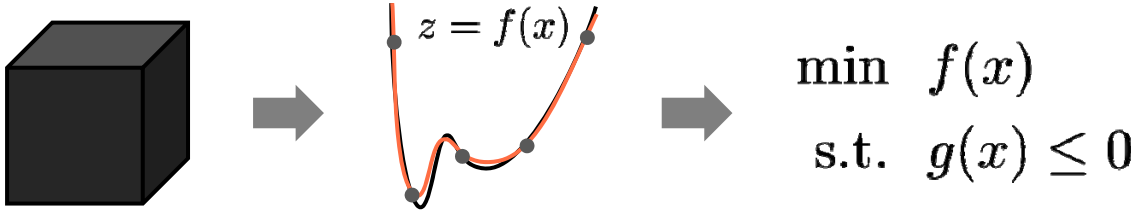
Learning Low-complexity Surrogate Models of Processes

Zach Wilson[1,2], Alison Cozad[1,2], Nick Sahinidis[1,2], David Miller[1]

[1]National Energy Technology Laboratory, Pittsburgh, PA,USA

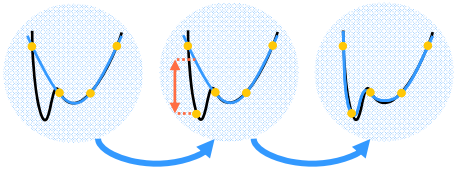
[2]Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA, USA

SURROGATE-BASED OPTIMIZATION



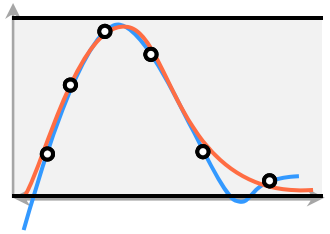
- **ALAMO:**

  Generate simple, accurate surrogate models from black-box data



- **Subset Selection:**

  Building surrogate models using a subset of potential basis functions

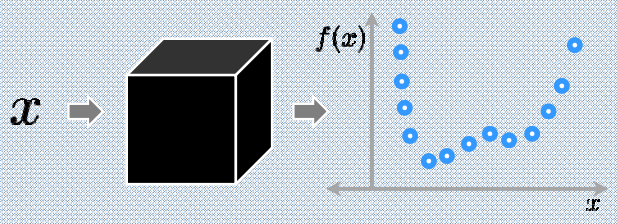


CHALLENGES

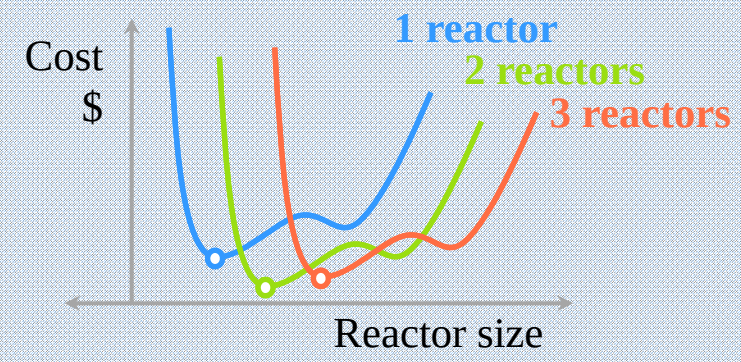
OPTIMIZER

SIMULATOR

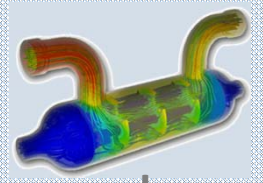
No algebraic model



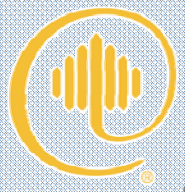
Complex process alternatives



Costly simulations



hours

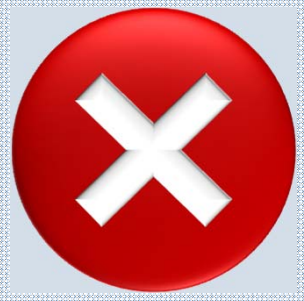


minutes

```
   AREA = SQ
   WRITE(6,60
   GO TO 10
50 WRITE(6,60
   STOP
90 WRITE(6,60
   STOP
```

seconds

Scarcity of fully robust simulations

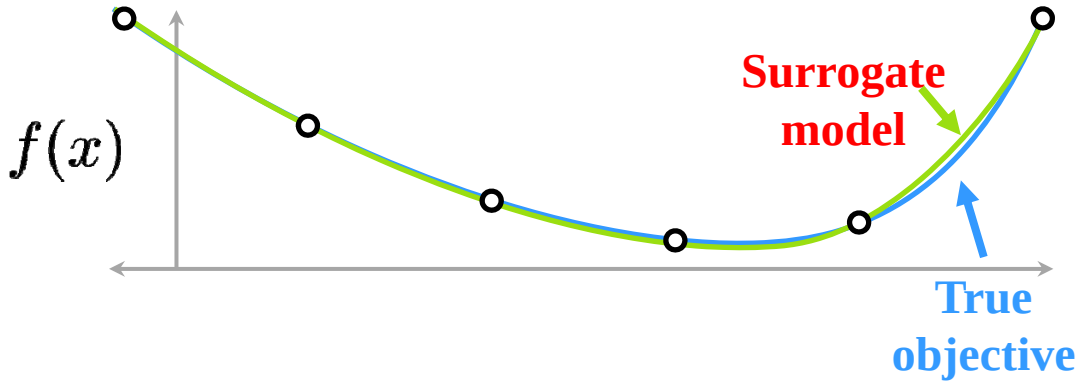


✗ Gradient-based methods

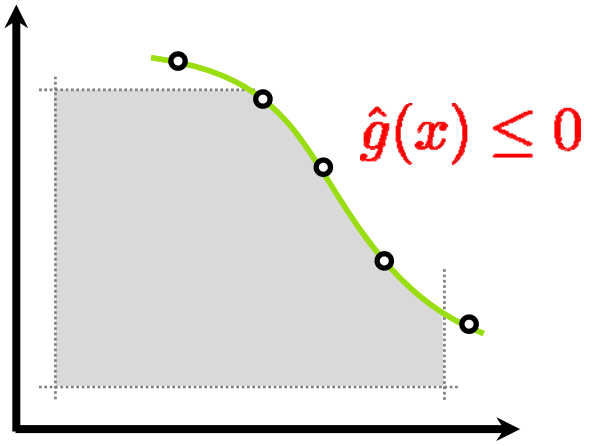
✗ Derivative-free methods

USE SURROGATE MODELS

- To replace black-box objectives
  - Generate surrogate models for the objective as a whole or in-parts



- To replace black-box constraints
  - Define the problem space
  - Generate equality or inequality constraints

