Appendix

A Hyperparameter Selection

For the arm and linear projection domains we mirror the hyperparameter selections from previous work [61, 19] and tuned manually the hyperparameters of our newly proposed algorithms: OG-MAP-Elites, OG-MAP-Elites (line), OMG-MEGA, and CMA-MEGA. For the latent space illumination domain, since the domain is new, we manually tuned each algorithm so that the learning rate was as small as possible while still have perceptible differences in the first 10 iterations of each algorithm. We chose small learning rates because for large learning rates, the search moved far away from the training distribution, resulting in unrealistic images. We report the parameters of each algorithm below. In all algorithms we used a batch size $\lambda = 36$ following previous work [19]. MAP-Elites and the algorithms derived from MAP-Elites (MAP-Elites (line), OG-MAP-Elites, OG-MAP-Elites (line), OMG-MEGA) had an initial population size of 100, sampled from a distribution $\mathcal{N}(\mathbf{0}, I)$. Initial solutions used to seed the initial archive do not count as an iteration in our experiments. We set the initial search position $\boldsymbol{\theta_0} = \mathbf{0}$ for CMA-ME, CMA-MEGA and CMA-MEGA (Adam) in all domains.

Linear Projection (Sphere, Rastrigin).

- MAP-Elites: $\sigma = 0.5$
- MAP-Elites (line): $\sigma_1 = 0.5$, $\sigma_2 = 0.2$
- CMA-ME: $\sigma = 0.5$
- OG-MAP-Elites: $\sigma = 0.5$, $\eta = 0.5$
- OG-MAP-Elites (line): $\sigma_1 = 0.5$, $\sigma_2 = 0.2$, $\eta = 0.5$
- OMG-MEGA: $\sigma_g = 10.0$
- CMA-MEGA: $\sigma_g = 10.0$, $\eta = 1.0$
- CMA-MEGA (Adam): $\sigma_g = 10.0$, $\eta = 0.002$

Arm Repertoire.

- MAP-Elites: $\sigma = 0.1$
- MAP-Elites (line): $\sigma_1 = 0.1$, $\sigma_2 = 0.2$
- CMA-ME: $\sigma = 0.2$
- OG-MAP-Elites: $\sigma = 0.1, \eta = 100$
- OG-MAP-Elites (line): $\sigma_1 = 0.1$, $\sigma_2 = 0.2$, $\eta = 100$
- OMG-MEGA: $\sigma_g = 1.0$
- CMA-MEGA: $\sigma_g = 0.05$, $\eta = 1.0$
- CMA-MEGA (Adam): $\sigma_g = 0.05$, $\eta = 0.002$

Latent Space Illumination of StyleGAN guided by CLIP.

- MAP-Elites: $\sigma = 0.2$
- MAP-Elites (line): $\sigma_1 = 0.1$, $\sigma_2 = 0.2$
- CMA-ME: $\sigma = 0.02$
- CMA-MEGA: $\sigma_g = 0.002$, $\eta = 1.0$
- CMA-MEGA (Adam): $\sigma_g = 0.002$, $\eta = 0.002$

Adam Hyperparameters. We use the same hyperparameters as the StyleGAN+CLIP implementation [48]. We configure Adam with the same hyperparameters for each domain.

- $\beta_1 = 0.9$
- $\beta_2 = 0.999$

B Domain Details.

Linear Projection. We use the linear projection domains sphere and Rastrigin from previous work [19], and selected $n = 1000$ for both domains. Each domain has a different objective function (Eq. 4, Eq. 5) but identical measure functions. As in previous work [18], we offset the objective to move the optimal location away from the center of the search space, to $x_i = 5.12 \cdot 0.4 = 2.048$.

$$sphere(x) = \sum_{i=1}^{n} {x_i}^2 \tag{4}$$

$$Rastrigin(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10cos(2\pi x_i)] \tag{5}$$

We define the 2D measure space with the projection of the first and second half of the components x_i (see Eq. 7). We bound the contribution of each component through a *clip* function (Eq. 6). which restricts the measure contribution of each x_i to the range $[-5.12, 5.12]$.

$$clip(x_i) = \begin{cases} x_i & \text{if } -5.12 \leq x_i \leq 5.12 \\ 5.12/x_i & \text{otherwise} \end{cases} \tag{6}$$

$$m(x) = \left(\sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} clip(x_i), \sum_{i=\lfloor \frac{n}{2} \rfloor + 1}^{n} clip(x_i) \right) \tag{7}$$

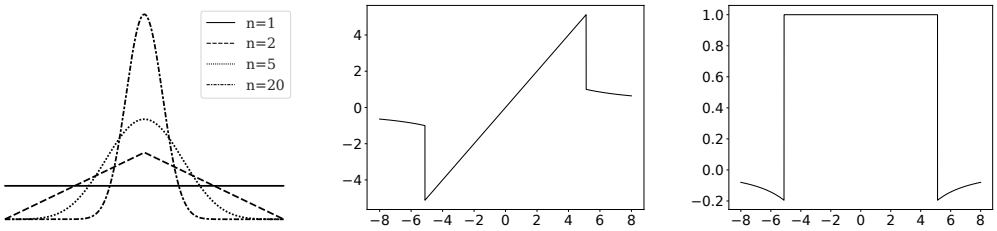
We observe that the partial derivatives for m_i are 1 for the range $[-5.12, 5.12]$. A constant derivative means that the gradient step will not shrink as OMG-MEGA approaches an extreme point in measure space, thus OMG-MEGA often overshoots the bounds; on the other hand, CMA-MEGA dynamically adapts the gradient steps, allowing for efficient coverage the measure space.

Fig. 4 visualizes the challenge of the linear projection domain. Observe that if we sample uniformly on the hypercube $[-5.12, 5.12]^n$, then each of our measure functions becomes a sum of random variables. If we normalize each measure by dividing by n, then our measure functions become an *average* of random variables. The average of n random variables forms the Bates distribution [34], a distribution that narrows as n increases (Fig. 4(a)). At $n = 1000$ the solutions sampled from the hypercube are very close to $\mathbf{0}$ in measure space with high probability. A QD algorithm could simply increase its step-size to move to extremes of measure space, but the *clip* function prevents this by implicitly bounding the extremes of the measure space; each component of $\boldsymbol{\theta}$ can contribute at most ± 5.12 to change the position in measure space. We note the heavy penalty in the *clip* function for a component leaving the range $[-5.12, 5.12]$. The combination of the narrowness of the Bates distribution and the implicit bounding of the *clip* function means that a QD algorithm must dynamically *adapt* its sampling distribution by shrinking step-sizes as the distribution gets close to the extremes of the measure space.

Arm Repertoire. We visualize example solutions for a $n = 7$ (7-DOF) planar arm in Fig. 5. The optimal solutions in this domain have zero joint angle variance from the mean (all angles are equal). We note that we selected as objective the variance, instead of the standard deviation used in previous work [61], so that the objective is differentiable at 0. The measures for the arm repertoire have range $[-n, n]$, since they are the positions of the end-effector, and the arm has n links of length $l_i = 1$. Thus, the reachable space forms a filled circle with radius n, and the maximum archive coverage becomes $\frac{\pi n^2}{4n^2} \approx 78.5\%$. We selected $n = 1000$ (1000-DOF) for the experiments.

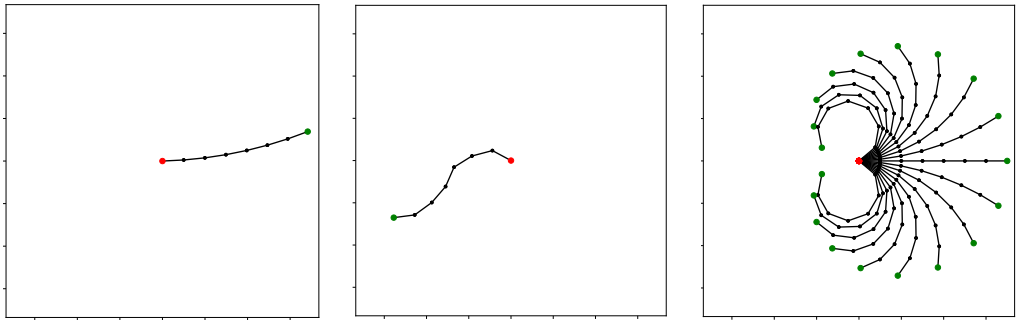
Latent Space Illumination. The latent space of StyleGAN [36] has size 512 where a latent code is repeated 18 times for each level of detail.

