., 2019; Rebuffi et al., 2017). Rehearsal retains a bounded memory from completed tasks and uses it to constrain the current policy toward its previous behaviour. Rehearsal has been found effective on Continual World, yet its memory is still constructed by uniformly sampling states from the replay buffer at each task boundary (Wołczyk et al., 2022).

The choice of which states to retain is therefore a data summarisation problem. Coresets provide a general approach to this problem by replacing a large data set with a smaller subset that approximately preserves a target objective (Agarwal et al., 2005; Langberg & Schulman, 2010; Feldman & Langberg, 2011). In supervised continual learning, several methods use structured memory selection instead of uniform sampling and report improved retention (Borsos et al., 2020; Yoon et al., 2022; Aljundi et al., 2019). These results suggest that rehearsal performance may depend not only on memory size, but also on which examples the memory contains.

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.

Applying this idea to continual reinforcement learning introduces an additional choice. A selection rule can only decide which states to retain after a representation defines which states are similar. Existing coreset and sample selection methods commonly combine these two choices by selecting examples using distances in a particular observation, feature, or gradient space (Sener & Savarese, 2018; Ash et al., 2020; Tiwari et al., 2022). As a result, a change in performance cannot generally be attributed to the selection rule or to the representation alone.

This distinction is especially important for policy rehearsal. The stored states are not used to reconstruct the state distribution itself. They are used to preserve the behaviour of a frozen policy through a cloning loss. Standard coreset guarantees are instead stated for objectives such as sums of distances to centres (Langberg & Schulman, 2010; Feldman & Langberg, 2011). Consequently, it is not known whether coreset based memory selection improves the cloning memory of a continual reinforcement learning agent, or which part of such a construction determines retention and transfer.

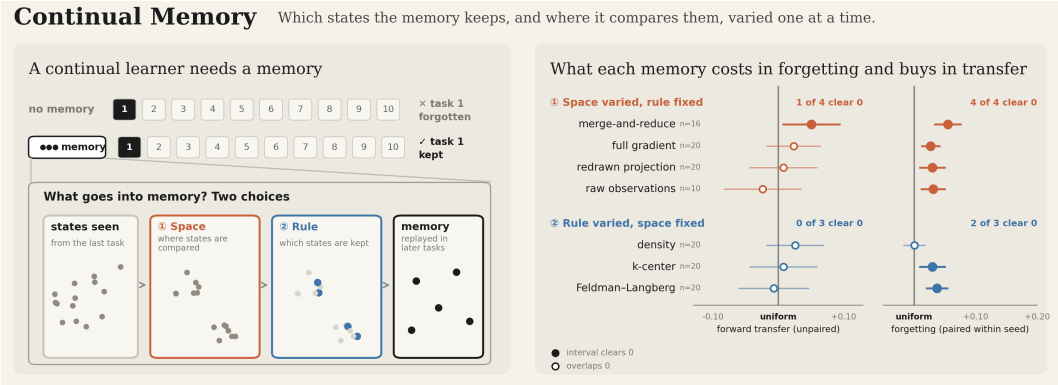


Figure 1: **Continual Memory.** Representation and selection shape forward transfer and forgetting.

In this work, we introduce **Continual Memory**, which formulates rehearsal memory construction as two separate decisions: a representation that defines similarity between states and a selection rule that determines which states are retained. We study these decisions separately on CW10 while holding the network, training budget, and rehearsal objective fixed. Figure 1 illustrates this decomposition: at each task boundary, a bounded set of states is selected from the visited states using a chosen representation and selection rule, then stored for rehearsal on subsequent tasks. We show three main results. First, the selection rule changes retention. Coverage based selection increases forgetting in every representation we test, whereas density based selection has no detectable increase in forgetting and forgets 0.029 less than coverage on the same 20 seeds. Second, with the selection rule fixed to coverage, the representation changes forward transfer. The merge and reduce gradient representation improves forward transfer by 0.075 relative to raw observations and reaches 0.47, compared with 0.42 for our ClonEx-SAC reproduction. Third, the resolved transfer improvement comes with increased forgetting, while density based selection preserves ClonEx-SAC's retention and reaches 0.44 forward transfer. Because we vary one decision at a time, the experiments separate their individual effects but do not estimate an interaction between representation and selection rule.

Our contributions are:

1. We formulate rehearsal memory construction in continual reinforcement learning as a coreset problem for policy preservation and separate the construction into a representation and a selection rule (§4).
2. We show that the selection rule changes retention when the representation is fixed. Coverage based selection increases forgetting, whereas density based selection has no detectable increase in forgetting and forgets 0.029 less than coverage on the same 20 seeds (§6.1).
3. We show that the representation changes forward transfer when the selection rule is fixed. The merge and reduce gradient representation improves forward transfer relative to raw observations and exceeds the uniform baseline under the unpaired analysis, at a retention cost (§6.2).

4. We identify which effects can be attributed to representation and selection separately, and which require crossed experiments to resolve (§6.3).

We formalise rehearsal memory construction in §4 and describe the experimental setup in §5. We then study the effect of the selection rule in §6.1, the effect of the representation in §6.2, and their consequences for retention and transfer in §6.3. Finally, §7 discusses hypotheses consistent with the observed results.

2 Related work

Memories in continual RL. Continual World (Wołczyk et al., 2021) compares regularisation (Kirkpatrick et al., 2017; Zenke et al., 2017; Aljundi et al., 2018; Nguyen et al., 2018; Li & Hoiem, 2018), parameter isolation (Mallya & Lazebnik, 2018; Rusu et al., 2016b; Zheng et al., 2025) and rehearsal (Rebuffi et al., 2017; Rolnick et al., 2019; Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019a); only rehearsal keeps a set of states. Holding a past policy in place on stored states descends from behavioural cloning (Pomerleau, 1988; Bain & Sammut, 1995) and policy distillation (Rusu et al., 2016a; Hinton et al., 2015; Schwarz et al., 2018); CLEAR (Rolnick et al., 2019) and Policy Consolidation (Kaplanis et al., 2019) are the RL precedents, and ClonEx-SAC (Wołczyk et al., 2022), which draws 10^4 states uniformly at each boundary, is the strongest on Continual World. That paper attributes forward transfer to exploration and to what crosses the boundary; its memory is a uniform draw throughout. Ahn et al. (2024) document negative transfer on the same benchmark under a different protocol; our memories keep the network and change only the memory. The nearest precedent to our question is Isele & Cosgun (2018), who compare filling rules for a lifelong-RL replay buffer of raw transitions trained on the RL loss and find distribution matching the most reliable for retention. Our retention result agrees with theirs in a different object, a cloning memory with a frozen teacher; what they do not measure, and we do, is forward transfer, and the space the set is chosen in as a separate decision from the rule.

Choosing what to store. In supervised continual learning the uniform draw, reservoir sampling (Vitter, 1985), is hard to beat (Chaudhry et al., 2019b; Prabhu et al., 2020); Dark Experience Replay (Buzzega et al., 2020) is the closest supervised analogue of the cloning memory. Selection rules have been proposed since iCaRL's herding (Welling, 2009; Rebuffi et al., 2017): gradient diversity (Aljundi et al., 2019), gradient coresets (Tiwari et al., 2022), bilevel optimisation (Borsos et al., 2020), online coreset selection (Yoon et al., 2022), inducing points (Titsias et al., 2020) and the memorable past (Pan et al., 2020). Outside continual learning, covering (Sener & Savarese, 2018) and gradient-matching (Mirzasoleiman et al., 2020; Killamsetty et al., 2021) criteria beat a uniform subset at small budgets, regime permitting (Sorscher et al., 2022; Zheng et al., 2023; Xia et al., 2023). Every one of these papers fixes an embedding, usually a gradient (Ash et al., 2020), together with a criterion, and evaluates the pair. The constructions we test are standard coresets and sketches (Agarwal et al., 2005; Feldman, 2020): farthest-point traversal (Gonzalez, 1985), sensitivity sampling (Langberg & Schulman, 2010; Feldman & Langberg, 2011; Axiotis et al., 2024), D^2 seeding (Arthur & Vassilvitskii, 2007), Carathéodory sets (Maalouf et al., 2019), random projections (Johnson & Lindenstrauss, 1984) and merge-and-reduce (Bentley & Saxe, 1980; Har-Peled & Mazumdar, 2004); none of their guarantees is stated for the divergence a cloning term poses (§4.1). We propose no new regulariser, architecture or exploration rule, we select stored states rather than synthesise them (Zhao et al., 2021; Lu et al., 2023), and we do not address plasticity loss (Nikishin et al., 2022; Dohare et al., 2024); reporting follows Henderson et al. (2018) and Agarwal et al. (2021).

3 Background

3.1 Continual World and its metrics

Continual World (Wołczyk et al., 2021) is a sequence of Meta-World manipulation tasks for a Sawyer arm (Yu et al., 2020). In CW10 a single network learns ten tasks one after another in a fixed order (Appendix A), with one million environment steps on each task, and never sees a finished task again. We denote by $N = 10$ the number of tasks and by $\Delta := 10^6$ the step count per task. Every $20{,}000$ steps the success rate of every task is measured, not only the one being trained, giving a complete curve $p_i(t)$ for each task i over the whole run. Write T for the end

of the run and τ_i for the end of task i's block. Performance is $P = \frac{1}{N}\sum_i p_i(T)$, forgetting is $F = \frac{1}{N}\sum_i \big(p_i(\tau_i) - p_i(T)\big)$, and plasticity is $P + F$ by definition and always derived. Forward transfer compares each task's learning curve in its own block with that of a single-task agent trained from scratch with the same budget: with A_i the normalised area under task i's curve in its block and A_i^{ref} the reference's, $\text{FT} = \frac{1}{N}\sum_i (A_i - A_i^{\text{ref}})/(1 - A_i^{\text{ref}})$ (Wołczyk et al., 2021). We also report the worst task $\min_i p_i(T)$, the tenth percentile $p10$, and steps to 0.8 success within a block.

3.2 Rehearsal-based continual RL

ClonEx-SAC (Wołczyk et al., 2022) is SAC (Haarnoja et al., 2018) with an exploration rule (a new task's first steps are collected with whichever previous head earns the highest return) and a behavioural-cloning term, which is the part we work on. When task i ends, the method draws $M = 10^4$ states uniformly from the replay buffer and stores them as a set C_i with the policy's Gaussian at each, $\pi_i(\cdot \mid s)$. For the rest of training the actor loss carries $\mathcal{L}_{\text{BC}} = \lambda \frac{1}{|C|}\sum_{s\in C} \text{KL}(\pi_\theta(\cdot \mid s) \,\|\, \pi_{i(s)}(\cdot \mid s))$ with $\lambda = 100$, $C = \bigcup_{j<i} C_j$ and $i(s)$ the task s came from; we write $\text{KL}_\theta(s)$ for the summand. The term acts on the actor only. The replay buffer is emptied at every task change, so C_i is chosen once, at the boundary, and never revised. We call C_i the rehearsal memory.

3.3 Coresets

A coreset is a small, usually weighted, subset of a data set chosen so that a loss evaluated on it stays within a known multiplicative error of the loss on the full data, for some class of functions. No coreset construction is known for the states of a policy directly, so we transform each state s in the buffer B into a vector $g(s)$ first; the choice of representation is one of the two decisions we study (§4.3). Take $X := \{g(s) : s \in B\}$ as the candidate pool, $C \subseteq X$ the subset to be stored, and X_c the pool points nearest to $c \in C$. The losses we work with are

$$\ell_{\beta,p}(C) \;=\; \sum_{c\in C} |X_c|^{-\beta} \sum_{x\in X_c} \|x - c\|^p,$$

with $p = 2$ unless stated otherwise. Two properties of a stored set are at stake and β chooses between them. A set has coverage if every pool point has a stored point near it, even one visited once; it matches the pool's density if the stored points are distributed the way the pool is. At $\beta = 1$ each cluster contributes its mean p-th power distance, so a cluster of a thousand points weighs no more than a cluster of one; at $\beta = 0$ every point counts once, so stored points go where the pool is crowded. Intermediate β interpolate; we run the two ends in training. We build the stored set greedily, adding at each step the candidate with the largest score $d^p m^{1-\beta}$, d its distance to the nearest chosen point and m the number of pool points it would take over. At $\beta = 1$ the score is the distance alone and the procedure is Gonzalez's farthest-point algorithm for k-center (Gonzalez, 1985), verified index for index; at $\beta = 0$ a candidate's distance is weighted by the number of points it would take over, so the set is drawn toward where the pool is crowded, which is what density means here. The score is a heuristic aimed at $\ell_{\beta,p}$, not its exact greedy decrement; the two ends are one procedure with one exponent changed, and $\ell_{\beta,p}$ is the loss the offline sweep of §7 and Appendix G measure.