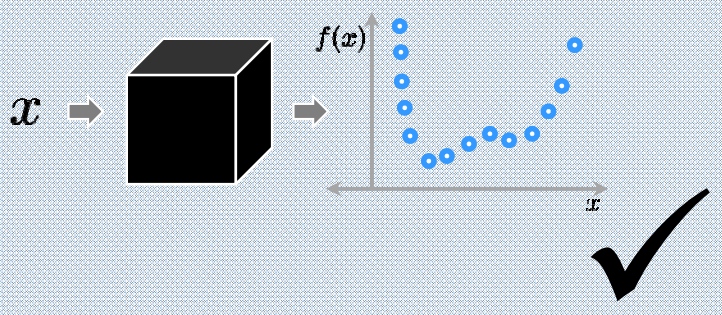


CHALLENGES

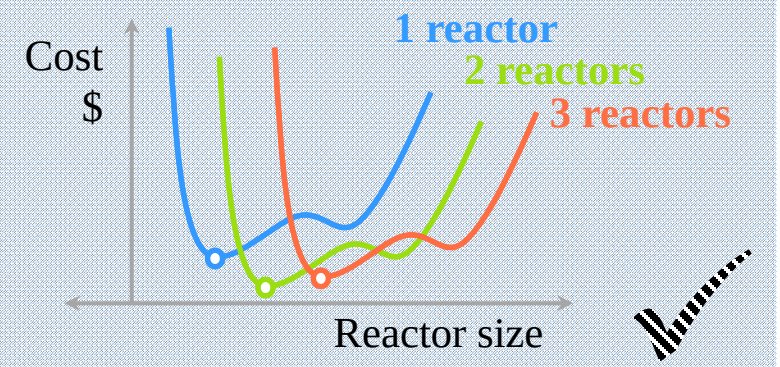
OPTIMIZER

SIMULATOR

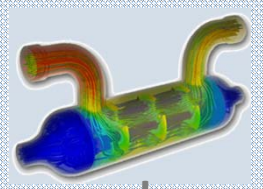
No algebraic model



Complex process alternatives



Costly simulations



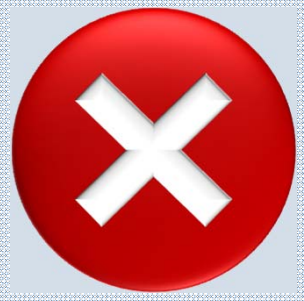
hours

minutes

seconds

```
   AREA = SQ
   WRITE(6,60
   GO TO 10
50 WRITE(6,60
   STOP
90 WRITE(6,60
   STOP
```

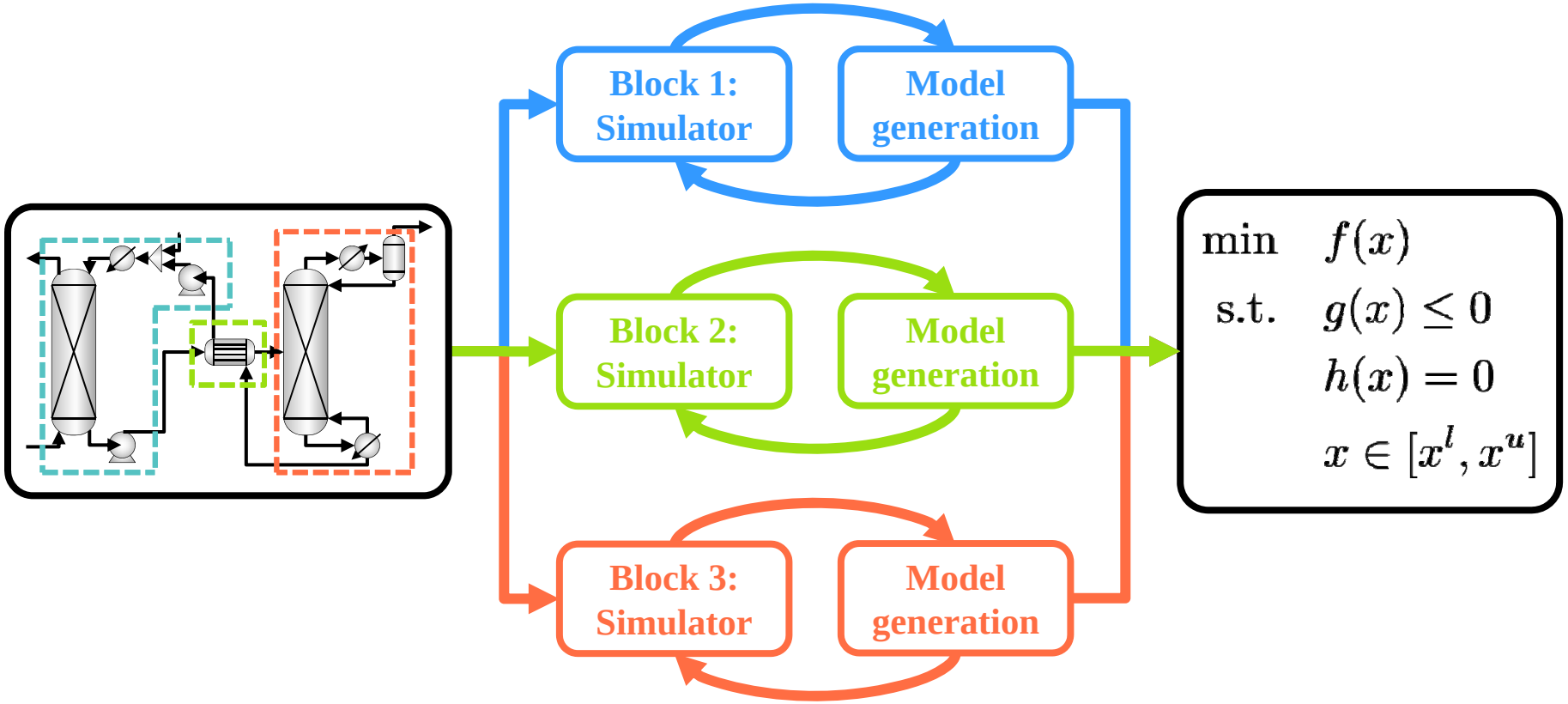
Scarcity of fully robust simulations



✗ Gradient-based methods

✗ Derivative-free methods

PROCESS DISAGGREGATION



Process Simulation

Disaggregate process into process blocks

Surrogate Models

Build simple and accurate models with a functional form tailored for an optimization framework

Optimization Model

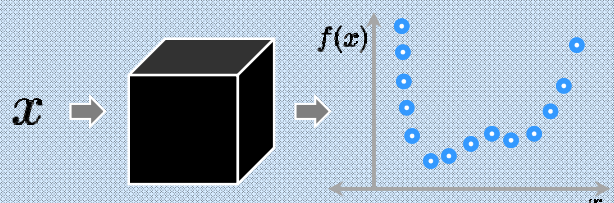
Add algebraic constraints design specs, heat/mass balances, and logic constraints