Transformations of the Objective Function. The QD-score metric, which we use to estimate the performance of QD algorithms, associates solution quality with maximizing the objective value f, and assumes a strictly positive objective value f for every solution. Therefore, in each domain we transform the objective through a linear transformation of the form $f' = af + b$ to map the objective values to the range $[0, 100]$.



(a) (b) (c)

Figure 4: (a) Bates distribution (reproduced from [19] with authors' permission). (b) $clip$ function for the linear projection domain. (c) Derivative of the $clip$ function.



(a) (b) (c)

4

Figure 5: (a) Example of an optimal solution (zero variance of joint angles) for a 7-DOF planar arm. The green dot indicates the robot's end-effector. (b) Example of a sub-optimal solution. (c) Ensemble of optimal solutions. A precise view of the filled archive at $n = 1000$ (1000-DOF), with the "characteristic swirls" found in lower dimensional archives, is shown in Fig. 1.

In the linear projection domain, we follow the objective transformation proposed in previous work [19]. The original objective is to minimize the sphere and Rastrigin functions. We compute an estimate on the largest sphere and Rastrigin values in the hypercube $[-5.12, 5.12]^n$ which contain only components in the linear portion of the clip function. We compute $f(-5.12, -5.12, ..., -5.12)$ as an estimated maximum of the function f_{max}. The minimum of each function is $f_{min} = 0$. We then remap the function values for both the sphere and Rastrigin objective functions through the linear transformation given by:

$$f'(\boldsymbol{\theta}) = 100 \cdot \frac{f(\boldsymbol{\theta}) - f_{max}}{f_{min} - f_{max}} \tag{8}$$

For the arm domain we estimate f_{max} to be 1 from an initial population of angles sampled from $\mathcal{N}(\mathbf{0}, I)$. For the LSI domain we picked $f_{max} = 10$ by observing the CLIP loss function values for the objective text prompt "Elon Musk with short hair.".

C Implementation

Archives. In the both the linear projection and arm repertoire domains, the measure space is 2D, with resolution 100×100. We normalize the objective value f so that it is in the range [0,100] and initialize empty cells of the archive with an objective value f of 0. For latent space illumination we form a 2D archive with the resolution 200×200. We double the resolution in each dimension as the examined QD algorithms fill roughly a quarter of possible cells in the archive.

Computational Resources. We ran 10,000 iterations in all algorithms. We ran all trials in parallel on an AMD Ryzen Threadripper 32-core (64 threads) processor and an GeForce RTX 3090 Nvidia GPU. A run of 20 trials in parallel required about 30' for the linear projection domains and 2 hours

for the arm repertoire domain. One trial run for the latent space illumination domain took about 2 hours. We note runtime increases at a higher logging frequency and algorithms which perform better may run slower due to iterating over more archive solutions when QD statistics are calculated.

Software Implementation. We use the publicly available Pyribs [59] library for all algorithms, where we implemented the OG-MAP-Elites, OG-MAP-Elites (line), OMG-MEGA and CMA-MEGA algorithms. We use the Adam implementation of ESTool. [1]

D Improvement Ranking as a Natural Gradient Approximation

We show for the first time that CMA-ME's improvement ranking optimizes a modified QD objective (eq. 10) via a natural gradient approximation. We then extend our derivation to show that the coefficient distribution of CMA-MEGA approximates natural gradient steps of the same objective with respect to the gradient coefficients.

We let the original QD objective for an archive $\mathcal{A}$ be:

$$\max \; J_1(\mathcal{A}) = \sum_{i=1}^{M} f(\boldsymbol{\theta_i}) \tag{9}$$

where $f(\boldsymbol{\theta_i}) = 0$ if a cell i is unfilled.

Consider that MAP-Elites solves a discrete optimization problem over *archives* $\mathcal{A}$ rather than a continuous optimization problem over solution parameters $\boldsymbol{\theta}$. To analyze the improvement ranking, we reparameterize the objective J_1 to include $\boldsymbol{\theta}$, the current search position for CMA-ES. By fixing the current archive $\mathcal{A}$, our goal is to show that improvement ranking approximates a natural gradient step of J_1 with respect to $\boldsymbol{\theta}$. In other words, we want to calculate the direction to change $\boldsymbol{\theta}$ that results in the steepest increase of J_1 after $\boldsymbol{\theta}$ is added to the archive.

Ranking candidate solutions $\boldsymbol{\theta}$ based on J_1 may prioritize solutions that improve existing cells on the archive over solutions that discover new cells. To show this possibility, let $\boldsymbol{\theta_i}$ and $\boldsymbol{\theta_j}$ be two arbitrary candidate solutions, where $\boldsymbol{\theta_i}$ replaces a cell with an existing occupant $\boldsymbol{\theta_p}$ and $\boldsymbol{\theta_j}$ discovers a new cell. If $f(\boldsymbol{\theta_i}) - f(\boldsymbol{\theta_p}) > f(\boldsymbol{\theta_j})$, then $\boldsymbol{\theta_i}$ will be ranked higher than $\boldsymbol{\theta_j}$.

To strictly prioritize exploration, CMA-ME performs a two-stage improvement ranking on the objective J_1, where it ranks first all solutions $\boldsymbol{\theta_j}$ that discover new cells based on their objective value $f(\boldsymbol{\theta_j})$, and subsequently all solutions $\boldsymbol{\theta_i}$ that improve existing cells based on the difference $f(\boldsymbol{\theta_i}) - f(\boldsymbol{\theta_p})$ between the f value of the new and previous solution $\boldsymbol{\theta_p}$.

To show that the ranking rules of CMA-ME are equivalent to a natural gradient of a QD objective, we specify a function J_2 (see Eq. 10), and we will show that *sorting solutions purely by J_2 results in the same order as the improvement ranking by CMA-ME on the objective J_1.*

$$\max \; J_2(\mathcal{A}) = \sum_{i=1}^{M} [f(\boldsymbol{\theta_i}) + b_i C] \tag{10}$$

$$C > 2 \cdot [\max_{\boldsymbol{\theta_i} \in \mathbb{R}^n} f(\boldsymbol{\theta_i}) - \min_{\boldsymbol{\theta_i} \in \mathbb{R}^n} f(\boldsymbol{\theta_i})] \tag{11}$$

where C is larger than two times the largest gap between any pair of $f(\boldsymbol{\theta}_i)$ and $f(\boldsymbol{\theta}_j)$ (see Eq. 11), b_i is 1 if a cell is occupied and 0 otherwise, and $f(\boldsymbol{\theta_i}) = 0$ if a cell i is unfilled.[2]

Let $\boldsymbol{\theta_i}$ and $\boldsymbol{\theta_j}$ be two arbitrary candidate solutions whose addition to the archive changes the archive. We define $\Delta_i^{\mathrm{J}_2} = J_2(\mathcal{A} + \boldsymbol{\theta_i}) - J_2(\mathcal{A})$ and $\Delta_j^{\mathrm{J}_2} = J_2(\mathcal{A} + \boldsymbol{\theta_j}) - J_2(\mathcal{A})$, and we consider the following three cases:

1: Without loss of generality θ_i improves an occupied cell and θ_j discovers a new cell.

Based on the improvement ranking of CMA-ME, $\boldsymbol{\theta_j}$ will always be ranked higher than $\boldsymbol{\theta_i}$. We will show that the same ordering holds if we sort based on the objective J_2.

[1] `https://github.com/hardmaru/estool/blob/master/es.py` (line 70)

[2] We note that this is an initialization value, since an empty cell does not yet contain a $\boldsymbol{\theta_i}$.

