AUTOMATIC HYPERPARAMETER TUNING FOR DEEP LEARNING

LMU MUNICH

WS 16/17

SPEAKER : CHUNG SHING REX HA

AUTOMATIC HYPERPARAMETER TUNING FOR DEEP LEARNING

- Introduction
- Hyperparameter tuning
- Methods of automatic tuning
    - Grid Search
    - Random Search
    - Model-based Optimisation
- Hyperparameter tuning for neural network
- Practical Evaluations
- Conclusion

Introduction

What is a hyperparameter?

- Model parameters → are optimised during training phase

  e.g. β in $y = X\beta$

- Hyperparameters →
  - are specified before the training algorithm starts
  - can't be optimised inside the training algorithm itself

  e.g. number of hidden units each layer

Model parameter

Hyperparameter

Polynomial regression $$y = \sum \beta_i x^i$$

Types of hyperparameters

- Nummerical
  - depth of decision tree
  - Regularisation coefficient
- Categorial
  - Types of kernel
  - Split criterion for trees
- Ordinal
  - {low, medium, high}
- Conditional
  - Kernel parameter, depending on selected kernel e.g. degree of polynomial kernel

Why should hyperparameters be tuned?

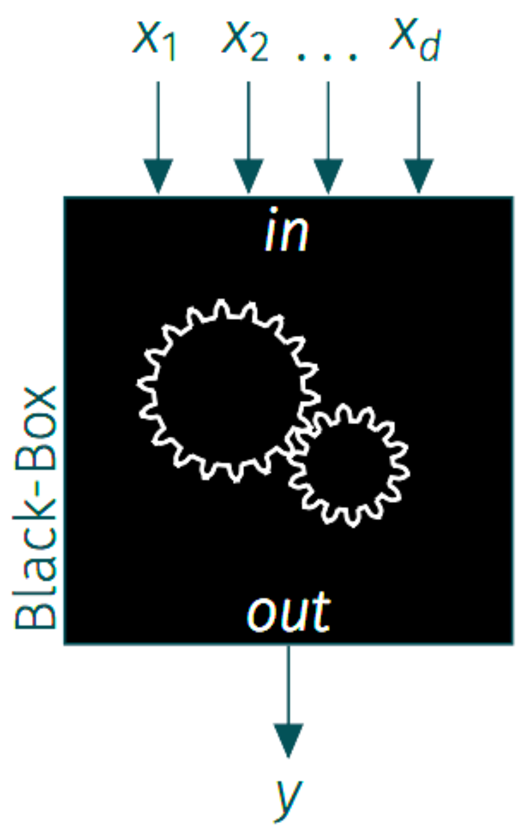
- They control the behaviour of the training algorithm and have significant effect on he performance of the models being trained.
- Some control the ***capacity*** of the model, i.e. ***flexibility*** of models, e.g. max. depth of a decision tree
- If the models are too flexible, it leads to ***overfitting***.
- In order to prevent overfitting, proper setting of hyperparameter, i.e. tuning, is required.

Hyperparameter Tuning

Concept of Hyperparameter Tuning

$$y = f(\boldsymbol{x}) \ , \quad f : \mathbb{X} \to \mathbb{R}$$

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathbb{X}} f(\boldsymbol{x})$$



- y, target value
- $\boldsymbol{x} \in \mathbb{X} \subset \mathbb{R}^d$, domain
- $f(\boldsymbol{x})$ function with considerably long runtime
- Goal: Find optimum $\boldsymbol{x}^*$

Goal of tuning : to find the lowest generalization error subject to some runtime and memory budget

https://www.researchgate.net/profile/Jakob_Richter/publication/308874347_mlrMBO_A_Toolbox_for_Model-Based_Optimization_of_Expensive_Black-Box_Functions/links/57f3c6d108ae280dd0b73154.pdf

Manual Tuning vs Automatic Tuning

Manual Tuning

- trying out some hyperparameter combinations based on prior experience until settling on a good one
- not efficient for multiple hyperparameter tuning, because it is difficult for human to imagine high dimensional space
- reproducibility of hyperparameters is weak

Automatic Tuning

- Hyperparameters are tuned by an algorithm without hand-tuning
- Hyperparameter optimization algorithms warping a learning algorithm and choosing its hyperparameters

Methods of Automatic Tuning

Grid Search

- the most popular and easiest method
- a finite set of values is predefined for each parameter
- search the Cartesian product of all possible combinations

Two hyperparameters

$$\alpha \in \{1,2,3,4,5\} \qquad \beta \in \{1,2,3\}$$

Possible combinations:

$$\{\alpha,\beta\} \in \left(\begin{array}{c} \{1,1\}\,,\{1,2\}\,,\{1,3\}\,,\{2,1\}\,,\{2,2\}\,,\{2,3\}\,,\{3,1\}\,, \\ \{3,2\}\,,\{3,3\}\,,\{4,1\}\,,\{4,2\}\,,\{4,3\}\,,\{5,1\}\,,\{5,2\}\,,\{5,3\} \end{array} \right)$$