CHALLENGES

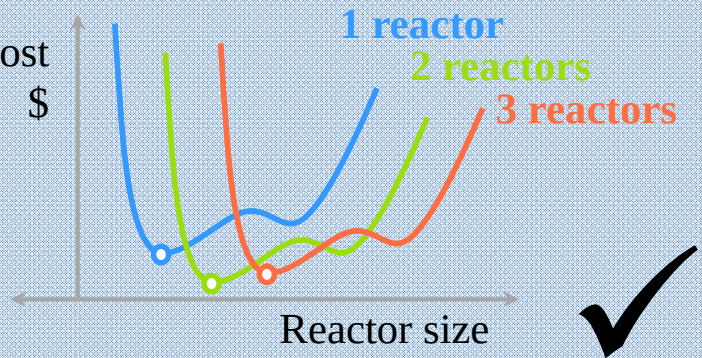
OPTIMIZER

SIMULATOR

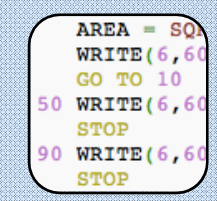
No algebraic model

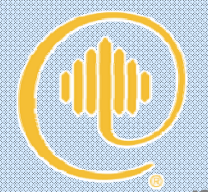


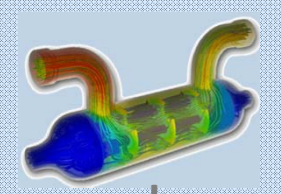
Complex process alternatives



Costly simulations







seconds

minutes

hours

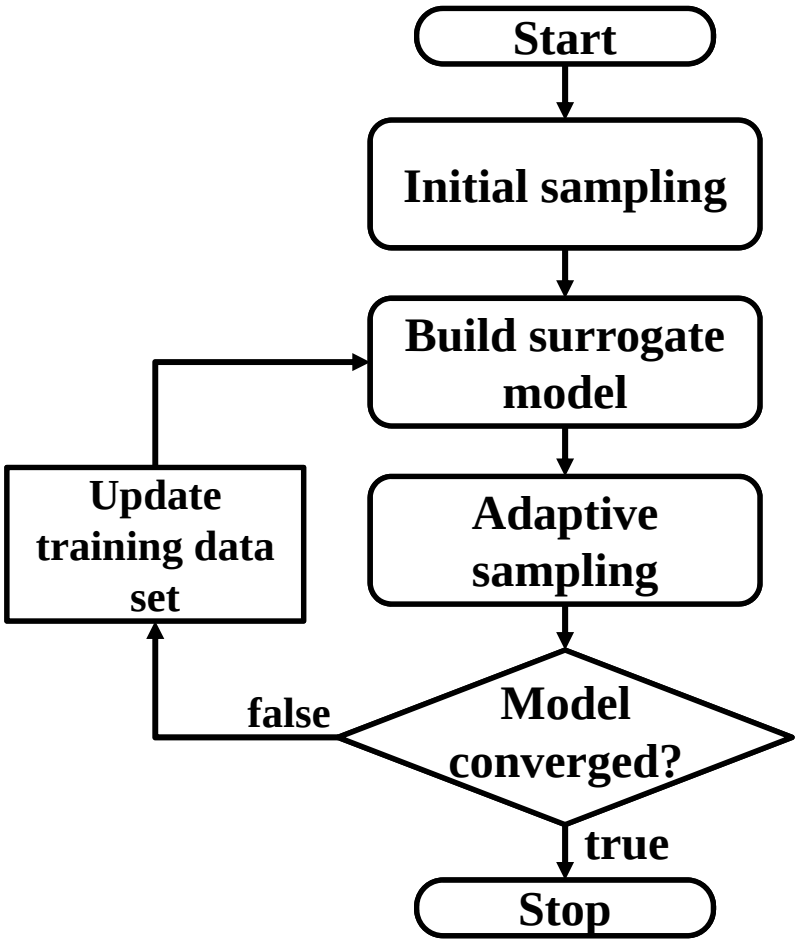
Scarcity of fully robust simulations

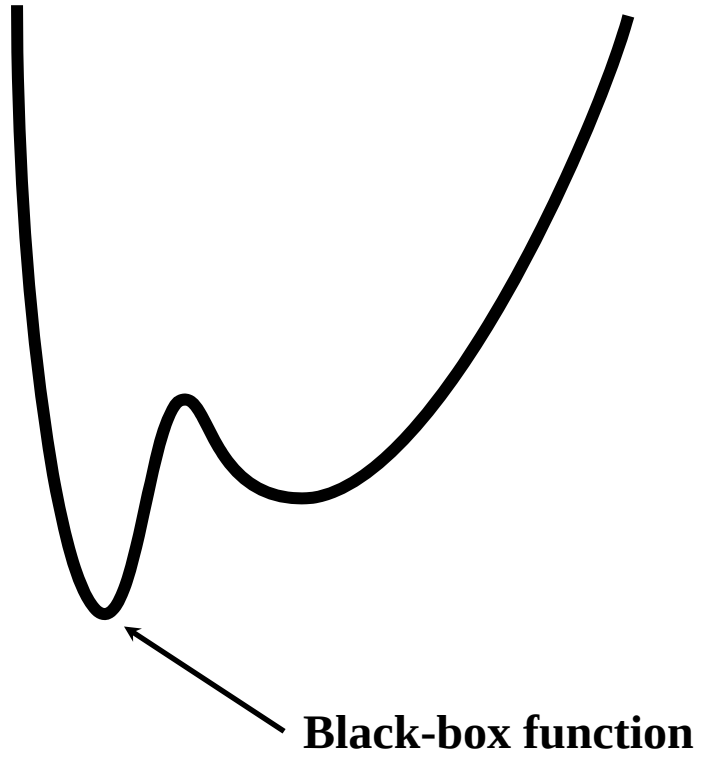
✗ Gradient-based methods

✗ Derivative-free methods

ALAMO

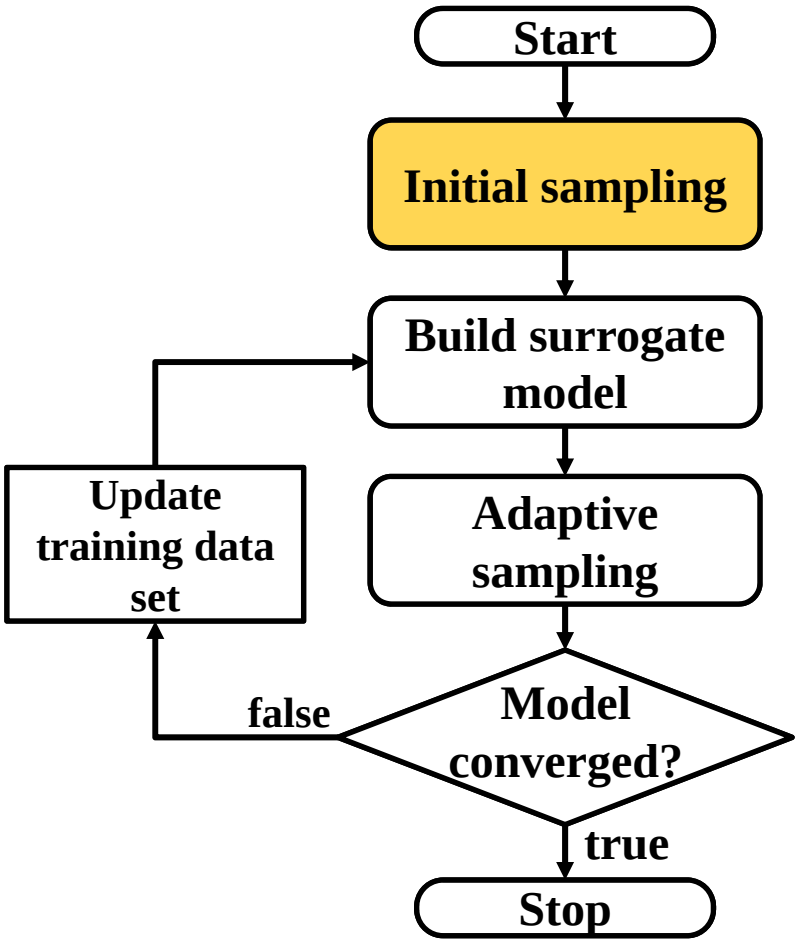
Automated Learning of Algebraic Models for Optimization

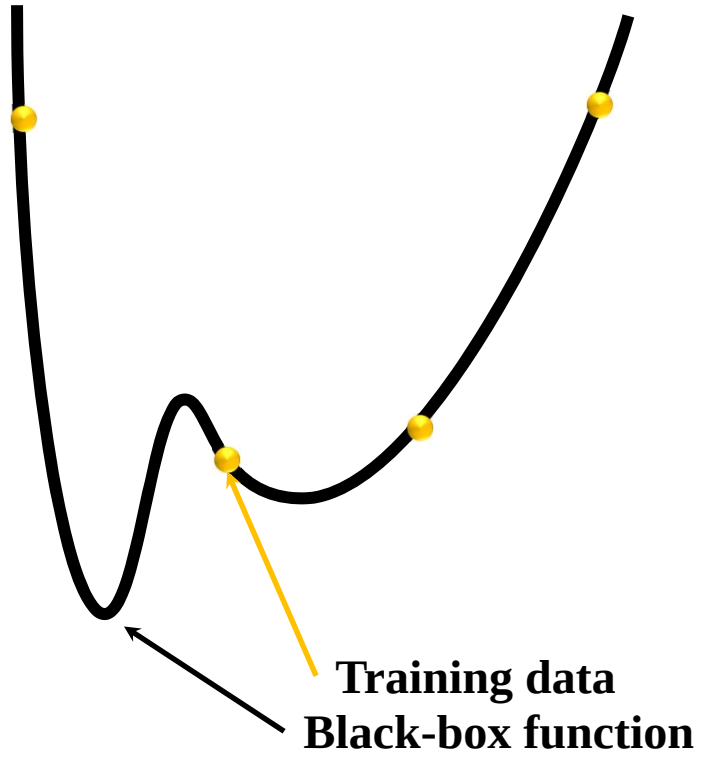




ALAMO

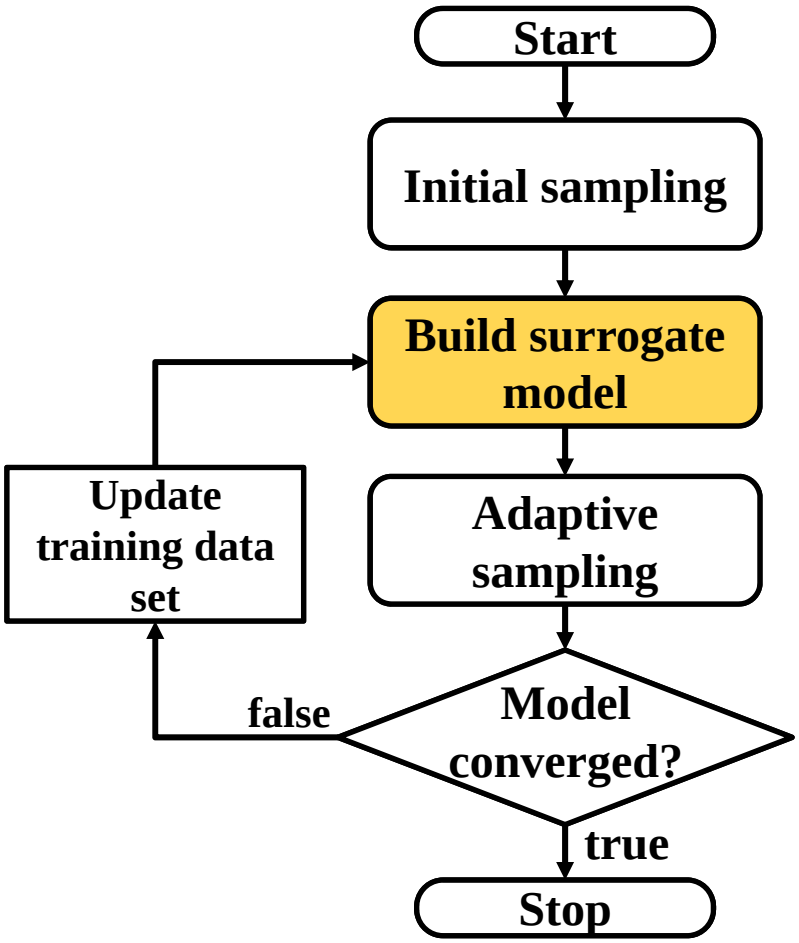
Automated Learning of Algebraic Models for Optimization