We let $\boldsymbol{\theta_p}$ be the occupant replaced by $\boldsymbol{\theta_i}$.

We show that $\Delta_j^{\mathrm{J}_2} > \Delta_i^{\mathrm{J}_2}$:

$$\begin{aligned}
\Delta_j^{\mathrm{J}_2} &= f(\boldsymbol{\theta_j}) + C \\
&= f(\boldsymbol{\theta_j}) + \frac{C}{2} + \frac{C}{2} \\
&> f(\boldsymbol{\theta_j}) + \frac{C}{2} + f(\boldsymbol{\theta_i}) - f(\boldsymbol{\theta_p}) \\
&> f(\boldsymbol{\theta_i}) - f(\boldsymbol{\theta_p}) \\
&= \Delta_i^{\mathrm{J}_2}
\end{aligned} \tag{12}$$

Where inequalities hold from the definition of C (Eq. 11).

2: Both $\boldsymbol{\theta_i}$ and $\boldsymbol{\theta_j}$ discover new cells.

From the improvement ranking rule, $\boldsymbol{\theta_i}$ will be ranked higher than $\boldsymbol{\theta_j}$ iff $\Delta_i^{\mathrm{J}_1} > \Delta_j^{\mathrm{J}_1}$.

We show that this is equivalent to $\Delta_i^{\mathrm{J}_2} > \Delta_j^{\mathrm{J}_2}$.

$$\Delta_i^{\mathrm{J}_1} > \Delta_j^{\mathrm{J}_1} \Leftrightarrow f(\boldsymbol{\theta_i}) > f(\boldsymbol{\theta_j}) \Leftrightarrow f(\boldsymbol{\theta_i}) + C > f(\boldsymbol{\theta_j}) + C \Leftrightarrow \Delta_i^{\mathrm{J}_2} > \Delta_j^{\mathrm{J}_2} \tag{13}$$

3: Both $\boldsymbol{\theta_i}$ and $\boldsymbol{\theta_j}$ improve existing cells.

We let $\boldsymbol{\theta_p}$ the occupant replaced by $\boldsymbol{\theta_i}$ and $\boldsymbol{\theta_r}$ the occupant replaced by $\boldsymbol{\theta_j}$.

$$\Delta_i^{\mathrm{J}_1} > \Delta_j^{\mathrm{J}_1} \Leftrightarrow f(\boldsymbol{\theta_i}) - f(\boldsymbol{\theta_p}) > f(\boldsymbol{\theta_j}) - f(\boldsymbol{\theta_r}) \Leftrightarrow \Delta_i^{\mathrm{J}_2} > \Delta_j^{\mathrm{J}_2} \tag{14}$$

Therefore, the CMA-ME improvement ranking is identical to ranking purely based on J_2.

Previous work [1] has shown that CMA-ES approximates the natural gradient of its provided optimization objective function to a scale. At the same time, CMA-ES is invariant to order-preserving transformations of its objective function, since it is a comparison-based optimization method [30]. We provide an explicit objective function J_2, and we have shown that maximizing this function results in the same ordering as the one specified implicitly by the improvement ranking rules. Therefore, CMA-ME, which uses improvement ranking to update the CMA-ES parameters, approximates a natural gradient of J_2 to a scale.

CMA-MEGA sorts solutions via the same improvement ranking rules as CMA-ME to update an internal sampling distribution maintained by CMA-ES. Unlike CMA-ME, CMA-MEGA updates a sampling distribution of gradient step coefficients, and not solution parameters directly. Therefore, CMA-MEGA performs a form of adaptive search, where the algorithm dynamically changes the search hyperparameters (gradient step coefficients) to maximize J_2.

By connecting CMA-ME to previous work that connects natural gradient descent and CMA-ES [1], we gain a new perspective on both CMA-ME and CMA-MEGA. In each iteration, CMA-ME optimizes for a single solution whose addition to the archive results in the largest increase in J_2. However, after sampling, we update the archive which results in a small change to the optimization landscape. In other words, CMA-ME assumes that CMA-ES is a robust enough optimizer to handle small changes to its target objective.

To apply this new perspective to CMA-MEGA, we observe that CMA-MEGA solves the optimization problem of finding the chosen gradient coefficients that yield the largest increase in J_2 for a fixed $\boldsymbol{\theta}$ and archive $\mathcal{A}$. In other words, CMA-MEGA optimizes the objective J_2, as CMA-ME, but it does so directly in objective-measure space. However, after taking one optimization step, we change the single solution optimization problem by updating the archive and moving $\boldsymbol{\theta}$. Just like CMA-ME, we assume that CMA-ES is a robust enough optimizer to handle changes in the optimization landscape brought about by changing the archive and $\boldsymbol{\theta}$.

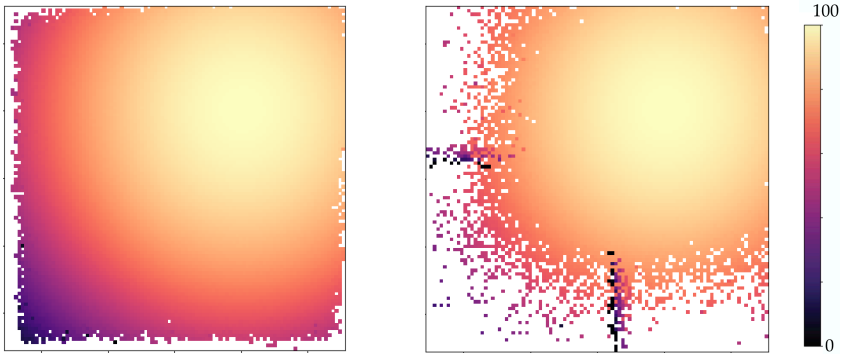


Figure 6: Example archive of OMG-MEGA for normalized (left) and unnormalized (right) gradients for the ablation run in the Linear Projection (sphere) domain.

	LP (sphere)		
Algorithm	QD-score	Coverage	Best
OMG-MEGA (norm)	71.58 ± 0.10	$92.09 \pm 0.21\%$	100.00 ± 0.00
OMG-MEGA (unnorm)	56.96 ± 0.24	$67.30 \pm 0.38\%$	100.00 ± 0.00
	LP (Rastrigin)		
Algorithm	QD-score	Coverage	Best
OMG-MEGA (norm)	55.90 ± 0.21	$77.00 \pm 0.34\%$	97.55 ± 0.06
OMG-MEGA (unnorm)	16.56 ± 0.11	$25.97 \pm 0.18\%$	89.76 ± 0.42
	Arm Repertoire		
Algorithm	QD-score	Coverage	Best
OMG-MEGA (norm)	44.12 ± 0.06	$44.13 \pm 0.06\%$	100.00 ± 0.00
OMG-MEGA (unnorm)	0.85 ± 0.04	$7.02 \pm 0.21\%$	12.58 ± 0.36

Table 2: OMG-MEGA with normalized (norm) and unnormalized (unnorm) gradients.

E On the Importance of Normalizing Gradients

We discuss the importance of normalizing gradients in the MAP-Elites via Gradient Arborescence algorithm and include results from an ablation over gradient normalization. We explore the importance of normalizing gradients through an ablation study on OMG-MEGA, where we compare OMG-MEGA with the normalization step to OMG-MEGA without the normalization step in the linear projection and arm repertoire domains.

To account for the scaled step-size after normalization in the linear projection domain, we set the σ_g to 10 in the normalized variant and to 0.5 in the unnormalized variant, since the average gradient magnitude was approximately 20. For the arm repertoire domain, we note that the objective gradient magnitude was approximately 10^{-2}, while the measure gradient magnitudes were approximately 10^2. We ran tuning experiments for σ_g at scales 10^2, 10^1, 1, 10^{-1}, 10^{-2}, 10^{-3}, and 10^{-4} to scale the unnormalized gradient and selected $\sigma_g = 10^{-4}$ as the best performing hyperparameter.