Grid Search

Advantages

- Simple to be implemented
- Parallelization

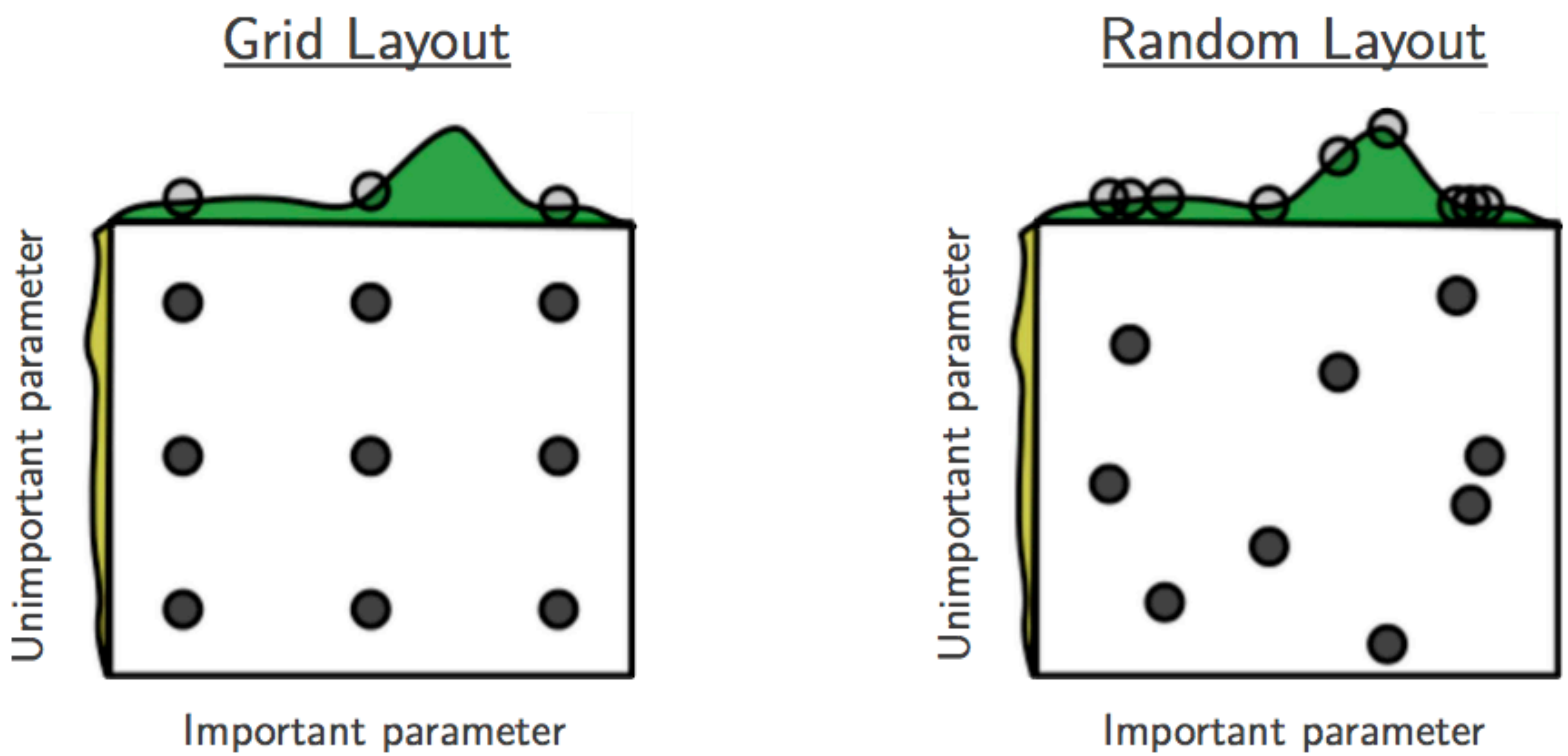
Disadvantages

- High computational cost with high hyperparameter dimension
  - 4 hyperparameters with each 10 values = 10^4 = 10,000 trials
- The search may fall on irrelevant areas

Random Search

- Variation of Grid Search
- Instead of evaluation of all parameter combination, samples are randomly selected independently sampling each hyperparameter from a prior distribution
- The optimization time is greatly reduced and the performances are as good as or even better than Grid Search

Grid Search vs Random Search



Bergstra, J. and Bengio, Y. Random search for hyper-parameter optimization. JMLR, 13:281–305, 2012.

Model-based Optimisation

Evaluation of black box functions are expensive, so we are trying to keep number of evaluations low

- Try to predict function values by regression model → surrogate model
- Search for points leading to finding the optimum on the surrogate model.
- Update surrogate model with evaluated points

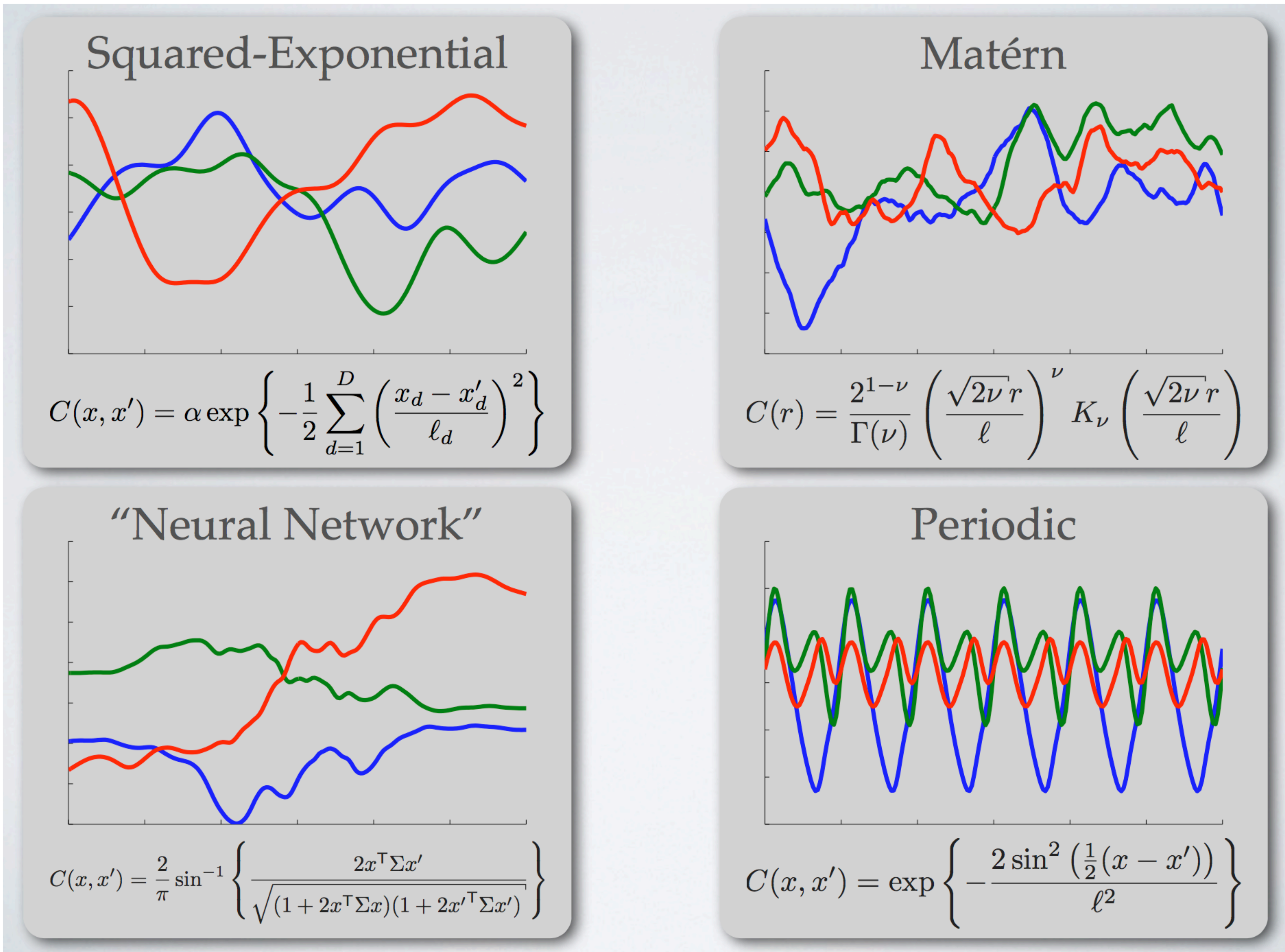
Model-based Optimisation - Gaussian Process

- GP is a distribution on functions
- Any finite set of N points $\{x_n \in \mathcal{X}\}_{n=1}^N$ induces a multivariate Gaussian distribution on $\mathbb{R}^N$, take to be the distribution on

$$\{y_n = f(x_n)\}_{n=1}^N$$

- Model scalar functions $f : \mathcal{X} \to \mathbb{R}$
- Mean function $m : \mathcal{X} \to \mathbb{R}$
- Positive definite covariance function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

Model-based Optimisation - Gaussian Process



https://www.iro.umontreal.ca/~bengioy/cifar/NCAP2014-summerschool/slides/Ryan_adams_140814_bayesopt_ncap.pdf

Model-based Optimisation - Acquisition Functions

- A function that acquires a next point to evaluate for a black-box function
- 3 common acquisition functions
  - Probability of Improvement (Kushner, 1964)
    - closed form under GP

$$a_{PI}\left(x;\{x_n, y_n\},\theta\right) = \Phi\left(\gamma\left(x\right)\right) \qquad \gamma\left(x\right) = \frac{f\left(x_{best}\right) - \mu\left(x;\{x_n, y_n\},\theta\right)}{\sigma\left(x;\{x_n, y_n\},\theta\right)}$$