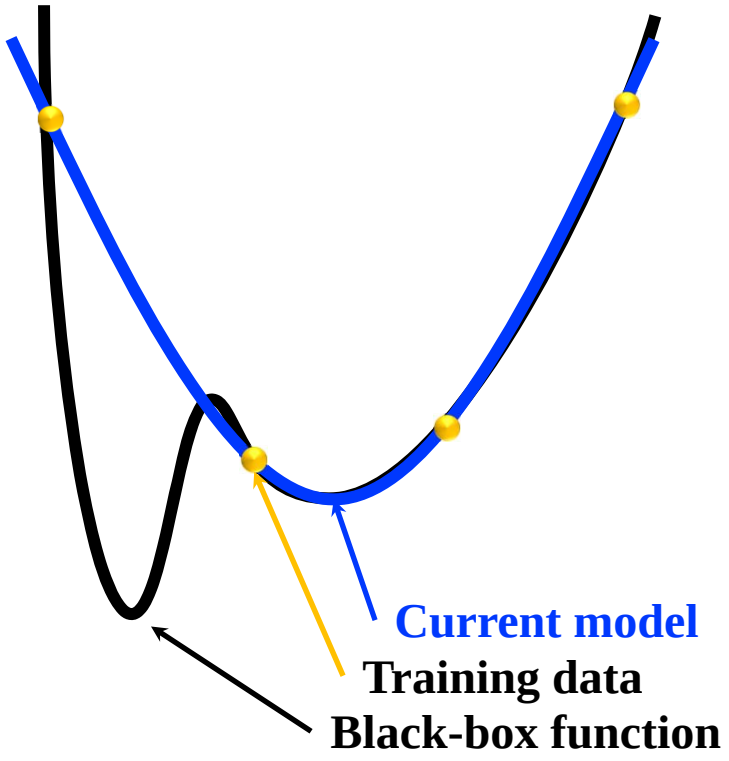




ALAMO

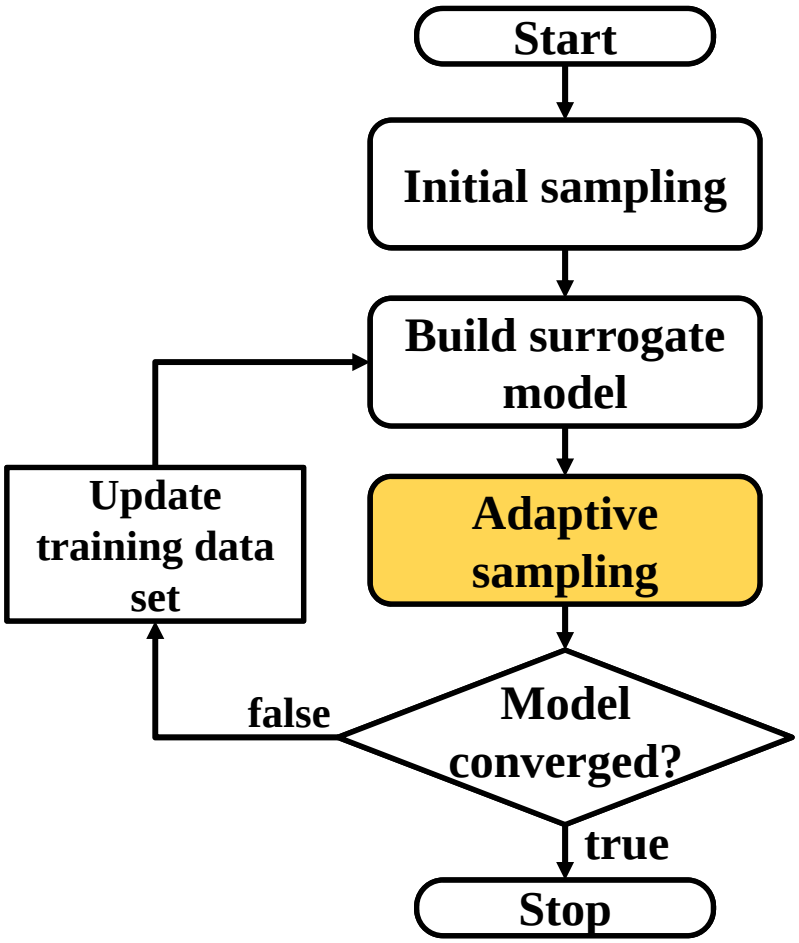
Automated Learning of Algebraic Models for Optimization

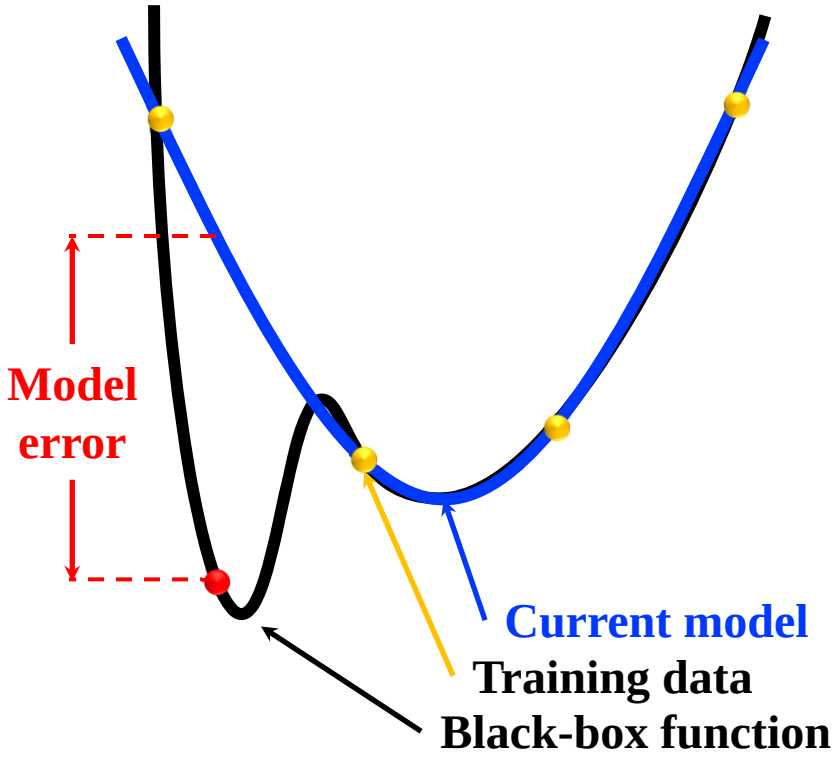




ALAMO

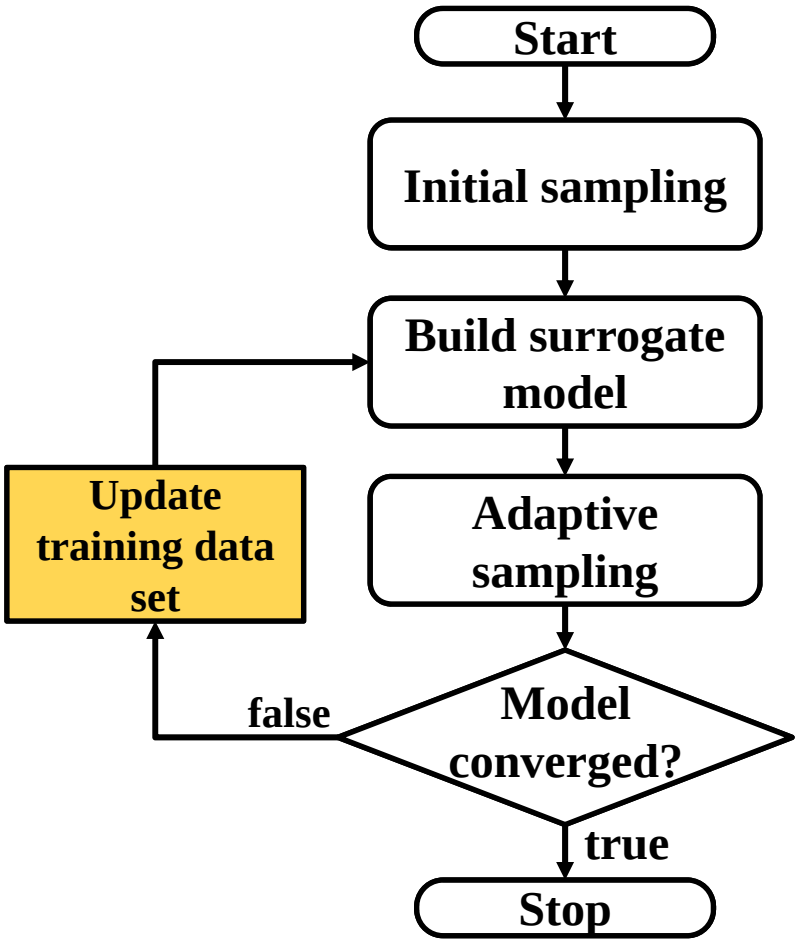
Automated Learning of Algebraic Models for Optimization