A weighted set (C, w) is an ε-coreset of X for a class $\mathcal{F}$ if $\big|\sum_{x\in X} f(x) - \sum_{c\in C} w_c f(c)\big| \le \varepsilon \sum_{x\in X} f(x)$ for every $f \in \mathcal{F}$. For $\ell_{\beta,p}$ the class is sums of p-th powers of distances to a set of centres, which is what sensitivity sampling gives coresets for (Langberg & Schulman, 2010; Feldman & Langberg, 2011). Appendix D.1 discusses why a distance family should say anything about a divergence; we take it as motivation. A Bi-Criteria construction with an (α, β)-approximation guarantee for the $\beta = 0$ end of the family, with proofs, is in Appendix G.

4 Continual Memory

Continual Memory constructs a rehearsal memory through two decisions: a representation that determines how states are compared and a selection rule that determines which states are retained. We first state the policy preservation objective that the memory should approximate, then separate these two decisions and instantiate the representations and selection rules used in our experiments.

Table 1: **Continual Memory configurations.** Each arm combines one representation and one selection rule; `clonex` uses the uniform draw and is the reference for every difference in the paper. Seeds are done | planned where an arm is incomplete, and arms with fewer than ten seeds are not reported (Appendix B).

arm	representation	selection rule	seeds
`clonex`	none (buffer order)	uniform draw	20
`bc`	none (buffer order)	uniform draw	10 | 20
`csgrad`	random projection of $g(s)$, redrawn per run	k-center	20
`nosketch`	full 1024-d $g(s)$	k-center	20
`mr`	merge-and-reduce basis of $g(s)$	k-center	16 | 20
`csobs`	raw observations	k-center	10 | 20
`fl`	random projection, redrawn	FL sensitivity sampling	20
`beta0`	random projection, redrawn	density, β=0	20

4.1 The problem, formally

Formally, let π_θ be the policy, π_i the copy frozen at the end of task i, and d_i the distribution of states visited while training on task i, which is what the buffer holds at the boundary. The retention loss the cloning term stands for is $\mathbb{E}_{s\sim d_i}[\mathrm{KL}_\theta(s)]$; rehearsal replaces it by an average over a stored set C_i of M states. The average has to serve a family of functions indexed by parameters that do not exist yet, $\mathcal{Q} = \{\, s \mapsto \mathrm{KL}(\pi_\theta(\cdot \mid s) \,\|\, \pi_i(\cdot \mid s)) : \theta \text{ reachable under training} \,\}$, and the condition we would like is the ε-coreset condition of §3.3 with $\mathcal{F} = \mathcal{Q}$:

$$\left| \tfrac{1}{|C_i|} \textstyle\sum_{s\in C_i} \mathrm{KL}_\theta(s) \;-\; \mathbb{E}_{s\sim d_i}\big[\mathrm{KL}_\theta(s)\big] \right| \;\le\; \varepsilon\, \mathbb{E}_{s\sim d_i}\big[\mathrm{KL}_\theta(s)\big] \qquad \text{for every reachable } \theta.$$

A uniform draw is unbiased for any one θ; what is asked is that one fixed set meet the bound for every θ at once. What keeps $\mathcal{Q}$ outside the available coreset bounds is that KL_θ is not a function of distance in the representation the constructions work in (Appendix D.1).

4.2 Two decisions, varied separately

Any construction that picks C_i fixes a representation in which two states count as close and a selection rule for which states to keep once closeness is settled; the rule only ever sees distances, and the representation produces them. The literature reports pairs (Sener & Savarese, 2018; Ash et al., 2020; Tiwari et al., 2022), which cannot tell which change produced which effect. We run a one-factor-at-a-time design around one anchor, `csgrad`: the k-center rule, the coverage criterion behind coreset-based selection, in a random 32-dimensional sketch of the gradient embedding, which is how that literature makes gradients cheap to cluster. We call each configuration of representation and selection rule an arm. Every other memory arm differs from `csgrad` in exactly one decision (Table 1); `bc` and `clonex` differ from each other only in the exploration rule.

4.3 Representations and selection rules

The gradient embedding. The rehearsal memory exists to constrain the actor's output layer at a set of states, so the question to ask of two states is whether they impose the same constraint on that layer. For a loss ℓ acting through the action mean $\mu(s) = Wh(s) + b$, with $h(s) \in \mathbb{R}^{256}$ the trunk feature, the gradient with respect to W is the outer product $(\partial\ell/\partial\mu)\otimes h(s)^\top$: two states with parallel h and equal push are one constraint, two with opposite pushes would cancel, two with orthogonal h constrain disjoint parts of W however alike they look; this is why supervised coreset selection uses gradient embeddings (Ash et al., 2020; Tiwari et al., 2022). The cloning term's own push is identically zero at the boundary, where teacher and student coincide, so we take the push from the SAC actor loss: up to the tanh Jacobian it is $-\nabla_a \min_k Q_k(s,a)$ at $a = \pi_\theta(s)$, the slope of the critic's value at the current action. We use

$$g(s) \;=\; \mathrm{vec}\Big(\nabla_a \min_k Q_k(s,a)\big|_{a=\pi(s)} \;\otimes\; h(s)^\top\Big) \;\in\; \mathbb{R}^{1024},$$

a surrogate for the per-state actor gradient at the output layer, with the tanh Jacobian and the entropy push (a function of the action alone) left out; all we ask of it is that its distances separate states the

actor is pushed in different directions at. At each boundary we draw a pool of $N = 5 \times 10^4$ states uniformly from the buffer, map each $g(s)$ to the arm's representation $x(s) = \Pi\, g(s)$, run the arm's selection rule on $\{x(s)\}$ with budget $M = 10^4$, and store the chosen states with $\pi_i(\cdot \mid s)$. Only Π and the selection rule differ between arms.

Representations (the map Π). `csgrad`: a Gaussian projection of $g(s)$ to 32 dimensions, redrawn from the run seed; the reference for every other arm. `nosketch`: the full 1024 dimensions. `mr`: a merge-and-reduce basis (Har-Peled & Mazumdar, 2004): the pool is read in blocks of 2,048 rows, each block is replaced by its rank-32 truncated SVD, two summaries of equal size are merged and re-truncated, and a final SVD of the surviving summaries gives 32 basis vectors; the basis is deterministic and follows the directions the gradients vary along. `csobs`: the raw observation, the one representation not built from $g(s)$.

Selection rules (in the `csgrad` representation unless the representation is varied). Uniform draw (`clonex`, `bc`): the published method, a rule in its own right since a uniform sample matches the visited density. k-center: Gonzalez's farthest-point traversal from a start point drawn uniformly with the run seed, the $\beta = 1$ end of the family. FL sensitivity (`fl`): the two-phase Feldman–Langberg construction (Feldman & Langberg, 2011): k-center at a tenth of the budget gives centres B; then M i.i.d. draws with probability $q(x) \propto d(x, B)^2 / \sum d^2 + 1/(|B|\, n_x)$, n_x the size of x's cluster, each weighted $1/(M\, q(x))$. The size term keeps every point's probability positive, so the weighted sum is unbiased for any loss over the pool; it is not a $(1 \pm \varepsilon)$-coreset (Appendix D.2). Density (`beta0`): the $\beta = 0$ end of the greedy score of §3.3, started from the same uniformly drawn point as k-center, with cluster sizes refreshed as the set grows; the appendix theory's Definition 1 with $f = 1$, Dist $= \| \cdot \|_2^6$ is this family at $\beta = 0$, $p = 6$. The three representations and the k-center and Feldman–Langberg selection rules each carry a guarantee for some class of functions of the embedded points; the density score carries none; and no guarantee is for $\mathcal{Q}$ (Appendix D.2).

5 Experimental setup

We evaluate Continual Memory by varying its two design decisions separately while keeping the underlying continual RL system fixed. We ask three questions:

- **RQ1:** With the representation fixed, does the selection rule change retention?
- **RQ2:** With the selection rule fixed, does the representation change forward transfer?
- **RQ3:** Does a memory that raises forward transfer pay for it in retention?

Benchmark. We use CW10 with $\Delta = 10^6$ steps per task and one task order (§3.1). Forward transfer is measured at the full budget against single-task references, with ten runs per task averaged.

Model and training budget. Every arm is ClonEx-SAC with a different rehearsal memory and nothing else changed. No arm receives extra parameters, environment steps, or gradient updates, and the critic is never masked, frozen, or penalised (Appendix A). Memory selection runs once per task boundary. Offline, k-center takes 1.9 seconds and the density selection rule 7.8 seconds at $M = 10^4$, compared with about 30 hours per training run; basis construction time and peak memory were not measured.

Baselines. We rerun ClonEx-SAC (`clonex`) and BC (`bc`) on our seeds. Every reported difference is taken against our `clonex`; published ClonEx-SAC, BC, and SSDE results are reported alongside ours in Appendix E.

Seeds. We use environment seeds 0 to 19; planned and completed counts are reported in Table 1. Seeds were launched as compute became available, and no run was stopped or removed based on its outcome. Arms below their planned count had seeds still running (`mr`) or not yet launched (`bc`, `csobs`) at submission and therefore carry their completed n. We report only arms with at least ten complete seeds. Arms below this threshold, a 200k ablation grid at three seeds, and CW20 at seven seeds are listed in Appendix B and included in the released logs. The reported arms contain 136 complete 1M runs.

Metrics and statistical protocol. For P, F, plasticity, worst task, and $p10$, we report the within-seed difference against `clonex` over shared seeds, together with its t statistic and a 90% interval, mean ± 1.83 standard errors (Appendix E, Table 8). Forward transfer is reported both paired and unpaired.

Table 2: Upper half: every arm uses the same 32-dimensional gradient representation, so only the selection rule changes. Lower half: the same k-center selection rule is used throughout, so only the representation changes. ΔF and ΔP are paired within seed; ΔFT is given both unpaired (primary, §5) and paired, each with its t. * marks $|t| > 1.83$. Intervals are in Table 8.

	n	ΔF (paired)	ΔP (paired)	ΔFT unpaired (t)	ΔFT paired (t)
vary the selection rule, representation fixed at the redrawn sketch					
uniform (`clonex`)	20	0	0	0	0
density, β=0 (`beta0`)	20	$+0.000$	$+0.005$	$+0.026$ (1.09)	$+0.026$ (1.07)
k-center (`csgrad`)	20	$+0.030^*$	-0.033^*	$+0.008$ (0.28)	$+0.008$ (0.25)
FL sensitivity (`fl`)	20	$+0.037^*$	-0.046^*	-0.007 (-0.24)	-0.007 (-0.22)
vary the representation, selection rule fixed at k-center: built from $g(s)$					
merge-and-reduce basis (`mr`)	16	$+0.055^*$	-0.054^*	$+0.051^*$ (2.15)	$+0.043$ (1.62)
full 1024-d (`nosketch`)	20	$+0.027^*$	-0.017	$+0.024$ (1.06)	$+0.024$ (1.12)
redrawn projection (`csgrad`)	20	$+0.030^*$	-0.033^*	$+0.008$ (0.28)	$+0.008$ (0.25)
not built from $g(s)$					
raw observations (`csobs`)	10	$+0.031^*$	-0.031^*	-0.024 (-0.75)	-0.025 (-0.84)

The unpaired estimate is the difference of arm means with a t statistic computed from the two arms' standard errors. It is primary because, across environment seeds, an arm's forward transfer does not move with the baseline's (mean $r = -0.04$ over eight arms, Appendix C). This choice was made after inspecting those correlations, and the one comparison it changes, the merge-and-reduce basis ($+0.043$ ($t\,1.62$, n.s.) paired versus $+0.051$ ($t\,2.15$, unpaired) unpaired), is therefore presented as exploratory. A star marks $|t| > 1.83$, the two-sided 90% value at ten degrees of freedom and above every comparison's Welch value. The family contains seven arms evaluated on three metrics; we do not correct for multiplicity. Every retention star has $|t| \geq 2.7$ and would survive a Bonferroni correction, while the one attributable transfer star ($t\,2.15$) would not.

