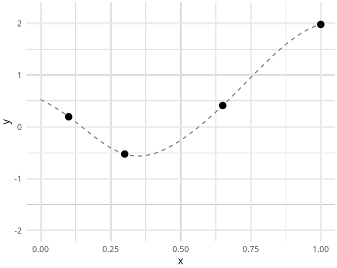
Optimization in Machine Learning

Bayesian Optimization
Basic BO Loop and Surrogate Modelling



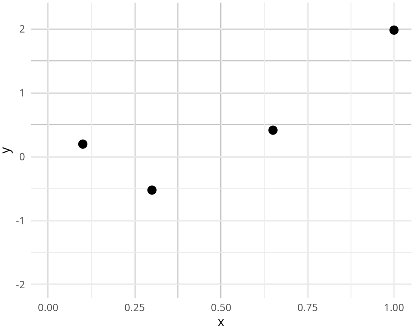
Learning goals

- Initial design
- Surrogate modeling
- Basic loop

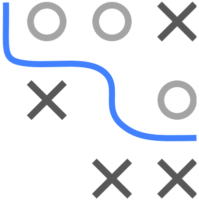
OPTIMIZATION VIA SURROGATE MODELING

Starting point:

- We do not know the objective function $f : \mathcal{S} \to \mathbb{R}$
- But we can evaluate f for a few different inputs $\mathbf{x} \in \mathcal{S}$
- For now we assume that those evaluations are noise-free
- **Idea:** Use the data $\mathcal{D}^{[t]} = \{(\mathbf{x}^{[i]}, y^{[i]})\}_{i=1,\dots,t}$, $y^{[i]} := f(\mathbf{x}^{[i]})$, to derive properties about the unknown function f



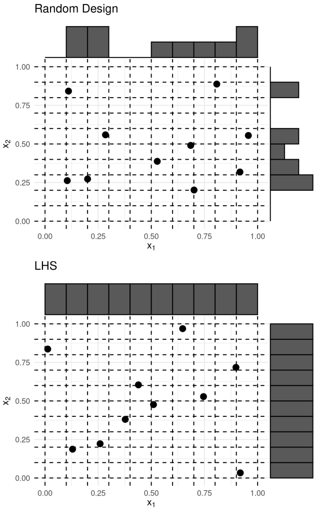
INITIAL DESIGN



- Should cover / explore input space sufficiently:
    - Random design
    - Latin hypercube sampling
    - Sobol sampling
- Type of design usually has not the largest effect
- A more important choice is the **size** of the initial design
    - Should neither be too small (bad initial fit) nor too large (spending too much budget without doing “intelligent” optimization)
    - Rule of thumb: $4d$

LATIN HYPERCUBE SAMPLING

- LHS partitions the search space into bins of equal probability
- Goal is to attain a more even distribution of sample points than random sampling
- Allow at most one sample per bin; exactly one sample per row and column

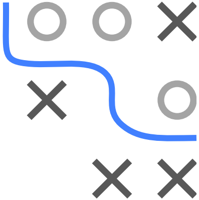


Marginal histograms RS vs. LHS

LATIN HYPERCUBE SAMPLING

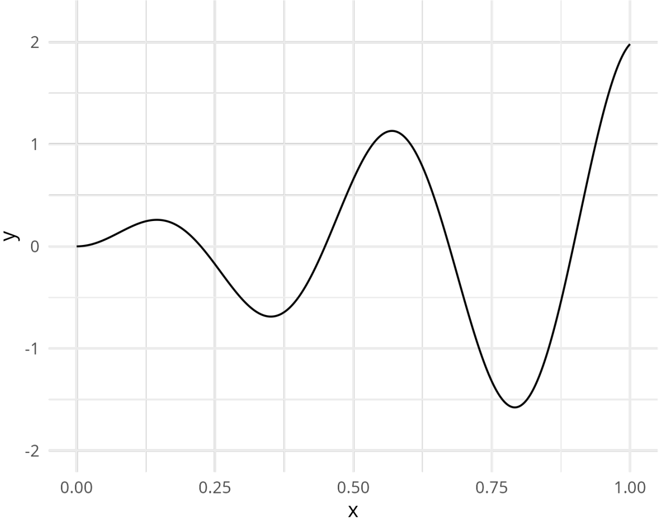
Actual sampling of points, e.g., constructed via **Maximin**:

- The minimum distance between any two points in $\mathcal{D}$ is $2q = \min_{\mathbf{x} \in \mathcal{D}, \mathbf{x}' \in \mathcal{D}} \rho(\mathbf{x}, \mathbf{x}')$ (ρ any metric, e.g., Euclidean distance)
- q is the packing radius - the radius of the largest ball that can be placed around every design point such that no two balls overlap
- Goal: Find $\mathcal{D}$ that maximizes $2q$: $\max_{\mathcal{D}} \min_{\mathbf{x} \in \mathcal{D}, \mathbf{x}' \in \mathcal{D}} \rho(\mathbf{x}, \mathbf{x}')$
- Ensures that the design points in $\mathcal{D}$ are as far apart from each other as possible



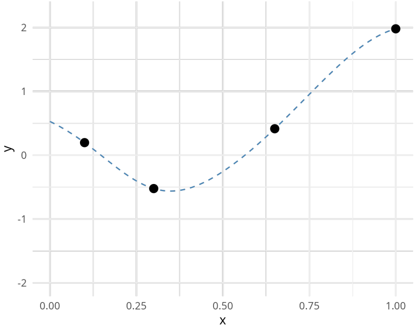
SURROGATE MODELING

Running example = minimize this “black-box”:



SURROGATE MODELING

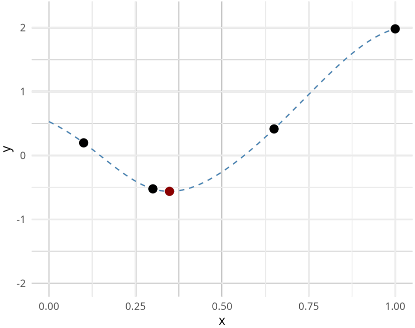
1. **Fit** a **regression model** $\hat{f} : \mathcal{D}^{[t]} \rightarrow \mathbb{R}$ (blue) to extract maximum information from the design points (black) and learn properties of f



As we can eval f without noise, we fit an interpolator

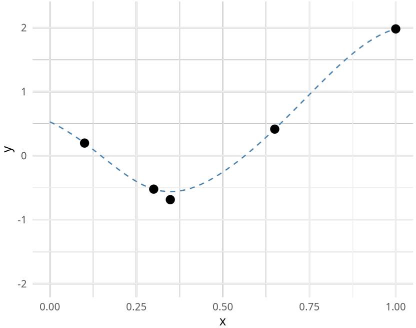
SURROGATE MODELING / 2

2. Instead of the expensive f, we optimize the cheap surrogate $\hat{f}$ (blue) to **propose** a new point (red) for evaluation



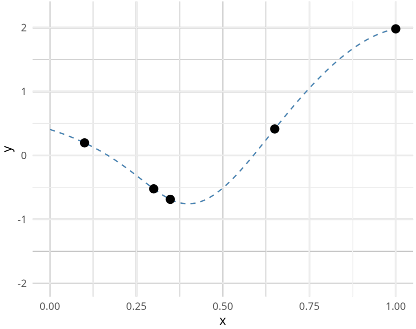
SURROGATE MODELING / 3

3. We finally evaluate the newly proposed point



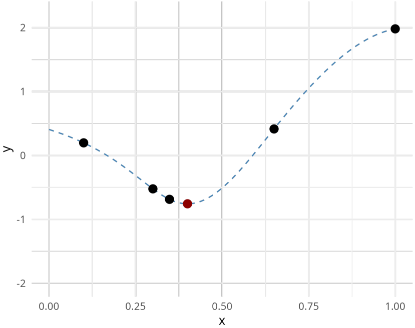
SURROGATE MODELING

- After evaluation of the new point, we **adjust** the model on the expanded dataset via (slower) refitting or a (cheaper) online update