Model-based Optimisation - Acquisition Functions

- Expected Improvement (Mockus, 1978)
  - closed form under GP

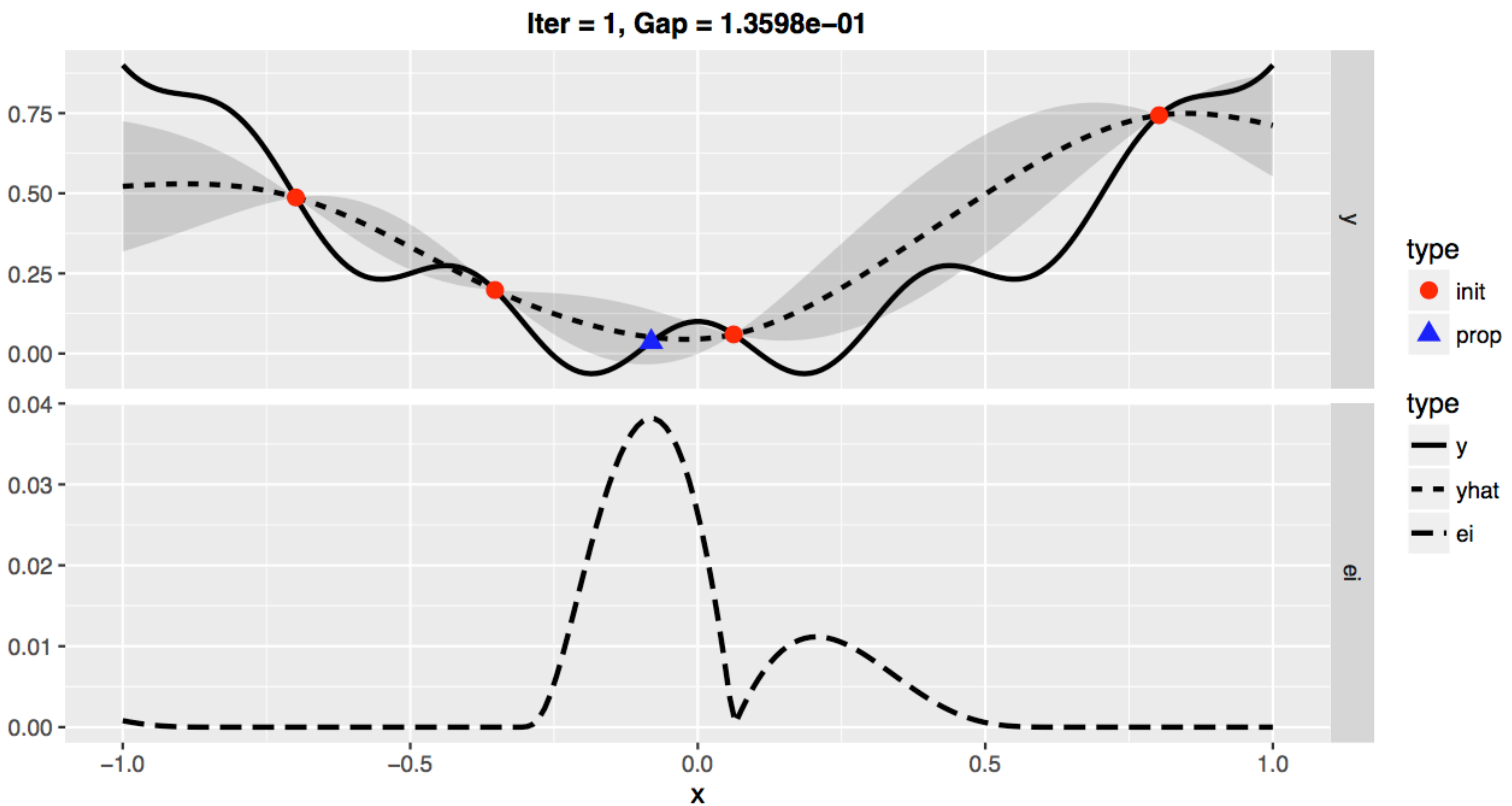
$$a_{EI}\left(x;\left\{x_n,y_n\right\},\theta\right)=\sigma\left(x;\left\{x_n,y_n\right\},\theta\right)\left(\gamma\left(x\right)\phi\left(\gamma\left(x\right)\right)+\mathcal{N}\left(\gamma\left(x\right);0,1\right)\right)$$

- GP Upper Confidence Bond (Srinivas et al. 2010)

$$a_{UCB}\left(x;\left\{x_n,y_n\right\},\theta\right)=\mu\left(x;\left\{x_n,y_n\right\},\theta\right)-\kappa\sigma\left(x;\left\{x_n,y_n\right\},\theta\right)$$

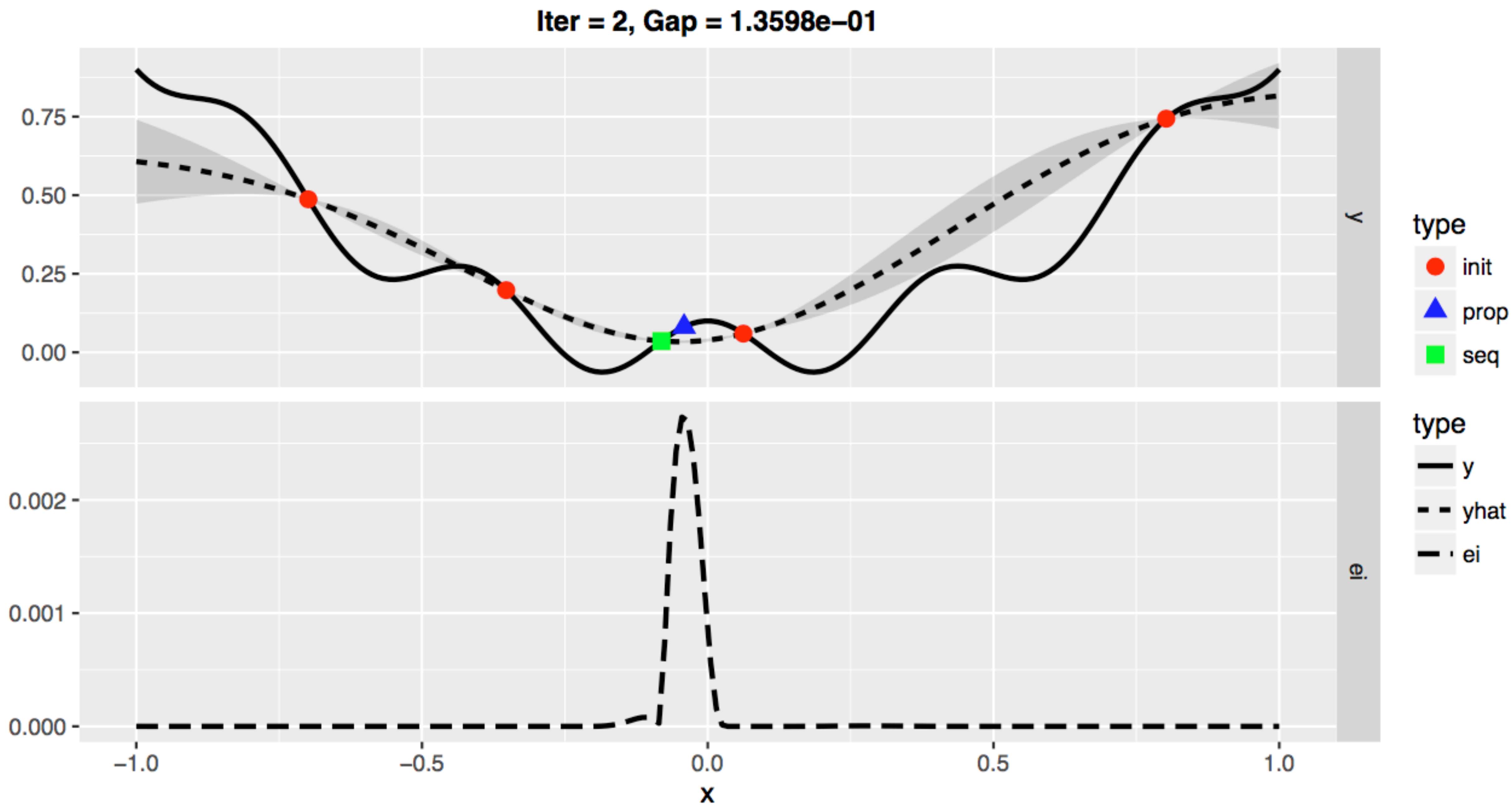
$$\kappa : tunable\ parameter\ to\ balance\ exploitation\ against\ exploration$$