6 Results

Baselines. Our ClonEx-SAC reaches $P = 0.865$ against a published 0.86 $[0.84, 0.87]$ and forward transfer 0.416 $[0.388, 0.443]$ against 0.44 $[0.42, 0.46]$; our BC reaches 0.831 and 0.379 against 0.84 and 0.41. The absolute table is Table 7 and every difference with its interval is Table 8 (Appendix E). Every difference below is against our own `clonex` (Table 2). Table 2 isolates the two Continual Memory decisions by varying one while holding the other fixed.

6.1 RQ1: The selection rule changes retention

Call the extra forgetting an arm pays relative to the uniform draw its fee. With the selection rule fixed at k-center, the fee is about the same in three of the four representations, $+0.030$ in the sketch, $+0.027$ in the full gradient and $+0.031$ in raw observations, each with $|t| \geq 2.7$; the merge-and-reduce basis pays more, $+0.055$ ($t\,4.85$) at $n = 16$, and also loses the most performance (-0.054 ($t\,2.92$)). So the representation is not irrelevant to retention: coverage costs retention everywhere, and how much depends on the representation. With the representation fixed, the fee moves with the selection rule. k-center and Feldman–Langberg pay $+0.030$ and $+0.037$ at $n = 20$; the Feldman–Langberg memory holds only about $4{,}500$ distinct states, so its fee is confounded with support size (Appendix D.2). The density rule pays $+0.000$ (n.s.) at $n = 20$, interval $[-0.017, +0.018]$: no difference from the uniform draw at that precision, which is not a proof of equality. The direct comparison carries the claim: against k-center on the same twenty seeds the density rule forgets -0.029 ($t\,2.64$) less and keeps $+0.038$ ($t\,2.25$) more performance, both with $|t| > 2.2$. Performance follows the fee: k-center loses 0.033 of P and Feldman–Langberg 0.046.

6.2 RQ2: The representation changes forward transfer

With the selection rule fixed, the representation not built from $g(s)$, raw observations, is lowest (-0.024 against the uniform draw, $n = 10$), and the three built from it are above it: $+0.008$, $+0.024$, and $+0.051$. The direct contrasts between representations, unpaired, are what support the claim: the

merge-and-reduce basis is $+0.075$ above raw observations ($t\,2.36$), the full gradient $+0.048$ ($t\,1.55$), and the redrawn sketch $+0.032$ ($t\,0.91$); the basis against the sketch is $+0.043$ ($t\,1.56$). Against the uniform draw, one representation clears the threshold, the basis at $+0.051$ ($t\,2.15,$ unpaired) with $n = 16$, and its paired estimate, $+0.043$ ($t\,1.62,$ n.s.), does not; the other two do not clear it either way. Pooling the three gradient representations (56 runs) against the observation representation (10) gives $+0.050$ ($t\,1.71$), but the pooled runs share seeds across arms, so that t is description only. That is the extent of the claim: one representation significantly above the observation representation and, unpaired, above the uniform draw; the other two are ordered consistently but not significantly. With the representation fixed, no selection rule differs from zero on transfer; the density rule reads $+0.026$ ($t\,1.09,$ n.s.) at $n = 20$.

6.3 RQ3: DOES TRANSFER COST RETENTION?

The one memory whose transfer gain we can resolve pays for it: the merge-and-reduce basis gains $+0.051$ ($t\,2.15,$ unpaired) and forgets $+0.055$ ($t\,4.85$) more, with P down -0.054 ($t\,2.92$). That is a trade, and it is the trade the rehearsal literature assumes. The other arms do not follow it: the four covering arms that pay a fee of about 0.03 span transfer from -0.024 to $+0.024$, and the density rule, which pays no fee we can detect, sits at $+0.026$, not significant. Across the seven arms the fee-transfer correlation is $r = +0.18$ ($t\,0.42$); seven points cannot rule out a relationship and we draw nothing from it (Figure 2, Appendix E). What the map shows is that no memory in our set raises transfer significantly without a retention cost. The candidate for that is the density rule, and whether its edge is real, and whether it belongs to the selection rule or to the representation in which it runs, is decided by configurations we have not run: density selection with the observation representation and with the merge-and-reduce representation. In absolute terms the density rule reaches $P = 0.870$, $F = 0.007$, and forward transfer 0.442 against the published 0.86, 0.02, and 0.44: it matches the published transfer and does not exceed it. The highest transfer in our table, 0.470 at $n = 19$ ($+0.054$, $t\,2.35$), belongs to `int`, `csgrad` with four further changes (Appendix B), and its memory mix alone (`intmix`, $n = 10$) reads 0.450 ($+0.035$ ($t\,0.79,$ n.s.)); both are listed, not attributed. The basis and the density rule reach 0.8 success sooner (Figure 3).

7 HYPOTHESES FOR THE OBSERVED EFFECTS

The experiments show two patterns. The selection rule changes retention, and the representation changes forward transfer under coverage selection. We do not yet have a complete mechanism for either effect. In this section we discuss the explanations that best fit the evidence and the experiments that would separate them.

Selection moves from coverage to density. The exponent β in §3.3 gives us a continuous family of selection rules. At $\beta = 1$, k-center prioritises coverage. At $\beta = 0$, the rule moves toward regions where the visited-state distribution is dense. On one saved boundary pool at budget 600, moving from coverage to density reduces the stored set's relative error in estimating a quadratic loss over the pool by 46%, while the covering-radius ratio increases by $2.5\times$ (Figure 4, Appendix E). This tells us that the two ends of the family make a real trade. Coverage spreads the memory, while density makes it follow the data more closely.

We see the same ordering in training. The coverage end adds 0.030 forgetting, whereas the density end adds $+0.000$ (n.s.). Against $\beta = 1$ on the same twenty seeds, density selection forgets -0.029 ($t\,2.64$) less and keeps $+0.038$ ($t\,2.25$) more performance. The corresponding change in forward transfer, ΔFT $+0.018$ ($t\,0.65,$ n.s.), remains unresolved. What we still do not know is whether the offline approximation result explains the retention result. The direct test is to measure, at later checkpoints, how accurately each rehearsal memory estimates the cloning divergence on held-out states across several task boundaries and seeds.

Why density selection may retain better. Our first hypothesis is that coverage produces a rehearsal set that is easier to memorise than to generalise from. The cloning loss constrains the policy only at stored states. A coverage rule deliberately spreads those states across the visited set, including regions with little probability mass. The actor may therefore satisfy those stored constraints without preserving the frozen policy well between them. Density selection places more of the memory where

the policy spent more of its time, so the stored constraints may better represent the behaviour that matters under the training distribution.

The retention result is consistent with this explanation, but it does not show that memorisation is the cause. Coverage also changes the rehearsal distribution itself. It places more weight on rare states, which can change the magnitude of the cloning gradient, how often clipping is active, and the effective strength of a fixed λ. We can separate these explanations with a direct control. We keep the states selected by the coverage rule and weight their rehearsal loss by cluster mass. If retention improves, the distributional weighting matters. If it does not, the geometry of the stored set becomes a stronger explanation. Sweeps over memory size and λ test the remaining alternatives. These controls have not yet reached the reporting threshold used in this paper.

Why the representation may affect transfer. Our working hypothesis is that transfer depends on whether the memory preserves directions in which the actor is likely to move. Two states are close in $g(s)$ when their actor updates point in similar directions. Under k-center, selecting states in this representation spreads the rehearsal memory across distinct update directions. When later tasks move the policy, the cloning loss can then constrain those same directions.

This interpretation is consistent with the observation representation. Raw observations are the only representation we test that discards the gradient geometry, and they give the lowest forward transfer among the representations under k-center. The explanation is less complete within the gradient-based representations. The random sketch, the full gradient, and the merge-and-reduce basis all preserve $g(s)$ in different ways, but no statistic we measured explains their ordering in forward transfer (Table 10). The same explanation also does not account for why the merge-and-reduce representation pays a larger retention cost than the random sketch. To separate these possibilities, we need to measure which actor-update directions each memory preserves and how those directions are used during later tasks.

8 Discussion and conclusion

Summary. We introduced **Continual Memory** to separate two decisions that are usually combined in rehearsal memory construction, the representation used to compare states and the selection rule used to retain them. By varying these decisions separately on CW10, we find that the selection rule changes retention, while the representation changes forward transfer under coverage selection. Coverage based selection increases forgetting in every representation we test, by about 0.03 in three and 0.055 in the merge-and-reduce representation. With the representation fixed, density selection shows no detectable retention cost and retains 0.029 more than coverage over the same 20 seeds. With the selection rule fixed, the merge-and-reduce representation transfers more than raw observations and, under the unpaired analysis, more than the uniform draw, although this comes with a retention cost. Continual Memory with density selection preserves ClonEx-SAC's performance and retention and reaches its published forward transfer, with the improvement over our reproduction still unresolved. **The open problem.** Continual Memory is motivated by a coreset objective, but the constructions we evaluate do not provide a guarantee for the policy divergence class $\mathcal{Q}$. Appendix G proves a Bi-Criteria guarantee for the $\beta = 0$ end of the distance based loss family, not for $\mathcal{Q}$ and not for the greedy score used in training. Extending coreset guarantees to the divergence induced by future policy updates therefore remains open. **Limitations.** Our experiments vary one decision at a time around a common anchor, so they do not estimate interactions between representation and selection rule. In particular, we have not evaluated density selection with the observation or merge-and-reduce representations, so the representation result is scoped to coverage selection and the selection result to the sketch representation. The attributable forward transfer result depends on the unpaired estimator, which was selected after inspecting seed correlations. The density result means that we detect no difference at the precision of the reported interval, not that the two methods are equivalent. We also study one task order and one training budget, and several ablations remain below the ten-seed reporting threshold. **Broader impact.** The two decisions in Continual Memory are not specific to Meta-World or SAC. Any learning system that compresses past experience into a bounded memory and trains against that memory must decide how examples are represented and which examples are retained. The same structure appears in language-model replay (Gupta et al., 2023; Ibrahim et al., 2024) and in retrieval of demonstrations for a bounded context. Appendix F develops corresponding hypotheses for these settings, which we leave to future work.

AI USE STATEMENT

Generative AI tools were used to draft and edit prose and to write and debug analysis and plotting scripts. Every reported number was computed by scripts from the run logs and checked by the authors, who take responsibility for the whole content of this work.

ETHICS STATEMENT

This work trains simulated robot-arm policies on a public benchmark, involves no human subjects and releases no new data; null results are reported in full.

REPRODUCIBILITY STATEMENT

Appendix A lists every hyperparameter and the task order, Appendix B every run family with seeds completed against planned, Appendix C the estimator, and Appendix D the gradient embedding. Every number is recomputable from the logged per-task success grid $p_i(t)$; code, run configurations and per-run logs are provided as supplementary material.

REFERENCES

Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Geometric approximation via coresets. In Combinatorial and Computational Geometry. Cambridge University Press, 2005.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

Hongjoon Ahn, Jinu Hyeon, Youngmin Oh, Bosun Hwang, and Taesup Moon. Reset & distill: A recipe for overcoming negative transfer in continual reinforcement learning. arXiv preprint arXiv:2403.05066, 2024.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In European Conference on Computer Vision (ECCV), pp. 144–161. Springer, 2018.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In Advances in Neural Information Processing Systems (NeurIPS), 2019.

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1027–1035, 2007.

Jordan T. Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. In International Conference on Learning Representations (ICLR), 2020.

Kyriakos Axiotis, Vincent Cohen-Addad, Monika Henzinger, Sammy Jerome, Vahab Mirrokni, David Saulpic, David Woodruff, and Michael Wunder. Data-efficient learning via clustering-based sensitivity sampling: Foundation models and beyond. In International Conference on Machine Learning (ICML), 2024.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In Machine Intelligence 15, Intelligent Agents, pp. 103–129. Oxford University Press, 1995.

Peter L. Bartlett and Wolfgang Maass. Vapnik–Chervonenkis dimension of neural nets. In The Handbook of Brain Theory and Neural Networks, pp. 1188–1192. MIT Press, 2nd edition, 2003.