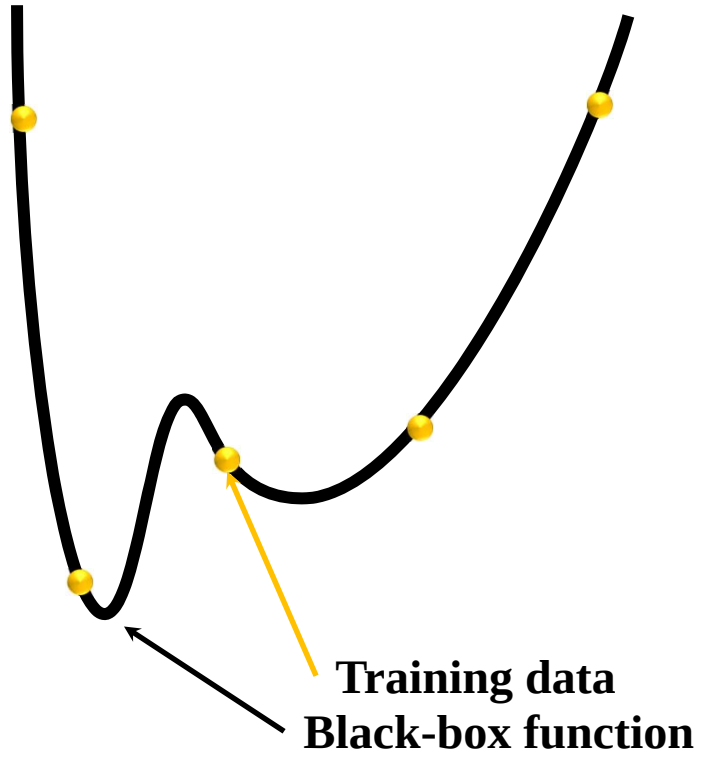




ALAMO

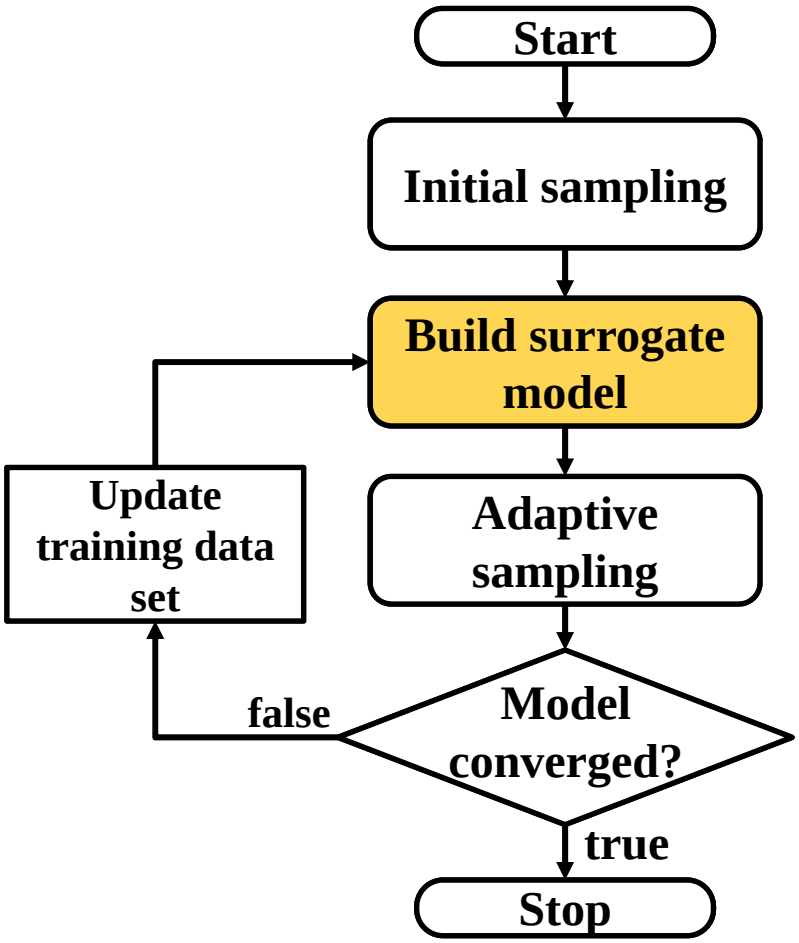
Automated Learning of Algebraic Models for Optimization

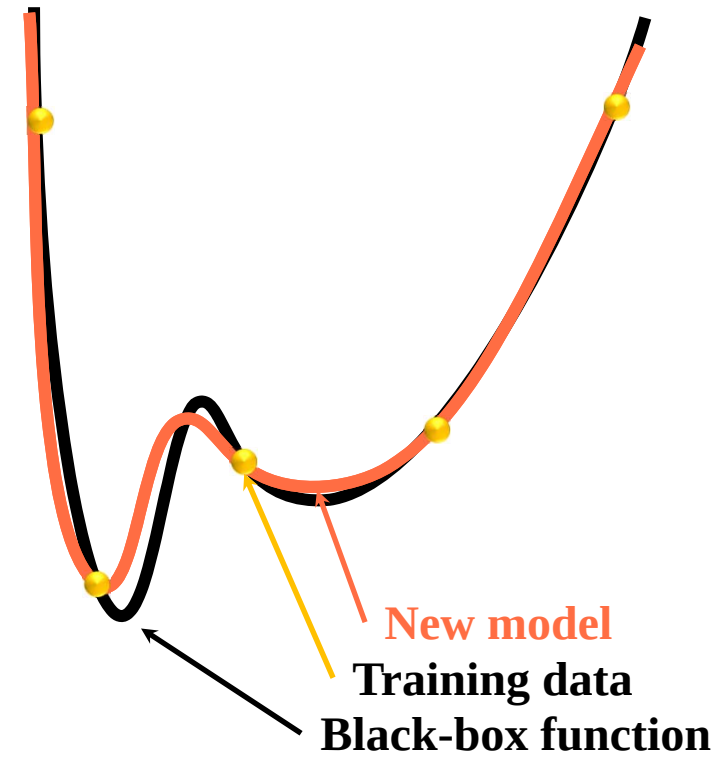




ALAMO

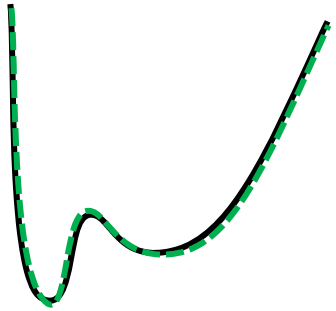
Automated Learning of Algebraic Models for Optimization





HOW TO BUILD THE SURROGATES

- **We aim to build surrogate models that are**
  - **Accurate**
    - *We want to reflect the true nature of the simulation*
  - **Simple**
    - *Tailored for algebraic optimization*



$$\hat{f}(x) = \sum_{i=1}^{n} \gamma_i \exp\left(\frac{\|x\|}{\sigma^2}\right) + \beta_0 + \beta_1 x + \dots$$

$$\hat{f}(x) = \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 e^x$$

  - **Generated from a minimal data set**
    - *Reduce experimental and simulation requirements*

MODEL IDENTIFICATION

- **Goal: Identify the functional form and complexity of the surrogate models**

$$z = f(x)$$

- **Functional form:**
  - General functional form is unknown: Our method will identify models with combinations of simple basis functions

Category		$X_j(x)$
I.	Polynomial	$(x_d)^{\alpha}$
II.	Multinomial	$\prod_{d \in \mathcal{D}' \subseteq \mathcal{D}} (x_d)^{\alpha_d}$
III.	Exponential and logarithmic forms	$\exp\left(\frac{x_d}{\gamma}\right)^{\alpha}$, $\log\left(\frac{x_d}{\gamma}\right)^{\alpha}$
IV.	Expected bases	From experience, simple inspection, physical phenomena, etc.

OVERFITTING AND TRUE ERROR

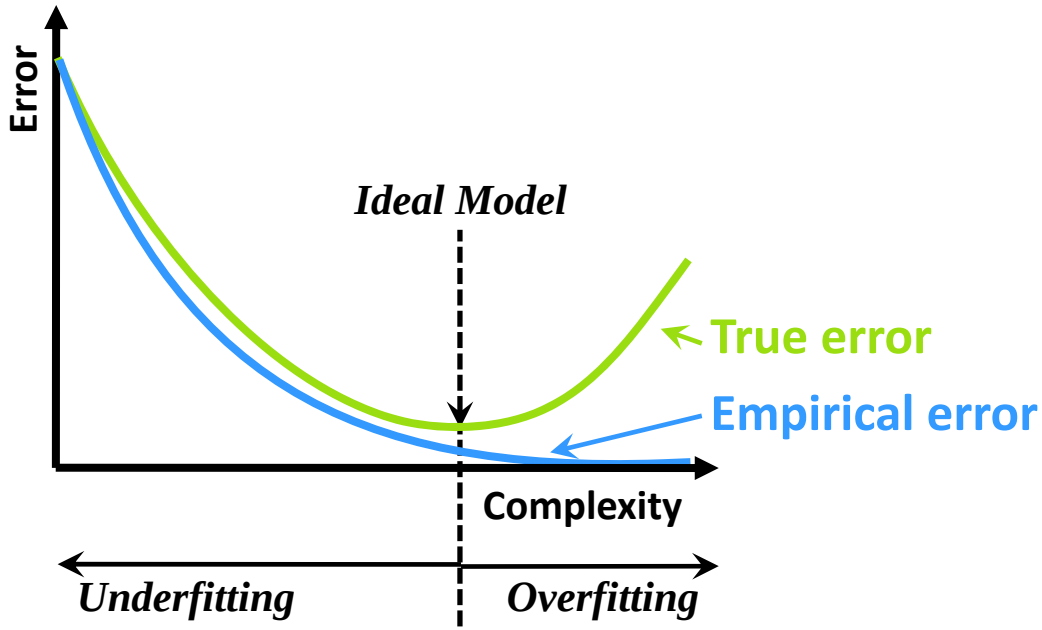
Step 1: Define a large set of potential basis functions

$$\hat{z}(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 \frac{x_1}{x_2} + \beta_5 \frac{x_2}{x_1} + \beta_6 e^{x_1} + \beta_7 e^{x_2} + \dots$$

OVERFITTING AND TRUE ERROR

Step 1: Define a large set of potential basis functions

$$\hat{z}(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 \frac{x_1}{x_2} + \beta_5 \frac{x_2}{x_1} + \beta_6 e^{x_1} + \beta_7 e^{x_2} + \ldots$$