Table 2 presents the results of the ablation study in the linear projection and arm domains. The quantitative results suggest that normalization greatly affects performance of the OMG-MEGA algorithm. Below we give our interpretation of why normalization plays such a significant factor on performance in each domain.

Consider the linear projection sphere domain. In this domain the objective is quadratic. As we move further away from the optimum, the magnitude of the gradient increases linearly. In OMG-MEGA, the gradient coefficients are sampled from a fixed distribution. This means the gradient of our objective will dominate the gradients of the measures for positions in measure space far away from the optimum. Fig. 6 visualizes a final archive from this domain. We see that the unnormalized OMG-MEGA easily fills cells near the global optimum, but struggles to fill cells with a lower objective value. In contrast OMG-MEGA with normalization fills the archive evenly.

In the arm repertoire domain, we observe that the objective gradient is on a different scale than the measure gradients and this affects the performance of the unnormalized OMG-MEGA. The average objective gradient magnitude was 10^{-2}. In contrast, the average measure gradient magnitude was 10^{2}. This means that the objective gradient barely contributes to the search for a fixed coefficient distribution. However, the fact that OG-MAP-Elites performs well in this domain indicates that the objective gradient is an important factor in filling the archive. Because of the different scale, the objective gradient is largely ignored by the OMG-MEGA variant that does not normalize gradients.

While in OMG-MEGA the benefit of normalizing gradients is clear, we speculate that normalizing gradients is beneficial also for CMA-MEGA. In CMA-MEGA, the CMA-ES subroutine solves a non-stationary optimization problem for gradient coefficients that maximize the QD objective. The coefficients can be viewed as a learning rate for each gradient. In this sense, CMA-MEGA has adaptive learning rates, where CMA-ES acts as the adaptation mechanism. However, in standard gradient ascent the magnitude of the gradient acts as a step size control. If the gradients are not normalized, the magnitudes and the adaptive step size control may end up fighting each other. This may cause oscillations in CMA-ES's adaptation mechanisms and lead to instability. We leave a thorough study to evaluate the exact benefits of normalization in CMA-MEGA as future work.

Finally, we note a conceptual difference for how we leverage measure gradients in DQD versus how objective gradients are used in optimization. In quality diversity, a goal of the algorithm is to cover the entire measure space. Intermediate solutions are of equal importance to solutions at the extremes of measure space. By normalizing gradients and giving control over the step-size to the DQD algorithm, CMA-MEGA can adapt its search distribution to more easily fill intermediate regions of the measure space, rather than *only* optimize for extrema.

F On the Differences between OG-MAP-Elites (line) and PGA-MAP-Elites

We explore the design decisions between PGA-MAP-Elites [45], an reinforcement learning (RL) algorithm, and OG-MAP-Elites (line), a DQD baseline that draws insights from PGA-MAP-Elites.

Specifically, PGA-MAP-Elites is an actor-critic reinforcement learning algorithm. Actor-critic methods combine elements of policy gradient methods, which approximate the gradient of the reward function with respect to a policy's actions, and Q-learning methods, which approximate Q-values for every state-action pair. The computationally expensive part in this setting is the evaluation of a policy, while gradient computations do not factor into the running time of the algorithm.

On the other hand, the DQD problem assumes knowledge of the exact gradients of the objective and measure functions, which makes gradient approximation steps unnecessary. Thus, the most expensive part of the computation in many applications becomes the gradient computation for a given solution, rather than the evaluation of that solution. For example, if the function is a neural network, evaluation of a solution can be done efficiently with a forward pass. However, gradient computation requires an often expensive backpropagation of the loss function through the network. For efficient performance, a designer of a DQD algorithm would aim towards minimizing the number of gradient computations to achieve good results, instead of *only* the number of evaluations.

As an actor-critic method, PGA-MAP-Elites trains a Q-value approximator (a critic). As a quality diversity algorithm, the critic network can be trained from the rollout data of the entire *archive* of policies to form a better Q-value approximation. Since an accurate gradient approximation is important in RL settings, the shared critic network allows for efficient and accurate policy gradient calculation *on demand* for any new candidate policy evaluated. PGA-MAP-Elites consists of two independent operators, (1) the Iso+LineDD operator and (2) a policy gradient step.

In our implementation of the DQD algorithms in Pyribs' ask-tell interface – adopted from Pycma [32], we evaluate a solution and pass the gradient simultaneously. If we directly implemented independent operators in OG-MAP-Elites (line), when using operator (2) we would need to evaluate a solution once to perform a gradient step and evaluate the new solution again after the gradient step. Instead, we apply in OG-MAP-Elites (line) sequentially the operators (1) and (2). This allows us to use the existing evaluation (and gradient computation) from (1) to perform a gradient step (2), reducing the number of evaluations needed per gradient step.

We implemented OG-MAP-Elites and OG-MAP-Elites (line) with the two independent operators to evaluate the effect of this design decision. We can divide our computation budget per iteration

	LP (sphere)		
Algorithm	QD-score	Coverage	Best
OG-MAP-Elites	1.36 ± 0.08	$1.50 \pm 0.09\%$	100.00 ± 0.00
OG-MAP-Elites (line)	14.91 ± 0.08	$17.29 \pm 0.09\%$	100.00 ± 0.00
OG-MAP-Elites†	1.15 ± 0.10	$1.27 \pm 0.11\%$	100.00 ± 0.00
OG-MAP-Elites (line)†	13.31 ± 0.09	$15.42 \pm 0.10\%$	100.00 ± 0.00
	LP (Rastrigin)		
Algorithm	QD-score	Coverage	Best
OG-MAP-Elites	0.83 ± 0.03	$1.28 \pm 0.04\%$	74.38 ± 0.04
OG-MAP-Elites (line)	6.09 ± 0.04	$8.84 \pm 0.07\%$	76.67 ± 0.05
OG-MAP-Elites†	0.76 ± 0.03	$1.14 \pm 0.03\%$	74.43 ± 0.04
OG-MAP-Elites (line)†	5.14 ± 0.04	$7.46 \pm 0.06\%$	76.17 ± 0.04
	Arm Repertoire		
Algorithm	QD-score	Coverage	Best
OG-MAP-Elites	57.21 ± 0.03	$58.11 \pm 0.03\%$	98.63 ± 0.00
OG-MAP-Elites (line)	59.57 ± 0.03	$60.19 \pm 0.03\%$	99.17 ± 0.00
OG-MAP-Elites†	65.72 ± 0.06	$65.95 \pm 0.06\%$	100.00 ± 0.00
OG-MAP-Elites (line)†	65.73 ± 0.04	$65.96 \pm 0.04\%$	100.00 ± 0.00

Table 3: OG-MAP-Elites with independent gradient and perturbation operators (denoted with †) compared against our sequential operators derivation from the main paper.

into thirds. The first third of operators will perturb existing solutions with Iso+LineDD. The second third will re-evaluate an existing archive solution and compute an objective gradient. The final third will evaluate the solution after applying the computed gradient step. With our current derivation, we waste a third of our evaluation budget on computing gradients. We denote implementations with independent operators as OG-MAP-Elites† and OG-MAP-Elites (line)†.

We run our ablation on the arm repertoire domain and both variants of the linear projection domain. Table 3 contains the results of the ablation. On both linear projection domains, the variants of OG-MAP-Elites and OG-MAP-Elites (line) with sequential operators outperform their counterparts with independent operators. However, in the arm repertoire domain, the independent operators implementation outperforms the sequential operator counterpart. We conjecture that because exploration is more difficult in the linear projection domain, the smaller number of perturbation operations in the independent operators variants hurts performance. On the other hand, in the arm repertoire domain, optimal solutions are concentrated in a small elite hypervolume. The independent operators can apply two or more gradient steps sequentially, directing the search efficiently towards the elite hypervolume.

G On the Effect of the Archive Resolution on the Performance of CMA-ME

In this section, we comment on the performance of CMA-ME on the linear projection domain in the main paper and include an extra experiment with a different archive resolution.

