Neural Network Hyperparameter Optimization with Sparse Grids

Maximilian Michallik

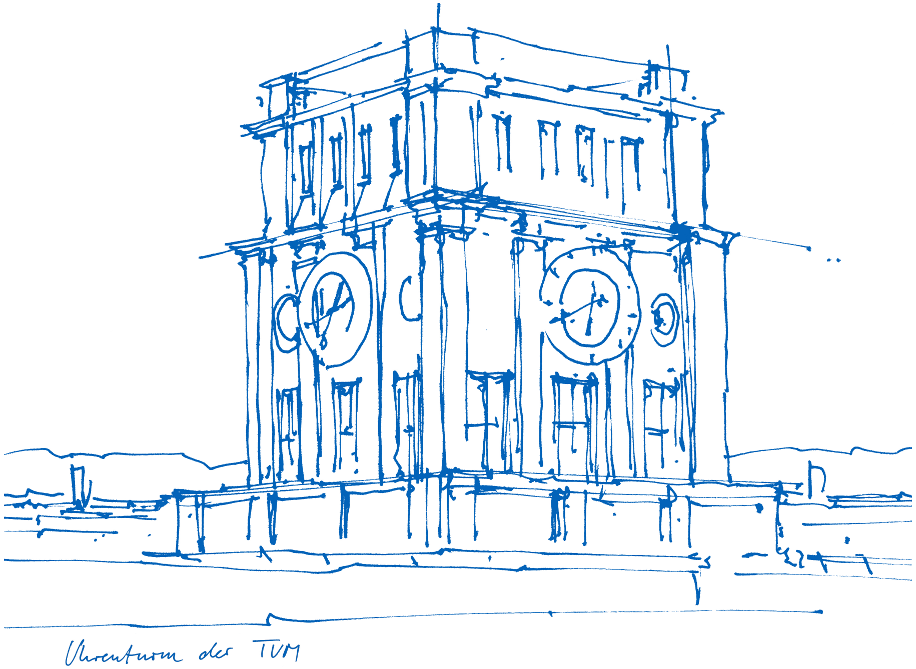
Technichal University Munich

Chair of Scientific Computing

Department Computer Science

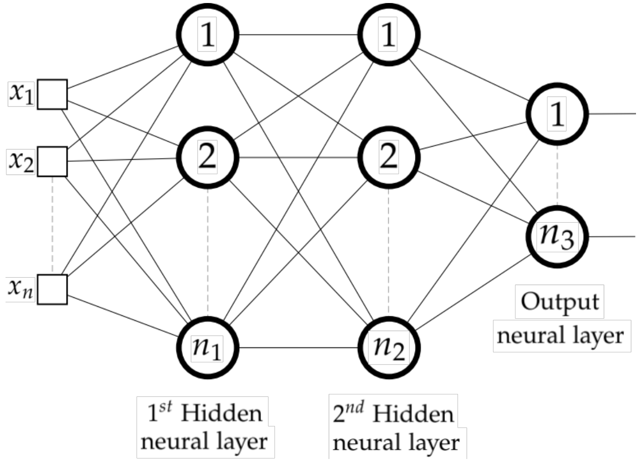
TUM School of Computation, Information and Technology

Garching, 02. August 2023



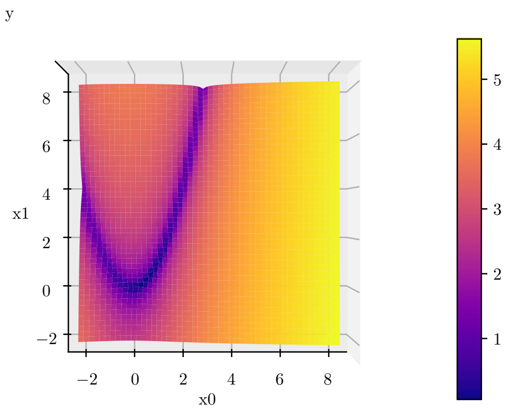
Hyperparameter Optimization

- **Hyperparameters:** Paramaters of a ML model that are fixed before training



Number of layers/ neurons, epochs, learning rate, ...

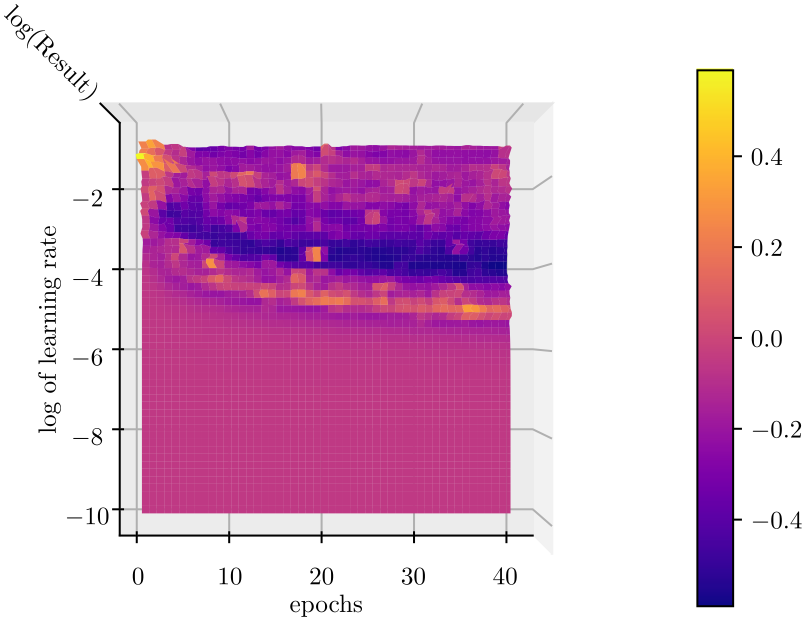
- **Optimization:** finding the optimum of a function



Rosenbrock: Optimum at (1, 1)

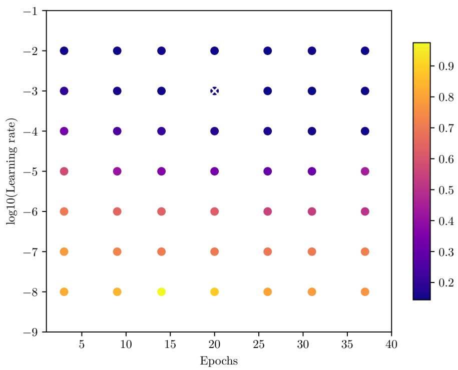
Hyperparameter Optimization

Regression of 2-layer neural network:

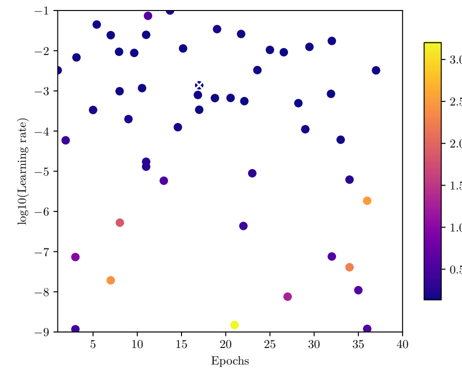


Hyperparameter Optimization

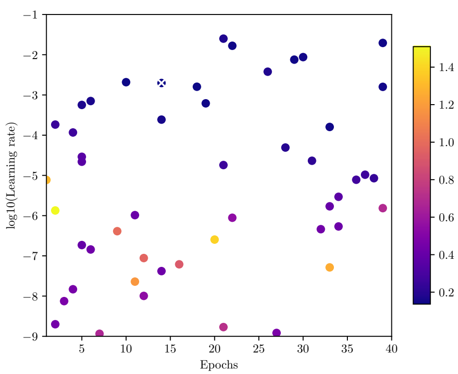
Grid search:



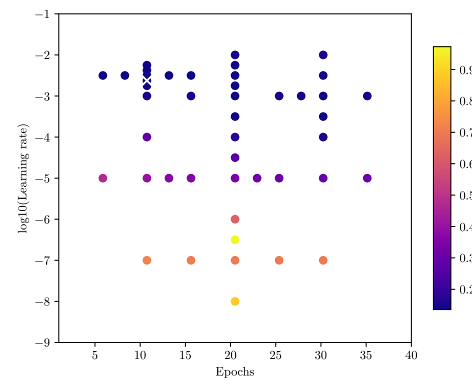
Bayesian optimization:



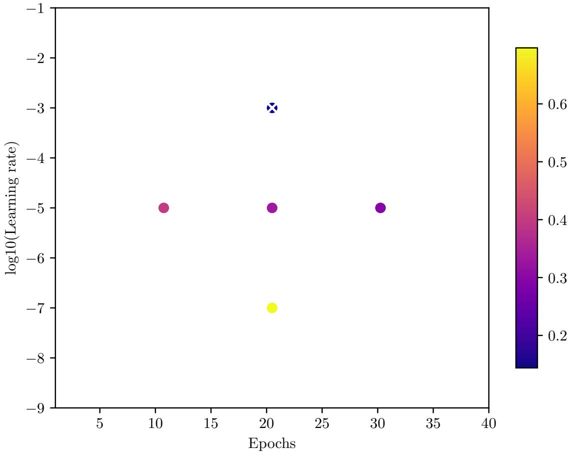
Random search:

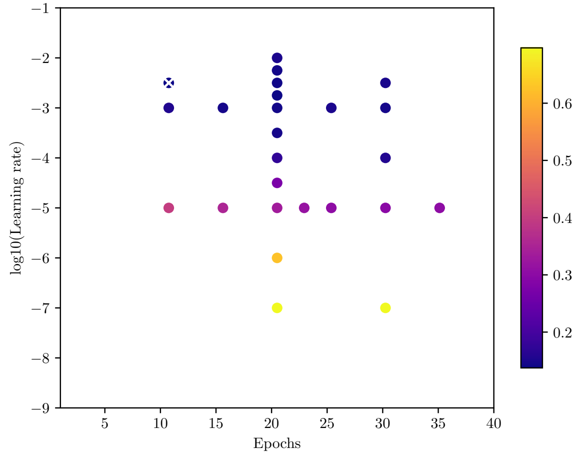


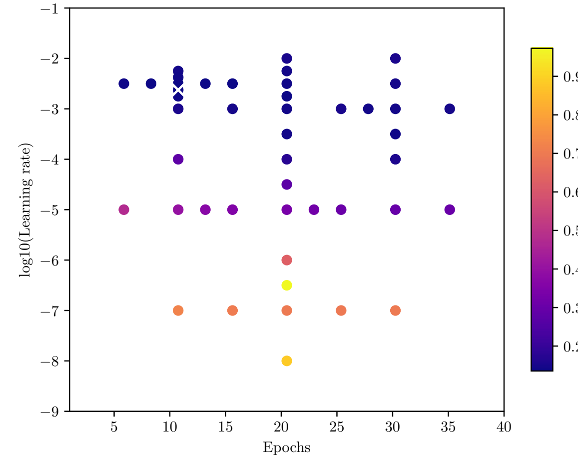
Sparse grid search:



Sparse Grid Hyperparameter Optimization







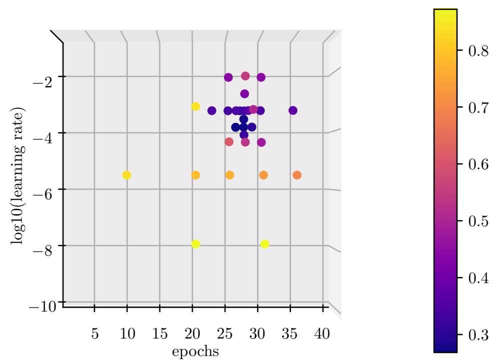
With Novak-Ritter refinement criterion: $\left(r_{l,i}+1\right)^{1-\gamma}\cdot\left(\|l_1\|+d_{l,i}+1\right)^{\gamma}$