Jon Louis Bentley and James B. Saxe. Decomposable searching problems I: Static-to-dynamic transformation. Journal of Algorithms, 1(4):301–358, 1980.

Zalán Borsos, Mojmír Mutný, and Andreas Krause. Coresets via bilevel optimization for continual learning and streaming. In Advances in Neural Information Processing Systems (NeurIPS), 2020.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: A strong, simple baseline. In Advances in Neural Information Processing Systems (NeurIPS), 2020.

Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In International Conference on Learning Representations (ICLR), 2019a.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. arXiv preprint arXiv:1902.10486, 2019b.

Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. Nature, 632:768–774, 2024.

Dan Feldman. Core-sets: Updated survey. Sampling Techniques for Supervised or Unsupervised Tasks, pp. 23–44, 2020.

Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In ACM Symposium on Theory of Computing (STOC), pp. 569–578, 2011.

Dan Feldman and Leonard J Schulman. Data reduction for weighted and outlier-resistant clustering. In ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1343–1354. Society for Industrial and Applied Mathematics, 2012.

Dan Feldman, Melanie Schmidt, and Christian Sohler. Turning big data into tiny data: Constant-size coresets for k-means, PCA and projective clustering. In ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1434–1453. Society for Industrial and Applied Mathematics, 2013.

Dan Feldman, Zahi Kfir, and Xuan Wu. Coresets for Gaussian mixture models of any shape. arXiv preprint arXiv:1906.04895, 2019.

Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. Theoretical Computer Science, 38:293–306, 1985.

Kshitij Gupta, Benjamin Thérien, Adam Ibrahim, Mats L. Richter, Quentin Anthony, Eugene Belilovsky, Irina Rish, and Timothée Lesort. Continual pre-training of large language models: How to (re)warm your model? arXiv preprint arXiv:2308.04014, 2023.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In International Conference on Machine Learning (ICML), 2018.

Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In ACM Symposium on Theory of Computing (STOC), pp. 291–300, 2004.

Sariel Har-Peled and Micha Sharir. Relative (p, ε)-approximations in geometry. Discrete & Computational Geometry, 45(3):462–496, 2011.

Haibo He and Edwardo A. Garcia. Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9):1263–1284, 2009. doi: 10.1109/TKDE.2008.239.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In AAAI Conference on Artificial Intelligence, 2018.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

C. A. R. Hoare. Algorithm 65: Find. Communications of the ACM, 4(7):321–322, 1961.

Adam Ibrahim, Benjamin Thérien, Kshitij Gupta, Mats L. Richter, Quentin Anthony, Timothée Lesort, Eugene Belilovsky, and Irina Rish. Simple and scalable strategies to continually pre-train large language models. arXiv preprint arXiv:2403.08763, 2024.

David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In AAAI Conference on Artificial Intelligence, 2018.

William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. Contemporary Mathematics, 26:189–206, 1984.

Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Policy consolidation for continual reinforcement learning. In International Conference on Machine Learning (ICML), 2019.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. Journal of Artificial Intelligence Research, 75:1401–1476, 2022.

Krishnateja Killamsetty, S. Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. GRAD-MATCH: Gradient matching based data subset selection for efficient deep model training. In International Conference on Machine Learning (ICML), 2021.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. Proceedings of the National Academy of Sciences, 114(13):3521–3526, 2017.

Thomas Kleine Buening, Jonas Hübotter, Barna Pásztor, Idan Shenfeld, Giorgia Ramponi, and Andreas Krause. Aligning language models from user interactions. arXiv preprint arXiv:2603.12273, 2026.

Michael Langberg and Leonard J. Schulman. Universal ε-approximators for integrals. In ACM-SIAM Symposium on Discrete Algorithms (SODA), 2010.

Yi Li, Philip M. Long, and Aravind Srinivasan. Improved bounds on the sample complexity of learning. Journal of Computer and System Sciences, 62(3):516–527, 2001.

Zhizhong Li and Derek Hoiem. Learning without forgetting. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(12):2935–2947, 2018.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In Advances in Neural Information Processing Systems (NeurIPS), 2017.

Cong Lu, Philip J. Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. In Advances in Neural Information Processing Systems (NeurIPS), 2023.

Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. Training Gaussian mixture models at scale via coresets. The Journal of Machine Learning Research, 18(1):5885–5909, 2017.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. IEEE Transactions on Audio, Speech and Language Processing, 2025. doi: 10.1109/TASLPRO.2025.3606231.

Alaa Maalouf, Ibrahim Jubran, and Dan Feldman. Fast and accurate least-mean-squares solvers. In Advances in Neural Information Processing Systems (NeurIPS), 2019.

Arun Mallya and Svetlana Lazebnik. PackNet: Adding multiple tasks to a single network by iterative pruning. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In International Conference on Machine Learning (ICML), 2020.

Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In International Conference on Learning Representations (ICLR), 2018.

Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In International Conference on Machine Learning (ICML), 2022.

Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard E. Turner, and Mohammad Emtiyaz Khan. Continual deep learning by functional regularisation of memorable past. In Advances in Neural Information Processing Systems (NeurIPS), 2020.

Dean A. Pomerleau. ALVINN: An autonomous land vehicle in a neural network. In Advances in Neural Information Processing Systems (NeurIPS), 1988.

Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. GDumb: A simple approach that questions our progress in continual learning. In European Conference on Computer Vision (ECCV), 2020.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. Experience replay for continual learning. In Advances in Neural Information Processing Systems (NeurIPS), 2019.

Andrei A. Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. In International Conference on Learning Representations (ICLR), 2016a.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. arXiv preprint arXiv:1606.04671, 2016b.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In International Conference on Machine Learning (ICML), 2018.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In International Conference on Learning Representations (ICLR), 2018.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S. Morcos. Beyond neural scaling laws: Beating power law scaling via data pruning. In Advances in Neural Information Processing Systems (NeurIPS), 2022.

Michalis K. Titsias, Jonathan Schwarz, Alexander G. de G. Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with Gaussian processes. In International Conference on Learning Representations (ICLR), 2020.

Rishabh Tiwari, Krishnateja Killamsetty, Rishabh Iyer, and Pradeep Shenoy. GCR: Gradient coreset based replay buffer selection for continual learning. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2022.

Jeffrey S. Vitter. Random sampling with a reservoir. ACM Transactions on Mathematical Software, 11(1):37–57, 1985.

Max Welling. Herding dynamical weights to learn. In International Conference on Machine Learning (ICML), 2009.

Maciej Wołczyk, Michał Zajac, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Continual world: A robotic benchmark for continual reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS), 2021.

Maciej Wołczyk, Michał Zajac, Razvan Pascanu, Łukasz Kuciński, and Piotr Miłoś. Disentangling transfer in continual reinforcement learning. In Advances in Neural Information Processing Systems (NeurIPS), 2022.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In International Conference on Machine Learning (ICML), 2024.

Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In International Conference on Learning Representations (ICLR), 2023.

Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In International Conference on Learning Representations (ICLR), 2022.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Conference on Robot Learning (CoRL), 2020.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In International Conference on Machine Learning (ICML), pp. 3987–3995. PMLR, 2017.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In International Conference on Learning Representations (ICLR), 2021.

Chengqi Zheng, Haiyan Yin, Jianda Chen, Terence Ng, Yew-Soon Ong, and Ivor Tsang. Mastering continual reinforcement learning through fine-grained sparse network allocation and dormant neuron exploration. arXiv preprint arXiv:2503.05246, 2025.

Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. Coverage-centric coreset selection for high pruning rates. In International Conference on Learning Representations (ICLR), 2023.

A SETUP

Every arm uses the published ClonEx-SAC configuration; nothing below is varied across arms except the selection rule and the space it runs in. The values are read from the run configuration files. CW10 task order: hammer, push-wall, faucet-close, push-back, stick-pull, handle-press-side, push, shelf-place, window-close, peg-unplug-side (Meta-World v1 variants).

Table 3: **Hyperparameters, shared by every arm.**

steps per task	10^6
actor and critic trunk	4×256, leaky ReLU, layer norm, task one-hot hidden from the trunk
heads	one Gaussian head per task
optimiser, learning rate	Adam, 10^{-3}; optimiser reset at each task change
batch size	128
discount γ	0.99
entropy coefficient α	automatic, target output std 0.089
replay buffer	FIFO, 10^6, emptied at each task change; critic not reset
warm-up	10^4 random steps per task, updates from step 10^3, every 50 steps
evaluation	every 20,000 steps, 10 stochastic and 1 deterministic episodes per task
cloning term	KL on the squashed Gaussian, coefficient $\lambda = 100$ on the actor, 0 on the critic
cloning gradient clip	norm 0.1
memory per task M	10^4 states, 128 drawn per gradient step
candidate pool N	5×10^4 states, uniform from the buffer at the boundary
sketch width	32 (`csgrad`, `fl`, `beta0`; the `mr` basis has the same width); none for `nosketch`
exploration	best-return head for the first steps of each task (ClonEx); off for `bc`

B RUN MANIFEST

Table 4 lists every run family behind a number in this paper, with seeds completed against planned. A run counts as complete when all its blocks finished; incomplete runs are not averaged in. Families excluded from the claims, and why: `int` (`csgrad` with four further changes: half the budget drawn uniformly, the cloning coefficient controlled to hold the cloning gradient at 0.3 of the actor update,

the new task's head initialised from a previous head, and that head chosen by a return probe; the parts cannot be attributed separately; it is listed in Table 7 because it holds the highest forward transfer in the table) and `intmix`, that bundle's memory mix alone, listed beside it; below the ten-seed floor of §5, a pinned projection, the $h(s)$ half of the embedding and a Carathéodory rule at five seeds, GSS and k-means++ at four, the $\nabla_a Q$ half of the embedding at three seeds and 200k steps, a 200k ablation grid over memory size, sketch width, pool size, the exponents β and p and mixtures with the uniform draw at three seeds per configuration, CW20 at seven seeds and `fd`, a Frequent Directions sketch, at one; and `teach`, `oracle` and other research lines that vary something other than the memory.

Table 4: **Run manifest.** Seeds complete | planned per family.

family	budget	seeds	reported in
`clonex`, `csgrad`, `nosketch`, `fl`	CW10, 1M	20 | 20 each	Tables 2, 7
`bc`, `csobs`	CW10, 1M	10 | 20 each	Tables 2, 7
`mr`	CW10, 1M	16 | 20	Tables 2, 7
`beta0`	CW10, 1M	20 | 20	Tables 2, 7
`int`	CW10, 1M	19 | 20	Table 7, listed only
`intmix`	CW10, 1M	10 | 10	Table 7, listed only
single-task references	CW10, 1M	10 tasks $\times$ 10 runs	FT denominator, averaged per task

C WHY FORWARD TRANSFER IS COMPARED UNPAIRED

Pairing within seed lowers variance only when the two arms co-vary across the pairing variable. For P, F, plasticity, worst-task and $p10$ they do. For forward transfer they do not (Table 5): the correlation between an arm's per-seed forward transfer and the baseline's, across environment seeds, averages $r = -0.04$ over eight arms. The seed effect on forward transfer is real, about 0.09, but it is not shared between arms, so subtracting the baseline seed by seed adds its variance rather than removing it. The paired standard error exceeds the unpaired one for five of the eight arms.

Table 5: **Per-seed correlation between each arm's forward transfer and the uniform baseline's.** Computed over the environment seeds the two shared when the estimator was chosen, at the counts shown. Pairing tightens an estimate only where r is positive and large. `teach` and `int` are research lines not reported elsewhere (Appendix B); they are kept here because the correlation is a property of the estimator, not of the arm.

arm	shared seeds	r
`teach`	10	-0.53
`csgrad`	20	-0.24
`fl`	20	-0.21
`bc`	10	-0.11
`mr`	10	-0.07
`nosketch`	20	$+0.12$
`int`	19	$+0.23$
`csobs`	10	$+0.48$
mean		-0.04

D THE GRADIENT EMBEDDING AND THE CONSTRUCTIONS

The construction is in §4.3. Three implementation notes belong here. The critic gradient comes from one batched backward pass, since Q is a scalar per sample and the gradient of the sum gives each sample its own $\partial Q/\partial a$. The trunk sees only the twelve Meta-World dimensions, not the task one-hot. And the cloud is low rank: its participation ratio is 11.1 of 32 dimensions, which is why a rank-32 data-dependent basis is not obviously worse than the full 1024.