The linear projection domain was first introduced by the authors of CMA-ME [19] to show the importance of adaptation mechanisms in quality diversity algorithms. In the paper, each domain was run with search dimensions of $n = 20$ and $n = 100$ for both the sphere and the Rastrigin objective. Experiments were run for 2.5×10^6 evaluations ($\approx 65,000$ iterations) and, due to the large number of evaluations, the archive resolution was 500×500. The paper reports CMA-ME outperforming both MAP-Elites and MAP-Elites (line) in this setting. In contrast, our experiments were run for 10,000 iterations with a search space dimension of $n = 1000$ and an archive resolution of 100×100.

We ran two additional experiments where we evaluated MAP-Elites, MAP-Elites (line), and CMA-ME on both linear projection domains with archive resolution of 500×500. Table 4 contains the results of the extra experiments and Fig. 7 visualizes a final archive of each algorithm for each experiment. We observe that under increased archive resolution CMA-ME outperforms both variants of MAP-Elites. We also observe that coverage increases for both MAP-Elites and CMA-ME when

	LP (sphere)		
Algorithm	QD-score	Coverage	Best
MAP-Elites	2.37 ± 0.01	$2.86 \pm 0.01\%$	90.56 ± 0.02
MAP-Elites (line)	4.96 ± 0.03	$5.82 \pm 0.04\%$	92.87 ± 0.03
CMA-ME	6.82 ± 0.05	$7.74 \pm 0.06\%$	94.80 ± 0.07
	LP (Rastrigin)		
Algorithm	QD-score	Coverage	Best
MAP-Elites	1.87 ± 0.01	$2.84 \pm 0.01\%$	74.46 ± 0.03
MAP-Elites (line)	3.75 ± 0.02	$5.45 \pm 0.03\%$	76.25 ± 0.04
CMA-ME	4.92 ± 0.04	$7.26 \pm 0.05\%$	76.15 ± 0.08

Table 4: Results for each derivative-free QD algorithm evaluated on the linear projection domains. These experiments were run with archives of resolution 500×500 instead of 100×100.

the archive is higher resolution. In contrast, coverage decreases for the MAP-Elites (line) algorithm when the archive resolution increases.

We explain why the low resolution of the archive negatively affects the performance of CMA-ME. For archives of resolution 100×100 there is not enough granularity in the archive to easily move through measure space, negatively affecting both MAP-Elites and CMA-ME. Moreover, CMA-ME restarts frequently due to the rule that the algorithm will restart from a random elite if none of the λ sampled solutions in an iteration improve the archive. Everytime CMA-ME restarts, it samples λ solutions by perturbing an elite chosen uniformly at random, with isotropic Gaussian noise. This limit case is nearly equivalent to the MAP-Elites algorithm, with the caveat that all generated solution come from one elite rather than λ different elites. In the 100×100 setting, CMA-ME performs similarly to MAP-Elites.

However, at a higher archive resolution of 500×500 restarts become less frequent and generated solutions in each iteration fall into different cells in the archive. The higher granularity of the archive causes CMA-ME to restart fewer times. Under this setting we see performance of CMA-ME consistent with the CMA-ME paper.

Overall, this study indicates the effect of archive resolution on the performance of QD algorithms.

H Additional Results

H.1 Generated Archives and Additional Metrics.

Fig. 10 presents example archives for each algorithm and domain combination. Table 5 presents the values of the QD-score, coverage, and best solution for each algorithm and domain.[3] We additionally ran MAP-Elites, MAP-Elites (line), and CMA-ME for 10^6 iterations, 20 trials, in the linear projection and arm repertoire domains, and present the results as MAP-Elites*, MAP-Elites (line)*, and CMA-ME*. We observe that running CMA-MEGA for only 10,000 iterations outperforms each algorithm in both domains.

We note that the QD-score metric combines both the number of cells filled in the archive (diversity) and the objective value of each occupant (quality). To disambiguate the two, we show in Fig. 11 the percentage of cells (y-axis) that have objective value greater than the threshold specified in the x-axis.

Qualitative Results. We can also compare the quality of the generated images with latent space illumination, with the images generated when using a single-objective Adam optimizer, where we optimize StyleGAN+CLIP [48] with Adam as a baseline (instead of running a QD algorithm). We run 5 different trials for 10,000 iterations, and for each trial we present the image that the algorithm

[3] We note that the QD-score and coverage values of OG-MAP-Elites are slightly different between Table 1 and Table 5. Table 5 contains the correct values and we will update Table 1 in the revised version. The change does not affect the tests for statistical significance or any of our quantitative and qualitative findings of the experiments in section 6.

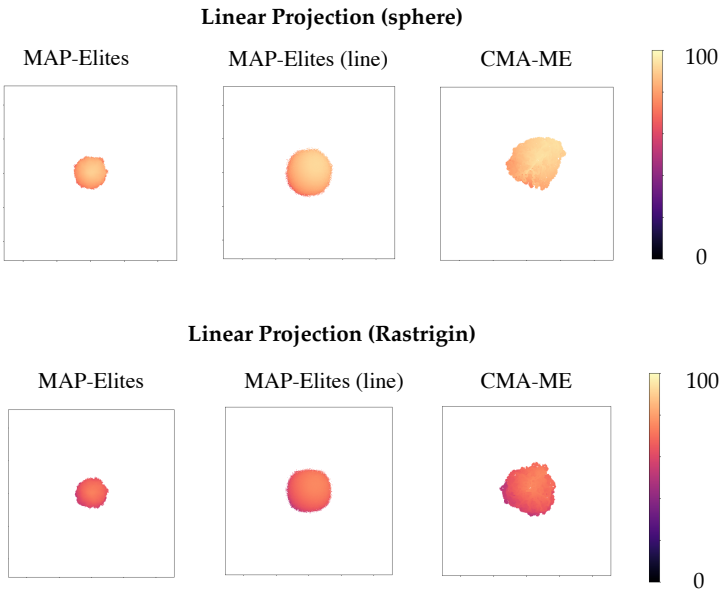


Figure 7: Example archives for each derivative-free QD algorithm evaluated in the linear projection domains. These experiments were run with archives of resolution 500×500 instead of 100×100.

converged to. StyleGAN+CLIP with Adam gradually drifted towards unrealistic images, so we excluded from our selection images that were unrealistic or with large artifacts. We used the same prompt as the objective of section 6: "Elon Musk with short hair.". We compare these images with the best image, according to the objective, of each of the 5 generated archives of CMA-MEGA (Adam). We used in both conditions the Adam implementation of ESTool (see accompanying source code) with the same hyperparameters. We observe that the quality of the images is comparable (Fig. 12).

While running the LSI experiments, we discovered that several intermediate solutions were of better visual quality than the final images present in the generated collage. Fig. 13 shows two such images cherry-picked from all intermediate solutions generated by CMA-MEGA (Adam) during the experiments. According to CLIP, these images are of lower quality than the images present in the collage of Fig. 3 in the main paper. We interpret this finding as CMA-MEGA overfitting to the CLIP loss function. Several methods exist in the generative art community for mitigating this affect. For example, CLIPDraw [21] augments the generated image via randomized data augmentations then evaluates the generated batch with CLIP. The average loss of the entire batch replaces evaluating a single generated image with CLIP. A similar method could be applied to our LSI approach to improve the perceptual quality of generated images.

H.2 Additional Experiments in the LSI Domain

We include additional experiments in the LSI domain to assess DQD's qualitative and quantitative performance on different prompts. The two additional experiments were run with the objective prompts "A photo of Beyonce" and a "A photo of Jennifer Lopez" with the measure prompts "A small child." and "A woman with long blonde hair.".

Table 6 presents quantitative metrics of the additional runs as well as the Elon Musk experiment from the main paper. We observe that CMA-MEGA (Adam) retains top quantitative performance against each baseline for each experiment.