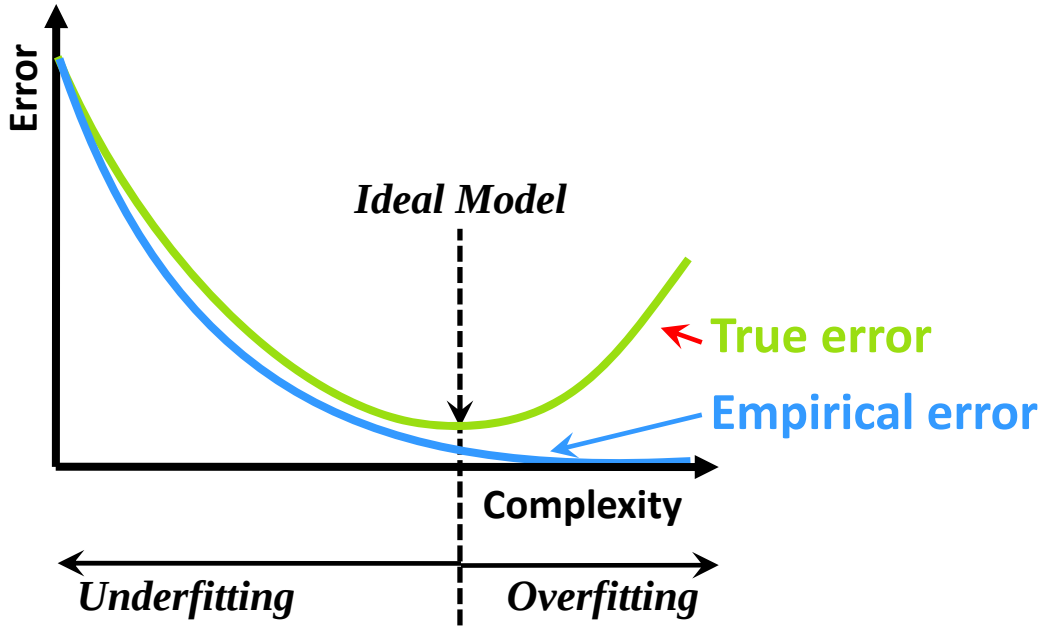
OVERFITTING AND TRUE ERROR

Step 1: Define a large set of potential basis functions

$$\hat{z}(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \beta_4 \frac{x_1}{x_2} + \beta_5 \frac{x_2}{x_1} + \beta_6 e^{x_1} + \beta_7 e^{x_2} + \ldots$$

Step 2: Model reduction

$$\hat{z}(x) = \beta_0 + \beta_2 x_2 + \beta_5 \frac{x_2}{x_1} + \beta_7 e^{x_2}$$



OBJECTIVES AND ALTERNATIVES

Goodness of fit Measure

- **Balances** model complexity with reduction in empirical error
- **Penalize directly** for the number of explanatory variables in the regression model

Alternatives to subset selection

- **Regularization** penalize regression models based on magnitude of regression coefficients
- **Stepwise heuristics** greedy search of explanatory variables misses synergistic affects

Corrected Akaike Information Criterion :

$$AIC_c = n\log\left(\frac{1}{n}\sum_{i=1}^{n}\left(b_i - \sum_{j\in S}\beta_j X_{ij}\right)^2\right) + 2T + \frac{2T(T+1)}{N-T-1}$$

Mallows' C_p :

$$C_p = \frac{\sum_{i=1}^{n}\left(b_i - \left(\sum_{j\in S}\beta_j X_{ij}\right)\right)^2}{\hat{\sigma}^2} + 2(T) - N$$

Mean Absolute Error :

$$MAE = \frac{\sum_{j\in S}\left|b_i - \left(\sum_{j\in J}\beta X_{ij}\right)\right|}{n-1-\sum_{j\in J}y_j}$$

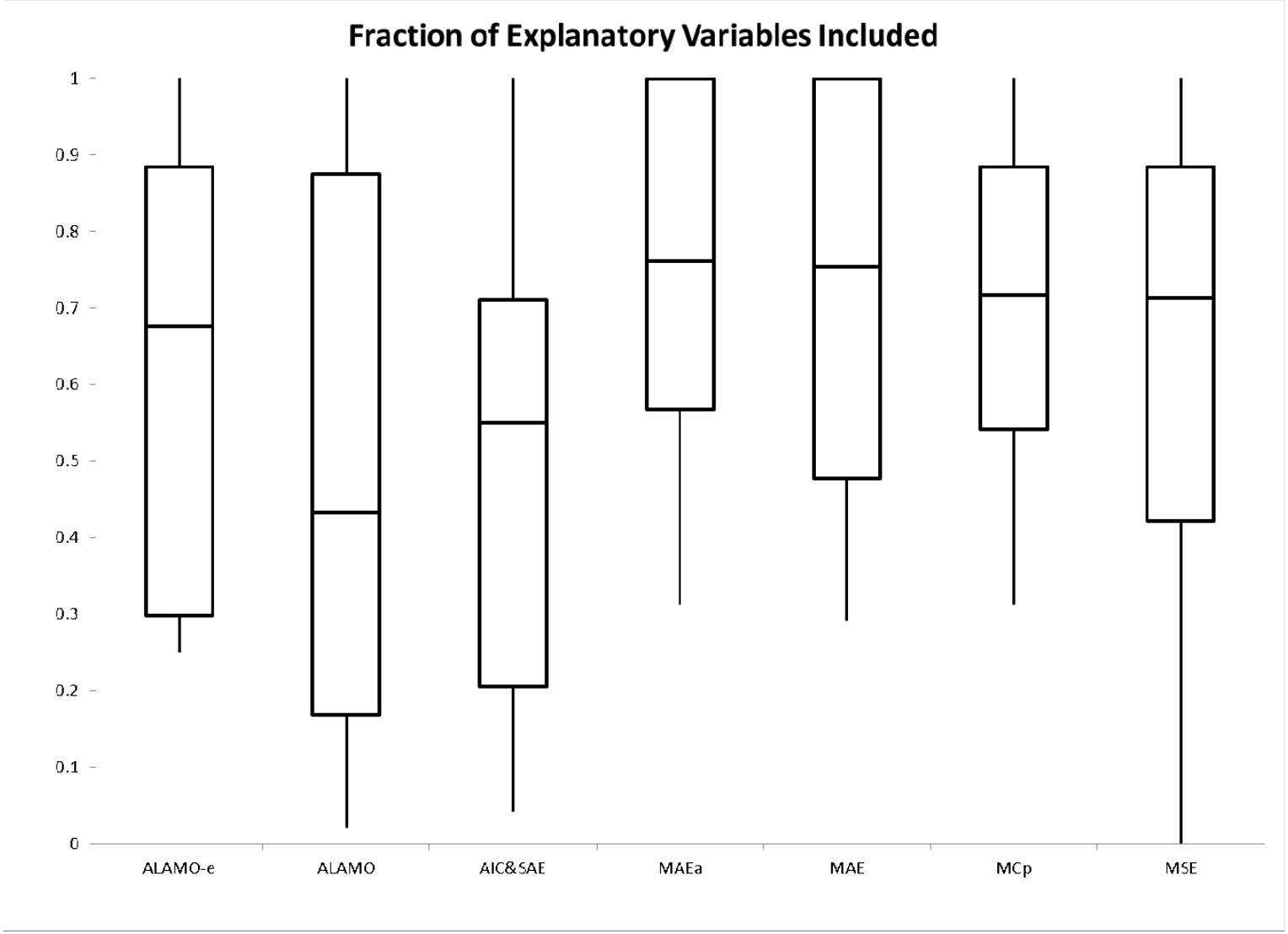
Mean Squared Error :

$$MSE = \frac{\sum_{i=1}^{n}\left(b_i - \left(\sum_{j\in J}\beta_j X_{ij}\right)\right)^2}{n-1-\sum_{j\in J}y_j}$$