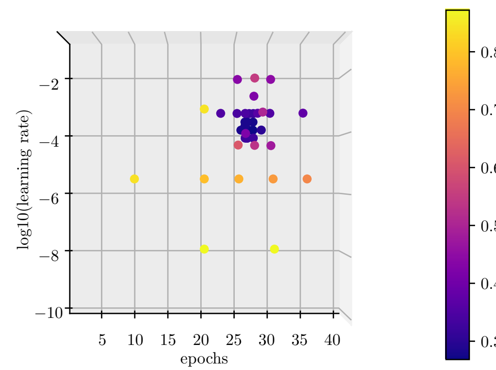
Adaptivity Parameter γ

Budget = 30

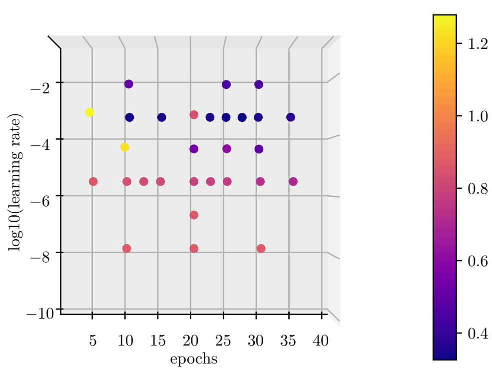
Budget = 50

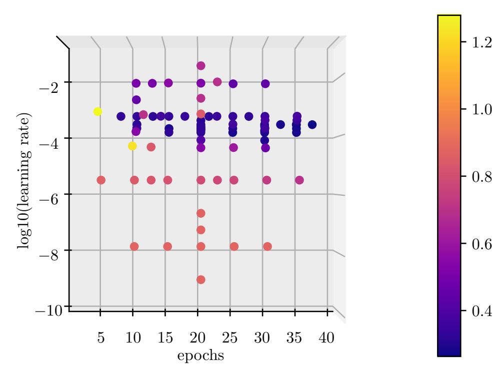
γ = 0.5





γ = 0.85

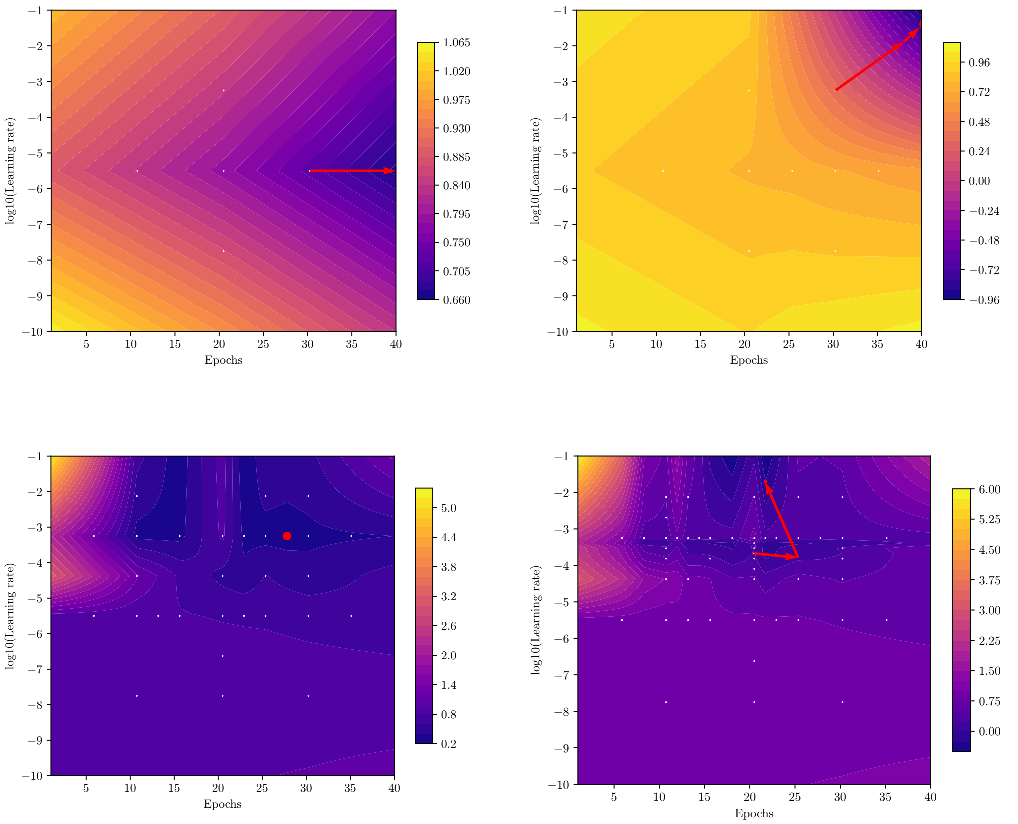




Sparse Grid Hyperparameter Optimization

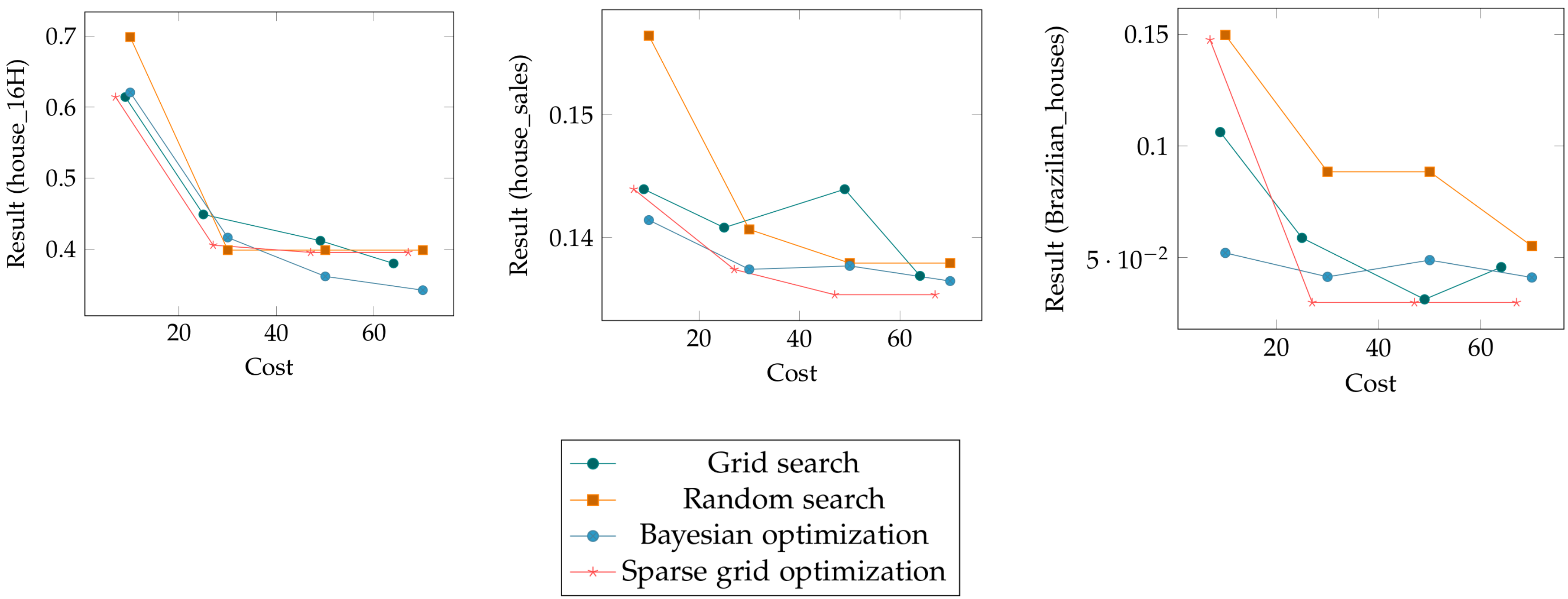
- **Gradient-free optimizers:** Nelder Mead, differential evolution, CMA-ES