Fig. 14 and Fig. 15 contain collages for the Beyoncé and Jennifer Lopez experiments, respectively. For the Jennifer Lopez and Elon Musk experiments, we report that all archive collages were of similar quality. However, for Beyoncé only one of the collages, the one presented in Fig. 14, did not have significant artifacts. CLIP guided CMA-MEGA (Adam) far out of the latent Gaussian distribution for the remaining runs of the Beyoncé prompt and became confident in low quality images that overwrote

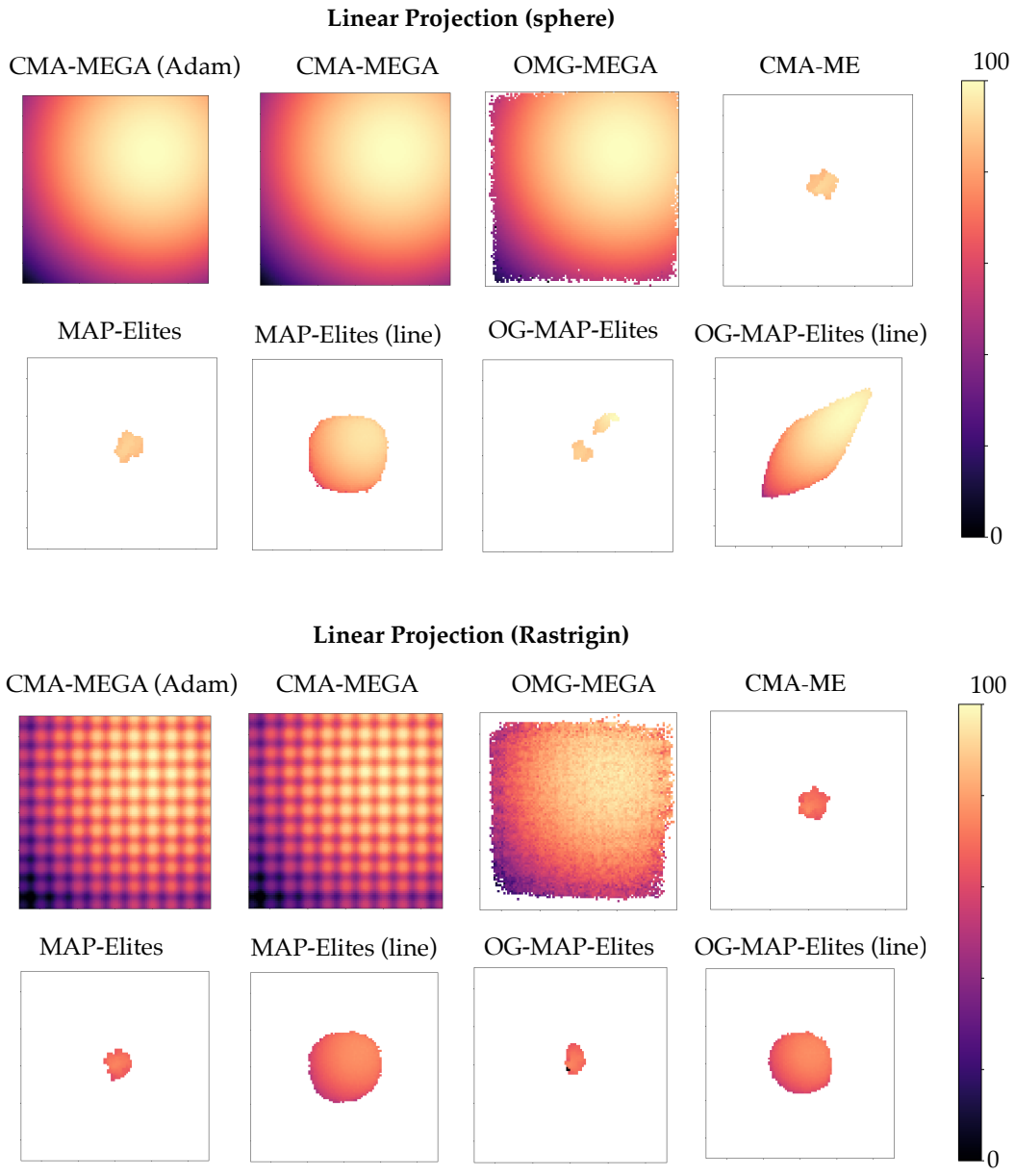


Figure 8: Example archives for each algorithm evaluated in the linear projection domain.

previously discovered high quality images in the archive. We highlight the concern that CLIP may perform differently for text prompts identifying celebrities of different ethnic backgrounds.

For Beyoncé, the collage contains faces of different ages for the measure "A small child", a proxy prompt for "age". However, for the Jennifer Lopez prompt, DQD constructs younger faces of Jennifer Lopez by places an older looking face on a younger looking head. We include an additional qualitative run of CMA-MEGA (Adam) on Jennifer Lopez (see Fig. 16), where we replace the measure prompt "A small child." with the prompt "A frowning person". In this prompt, we find faces of a younger Jennifer Lopez not discovered in Fig. 15. This result shows that younger faces of Jennifer Lopez can be generated by StyleGAN and identified by CLIP, but not discovered by DQD when the measure function approximated "age". We hypothesize that this is a limitation of using a proxy measure for age, rather than a predictive model that can directly measure the abstract concept of age.

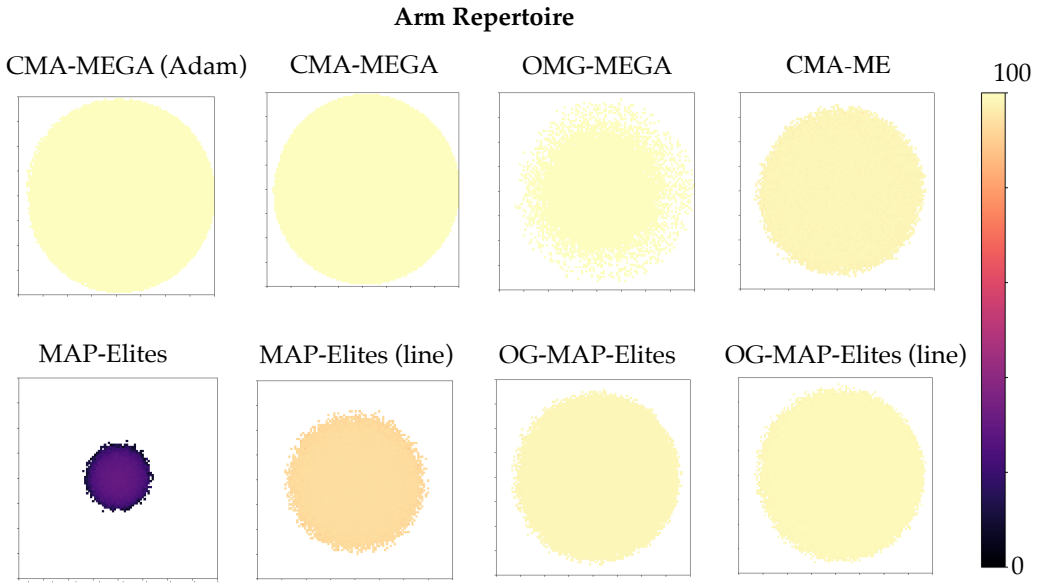


Figure 9: Example archives for each algorithm evaluated in the arm repertoire domain.

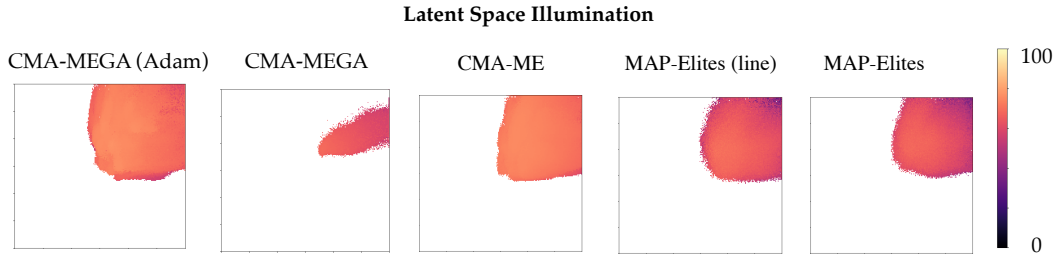


Figure 10: Example archives for each algorithm in the latent space illumination (LSI) domain.