D.1 WHY A DISTANCE-BASED FAMILY, FOR A DIVERGENCE

Why look at this family at all, when the quantity rehearsal has to preserve is a divergence and not a distance? Suppose distance in the embedding bounded the cloning divergence, in the sense that $|\mathrm{KL}_\theta(s) - \mathrm{KL}_\theta(s')| \le L\|g(s) - g(s')\|$ for every θ training can reach. Then a stored set with covering radius r would pin every state's divergence to within Lr of a stored one, so a covering rule such as k-center would bound the worst case of the divergence over the pool, and a density-matched set would bound its pool-weighted average. Choosing the rule would amount to choosing between approximating a maximum and approximating a mean. There is some reason to expect a bound of this shape. The divergence between two Gaussians is a smooth function of their means and standard deviations, and SAC clamps the log standard deviation to a fixed range, so KL_θ is Lipschitz in the two policies' outputs; the outputs are Lipschitz in the state, since both policies are small multilayer perceptrons. So the bound does hold with $\|s - s'\|$ on the right, with a constant that depends on the weight norms of both networks and therefore on θ. Whether it holds with $\|g(s) - g(s')\|$ depends on the embedding. For raw observations it does. For the gradient embedding it does not in general: two states with the same trunk activation and the same critic gradient have the same $g(s)$ and hence the same current output, but the frozen policy has its own trunk and can still assign them different means, so their divergences differ while the right-hand side is zero. We therefore use the bound as motivation for reading the two rules as approximating a maximum and a mean, no more, and §7 measures what happens without it.

D.2 WHAT EACH CONSTRUCTION GUARANTEES

Every construction in §4.3 comes with a guarantee, and every guarantee is a statement about some class of functions of the embedded points. Table 6 lists, for each construction, what it guarantees and for which class. The class is never $\mathcal{Q}$.

Table 6: **What each construction guarantees, and for which functions of the points.** The last column is the class $\mathcal{F}$ in the ε-coreset condition of §3.3: the functions of the embedded points $g(s)$ that the guarantee is proved for. In every row that class is made of distances or second moments of the points. None of them is $\mathcal{Q}$, the divergences $s \mapsto \mathrm{KL}_\theta(s)$ that the memory is actually asked to preserve, so no row's guarantee transfers to the rehearsal problem.

construction	guarantee	class it is proved for
spaces (the map Π)		
Gaussian projection (`csgrad`)	Johnson–Lindenstrauss: every pairwise distance within $\pm 50\%$, but only at $\ell \approx 519$ dimensions for 5×10^4 points; we use 32, below the bound	pairwise Euclidean distances between pool points
Merge-and-reduce (`mr`)	each block replaced by its rank-32 truncated SVD before merging; deterministic; exact only for a block of rank at most 32	the second-moment matrix $X^\top X$, block by block, up to each block's truncation error
Full 1024-d (`nosketch`)	no compression, so nothing to guarantee	—
Observations (`csobs`)	none: a different input to the rule, not a compression of $g(s)$	—
rules (the procedure RULE)		
k-center	covering radius within a factor 2 of the smallest possible	the largest distance from a pool point to the set
FL sensitivity (`fl`)	the weighted set is an unbiased estimate of the clustering cost; verified over 40 repeats	sums of squared distances to a set of centres
Density, $\beta = 0$ (`beta0`)	none; the $\beta = 0$ greedy score	—

Three remarks on that table. `fl` runs the Feldman–Langberg scheme in full with the estimator of §4.3: $q(x) \propto d(x, B)^2 / \sum d^2 + 1/(|B|\, n_x)$, M i.i.d. draws, weight $1/(M\, q(x))$ per draw. The size term keeps every point's probability positive, so the weighted sum is unbiased for any loss summed over the pool, which a unit test checks over 40 repeats. We do not call its output a $(1 \pm \varepsilon)$-coreset:

the sensitivity bound is the k-means one, the first phase approximates the covering radius rather than the clustering cost, and m is fixed at M by the memory budget, so there is no ε to quote.

Second, `fl` does not store M distinct states. It produces a weighted support smaller than M and fills the M slots by sampling from it with replacement, so the memory holds repeated states. On the saved $50{,}000 \times 32$ embedding of one boundary, the same one used for the offline sweep of §7, its support has about $7{,}100$ points and the 10^4 draws land on about $4{,}500$ distinct states (three sampler seeds: $4{,}640$, $4{,}411$, $4{,}485$). The training runs did not record these counts, so they are offline measurements on one boundary, not averages over the run, and `fl`'s fee is confounded with its support: a control matched on distinct states has not been run.

Third, the Bi-Criteria construction of Appendix G was timed offline on the same embedding at $M = 10^4$: exact greedy k-center takes 1.9 seconds, Bi-Criteria 37 seconds at $k = 128$ and 564 at $k = 1{,}024$, with a covering radius $13.6\times$ larger. Bi-Criteria exists for $n = 10^6$ and $k = 10^4$, where greedy k-center's $O(ndk)$ is prohibitive; our pool is 5×10^4 points, which is why we use exact greedy.

E Full results

Table 7 is the absolute CW10 table with the published rows; Table 8 every difference with its interval; Figure 2 plots every arm on the two decisions; Figure 3 the acquisition speed; Figure 4 the offline β sweep; and Table 10 the distance-fidelity test of §7.

Table 7: **CW10, absolute, with denominators.** Seeds done | planned and blocks done | total travel with every row. Plasticity is $P + F$ by identity and is derived, never measured. The published rows are the targets for the reproduction and the reference for §6.3; our `clonex` is inside the published interval on P and its own interval overlaps the published one on FT. `mr` is sixteen of twenty seeds. `int` and `intmix` are bundles (§6.3) and are listed, not attributed.

arm	seeds	blocks	P	F	plasticity	FT	worst task	$p10$
published								
ClonEx-SAC	20 | 20	10 | 10	0.86	0.02	0.88	0.44	—	—
BC	20 | 20	10 | 10	0.84	0.02	0.86	0.41	—	—
SSDE	—	10 | 10	0.95	0.00	0.95	0.30	—	—
this paper								
`clonex`	20 | 20	10 | 10	0.865	0.006	0.872	0.416	0.398	0.687
`int`	19 | 20	10 | 10	0.869	0.025	0.894	0.470	0.548	0.686
`mr`	16 | 20	10 | 10	0.813	0.065	0.878	0.467	0.279	0.554
`intmix`	10 | 10	10 | 10	0.863	0.016	0.879	0.450	0.498	0.653
`beta0`	20 | 20	10 | 10	0.870	0.007	0.877	0.442	0.452	0.689
`nosketch`	20 | 20	10 | 10	0.848	0.033	0.882	0.439	0.432	0.621
`csgrad`	20 | 20	10 | 10	0.832	0.036	0.868	0.424	0.358	0.560
`bc`	10 | 20	10 | 10	0.831	0.032	0.863	0.379	0.314	0.539
`fl`	20 | 20	10 | 10	0.819	0.043	0.862	0.409	0.298	0.573
`csobs`	10 | 20	10 | 10	0.824	0.038	0.862	0.392	0.310	0.549

F Broader impact, at length

We measured the structure on one benchmark, but nothing in it is specific to Meta-World or to SAC. The two decisions exist wherever a system summarises what it has seen into a fixed budget and then trains against the summary: which items to keep, and in what space two items count as similar. Continual learning for language models is the largest case. The current recipe for continually pre-training a model on new data, or fine-tuning it for a new domain, is to mix a fixed fraction of the old data back into training, and that fraction is what stands between the model and forgetting (Gupta et al., 2023; Ibrahim et al., 2024; Luo et al., 2025). The replayed sample is a continual memory. It is drawn uniformly, its size is set by a mixture ratio, and the one sample is expected both to hold the old capabilities and to help the new ones. Our result suggests those are two jobs with two decisions. The hypothesis it raises is that a sample which matches the density of the original training distribution

Table 8: **Every difference against the uniform draw, with its 90% interval.** ΔP and ΔF paired within seed, ΔFT unpaired. Intervals are mean $\pm$ 1.83 standard errors; * marks an interval excluding zero. Plasticity is $P + F$ and is not listed. The widths are the precision at which "no detectable difference" is meant. Worst task and $p10$ are in Table 9.

arm	n	ΔP	ΔF	ΔFT (unpaired)
beta0	20	$+0.005$ $[-0.023, +0.032]$	$+0.000$ $[-0.017, +0.018]$	$+0.026$ $[-0.018, +0.069]$
csgrad	20	-0.033 $[-0.067, +0.000]$	$+0.030$ $[+0.010, +0.050]^*$	$+0.008$ $[-0.043, +0.059]$
fl	20	-0.046 $[-0.074, -0.019]^*$	$+0.037$ $[+0.021, +0.054]^*$	-0.007 $[-0.061, +0.046]$
mr	16	-0.054 $[-0.088, -0.020]^*$	$+0.055$ $[+0.034, +0.075]^*$	$+0.051$ $[+0.007, +0.094]^*$
nosketch	20	-0.017 $[-0.037, +0.003]$	$+0.027$ $[+0.013, +0.041]^*$	$+0.024$ $[-0.017, +0.064]$
csobs	10	-0.031 $[-0.065, +0.003]$	$+0.031$ $[+0.012, +0.050]^*$	-0.024 $[-0.083, +0.035]$
bc	10	-0.024 $[-0.074, +0.025]$	$+0.025$ $[-0.002, +0.052]$	-0.037 $[-0.081, +0.008]$
int	19	$+0.004$ $[-0.024, +0.032]$	$+0.018$ $[-0.000, +0.036]$	$+0.054$ $[+0.012, +0.097]^*$
intmix	10	$+0.009$ $[-0.029, +0.046]$	$+0.009$ $[-0.005, +0.023]$	$+0.035$ $[-0.045, +0.114]$

Table 9: **Worst task and tenth percentile against the uniform draw, with 90% intervals.** Paired within seed; same convention as Table 8.

arm	n	Δworst task	$\Delta p10$
beta0	20	$+0.054$ $[-0.117, +0.225]$	$+0.002$ $[-0.079, +0.082]$
csgrad	20	-0.040 $[-0.222, +0.142]$	-0.126 $[-0.242, -0.011]^*$
fl	20	-0.100 $[-0.258, +0.058]$	-0.114 $[-0.198, -0.031]^*$
mr	16	-0.129 $[-0.270, +0.012]$	-0.128 $[-0.241, -0.014]^*$
nosketch	20	$+0.034$ $[-0.102, +0.170]$	-0.066 $[-0.149, +0.018]$
csobs	10	-0.066 $[-0.240, +0.108]$	-0.082 $[-0.214, +0.050]$
bc	10	-0.062 $[-0.298, +0.174]$	-0.093 $[-0.235, +0.050]$
int	19	$+0.147$ $[-0.008, +0.302]$	-0.006 $[-0.089, +0.077]$
intmix	10	$+0.122$ $[-0.109, +0.353]$	$+0.021$ $[-0.099, +0.142]$

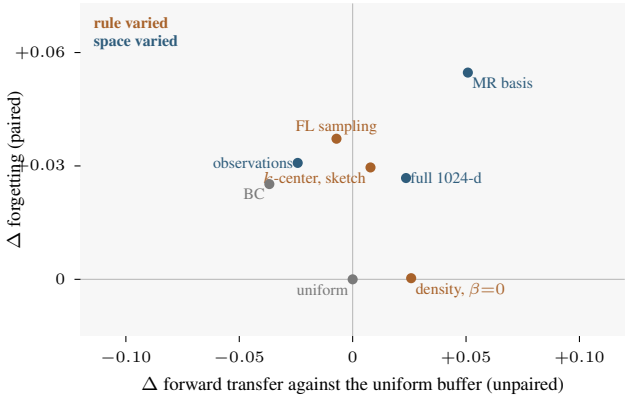


Figure 2: **Two decisions, two metrics.** Each point is one construction of the stored set with ten or more seeds, compared against the same uniform baseline. Horizontal: change in forward transfer (unpaired). Vertical: change in forgetting (paired within seed). Brown arms change the selection rule in a fixed space; blue arms change the space under a fixed rule. The arms that pay the covering fee span the whole horizontal range, and no arm lies in the lower-right quadrant. Seed counts and t statistics are in Table 2; mr is sixteen of twenty seeds.