Model-based Optimisation - Acquisition Functions



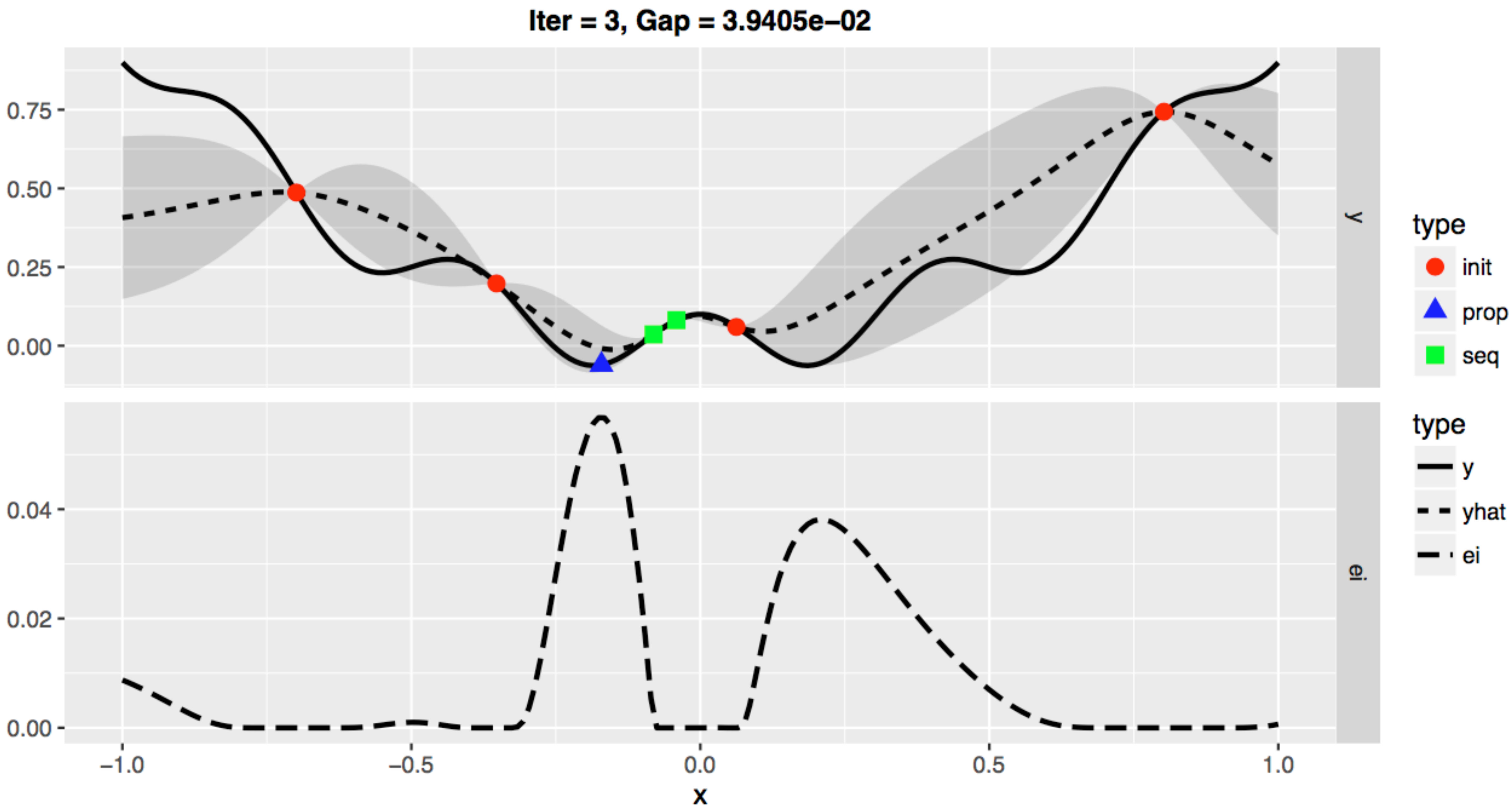
https://github.com/mlr-org/mlrMBO/blob/master/inst/examples/ex_multiobj_plots.R

Model-based Optimisation - Acquisition Functions



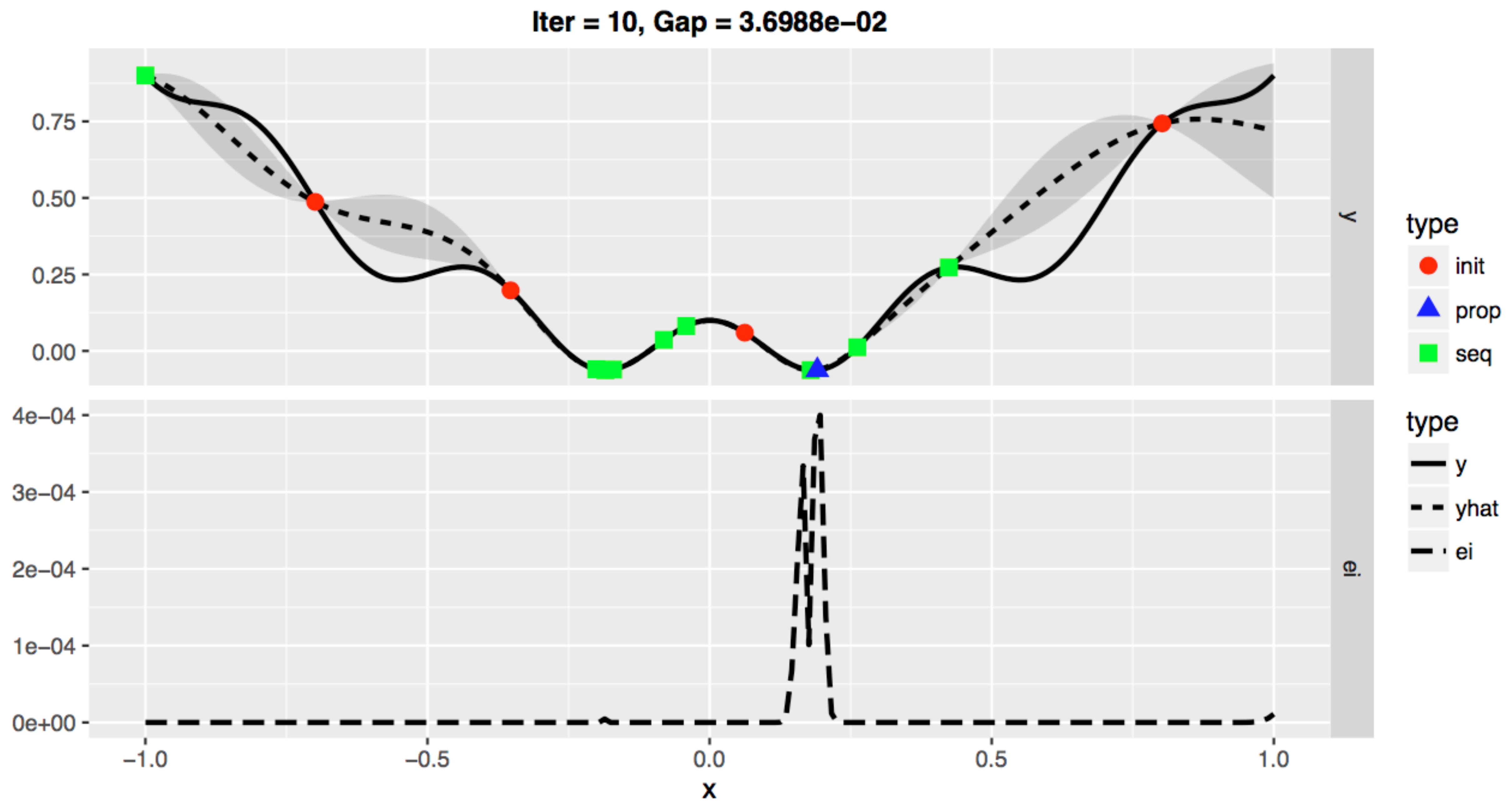
https://github.com/mlr-org/mlrMBO/blob/master/inst/examples/ex_multiobj_plots.R