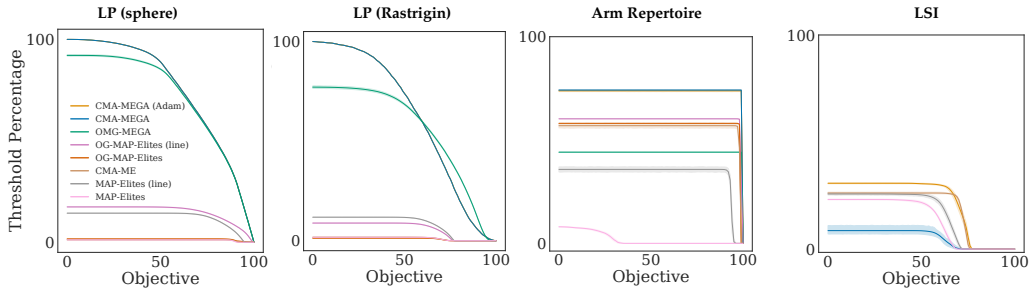


Figure 11: Percentage of cells in the y-axis with objective values larger than or equal to a threshold specified in the x-axis. The percentage is the number of filled cells (filtered by the threshold) over the archive size. Larger area under each curve indicates better performance.

	LP (sphere)		
Algorithm	QD-score	Coverage	Best
MAP-Elites	1.04 ± 0.03	$1.17 \pm 0.04\%$	90.60 ± 0.03
MAP-Elites*	5.19 ± 0.05	$5.93 \pm 0.06\%$	91.39 ± 0.02
MAP-Elites (line)	12.21 ± 0.06	$14.32 \pm 0.07\%$	94.89 ± 0.04
MAP-Elites (line)*	34.74 ± 0.05	$41.31 \pm 0.06\%$	98.83 ± 0.01
CMA-ME	1.08 ± 0.03	$1.21 \pm 0.04\%$	91.66 ± 0.01
CMA-ME*	5.30 ± 0.04	$6.03 \pm 0.05\%$	92.01 ± 0.02
OG-MAP-Elites	1.52 ± 0.09	$1.67 \pm 0.10\%$	$\mathbf{100.00 \pm 0.00}$
OG-MAP-Elites (line)	15.01 ± 0.06	$17.41 \pm 0.08\%$	$\mathbf{100.00 \pm 0.00}$
OMG-MEGA	71.58 ± 0.10	$92.09 \pm 0.21\%$	$\mathbf{100.00 \pm 0.00}$
CMA-MEGA	75.29 ± 0.00	$\mathbf{100.00 \pm 0.00}\%$	$\mathbf{100.00 \pm 0.00}$
CMA-MEGA (Adam)	$\mathbf{75.30 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}\%$	$\mathbf{100.00 \pm 0.00}$
	LP (Rastrigin)		
Algorithm	QD-score	Coverage	Best
MAP-Elites	1.18 ± 0.02	$1.72 \pm 0.04\%$	74.48 ± 0.06
MAP-Elites*	3.89 ± 0.03	$5.96 \pm 0.05\%$	74.60 ± 0.04
MAP-Elites (line)	8.12 ± 0.03	$11.79 \pm 0.05\%$	77.43 ± 0.05
MAP-Elites (line)*	22.65 ± 0.05	$33.19 \pm 0.08\%$	81.13 ± 0.03
CMA-ME	1.21 ± 0.02	$1.76 \pm 0.03\%$	75.61 ± 0.05
CMA-ME*	4.04 ± 0.02	$6.13 \pm 0.03\%$	75.71 ± 0.03
OG-MAP-Elites	0.83 ± 0.03	$1.26 \pm 0.03\%$	74.45 ± 0.03
OG-MAP-Elites (line)	6.10 ± 0.05	$8.85 \pm 0.07\%$	76.61 ± 0.06
OMG-MEGA	55.90 ± 0.21	$77.00 \pm 0.34\%$	97.55 ± 0.06
CMA-MEGA	62.54 ± 0.00	$\mathbf{100.00 \pm 0.00}\%$	99.98 ± 0.00
CMA-MEGA (Adam)	$\mathbf{62.58 \pm 0.00}$	$\mathbf{100.00 \pm 0.00}\%$	$\mathbf{99.99 \pm 0.00}$
	Arm Repertoire		
Algorithm	QD-score	Coverage	Best
MAP-Elites	1.97 ± 0.05	$8.06 \pm 0.11\%$	31.64 ± 0.44
MAP-Elites*	19.43 ± 0.30	$27.17 \pm 0.35\%$	75.42 ± 0.08
MAP-Elites (line)	33.51 ± 0.65	$35.79 \pm 0.66\%$	94.57 ± 0.10
MAP-Elites (line)*	61.59 ± 0.04	$62.73 \pm 0.04\%$	98.40 ± 0.00
CMA-ME	55.98 ± 0.60	$56.95 \pm 0.61\%$	99.10 ± 0.00
CMA-ME*	61.45 ± 0.22	$62.48 \pm 0.22\%$	99.10 ± 0.00
OG-MAP-Elites	57.17 ± 0.04	$58.08 \pm 0.04\%$	98.64 ± 0.00
OG-MAP-Elites (line)	59.66 ± 0.03	$60.28 \pm 0.03\%$	99.17 ± 0.00
OMG-MEGA	44.12 ± 0.06	$44.13 \pm 0.06\%$	$\mathbf{100.00 \pm 0.00}$
CMA-MEGA	$\mathbf{74.18 \pm 0.15}$	$\mathbf{74.18 \pm 0.15}\%$	$\mathbf{100.00 \pm 0.00}$
CMA-MEGA (Adam)	73.82 ± 0.20	$73.82 \pm 0.20\%$	$\mathbf{100.00 \pm 0.00}$
	LSI		
Algorithm	QD-score	Coverage	Best
MAP-Elites	13.88 ± 0.11	$23.15 \pm 0.14\%$	69.76 ± 0.07
MAP-Elites (line)	16.54 ± 0.28	$25.73 \pm 0.31\%$	72.63 ± 0.28
CMA-ME	18.96 ± 0.17	$26.18 \pm 0.24\%$	75.84 ± 0.10
CMA-MEGA	5.36 ± 0.78	$8.61 \pm 1.19\%$	68.74 ± 1.20
CMA-MEGA (Adam)	$\mathbf{21.82 \pm 0.18}$	$\mathbf{30.73 \pm 0.15}\%$	$\mathbf{76.89 \pm 0.15}$

Table 5: Results: Values with standard errors of QD-score, coverage, and best solution after 10,000 iterations for each algorithm and domain. Starred algorithms are run for 10^6 iterations.

Figure 12: Images generated with the single-objective Adam optimizer (top) and with CMA-MEGA (bottom) on the StyleGAN+CLIP model. Each algorithm optimizes the objective prompt "Elon Musk with short hair."

(a) (b)

Figure 13: Cherry-picked intermediate images generated from CMA-MEGA (Adam) during the LSI experiments for the prompt "Elon Musk with short hair.". These images were discovered intermediate solutions, but did not end up in the final archive returned by CMA-MEGA (Adam).