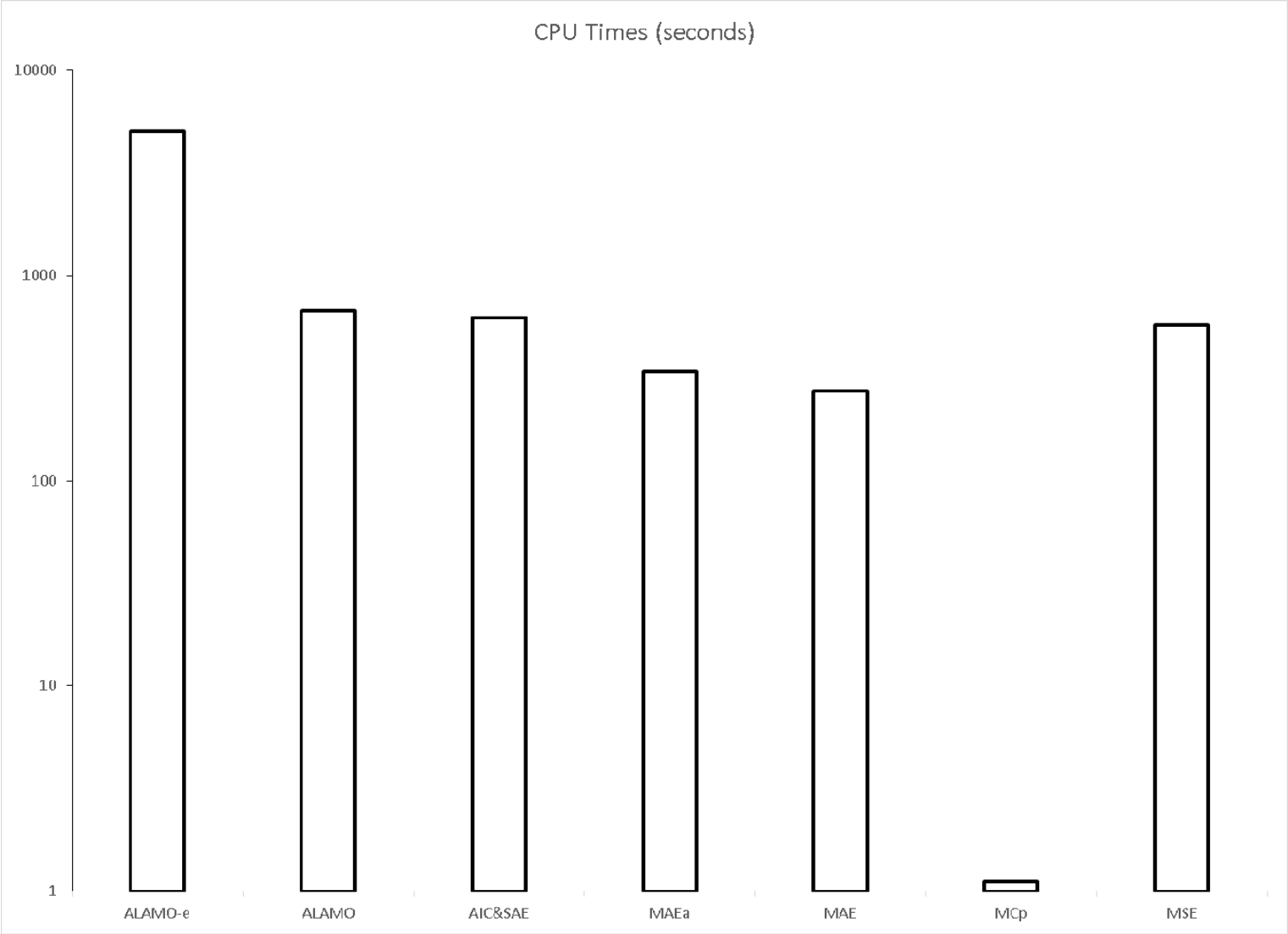
Adjusted Mean Absolute Error :

$$MAE_a = \frac{\sum_{i=1}^{n}\left|b_i - \left(\sum_{j\in J}\beta_j X_{ij}\right)\right| + \left(\frac{\sum_{j\in J}y_j}{n-1}\right)mae_0}{n-1-\sum_{j\in J}y_j}$$

COMPUTATIONAL RESULTS

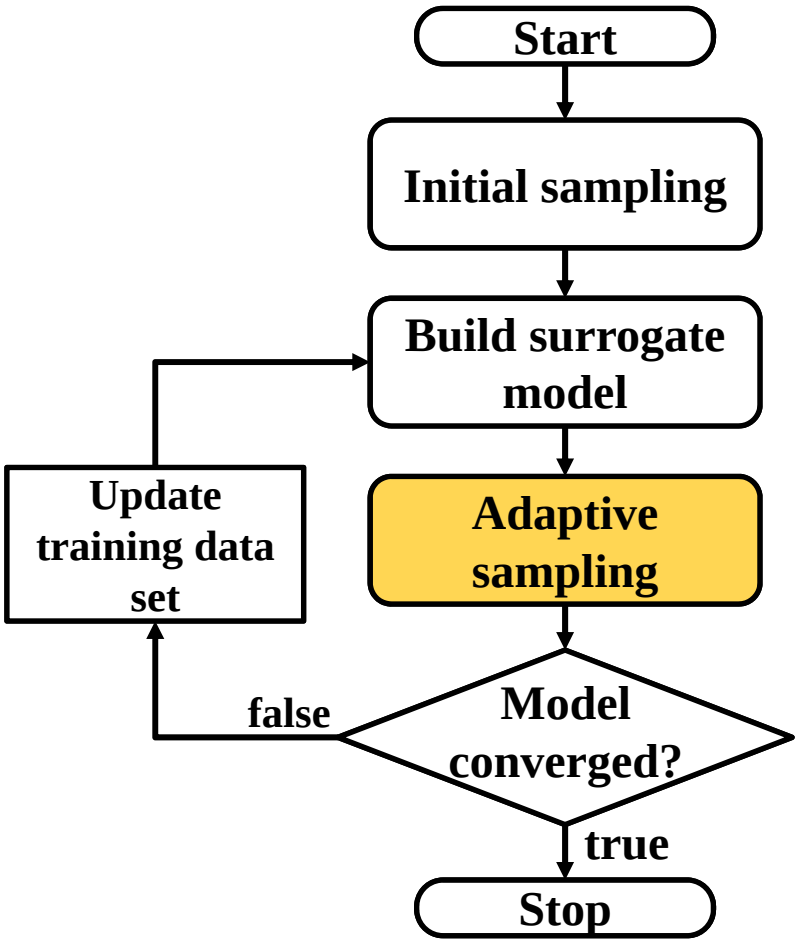


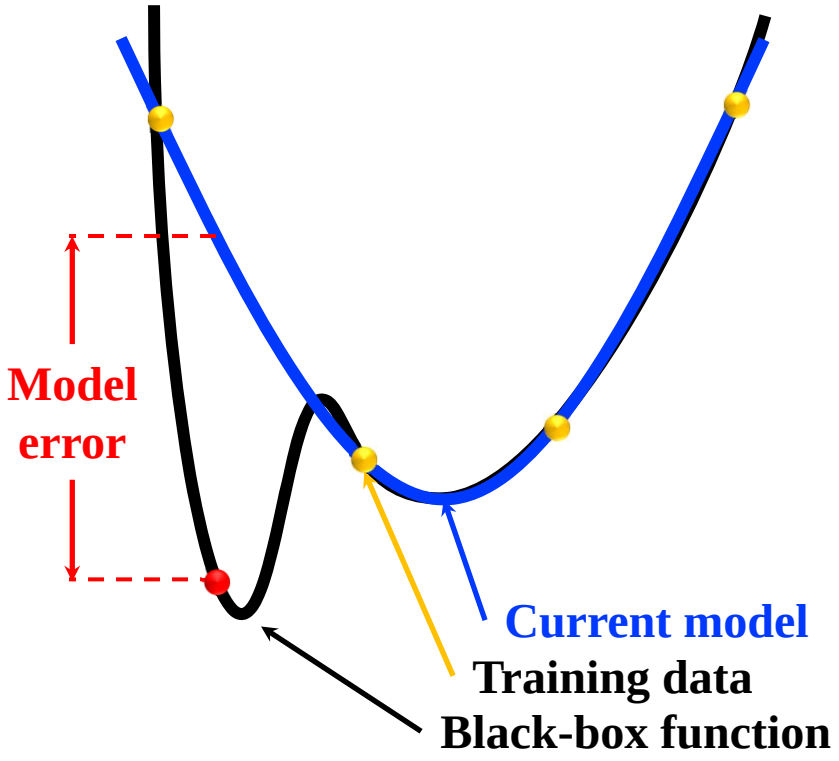
COMPUTATIONAL RESULTS



ALAMO

Automated Learning of Algebraic Models for Optimization





ERROR MAXIMIZATION SAMPLING

- **New goal: Search the problem space for areas of model inconsistency or model mismatch**