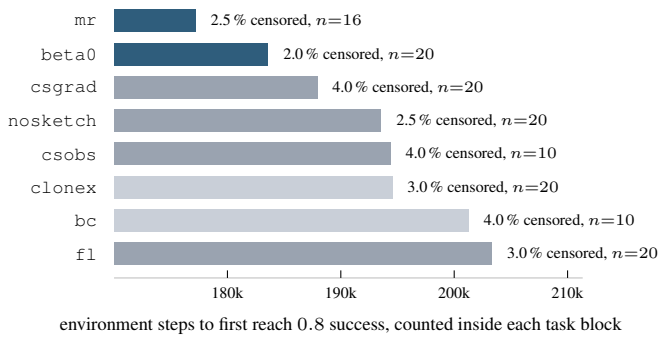


Figure 3: **Acquisition speed, with the censoring beside it.** Mean steps to the 0.8 threshold within a task block, over complete runs only. The number beside each bar is the share of task blocks that never reach it and are excluded from that mean, with the arm's seed count; the mean is therefore conditional on reaching the threshold, and the censored shares are within 2 to 4% for every arm. The merge-and-reduce basis and the density rule are the fastest, 17,000 and 11,000 steps ahead of the uniform draw. `mr` is sixteen of twenty seeds.

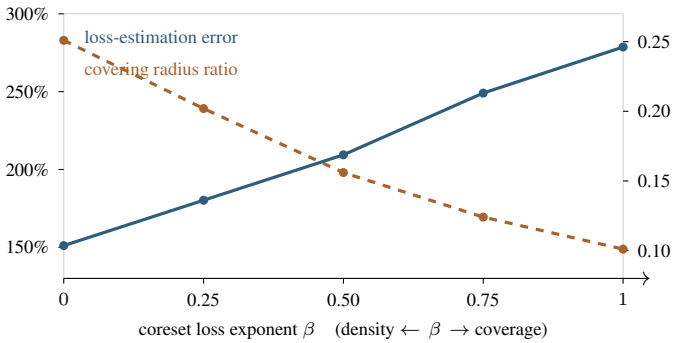


Figure 4: **The two ends of the rule, offline on one pool.** Sweeping the exponent β of $\ell_{\beta,p}(C) = \sum_c |X_c|^{-\beta} \sum_{x \in X_c} \|x - c\|^p$ across one saved 50,000 × 32 boundary pool at budget 600. Loss-estimation error is $|\ell(C) - \ell(X)|/\ell(X)$ for a quadratic loss over the pool that the rules never see; the covering radius ratio is the set's covering radius relative to the pool's. Both are properties of the procedure on this one pool, not of training. At β=1 the rule is k-center: it covers the extremes (covering radius ratio 0.10) and estimates the pool loss worst (278% error). At β=0 it keeps the density: the radius ratio rises 2.5× as the extremes are abandoned and estimation error falls 46%. The covering end is the arm that pays the forgetting fee (§6.1); the density end pays no fee we can detect. 15 repeats per configuration measure the procedure's own variability, not variability across boundaries or seeds. Table 2 is the same two ends in training.

Table 10: **Distance fidelity does not predict transfer.** Rank correlation of pairwise distances in each space with distances in the full gradient embedding, on 1,200 points from the 50,000 × 32 pool, beside each space's unpaired change in forward transfer at 1M (Table 2). The merge-and-reduce row is a proxy, the top singular directions of the already-sketched 32-dimensional embedding, where the arm itself runs merge-and-reduce on the full 1024 dimensions, so this row is not a clean diagnostic of the arm that produced the transfer result. The gradient cloud's participation ratio is 11.1 of 32 dimensions.

space	distance fidelity	ΔFT (unpaired)
merge-and-reduce basis (`mr`, proxy)	0.970	+0.051
full embedding (`nosketch`)	1.000	+0.024
random projection (`csgrad`)	0.970	+0.008
raw observations (`csobs`)	0.471	−0.024

protects the old capabilities, that how much the sample helps the new domain depends on the space it was chosen in, and that the gradient-based data-selection methods now used for instruction tuning (Xia et al., 2024; Killamsetty et al., 2021; Mirzasoleiman et al., 2020; Axiotis et al., 2024) act on the second decision and not the first.

Alignment from interaction data has the same shape. In self-distillation from user interactions (Kleine Buening et al., 2026, SDPO), the model conditioned on a user's follow-up message is the teacher, the model without that context is the student, and a distillation loss is applied over a stored set of conversations. That stored set is a continual memory, and the two decisions are which conversations to distil on and in what representation conversations are compared when the set is built. The teacher there moves with the student, since both are the same network, so the query the memory must serve is the moving divergence of §4.1 with the frozen side replaced by an in-context side. A coreset for that query is the same open problem we state for continual RL.

In-context learning makes the memory explicit. The context window is a bounded memory filled afresh for each query, the demonstrations placed in it are chosen by a rule, and they are retrieved in an embedding space. Retrieval by similarity chooses both at once, exactly as coreset selection did in §2. Our result predicts that the embedding sets how much the demonstrations help the query at hand, and that a demonstration set matching the density of the task distribution keeps the model's behaviour on the rest of that distribution intact. We state each of these as a hypothesis for its setting, with the same test in every case: vary the selection rule at a fixed embedding and read retention, then vary the embedding at a fixed rule and read learning speed on the new task.

Two things do not carry over on their own. Our fee mechanism is memorisation of a fixed set by a small trunk, and a model with billions of parameters may or may not memorise a replay sample of the sizes used in practice. Our transfer gate is the actor's gradient geometry; the analogue for a language model is the per-example gradient at its output layer, which is the object the influence-based selection methods already compute, so the test needs no new machinery. We have not run it.

For continual RL the result is a design rule with a stated scope: under the covering rule, the space the set is chosen in changes transfer; in the sketch space, a density-matched rule avoids the retention cost coverage pays. For the coreset literature it is a query class, $\mathcal{Q}$, defined by an optimisation trajectory rather than by a finite set, for which no standard construction is proved. The next experiments are `mr` at twenty seeds, the density rule in observation space and in the merge-and-reduce basis (§8), the ablations of memory size, sketch width, pool size and sequence length at ten seeds, the collaborators' $p = 6$ configuration at 1M, and a leverage-score construction derived from $\mathcal{Q}$ rather than borrowed.

G CORESET BASED APPROXIMATION

This appendix gives the construction and guarantee behind the loss family of §3.3, for the coverage end of that family: with $f = 1$ and $\mathrm{Dist} = \|\cdot\|_2^6$ the loss below is $\ell_{0,6}$, which Lemma 14 places within a factor $|P|^{1/6}$ of the k-center objective. The guarantee is a statement about distances in the embedding, not about the class $\mathcal{Q}$ of §4.1. Notation here is self-contained and differs from the main text: P is a point set (not performance), k the number of centres (the budget M of the main text), and β in "(α, β)-approximation" is the bi-criteria factor, unrelated to the loss exponent β of §3.3.

We generalize previous work on imbalanced clustering (He & Garcia, 2009), to obtain a fast approximation, at the price of more centroids returned. The worst-case analysis returns significantly more than k points, exclusively because of the loose bound on the sample size in Lemma 12; we believe that a significantly lower bound can be obtained, and in practice we set a lower sample size than the theory requires, which yields $O(k \log(n))$ points.

The construction and analysis closely follow the structure of He & Garcia (2009). We adapt their arguments to a more general setting and provide a complete proof of the resulting guarantee, which is not covered by their result.

Throughout, the statements are for $f \equiv 1$, the case used in §3.3 and by our implementation. The extension to a general non-decreasing f is not claimed: the pigeonhole step in the proof of Theorem 13 uses $f \equiv 1$, and f is kept in the notation only to mark where a general f would enter.

Notations. Throughout this section, we assume $k, d \geq 1$ are integers. Let $f : [1, \infty) \to [0, \infty)$ be a non-decreasing non-negative function, taken to be $f \equiv 1$ in every statement below; in He & Garcia (2009) the linear function was used. Let $\rho \geq 1$ and $\text{Dist} : \mathbb{R}^d \times \mathbb{R}^d \to [0, \infty)$ be a distance function (semi-metric) satisfying for every $a, b, c \in \mathbb{R}^d$ that $\text{Dist}(a, c) \leq \rho\text{Dist}(a, b) + \rho\text{Dist}(b, c)$, $\text{Dist}(a, b) = \text{Dist}(b, a)$, $\text{Dist}(a, a) = 0$, and $\text{Dist}(a, b) = 0$ only if $a = b$. Suppose there is, $t_d \geq d$, such that for every $a, b \in \mathbb{R}^d$ we can compute $\text{Dist}(a, b)$ with at most t_d arithmetic operations $+, -, \times, /$ on real numbers, and jumps conditioned on $>, \leq, <, \geq, =, \neq$ comparisons of real numbers. For simplicity, we denote $\log(x) := \log_2(x)$ and assume that all the input sets are finite and non-empty. For every integer $n \geq 1$, we denote $[n] = \{1, \ldots, n\}$.

Let a pair of sets $P, C \subset \mathbb{R}^d$, for every $c \in C$, we define the set P_c as the points in P that are closest to c from all of C (ties broken arbitrarily). Throughout this section, we define $M(P, C) := \{P_c\}_{c \in C}$ as a partition of P, for every such $P, C \subset \mathbb{R}^d$.

We define the loss function as follows. Note that we define all the loss functions for general sets $C \subset \mathbb{R}^d$, but aim to minimize the loss over such sets of size at most k.

Definition 1 (Loss function)**.** *Let $P \subset \mathbb{R}^d$ be a set of points. We define its fitting loss to a set $C \subset \mathbb{R}^d$, as*

$$\ell\big(P, C\big) = \sum_{c \in C, |P_c| \geq 1} \frac{1}{f(|P_c|)} \sum_{p \in P_c} \text{Dist}(p, c), \tag{1}$$

where $\{P_c\}_{c \in C} := M(P, C)$. Our goal is to minimize $\ell\big(P, C\big)$ over every set C of $|C| = k$ centers.

We define the following desired approximation. Note, as in He & Garcia (2009), that this approximation also includes a corresponding partition, unlike commonly in the coreset literature (Feldman, 2020). This is because assigning each point to its closest cluster in Definition 1 does not necessarily yield the lowest loss (consider two clusters where all the points in the cluster are close to arbitrary precision, and we introduce a new point), which is not the common case in the coreset literature (Feldman, 2020).

Definition 2 ((α, β)-approximation)**.** *Let $\alpha \geq 0$, and $\beta \geq 1$ be an integer. Let $P \subset \mathbb{R}^d$ be a finite non-empty set. A pair $(B, \mathcal{P})$ is an (α, β)-approximation of P, if $B := \{c_i\}_{i=1}^{k\beta} \subset \mathbb{R}^d$ is a set of size $k\beta$, and $\mathcal{P} := \{P_i\}_{i=1}^{k\beta}$ is a partition of P into $k\beta$ sets, such that*

$$\sum_{i=1}^{k\beta} \frac{1}{f(|P_i|)} \sum_{p \in P_i} \text{Dist}(p, c_i) \leq \alpha \cdot \min_{C \subset \mathbb{R}^d, |C| = k} \ell(P, C). \tag{2}$$

Remark: when $\mathcal{P} = M(P, B)$, the left-hand side of equation 2 simplifies to $\ell(P, B)$.

G.1 Bound on the VC-dimension

For the method we need the following bound on the dimension/complexity of the problem. We first define the classic notion of VC-dimension, which is used in Theorem 2.3 of Bartlett & Maass (2003), and is usually related to the PAC-learning theory (Li et al., 2001). The following definition is from Lucic et al. (2017).