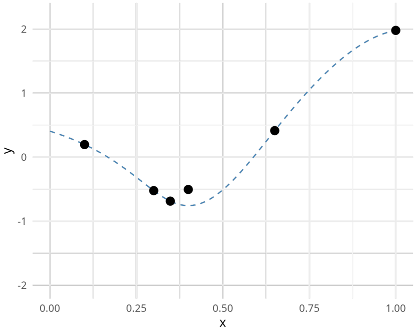
SURROGATE MODELING

- We again obtain a new candidate point (red) by optimizing the cheap surrogate model function (blue) ...



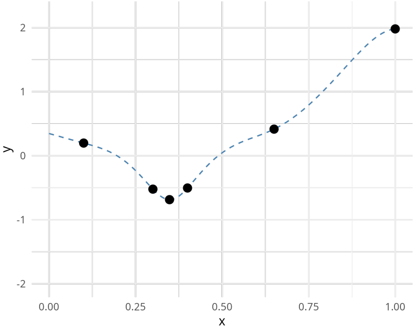
SURROGATE MODELING

- ... and evaluate that candidate



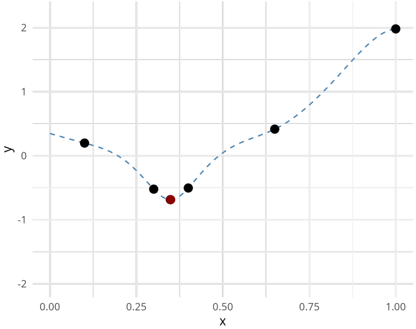
SURROGATE MODELING

- We repeat: (i) **fit** the model



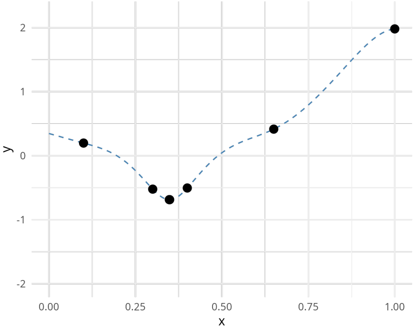
SURROGATE MODELING

- (ii) **propose** a new point



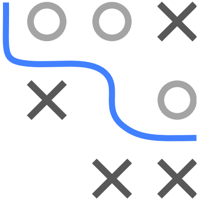
SURROGATE MODELING

- (iii) **evaluate** that point



- We observe that the algorithm converged

BASIC LOOP

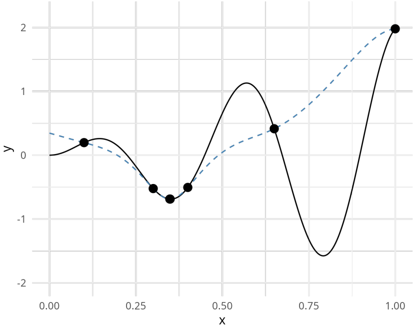


The basic loop of our sequential optimization procedure is:

1. Fit surrogate model $\hat{f}$ on previous evaluations $\mathcal{D}^{[t]} = \{(\mathbf{x}^{[i]}, y^{[i]})\}_{i=1,\dots,t}$
2. Optimize the surrogate model $\hat{f}$ to obtain a new point $\mathbf{x}^{[t+1]} := \arg\min_{\mathbf{x}\in\mathcal{S}} \hat{f}(\mathbf{x})$
3. Evaluate $\mathbf{x}^{[t+1]}$ and update data $\mathcal{D}^{[t+1]} = \mathcal{D}^{[t]} \cup \{(\mathbf{x}^{[t+1]}, f(\mathbf{x}^{[t+1]}))\}$

EXPLORATION VS. EXPLOITATION

We see: We ran into a local minimum. We did not "explore" the most crucial areas and **missed** the global minimum.



- Better ways to propose points based on our model exist, so-called **acquisition functions**
- Optimizing SM directly corresponds to raw / mean prediction as AQF
- Results in **high exploitation but low exploration**