These are notes on how to reproduce all tests conducted for our ICLR 2024 submission for PopDescent.

Benchmarks: All benchmark tests are run with the same parameters over 10 randomly selected seeds. *We decide to train all methods until they converge, and this is how we choose how many iterations to run each of them for.*

FMNIST:

PopDescent: All parameters are the same for the without/with regularization tests, except for changing the model by adding one l2 kernel regularizer to the second to last fully connected layer in the model with regularization.

Training Parameters:

Population Size	5
Replaced Individuals (in m-elitist)	2
Iterations	50
Batches	128
Batch Size	64
lr	0.001
Regularization Rate (in model with regularization)	0.001
Optimizer	Adam

Randomization Parameters:

lr_constant	10**(random.normal(mu=-4, sigma=2))
regularization_constant	10**(random.normal(mu=0, sigma=2))
randomization_amount (amount to randomize model, changes based on model's performance)	1 - (2 / (2 + model loss))
Gaussian Noise for Model Weights (sum of current weights and noise)	noise = random.normal(mu=0, sigma=0.01)*randomization_amount
Gaussian Noise for lr_constant (product of current lr_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))
Gaussian Noise for regularization_rate (product	2**(np.random.normal(mu=0,

of current regularization_constant and noise)	sigma=randomization_amount*15))

Basic Grid Search: All parameters are the same for the without/with regularization tests. Both without/with regularization tests use the model with l2 kernel regularization, but when training without regularization, the *Regularization Rates to Sample* is simply [0], meaning no regularization.

Training Parameters:

Learning Rates to Sample	[0.01, 0.001, 0.0001, 0.00001, 0.000001]
Regularization Rates to Sample	[0.01, 0.001, 0.0001, 0.00001, 0.000001]
Iterations	100
Batches	128
Batch Size	64
Optimizer	Adam

KT RandomSearch: All parameters are the same for the without/with regularization tests, except for changing the model by adding one l2 kernel regularizer to the second to last fully connected layer in the model with regularization. We used Keras Tuner's RandomSearch implementation.

Training Parameters:

Range of Learning Rates to Sample (continuous)	[0.0001, 0.01]
Range of Regularization Rates to Sample	[0.00001, 0.1]
max_trials	25
executions_per_trial	2
train_epochs (with early stopping, patience=2; train over the whole train dataset)	20
Batch Size	64
Optimizer	Adam

ESGD: We used *tqch*'s open-source implementation of ESGD on https://github.com/tqch/esgd-ws.

Training Parameters:

n_population	5

sgds_per_gen	1
evos_per_gen	1
reproductive_factor	4
m_elite	3
mixing_number	3
optimizer_class	SGD (lr=0.001)
n_generations	10
batch_size	64

CIFAR-10:

PopDescent: All parameters are the same for the without/with regularization tests, except for changing the model by adding one l2 kernel regularizer to the second to last fully connected layer in the model with regularization.

Training Parameters:

Population Size	5
Replaced Individuals (in m-elitist)	2
Iterations	20
Batches (trained over 2 epochs per iteration here)	128
Batch Size	64
lr	0.001
Regularization Rate (in model with regularization)	0.001
Optimizer	Adam

Randomization Parameters:

lr_constant	10**(random.normal(mu=-4, sigma=2))
regularization_constant	10**(random.normal(mu=0, sigma=2))
randomization_amount (amount to randomize model, changes based on model's performance)	1 - (2 / (2 + model loss))

Gaussian Noise for Model Weights (sum of current weights and noise)	noise = random.normal(mu=0, sigma=0.01)*randomization_amount
Gaussian Noise for lr_constant (product of current lr_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))
Gaussian Noise for regularization_rate (product of current regularization_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))

Basic Grid Search: All parameters are the same for the without/with regularization tests. Both without/with regularization tests use the model with l2 kernel regularization, but when training without regularization, the *Regularization Rates to Sample* is simply [0], meaning no regularization.

Training Parameters:

Learning Rates to Sample	[0.01, 0.001, 0.0001, 0.00001, 0.000001]
Regularization Rates to Sample	[0.01, 0.001, 0.0001, 0.00001, 0.000001]
Iterations	30
Batches	128
Batch Size	64
Optimizer	Adam

KT RandomSearch: All parameters are the same for the without/with regularization tests, except for changing the model by adding one l2 kernel regularizer to the second to last fully connected layer in the model with regularization. We used Keras Tuner's RandomSearch implementation.

Training Parameters:

Range of Learning Rates to Sample (continuous)	[0.0001, 0.01]
Range of Regularization Rates to Sample	[0.00001, 0.1]
max_trials	25
executions_per_trial	2
train_epochs (with early stopping, patience=2; train over the whole train dataset)	20
Batch Size	64
Optimizer	Adam

ESGD: We used *tqch*'s open-source implementation of ESGD on https://github.com/tqch/esgd-ws.