Definition 3. *Let F be a set of functions from $\mathbb{R}^d$ to $\{0, 1\}$ (i.e., classifiers) and let $X \subset \mathbb{R}^d$. Fix a set $S = \{x_1, \cdots, x_n\} \subset X$ and a function $f \in F$. We call $S_f = \{x_i \in S \mid f(x_i) = 1\}$ the induced subset of S by f. A subset $S = \{x_1, \cdots, x_n\}$ of X is shattered by F if $|\{S_f \mid f \in F\}| = 2^n$. The VC-dimension of F is the size of the largest subset of X shattered by F.*

Theorem 4 (Based on Theorem 2.3 of Bartlett & Maass (2003))**.** *Let h be a function from $\mathbb{R}^d \times \mathbb{R}^n$ to $\{0, 1\}$, and let*

$$\mathcal{H} = \{h_\theta : \mathbb{R}^n \to \{0, 1\} \mid \theta \in \mathbb{R}^d\}. \tag{3}$$

Suppose that h can be computed by an algorithm that takes as input the pair $(\theta, x) \in \mathbb{R}^d \times \mathbb{R}^n$ and returns $h_\theta(x)$ after no more than t of the following operations:

- *the arithmetic operations $+, -, \times$, and $/$ on real numbers,*
- *jumps conditioned on $>, \leq, <, \geq, =$, and $\neq$ comparisons of real numbers, and*

- *output 0, 1.*

Then the VC-dimension of $\mathcal{H}$ is at most $4d(t+2) \in O(dt)$.

For the method we utilize the following generalization of the previous definition of VC-dimension. This is commonly referred to as VC-dimension, but to differentiate this definition from the previous, and to be in line with the notations in (Feldman et al., 2019), we abbreviate it to dimension.

Definition 5 (range space of functions (Feldman et al., 2013; Har-Peled & Sharir, 2011; Feldman & Langberg, 2011))**.** *Let F be a finite set of functions from a set $\mathcal{Q}$ to $[0,\infty)$. For every $Q \in \mathcal{Q}$ and $r \geq 0$, let* $\text{range}(F,Q,r) = \{f \in F \mid f(Q) \geq r\}$.
Let $\text{ranges}(F) = \{\text{range}(F,Q,r) \mid Q \in \mathcal{Q}, r \geq 0\}$. *Let $\mathfrak{R}_{\mathcal{Q},F} = (F, \text{ranges}(F))$ be the range space induced by $\mathcal{Q}$ and F.*

In the following lemma, which is inspired by Theorem 12 in Lucic et al. (2017), we bound the VC-dimension that would be used to obtain the approximation to ℓ.

Lemma 6. *Let $\{p_1, \cdots, p_n\} := P \subset \mathbb{R}^d$, and let $\mathcal{Q} := [\mathbb{R}^d]^k$ be the family of all subsets of size k from $\mathbb{R}^d$. For every $i \in [n]$, let $f_i : \mathcal{Q} \to [0,\infty)$ denote the function that maps every $C \in \mathcal{Q}$ to $f_i(C) := \frac{1}{f(|P_c''|)} \min_{c \in C} \text{Dist}(p_i, c)$, where $\{P_c''\}_{c \in C} := M(P,C)$ and $p_i \in P_c''$. Let $F := \{f_1, \ldots, f_n\}$. We have that the dimension of the range space $\mathfrak{R}_{\mathcal{Q},F}$ that is induced by $\mathcal{Q}$ and F is in $O(k^2 dt_d)$.*

Proof. We introduce a weighted version (Q', F') to simplify the VC-dimension analysis. Let $\mathcal{Q}'$ be the family of all the weighted sets of size k. For every $i \in [n]$, let $f_i' : \mathcal{Q}' \to [0,\infty)$ denote the function that maps every $C \in \mathcal{Q}'$ to $f_i'(C) := \min_{c \in C} w(c)\text{Dist}(p_i, c)$. Let $F' := \{f_1', \ldots, f_n'\}$. For $f \equiv 1$ every f_i is the member of F' with all weights equal to one, so F embeds in F' and the dimension of the range space $\mathbb{R}_{\mathcal{Q},F}$ is upper bounded by the dimension of the range space $\mathfrak{R}_{\mathcal{Q}',F'}$. Hence, from now on, we prove that the dimension of the range space $\mathbb{R}_{\mathcal{Q}',F'}$ is in $O(k^2 dt_d)$.

Let $f' : \mathcal{Q}' \to \mathbb{R}^{k(d+1)}$ be the function that maps every $(C,w) \in \mathcal{Q}', \{c_i\}_{i=1}^d := C$ to $f(C,w) := \{(c_1|c_2|\cdots|c_d|w(c)\}_{c \in C}$. For every $(q,r) = \big((C,w),r\big) \in \mathcal{Q}' \times \mathbb{R}$, let $h_{(f(C,w)|r)} : \mathbb{R} \to \{0,1\}$ that maps every $p_i \in P$ to $h_{(f(C,w)|r)}(x) = 1$ if and only if $\min_{c \in C} w(c)\text{Dist}(p_i, c) \leq r$, and every $p \in \mathbb{R}^d \setminus P$ to $h_{(f(C,w)|r)}(x) := 0$. Let $\mathcal{H} = \{h_\theta \mid \theta \in \mathbb{R}^{k(d+1)+1}\}$. For every $(C,w) \in \mathcal{Q}$, and $p_i \in P$, we can calculate $\min_{c \in C} w(c)\text{Dist}(p_i, c)$ with $O\big(kt_d\big)$ arithmetic operations on real numbers and jumps conditioned on comparisons of real numbers.

Therefore, for every $p_i \in P$ and any $\theta \in \mathbb{R}^{k(d+1)+1}$ we can calculate $h_\theta(i)$ with $O(kt_d)$ arithmetic operations on real numbers and jumps conditioned on comparisons of real numbers. Hence, substituting $d := k(d+1)+1$, $n := d$, $h := h$, $\mathcal{H} := \mathcal{H}$, and $t \in O(kt_d)$ in Theorem 4 yields that the VC-dimension of $\mathcal{H}$ is in $O(k^2 dt_d)$. Hence, by the construction of $\mathcal{H}$ and the definition of range spaces in Definition 5, we have that the dimension of the range space $\mathbb{R}_{\mathcal{Q}',F'}$ that is induced by $\mathcal{Q}'$ and F' is in $O(k^2 dt_d)$. □

G.2 Robust median

In the following section, we will define the robust median and prove that it can be computed efficiently. This consists of two methods: the initial method and its speed-up. The initial method involves selecting the best center from a uniform sample of points and is inspired by (Feldman & Schulman, 2012). The speed-up entails computing the previous robust median on a uniform sample and is based on Section *From ϵ-approximation to robust medians* of Feldman & Langberg (2011).

Initial method. For the statement, we utilize the following definitions.

Definition 7 ((Feldman & Schulman, 2012))**.** *Let $P \subset \mathbb{R}^d$ be a non-empty finite set of points. For $q \in \mathbb{R}^d$ and $\gamma \in [0,1]$, we denote by* CLOSEST(P,q,γ) *the set that consists of the $\lceil \gamma |P| \rceil$ points $p \in P$ with the smallest values of* $\text{Dist}(p,q)$*, ties broken arbitrarily.*

Definition 8 ((Feldman & Schulman, 2012)). *For $\gamma \in [0,1]$, $\epsilon \in (0,1)$, and $\alpha > 0$, the point $q \in \mathbb{R}^d$ is a $(\gamma, \epsilon, \alpha)$-median of the non-empty finite set $Q \subset \mathbb{R}^d$ if*

$$\sum_{p \in \text{CLOSEST}(Q,q,(1-\epsilon)\gamma)} \text{Dist}(p,q) \leq \alpha \cdot \inf_{q' \in \mathbb{R}^d} \sum_{p \in \text{CLOSEST}(Q,q',\gamma)} \text{Dist}(p,q'). \tag{4}$$

Utilizing these definitions, we can state the main result of our initial method for computing the robust median.

Lemma 9. *Let $P \subset \mathbb{R}^d$ be a finite and non-empty set. There is $q \in P$, which is a $(15/(16k), 1/16, 2\rho)$-median of P.*

Proof. Let $q^* \in \mathbb{R}^d$ be a $(1/k, 0, 1)$-median of P. Let $q \in \arg\min_{p \in P} \text{Dist}(q^*, p)$. From the definition of Dist, for every $p \in P$, we have

$$\text{Dist}(p,q) \leq \rho \text{Dist}(p,q^*) + \rho \text{Dist}(q^*,q) \leq 2\rho \text{Dist}(p,q^*). \tag{5}$$

Let $C^* := \text{CLOSEST}(P, q^*, 15/16k)$. Summing this over every $p \in \text{CLOSEST}(P, q, 15/(16k))$ yields

$$\sum_{p \in \text{CLOSEST}(P,q,15/(16k))} \text{Dist}(p,q) \leq \sum_{p \in C^*} \text{Dist}(p,q) \leq 2\rho \sum_{p \in C^*} \text{Dist}(p,q^*). \tag{6}$$

Hence, q is a $(15/(16k)), 0, 2\rho)$-median of P, which is also a $(15/(16k), 1/16, 2\rho)$-median of P. □

Hence, as suggested by Lemma 9, our initial method for computing a robust median for a set involves an exhaustive search over all the points in the set.

Speed up. For our speedup, we utilize the following notation of function space from Feldman & Langberg (2011) (we have used the previous definition of dimension of the range space, instead of restating it as in the original definition).

Definition 10 (Function space, based on Feldman & Langberg (2011)). *Let F be a finite set of functions from a set X to $[0, \infty)$. Let $\mathcal{X}$ be a function that maps every subset $S \subseteq F$ to a set of items $\mathcal{X}(S) \subseteq X$. The pair $(F, \mathcal{X})$ is called a generalized function space if for any $S \subseteq S'$, it holds that $\mathcal{X}(S) \subseteq \mathcal{X}(S')$. The dimension of $(F, \mathcal{X})$ is the dimension of the range space $\Re_{X,F}$ that is induced by X and F.*

Lemma 4.6 of Feldman & Langberg (2011) states the following.

Lemma 11. *Let $(F, \mathcal{X})$ be a function space of dimension d. Let $\gamma \in (0,1], \epsilon \in (0, 1/10), \delta \in (0, 1/10)$, and $\alpha > 0$. Let S be a random sample of $s = \frac{c}{\epsilon^4 \gamma^2}(d + \log(1/\delta))$, i.i.d. functions from F, where c is a sufficiently large constant. Suppose that $x \in \mathcal{X}(S)$ is a $((1-\epsilon)\gamma, \epsilon, \alpha)$-median of S, and that $|F| \geq s$. Then, with probability at least $1 - \delta$, x is a $(\gamma, 4\epsilon, \alpha)$-median of F.*

We substitute in Lemma 11 the following:

- $\epsilon := 1/16, \gamma := 1/k, \alpha = 2\rho$.
- $X = P$, for some $\{p_1, \cdots, p_n\} := P \subset \mathbb{R}^d$, where $n \geq 1$.
- $F := \{f_i\}_{i=1}^n$ as stated in Lemma 6 for P. That is, let $\mathcal{Q} := [\mathbb{R}^d]^k$ be the family of all subsets of size k from $\mathbb{R}^d$. For every $i \in [n]$, let $f_i : \mathcal{Q} \to [0, \infty)$ denote the function that maps every $C \in \mathcal{Q}$ to $f_i(C) = \frac{1}{f(|P_c''|)} \min_{c \in C} \text{Dist}(p_i, c)$, where $\{P_c''\}_{c \in C} := M(P, C)$ and $p_i \in P_c''$.
- $d \in O(k^2 d t_d)$ the dimension of the range space $\Re_{X,F}$ as follows from Lemma 6.
- $\mathcal{X} : F \to P$ that maps every f_i to $\mathcal{X}(f_i) := p_i$.

This yields the following version for our case.

Algorithm 1 BI-CRITERIA(P, λ); see Theorem 13

Require: A finite set $P \subset \mathbb{R}^d$ of size $n \geq 1$, an integer $\lambda \geq 1$

Ensure: A pair $(B, \mathcal{P})$, where $B \subset \mathbb{R}^d$ and $\mathcal{P}$ is a partition of P, which, for sufficiently large $\lambda \geq 1$ (see Theorem 13) with corresponding $\delta \in (0,1)$, with probability at least $1-\delta$, is a $\big(O(k \log n), O(k \log n) + \lambda\big)$-approximation to P.