- **Gradient-based optimizers:** <u>Gradient descent</u>, NLCG, Newton, Rprop



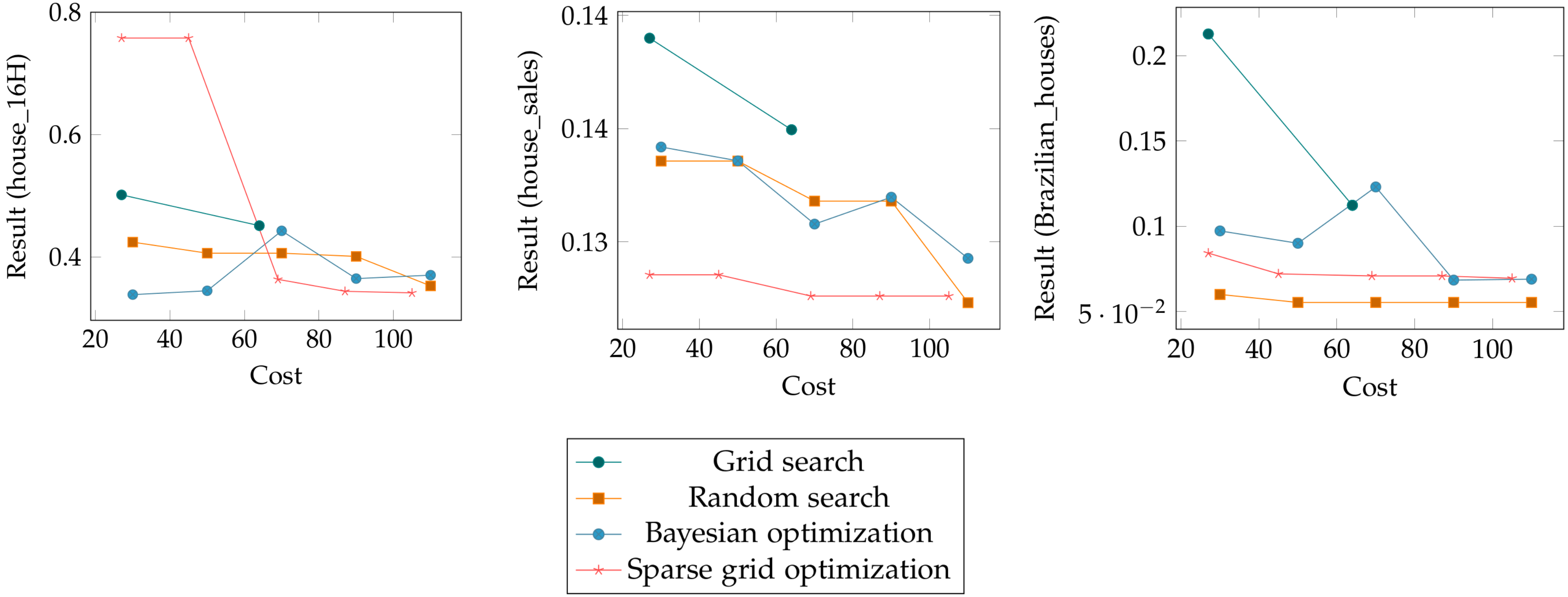
Numerical Results

2D Optimization: Epochs, Learning rate



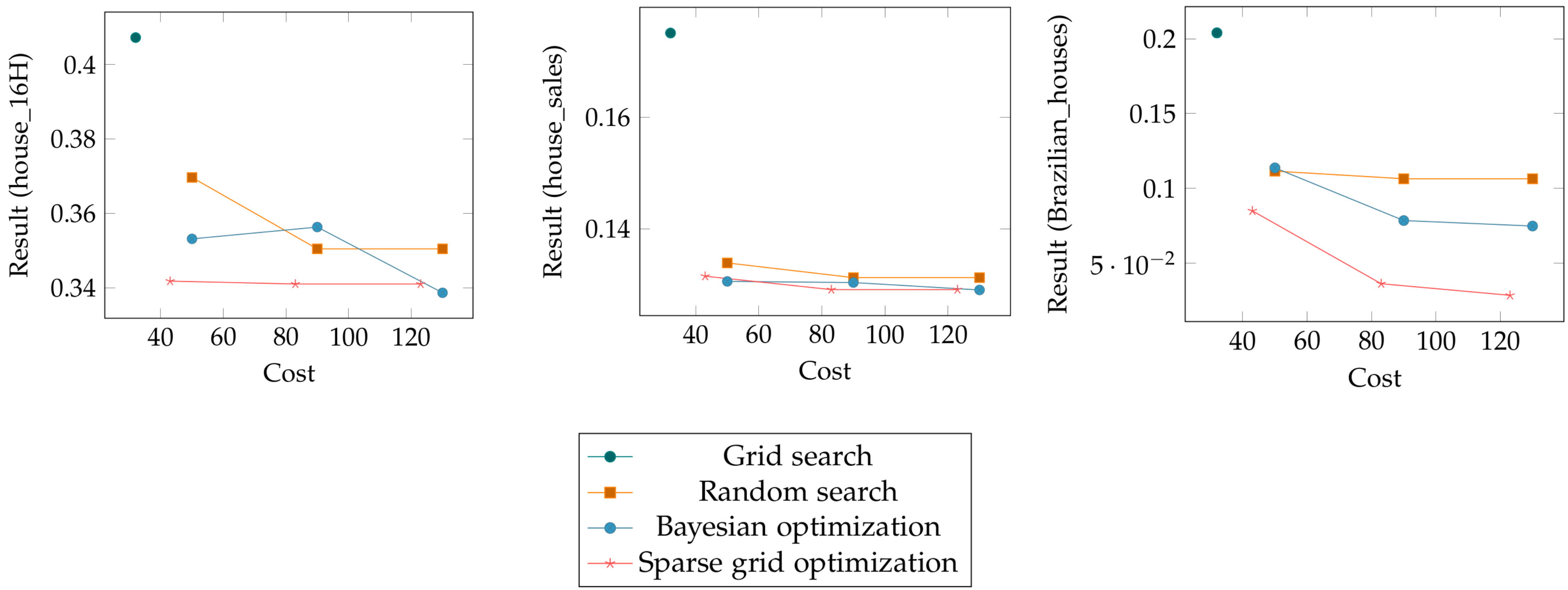
Numerical Results

3D Optimization: Epochs, Learning rate, Batch size



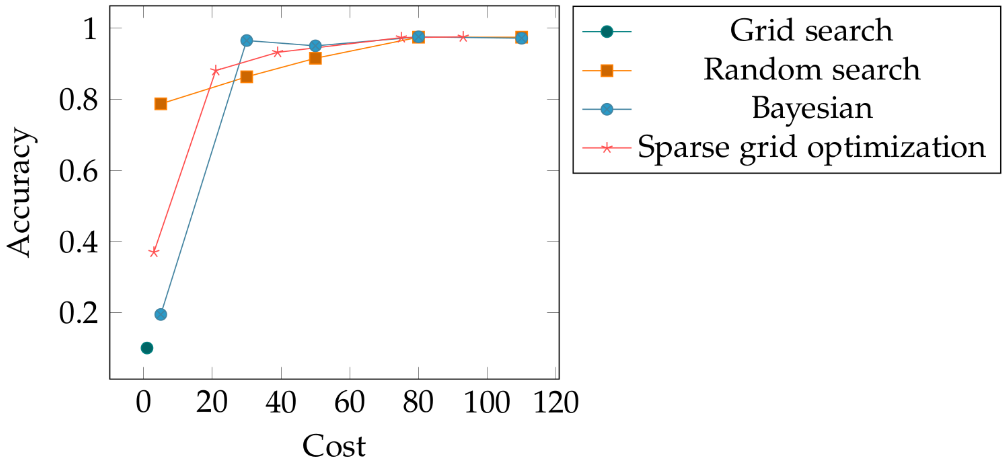
Numerical Results