Model-based Optimisation - Acquisition Functions



https://github.com/mlr-org/mlrMBO/blob/master/inst/examples/ex_multiobj_plots.R

Model-based Optimisation - Acquisition Functions



https://github.com/mlr-org/mlrMBO/blob/master/inst/examples/ex_multiobj_plots.R

Hyperparameter tuning for neural network

Types of hyperparameters for neural network

Hyperparameters
for neural network

associated with
optimizer

- variated with different optimization algorithms

associated with
model itself

- architectural parameters

Types of hyperparameters for neural network

	Hyperparameters associated with optimizer	**Hyperparameters associated with model itself**
stochastic gradient descent	Initial learning rate	Number of hidden units
	Mini-batch size	Weight decay regularization coefficient
	Number of training iteration	Sparsirty of activation regularization coefficient
	Momentum	Neuro non-linearity
Conjugate Gradient algorithms	Number of search steps	Weights initialisation scaling coefficient
	tolerance for stopping each line search	Random seeds

Goal of tuning

1. to minimize training and testing errors
2. to prevent form overfitting or underfitting

Tuning Technique - Validation

- For any hyperparameter that has an impact on the effective capacity of a learner.

→ Hyperparameters are always maximized to reach the maximum capacity, when the training data set is used for hyperparameter tuning

→ Overfitting

- So it makes more sense to select its value based on out-of-sample data (outside the training set)

e.g., a validation set performance, online error, or cross-validation error.

Tuning Technique - Layer-wise optimization of hyperparameters

- Hyperparameters in each layer are tuned separately
- It preforms greedy choice of the hyperparameters associated with lower layers (near the input) before training the higher layers.
- Optimal hyperparameter values in Nth layer will be considered as a starting point of tuning for (N+1)th layer.
- Keeping only the best configuration

Practical Evaluation

Data

Wine date set from University of California Irvine Machine Learning Repository

Description:

Classification of wines grown in the same region but from three different cultivars, through analysis of the quantities of 13 components, z.B. Alcohol, Malic acid, Magnesium, etc., found in each of the three types of wines.

Number of instances: 178

Programme and packages used: mlr and mlrMBO in R

Optimization algorithm used:

Support vector machines with Gaussian radial basis function kernel

Resampling methods used:

Cross-validation

Hyperparameter being tuned:

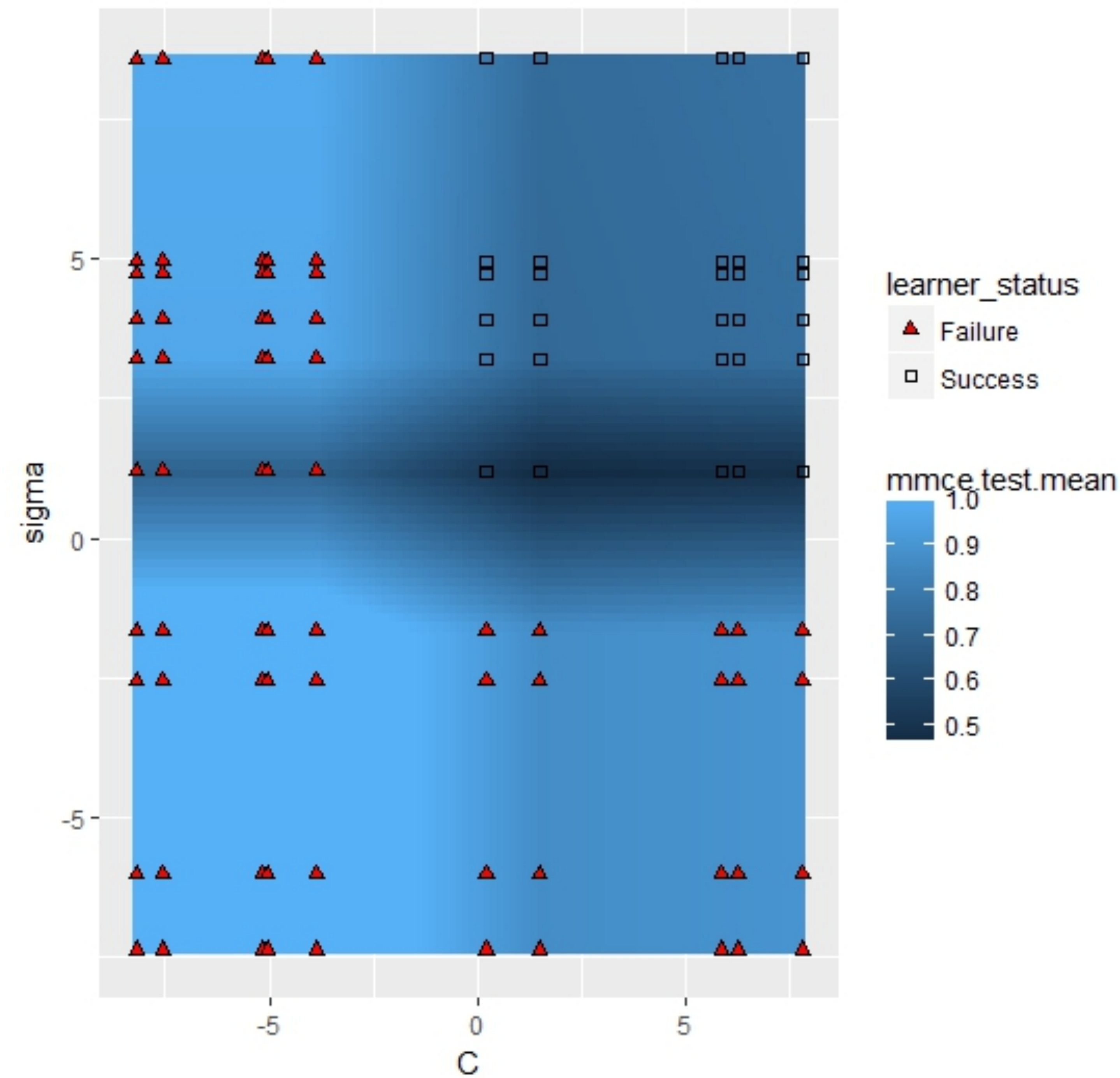
Regulation parameter C and kernel’s parameter sigma

Tuning Methods:

Grid Search, Random Search, MBO

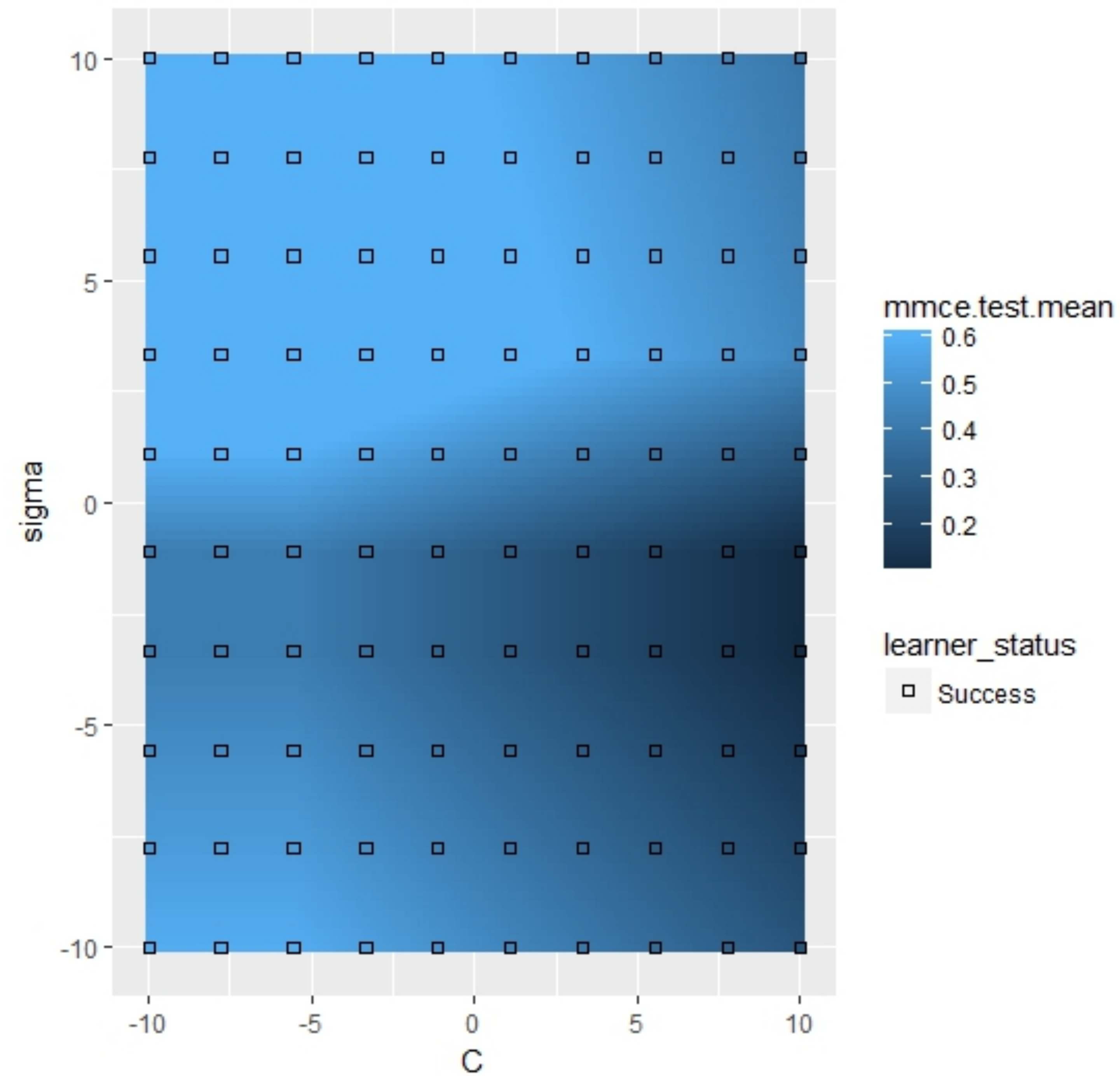
Grid Search - 1

- 10 values are randomly selected between -10 and 10 in advance and assigned to both hyperparameters
- 10 x 10 combinations (points) undergo the tuning