	LSI: Elon Musk		
Algorithm	QD-score	Coverage	Best
MAP-Elites	13.88 ± 0.11	23.15 ± 0.14%	69.76 ± 0.07
MAP-Elites (line)	16.54 ± 0.28	25.73 ± 0.31%	72.63 ± 0.28
CMA-ME	18.96 ± 0.17	26.18 ± 0.24%	75.84 ± 0.10
CMA-MEGA	5.36 ± 0.78	8.61 ± 1.19%	68.74 ± 1.20
CMA-MEGA (Adam)	**21.82 ± 0.18**	**30.73 ± 0.15%**	**76.89 ± 0.15**
	LSI: Beyonce		
Algorithm	QD-score	Coverage	Best
MAP-Elites	12.84 ± 0.10	19.41 ± 0.16%	71.42 ± 0.14
MAP-Elites (line)	14.40 ± 0.09	21.11 ± 0.16%	73.04 ± 0.05
CMA-ME	14.00 ± 0.62	19.57 ± 0.90%	74.11 ± 0.08
CMA-MEGA	3.37 ± 0.44	6.38 ± 0.71%	59.96 ± 1.40
CMA-MEGA (Adam)	**16.08 ± 0.37**	**22.58 ± 0.57%**	**74.95 ± 0.27**
	LSI: Jennifer Lopez		
Algorithm	QD-score	Coverage	Best
MAP-Elites	12.51 ± 0.28	19.18 ± 0.48 %	70.87 ± 0.27
MAP-Elites (line)	14.73 ± 0.06	21.60 ± 0.08%	73.50 ± 0.13
CMA-ME	15.24 ± 0.37	20.86 ± 0.50%	75.39 ± 0.09
CMA-MEGA	3.14 ± 0.14	5.58 ± 0.13%	64.88 ± 2.89
CMA-MEGA (Adam)	**17.06 ± 0.10**	**23.40 ± 0.14%**	**76.02 ± 0.08**

Table 6: Results from additional runs for Jennifer Lopez and Beyoncé. For comparison purposes, we include the original results for Elon Musk in the top table.

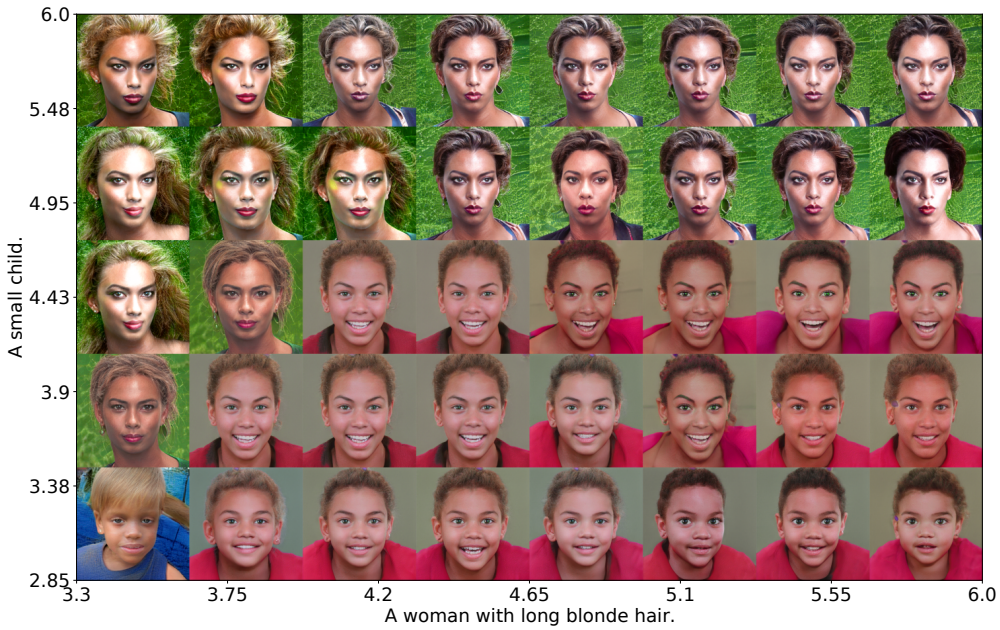


Figure 14: Result of latent space illumination for objective "A photo of Beyonce." and for measures "A small child." and "A woman with long blonde hair.". The axes values indicate the score returned by the CLIP model, where lower score indicates a better match.

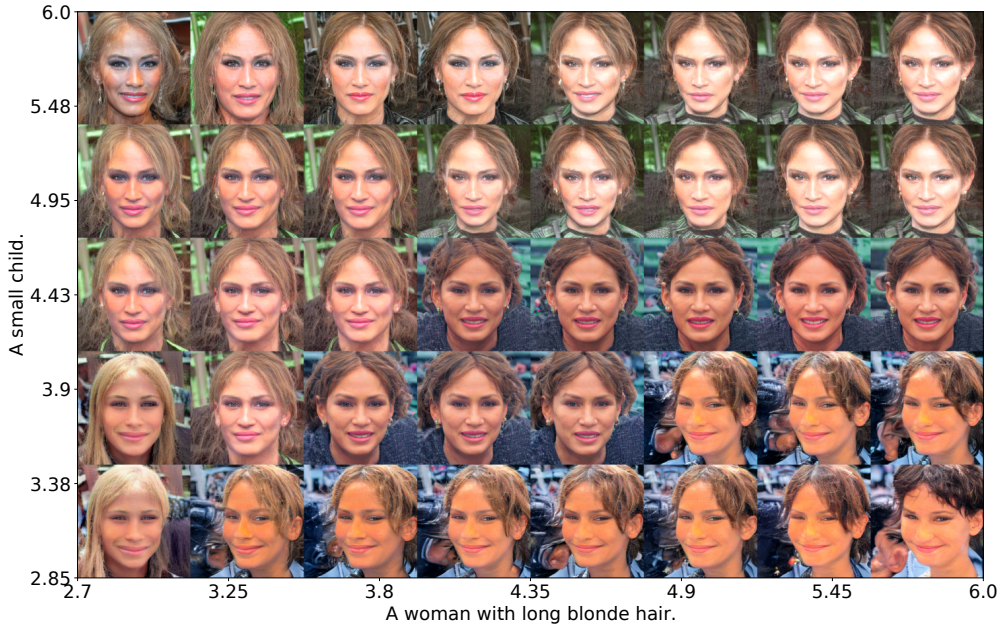


Figure 15: Result of latent space illumination for objective "A photo of Jennifer Lopez." and for measures "A small child." and "A woman with long blonde hair.". The axes values indicate the score returned by the CLIP model, where lower score indicates a better match.

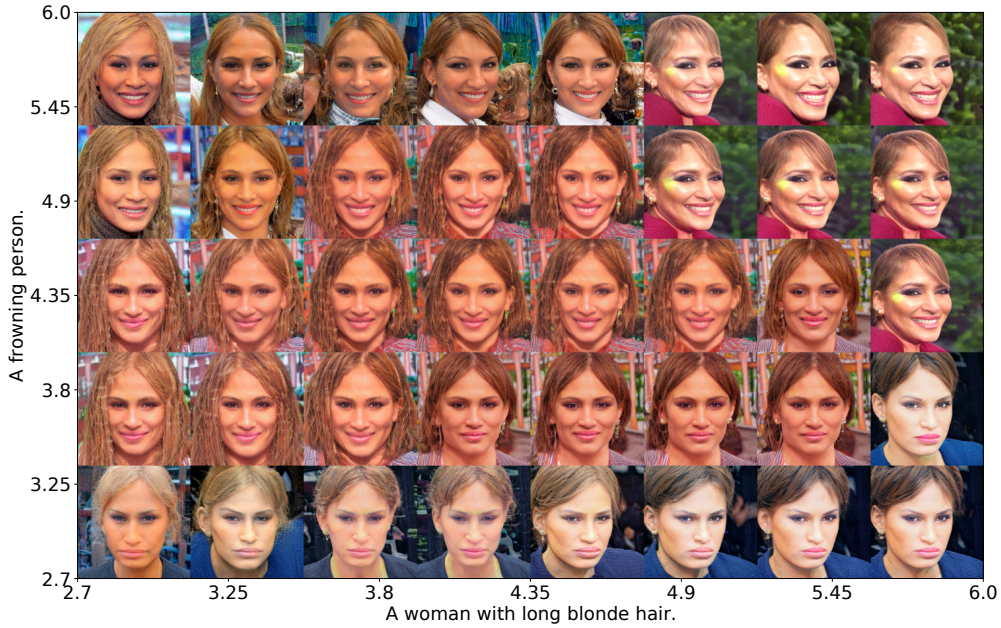


Figure 16: Result of latent space illumination for objective "A photo of Jennifer Lopez." and for measures "A frowning person." and "A woman with long blonde hair.". The axes values indicate the score returned by the CLIP model, where lower score indicates a better match.