5D Optimization: Epochs, Learning rate, Batch size, Number of layers & neurons per layer



Application: MNIST

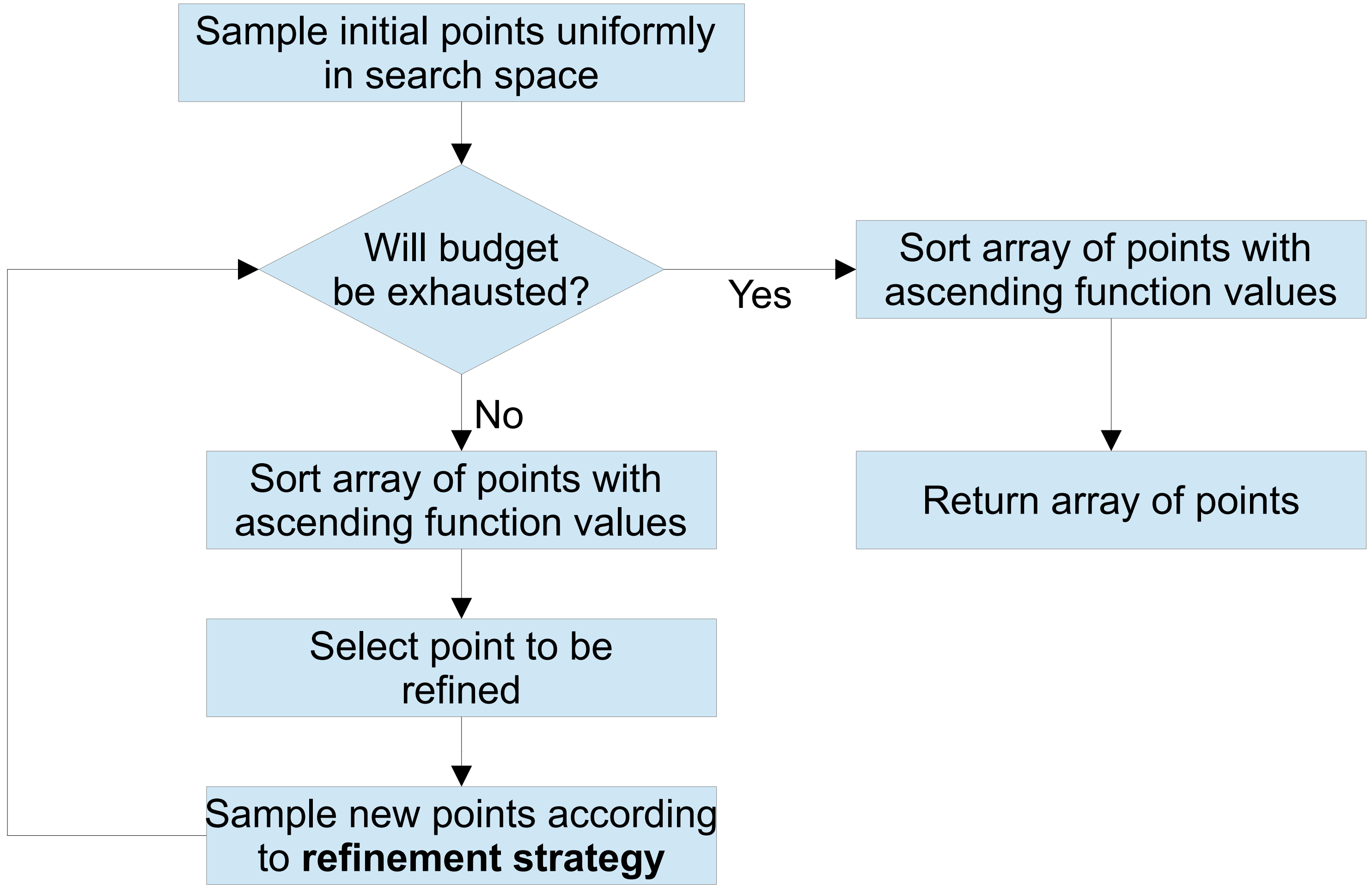
9D Optimization:
- Epochs
- Batch size
- Learning rate
- Number of convolutional Layers
- Number of fully connected layers
- Kernel size
- Pool size
- Neurons per fully connected
- Dropout probability