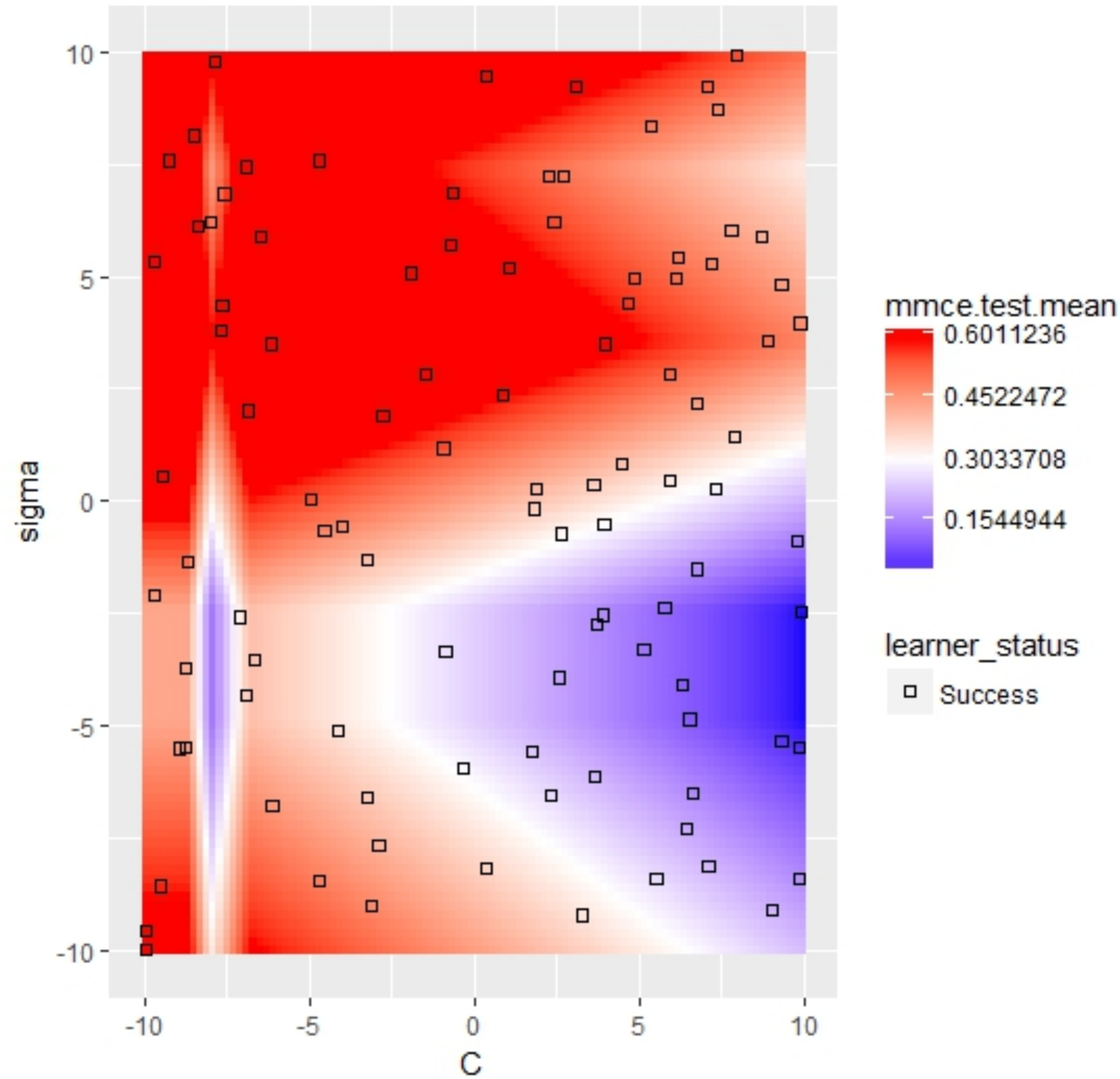
Grid Search - 2

- 10 x 10 combinations (points) are evenly selected from the parameter space



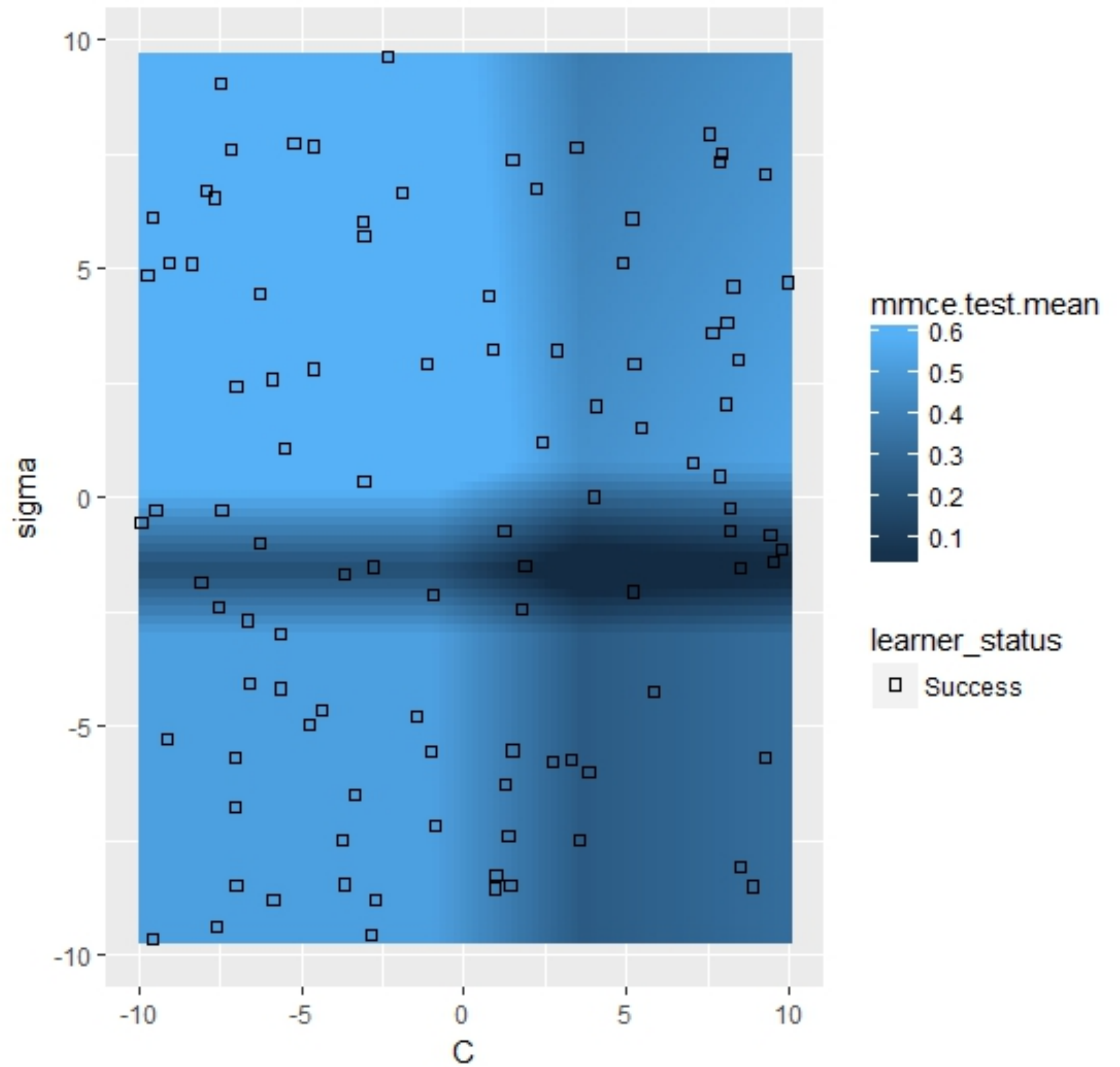
Random Search

- Range of both hyperparameter is set between -10 and 10
- 100 iterations are carried out

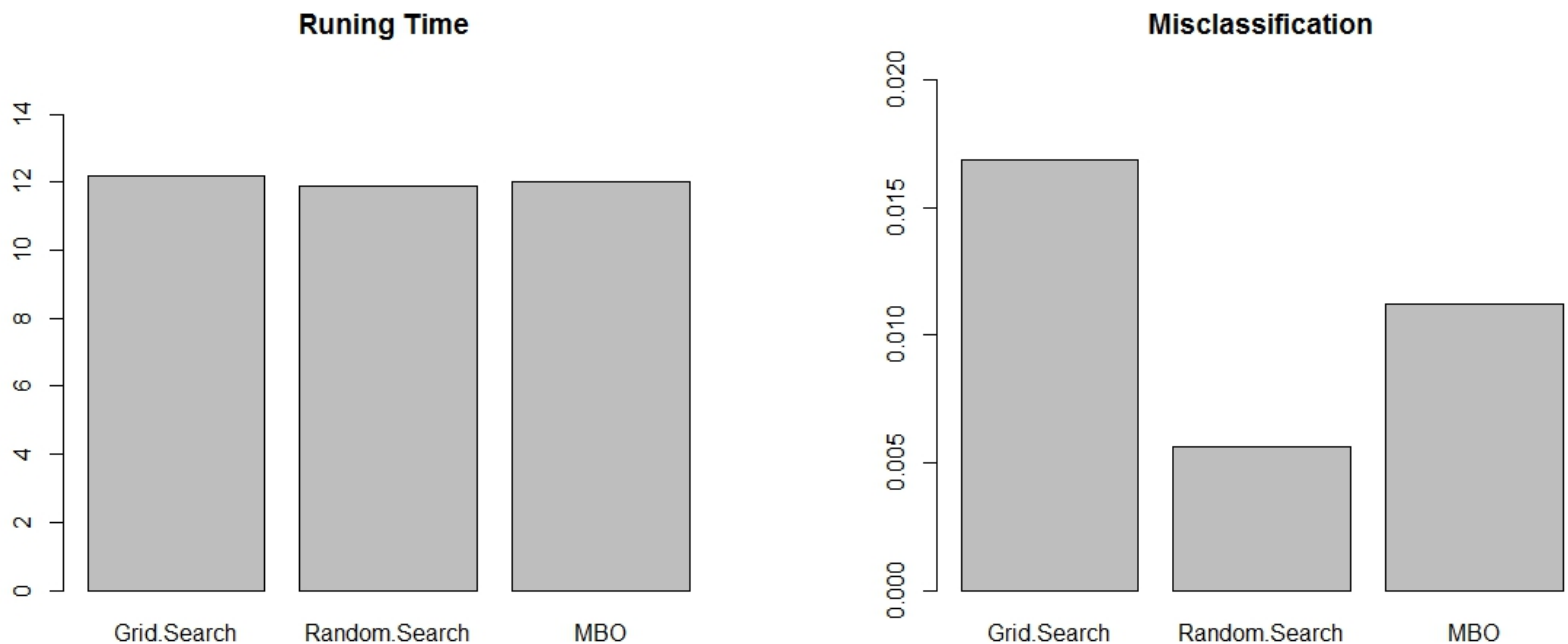


MBO

- Gaussian Process is used as prior distribution
- Expected Improvement is as selection criteria
- 100 iterations are carried out