$P' \leftarrow P$
$B \leftarrow \emptyset$
$\mathcal{P} \leftarrow \emptyset$
while $|P'| \geq 2\lambda$ **do**
    Pick a sample S of size λ from P', where each element in S is sampled i.i.d. and uniformly at random from P'.
    for $p \in S$ **do**
        Set S_p as the subset of S of the $\left\lfloor \frac{15|S|}{16k} \right\rfloor$ closest (under Dist; for Dist $= \|\cdot\|_2^6$ this is the Euclidean order) points to p, ties broken arbitrarily.
        $\ell_p \leftarrow \ell(S_p, \{p\})$
    Let $p \in \arg\min_{p \in S} \ell_p$.
    Set P'_p as the subset of P' of the $\left\lfloor \frac{3|P'|}{4k} \right\rfloor$ closest (with distance function Dist) points to p, ties broken arbitrarily.
    $B \leftarrow B \cup \{p\}$
    $P' \leftarrow P' \setminus P'_p$
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{P'_p\}$
$B \leftarrow B \cup P'$
$\mathcal{P} \leftarrow \mathcal{P} \cup \{\{p\} \mid p \in P'\}$ ▷ Put each point in P' in a partition containing only it.
return $(B, \mathcal{P})$

Lemma 12. *Let $P \subset \mathbb{R}^d$ be a finite and non-empty set of points. Let $\delta \in (0, 1/10]$ and put*

$$s := k^2\big(k^2 d t_d + \log(1/\delta)\big).$$

Let S be a uniform random sample of i.i.d. points from P, of size $\lambda := cs \in O\big(k^4 d t_d \log(1/\delta)\big)$, for a universal constant c that can be determined from the proof. With probability at least $1-\delta$, any $(15/(16k), 1/16, 2\rho)$-median of S is a $(1/k, 1/4, 2\rho)$-median of P.

G.3 THE APPROXIMATION

The input for the following algorithm is a set $P \subset \mathbb{R}^d$ of size n, along with an integer $k \geq 1$, and an upper limit $\delta \in (0, 1/10]$ on the allowed failure probability. Its output is a set $B \subset \mathbb{R}^d$ along a partition of P, which together are an $\big(O(k \log n), O(k \log n)\big)$-approximation of P with probability at least $1-\delta$.

While the algorithm may initially appear similar to the (α, β)-approximation of Feldman & Langberg (2011), it does not involve solving the original problem on the sampled subset (we use a single point, rather than k points, as the approximation for the sample). This modification enables a non-exponential dependency on k, unlike similar algorithms for k-means of Feldman & Langberg (2011). It should be emphasized that there are approximation algorithms for k-means that have non-exponential dependency on k, notably (Arthur & Vassilvitskii, 2007).

The following theorem states and proves the desired properties of Algorithm 1.

Theorem 13. *Let $P \subset \mathbb{R}^d$ be a set of size $n \geq 1$. Let $\delta \in (0, 1/10]$, let $k \geq 1$ be an integer, and let $\lambda := ck^4 d t_d \log(n/\delta)$ where c is as defined in Lemma 12, when substituting $\delta := \delta/n$. Let $(B, \mathcal{P})$ be the output of a call to* BI-CRITERIA(P, λ)*; see Algorithm 1. Then, with probability at least $1-\delta$, we have that $(B, \mathcal{P})$ is a $\big(O(k \log n), O(k \log n) + \lambda\big)$-approximation to P. Moreover, the computation time of B is in $O\big(dk(n + \lambda^2 \log n)\big)$.*

Proof. Consider a single iteration of the "while" loop in the call to BI-CRITERIA(P, λ).

By the choice of λ according to Lemma 12, with probability at least $1-\delta/n$, any $(15/16k, 1/16, 2\rho)$-median of S is a $(1/k, 1/4, 2\rho)$-median of P'. Assume that this event indeed occurs.

By Lemma 9, there is $p \in S$ which is a $(15/16k, 1/16, 2\rho)$-median of S. Hence, by the definition of $p \in S$ in the current "while" iteration it follows that p is a $(15/16k, 1/16, 2\rho)$-median of S. Thus, we have that p is a $(1/k, 1/4, 2\rho)$-median of P'.

Let P'_p as computed in this "while" iteration. We have that

$$\sum_{p' \in P'_p} \mathrm{Dist}(p', p) \leq 2\rho \min_{q^* \in \mathbb{R}^d} \sum_{p' \in \textsc{closest}(P', q^*, 1/k)} \mathrm{Dist}(p', q^*). \tag{7}$$

Observe that due to the pigeonhole principle, for any k centers there is a center whose cluster holds at least $|P'|/k$ points, and the summed distance of those points to their center is at least the minimum over q of the summed distance of the $\lceil |P'|/k \rceil$ points closest to q. As $f \equiv 1$, this yields

$$\frac{1}{f(|P'|/k)} \cdot \min_{q^* \in \mathbb{R}^d} \sum_{p' \in \textsc{closest}(P, q^*, 1/k)} \mathrm{Dist}(p', q^*) \leq \min_{C \subset \mathbb{R}^d, |C|=k} \ell(P', C). \tag{8}$$

Hence, we have that

$$\ell(P'_p, \{p\}) \leq 2\rho \min_{C \subset \mathbb{R}^d, |C|=k} \ell(P', C). \tag{9}$$

Observe that at each iteration of the "while" loop in Algorithm 1, the size of P' decreases by a multiplicative factor of $1 - 3/(4k) < 1 - 1/(2k)$. Hence, every $1/2k$ iterations, the size of P' decreases by a multiplicative factor of $(1 - 1/(2k))^{2k}$. Let $f : \mathbb{R} \to \mathbb{R}$ be the function that maps every $x \in \mathbb{R}$ to $f(x) := (1 - 1/x)^x$. A known property of the function is that for every $x \geq 1$, we have that $f(x) \leq 1/e$. Hence, due to k being a positive integer we have that $(1 - 1/(2k))^{2k} \leq 1/e$. As such, for every $2k$ iteration of the "while" loop, the size of P' decreases by a constant multiplicative factor. Hence, we have that there are $O\big(k \log n\big)$ iterations in the "while" loop.

Let B and $\mathcal{P}$ be as computed at the end of the "while" loop (Line 14 of Algorithm 1). Let $\{c_i\}_{i=1}^{|B|}$ and $\{P_i\}_{i=1}^{|B|}$ label the elements of B and $\mathcal{P}$ respectively. Since there are $O(k \log n)$ iterations of the "while" loop, we have

$$\sum_{i=1}^{|B|} \frac{1}{f(|P_i|)} \sum_{p \in P_i} \mathrm{Dist}(p, c_i) \in O(k \log n) \min_{C \subset \mathbb{R}^d, |C|=k} \ell(P, C). \tag{10}$$

Thus since the algorithm returns $(B, \mathcal{P})$, where $B := B \cup P'$ and $\mathcal{P} := \mathcal{P} \cup \{\{p\} \mid p \in P'\}$, it follows that $(B, \mathcal{P})$ is a $\big(O(k \log n), O(k \log n) + \lambda\big)$-approximation to P.

We assumed, that at each iteration of the "while" loop, there is $q \in S$ which is a $(15/16k, 1/16, 2\rho)$-median of S and any such median is also a $(1/k, 1/4, 2\rho)$-median of P'. This event occurs with a probability of at least $1 - \delta/n$. Hence, by the union bound and that $\delta \leq 1/10$, we have that all the events occur with probability at least $\left(1 - \dfrac{\delta}{n}\right)^n \leq 1 - \delta$.

Thus, with probability at least $1-\delta$, we have that B is a $\big(O(k \log n), O(k \log n)+\lambda\big)$-approximation to P.

Running time. Consider a single iteration of the "while" loop in the call to BI-CRITERIA(P, λ).

It immediately follows that every "for" iteration can be done in $O(\lambda d)$; note that the choice of S_p does not require sorting S; a linear-time selection algorithm suffices (Hoare, 1961). Hence, since there are λ iterations, it immediately follows that Lines 4–9 can be computed in $O\big(\lambda^2 d\big)$ time.

Let η be the size of P' in the "while" iteration considered. It can be seen that all the computations in Lines 10–13 can be done in $O\big(\eta d\big)$ time; note that the choice of P'_p does not require sorting P'; a linear-time selection algorithm suffices (Hoare, 1961).

Hence, each "while" iteration in the call to BI-CRITERIA(P, k, δ), where $n' := |P'|$, can be computed in $O\big(\eta d + \lambda^2 d\big)$ time.

Hence, since there are $O\big(k \log n\big)$ iterations in the "while" loop, and in each iteration the size of P' decreases by a factor of $\left(1 - \frac{1}{2k}\right)$, i.e., we have a geometric sum, the total time of the "while" loop is in

$$\frac{O\big(nd\big)}{1 - \left(1 - \frac{1}{2k}\right)} + O\big(k \log n\big) \cdot \lambda^2 d. \tag{11}$$

Hence, the total time of the "while" loop is in $O\big(dk(n + \lambda^2 \log n)\big)$. By the construction of Algorithm 1, the computation time is dominated by the computation time of the "while" loop. Thus, the call to BI-CRITERIA(P, λ) can be computed in $O\big(dk(n + \lambda^2 \log n)\big)$ time. □

G.4 UTILIZATION OF ALGORITHM 1

In this section we show how Algorithm 1 can be used to efficiently approximate the k-center objective, the coverage end of the family of §3.3: given a finite, nonempty set $P \subseteq \mathbb{R}^d$, find (or approximate) a set $C \subseteq \mathbb{R}^d$, with $|C| = k$, minimizing

$$\ell_\infty(P, C) := \max_{p \in P} \min_{c \in C} \|p - c\|_2.$$

To this end, we set $f(x) := 1$ for every $x \in [1, \infty)$, and define $\mathrm{Dist}(a, b) := \|a - b\|_2^6$ for every $a, b \in \mathbb{R}^d$. This gives $\rho = 2^{6-1} = 32$.

Observe that the choice of constant f brings into consideration other algorithms, with similar running time as González's farthest-point algorithm (Gonzalez, 1985), e.g., (Arthur & Vassilvitskii, 2007). Nonetheless, those alternatives suffer from the same prohibitively slow time for our parameter choice that motivated this alternative. The choice of f does not affect the size of the resulting set, but the guarantee below is for $f \equiv 1$ only; a different f applied after construction carries no guarantee from this section.

Let ℓ' denote the loss resulting from these choices. The following lemma relates ℓ' and ℓ_∞.

Lemma 14. *Let $P \subseteq \mathbb{R}^d$ be finite and nonempty, and let $C \subseteq \mathbb{R}^d$ satisfy $|C| = k$. Then*

$$\ell_\infty(P, C) \le \ell'(P, C)^{1/6} \le |P|^{1/6} \ell_\infty(P, C).$$

In particular, if $|P| = 10^6$, then

$$\ell_\infty(P, C) \le \ell'(P, C)^{1/6} \le 10 \ell_\infty(P, C).$$

Proof. Let $(P_c)_{c \in C} := M(P, C)$. By the definitions of f and Dist,

$$\ell'(P, C) = \sum_{\substack{c \in C \\ |P_c| \ge 1}} \sum_{p \in P_c} \|p - c\|_2^6 = \sum_{p \in P} \min_{c \in C} \|p - c\|_2^6.$$

For each $p \in P$, write $r_p := \min_{c \in C} \|p - c\|_2$. Since the map $x \mapsto x^6$ is increasing on $[0, \infty)$, we have $\ell'(P, C) = \sum_{p \in P} r_p^6$. For the upper bound,

$$\ell'(P, C) = \sum_{p \in P} r_p^6 \le |P| \max_{p \in P} r_p^6 = |P| \ell_\infty(P, C)^6.$$

Taking sixth roots gives $\ell'(P, C)^{1/6} \le |P|^{1/6} \ell_\infty(P, C)$.

For the lower bound,

$$\ell'(P, C) = \sum_{p \in P} r_p^6 \ge \max_{p \in P} r_p^6 = \ell_\infty(P, C)^6.$$

Taking sixth roots completes the proof. □

For our approximation we therefore use the following two-step procedure. Given $P \subset \mathbb{R}^d$ we run González's farthest-point algorithm (Gonzalez, 1985), to obtain 1024 points denoted by C_1. We apply Algorithm 1, with $P := P \setminus C_1$, $\lambda := 1024$ and $k := 128$ ($\lambda = 8k$), and denote its output by C_2. We return $C := C_1 \cup C_2$.