Algorithm	Configuration	Accuracy	Cost
GS	$(5, 600, 10^{-6}, 2, 2, 2, 2, 4, 0.5)$	10.1%	1
RS	$(9, 975, 0.0173, 2, 1, 3, 1, 6, 0.619)$	97.4%	80
BO	$(6, 584, 10^{-2.17}, 2, 1, 3, 1, 5, 0.281)$	97.5%	80
SG	$(7, 400, 10^{-2}, 2, 2, 2, 2, 5, 0.5)$	97.5%	93

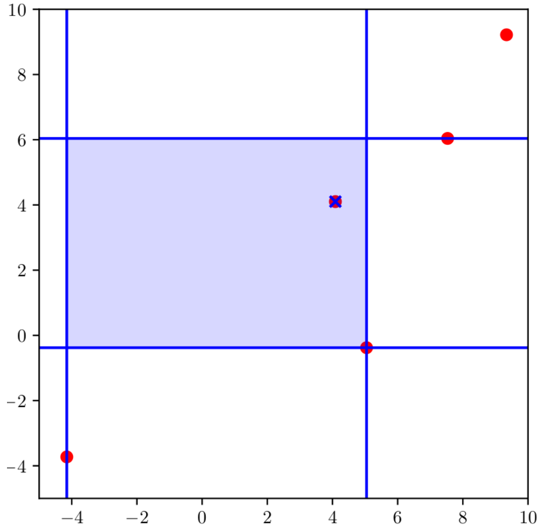
Iterative Adaptive Random Search

Algorithm:

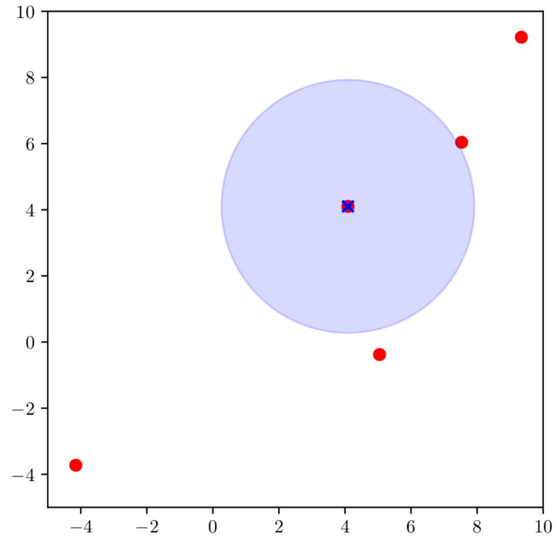


Refinement Strategies

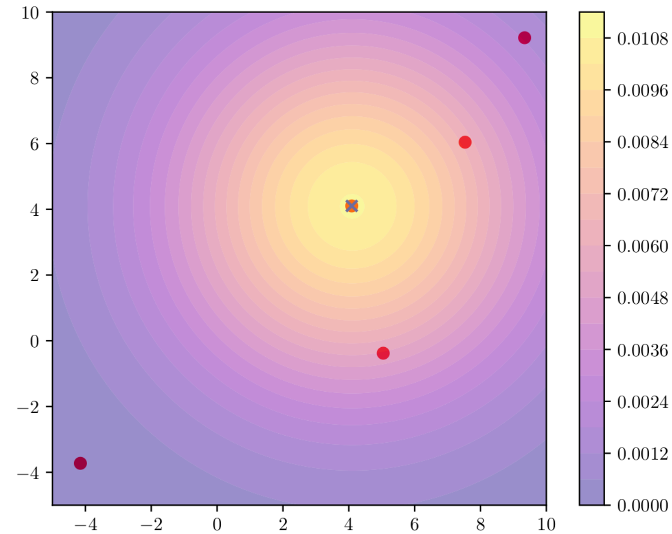
Interval based refinement



Uniform d-ball sampling



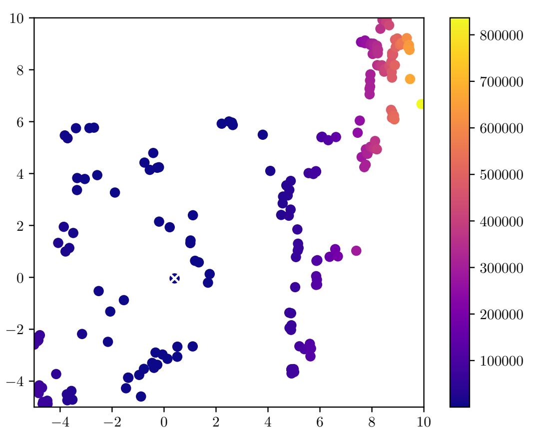
Normal distribution sampling



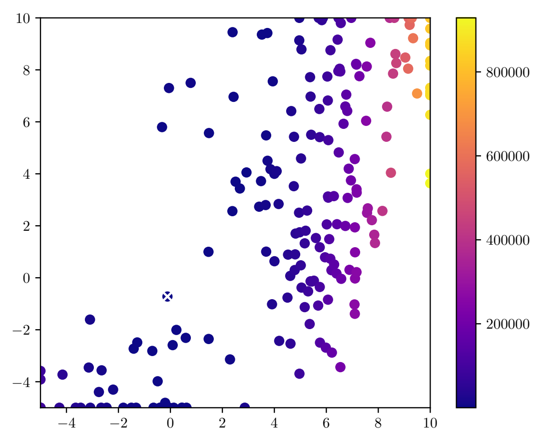
Refinement criterion: $(rank_i+1)^{1-\gamma}\cdot(level_i+refinements_i+1)^{\gamma}$