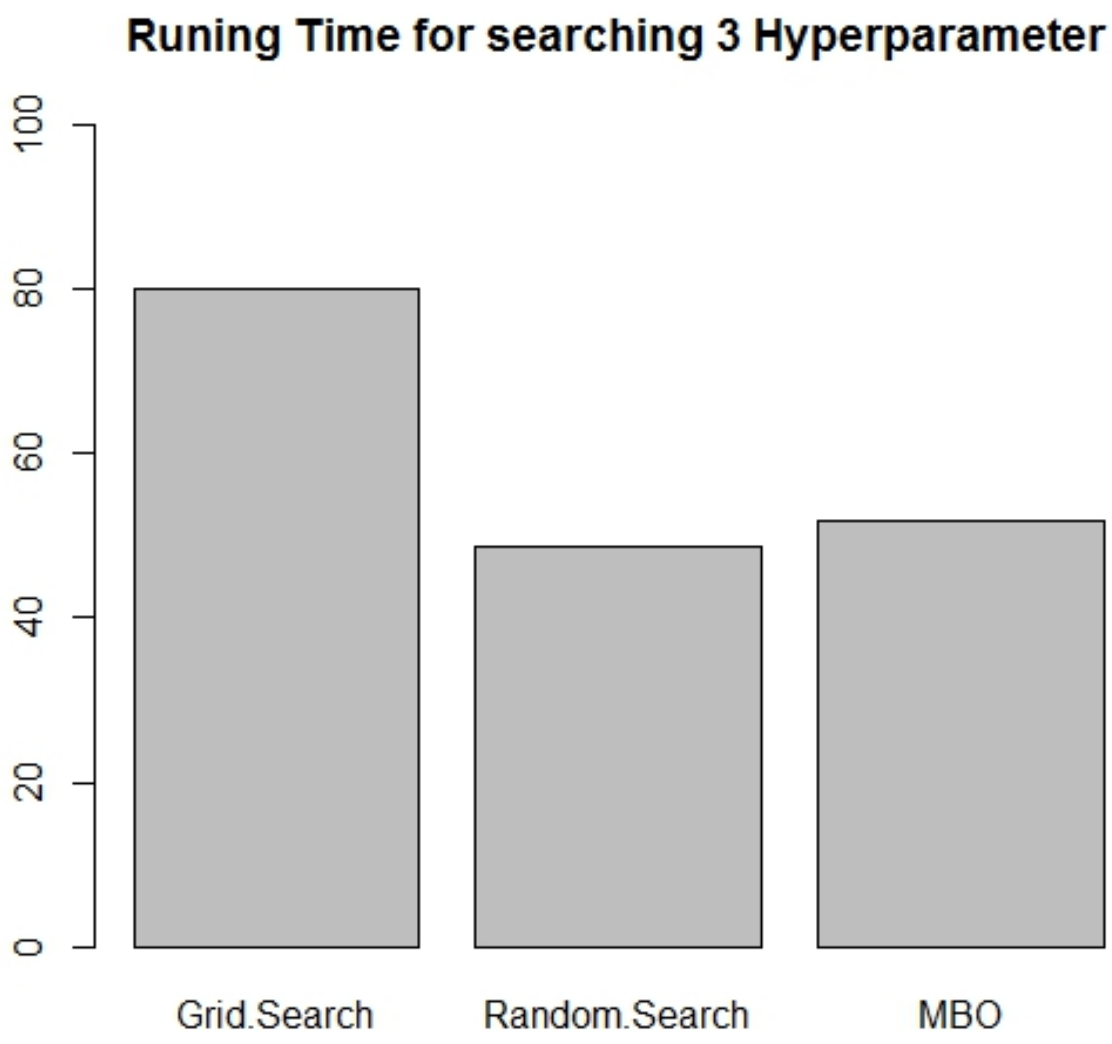


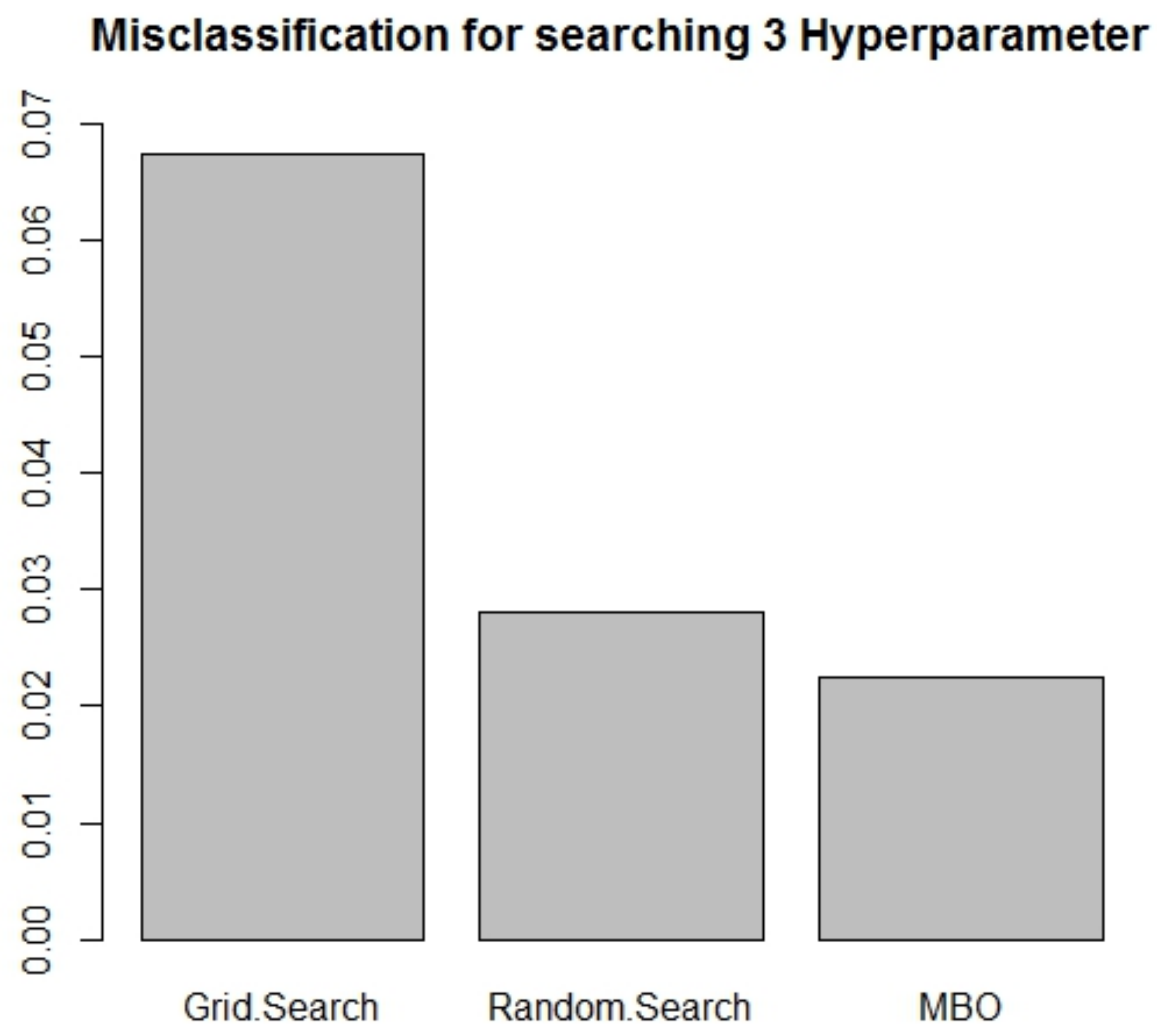
Comparison of running time and accuracy - 1



Comparison of running time and accuracy - 2

Replacing kernel with Bessel kernel and adding hyperparameter degree of kernel





Conclusion

Conclusion

Hyperparameter tuning is a problem of optimization of black box functions

Hyperparameters can be tuned manually or automatically by algorithms:

- Grid Search
- Random Search
- Model-based Optimization - GP

Two types of hyperparameters in neural network: associated with optimizer & with model

Tuning Technique: Using validation data set & layer-wise optimization of hyperparameters

Conclusion

No universal optimal hyperparameters for all tasks or models

Further discussions:

Instead of mentioned 3 methods, there are lots of tuning methods, like evolutionary algorithm or different surrogate models

Hyper-hyperparameter?