Training Parameters:

n_population	5
sgds_per_gen	1
evos_per_gen	1
reproductive_factor	4
m_elite	3
mixing_number	3
optimizer_class	SGD (lr=0.001)
n_generations	3
batch_size	8

Convergence Test: All convergence tests are run with the same parameters over 6 randomly selected seeds. All tests are run on the same model without regularization on the FMNIST dataset.

PopDescent:

Training Parameters

Population Size	5
Replaced Individuals (in m-elitist)	2
Iterations	115
Batches	128
Batch Size	64
lr	0.001
Optimizer	Adam

Randomization Parameters:

lr_constant	10**(random.normal(mu=-4, sigma=2))
randomization_amount (amount to randomize model, changes based on model's performance)	1 - (2 / (2 + model loss))
Gaussian Noise for Model Weights (sum of current weights and noise)	noise = random.normal(mu=0, sigma=0.01)*randomization_amount
Gaussian Noise for lr_constant (product of current lr_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))

Basic Grid Search:

Training Parameters

Learning Rates to Sample	[0.01, 0.001, 0.0001, 0.00001, 0.000001]
Regularization Rates to Sample	[0]
Iterations	100
Batches	128
Batch Size	64
Optimizer	Adam

KT RandomSearch:

Training Parameters

Range of Learning Rates to Sample (continuous)	[0.0001, 0.01]
Range of Regularization Rates to Sample	[0.00001, 0.1]
max_trials	25
executions_per_trial	2
train_epochs (with early stopping, patience=2; train over the whole train dataset)	20
Batch Size	64
Optimizer	Adam

ESGD:

Training Parameters

n_population	5
sgds_per_gen	1
evos_per_gen	1
reproductive_factor	4
m_elite	3
mixing_number	3
optimizer_class	SGD (lr=0.001)
n_generations	15
batch_size	64

Ablation Study: To emphasize the differences of PopDescent's features, we train only on the first 10k images in the FMNIST dataset, and add l2 kernel regularization to every layer in the benchmark FMNIST model, initialized with a default value of 0.001 in each layer. All training parameters are the same in each of the four tests listed in the paper. We only change whether PopDescent's randomization scheme is on or off, with CV selection turned on, and using the model with regularization for the first two tests listed ("Ablation Study Over Randomization"). We only change the selection process when comparing without CV selection vs with CV selection, both on a model without regularization for the second two tests listed ("Ablation Study Over Cross-Validation Fitness").

Training Parameters:

Population Size	10
Replaced Individuals (in m-elitist)	5
Iterations	35
Batches	128
Batch Size	64
lr	0.001

Regularization Rate	0.001
Optimizer	Adam

Randomization Parameters:

lr_constant	10**(random.normal(mu=-4, sigma=2))
regularization_constant	10**(random.normal(mu=0, sigma=2))
randomization_amount (amount to randomize model, changes based on model's performance)	1 - (2 / (2 + model loss))
Gaussian Noise for Model Weights (sum of current weights and noise)	noise = random.normal(mu=0, sigma=0.01)*randomization_amount
Gaussian Noise for lr_constant (product of current lr_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))
Gaussian Noise for regularization_rate (product of current regularization_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))

Hyperparameter Sensitivity Study: We conduct this study on the CIFAR-10 dataset, on the same model used in the CIFAR-10 benchmark tests without regularization. When testing for the effects of changing the number of iterations vs different learning rates, we keep everything constant, except for changing the hyperparameter we are testing by sampling each possibility in the arrays listed for that category.

Training Parameters:

Population Size	10
Replaced Individuals (in m-elitist)	5
Iterations	**[10, 30, 50]**
Batches	128
Batch Size	64
lr	**[0.001, 0.01, 0.05]**
Regularization Rate	0.001

Optimizer	Adam

Randomization Parameters:

lr_constant	10**(random.normal(mu=-4, sigma=2))
regularization_constant	10**(random.normal(mu=0, sigma=2))
randomization_amount (amount to randomize model, changes based on model's performance)	1 - (2 / (2 + model loss))
Gaussian Noise for Model Weights (sum of current weights and noise)	noise = random.normal(mu=0, sigma=0.01)*randomization_amount
Gaussian Noise for lr_constant (product of current lr_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))
Gaussian Noise for regularization_rate (product of current regularization_constant and noise)	2**(np.random.normal(mu=0, sigma=randomization_amount*15))

ESGD:

Training Parameters:

n_population	5
sgds_per_gen	1
evos_per_gen	1
reproductive_factor	4
m_elite	3
mixing_number	3
optimizer_class	SGD
learning_rate	**[0.001, 0.01, 0.05]**
n_generations	**[1, 3, 5]**
batch_size	8