Adaptivity parameter γ = 1.0

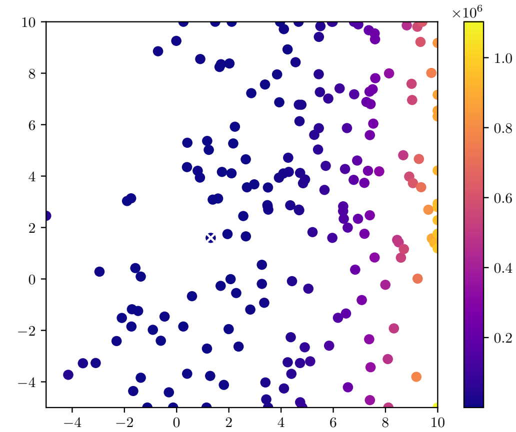
Interval based refinement



Uniform d-ball sampling



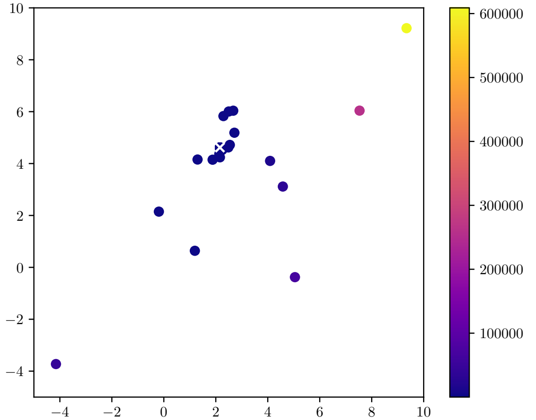
Normal distribution sampling



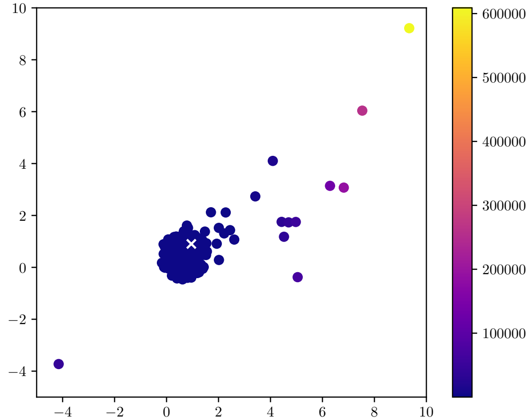
Refinement criterion: $$\underbrace{(rank_i+1)^{1-\gamma}}_{1} \cdot (level_i + refinements_i + 1)^{\gamma}$$

Adaptivity parameter γ = 0.0

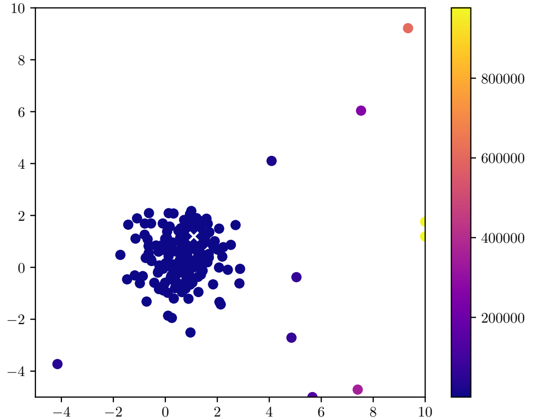
Interval based refinement



Uniform d-ball sampling



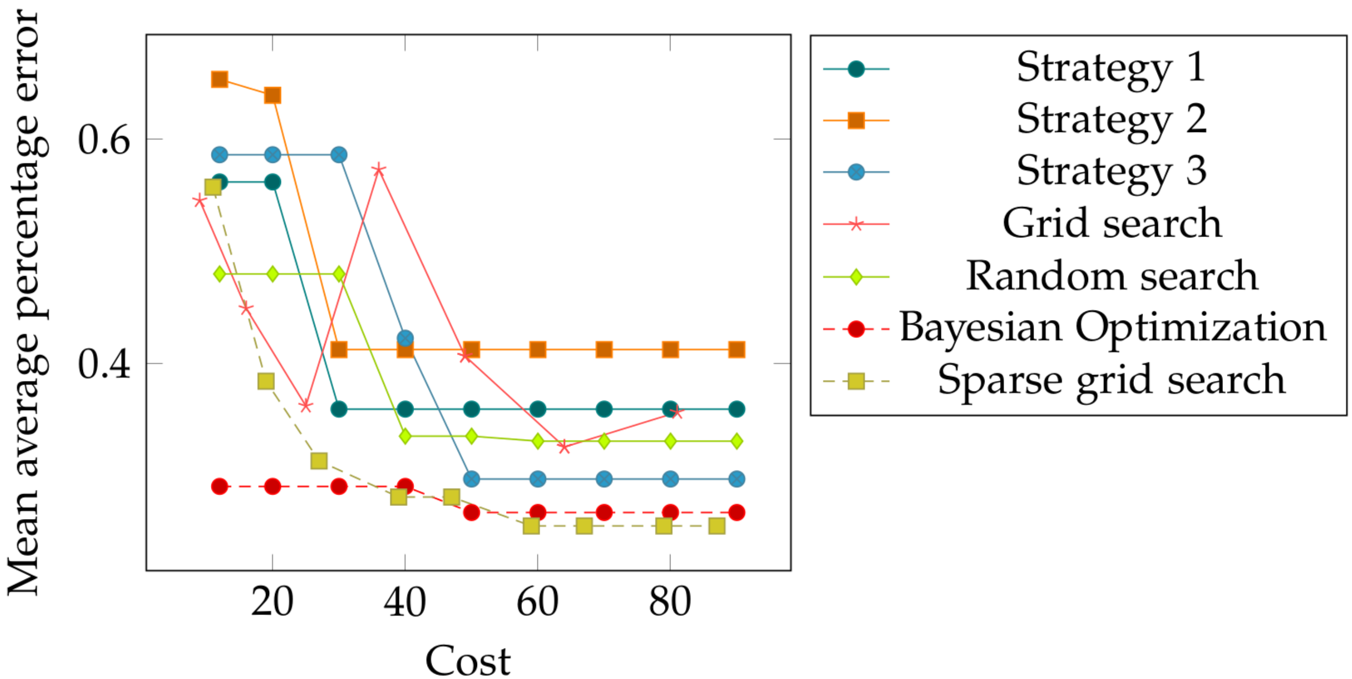
Normal distribution sampling



Refinement criterion: $(rank_i + 1)^{1-\gamma} \cdot \underbrace{(level_i + refinements_i + 1)^{\gamma}}_{\mathbf{1}}$

Comparison of Optimization Algorithms

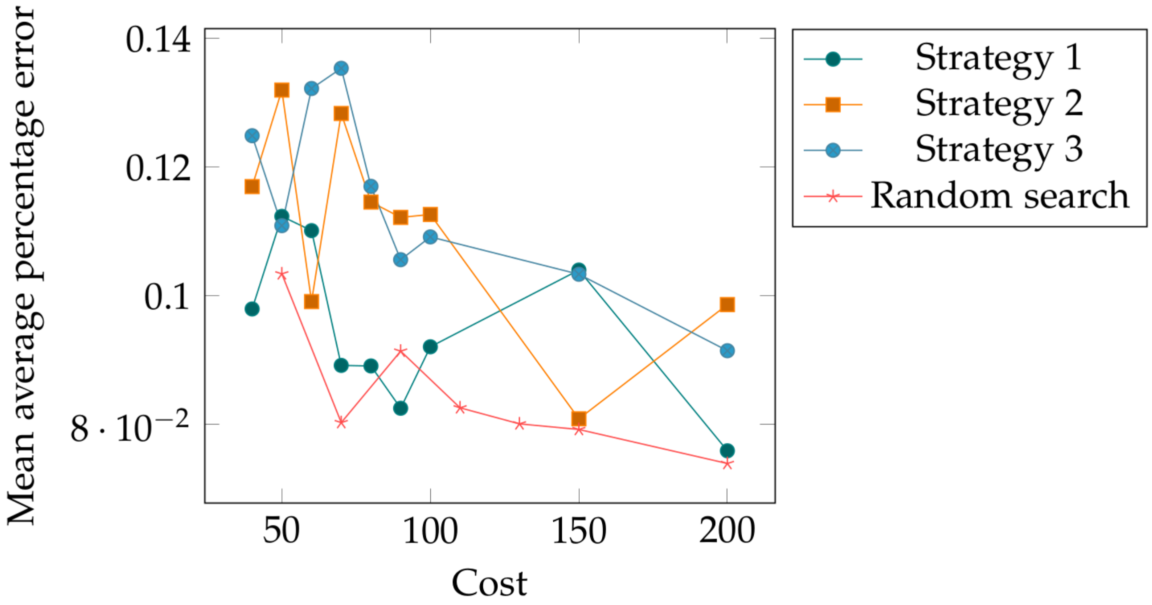
2D Optimization: Epochs & Learning rate



Comparison of Optimization Algorithms

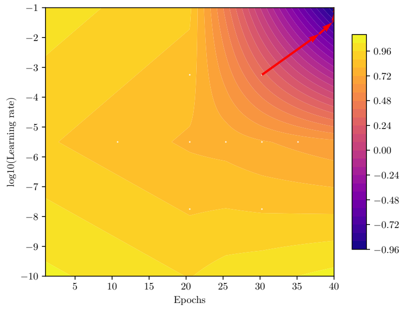
6D Optimization:

- Epochs
- Batch size
- Learning rate
- Number of layers
- Neurons per layer
- Dropout probability

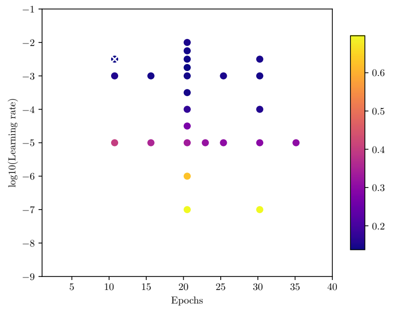


Discussion – Sparse Grid Hyperparameter Optimization

- **Often not enough grid points for optimizer**
  → apply global and local optimizer and compare with the best grid point



- **Problem if optimum at the boundary**
  → optimizer can sometimes help or use sparse grid with boundary points

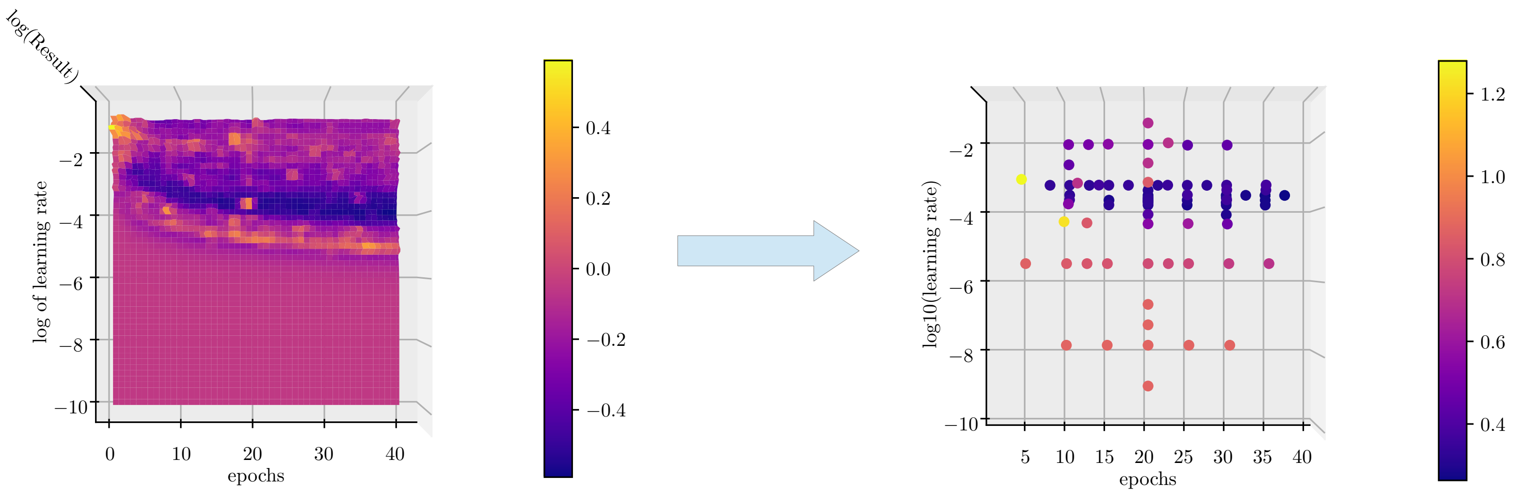


- **BUT: time to solution most important**

Discussion – Iterative Adaptive Random Search

- Promising results in small dimension
- **Problem:** Volume of d-ball compared to d-hypercube decreases rapidly for increasing d
  → worse performance than conventional random search for higher dimensions
- More analysis:
  - Convergence?
  - Adapt algorithm?
  - Other search radius?
  - Change refinement criterion (e.g. add function value)?
  - Other refinement strategy?

Conclusion


log(Result)
log of learning rate
−2
−4
−6
−8
−10
0
10
20
30
40
epochs
0.4
0.2
0.0
−0.2
−0.4
log10(learning rate)
−2
−4
−6
−8
−10
5
10
15
20
25
30
35
40
epochs
1.2
1.0
0.8
0.6
0.4