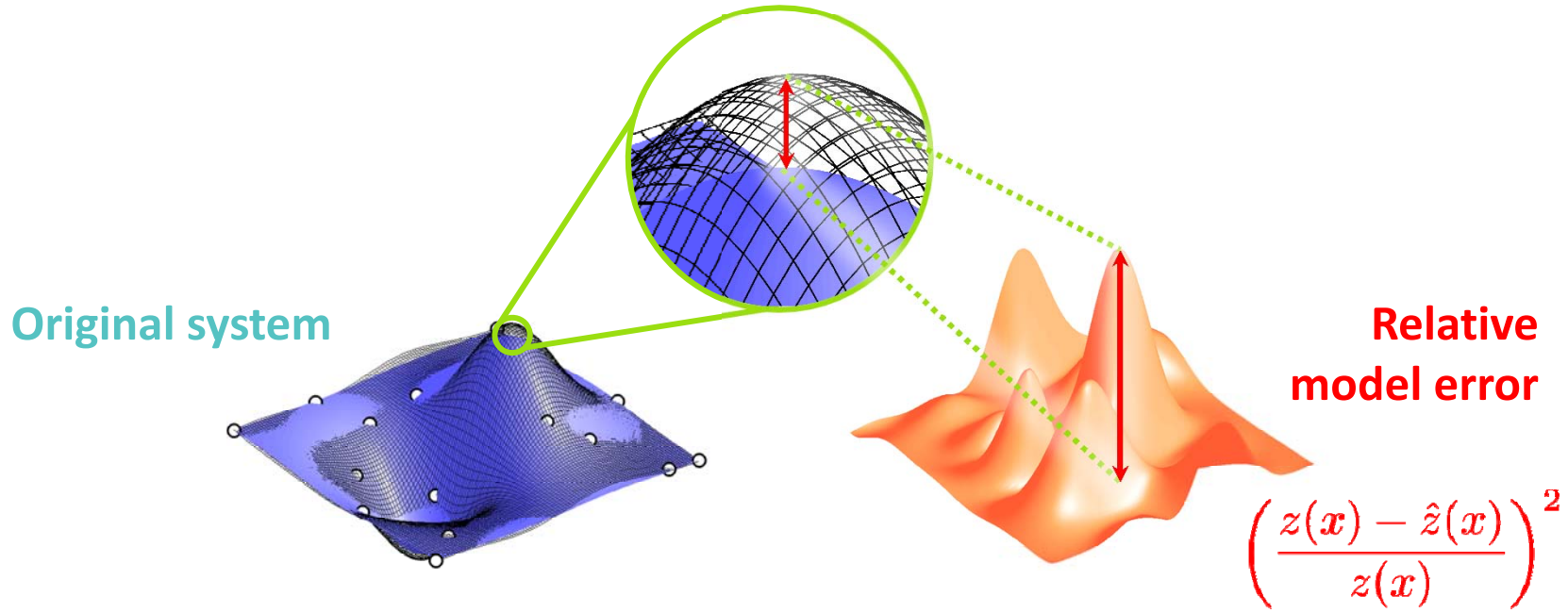
- **More succinctly, we are trying to find points that maximizes the model error with respect to the independent variables**

Surrogate model

$$\max_{x} \left(\frac{z(x) - \hat{z}(x)}{z(x)} \right)^2$$

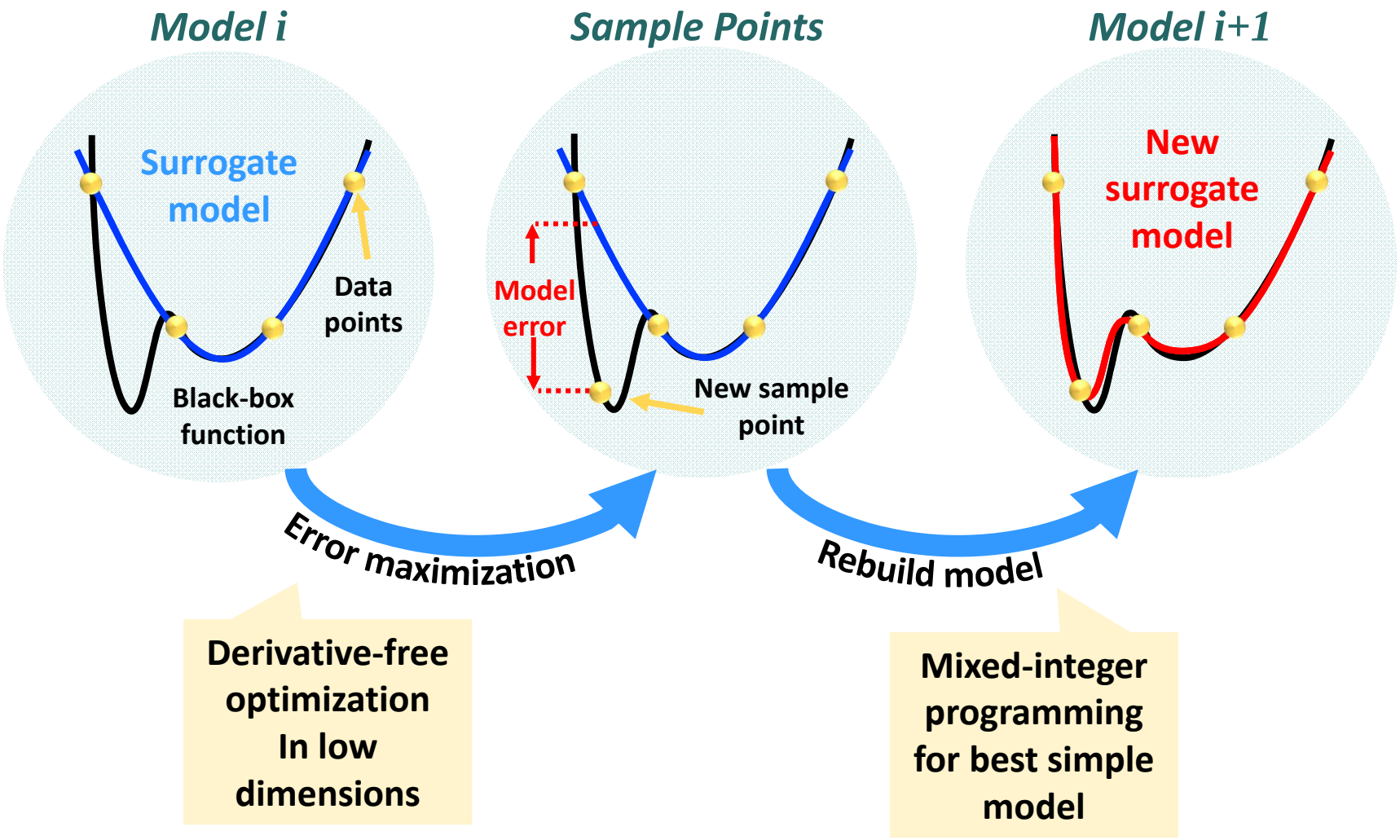
  - **Optimized using a black-box or derivative-free solver (SNOBFIT)** [Huyer and Neumaier, 08]
  - **Derivative-free solvers work well in low-dimensional spaces** [Rios and Sahinidis, 12]

ERROR MAXIMIZATION SAMPLING



- **Information gained using error maximization sampling:**
  - New data point locations that will be used to better train the next iteration's surrogate model
  - Conservative estimate of the true model error
    - *Defines a stopping criterion*
    - *Estimates the final model error*

SYNOPSIS

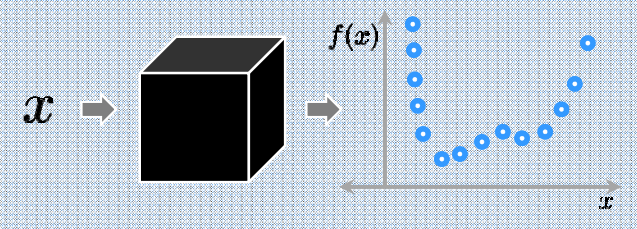

Model i
Surrogate model
Data points
Black-box function
Sample Points
Model error
New sample point
Model i+1
New surrogate model
Error maximization
Rebuild model
Derivative-free optimization In low dimensions
Mixed-integer programming for best simple model

CHALLENGES

OPTIMIZER

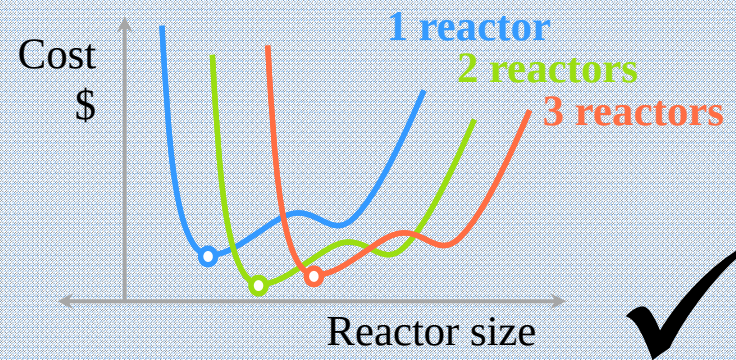
SIMULATOR

No algebraic model



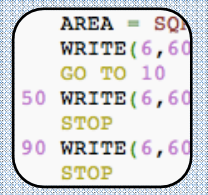
✓

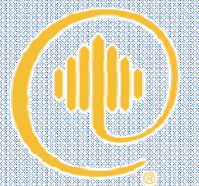
Complex process alternatives

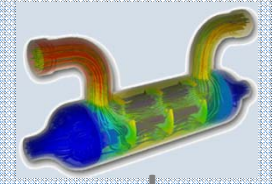


✓

Costly simulations







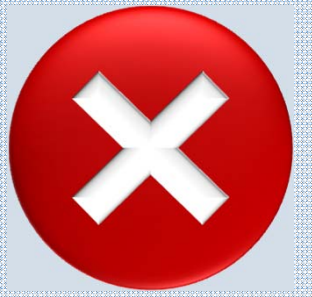
hours

minutes

seconds

✓

Scarcity of fully robust simulations



✓

✗ Gradient-based methods

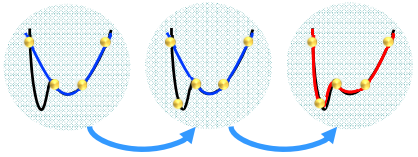
✗ Derivative-free methods

CONCLUSIONS

- **Expanding the scope of MINLPs**
  - Using low-complexity surrogate models to strike a balance between optimal decision-making and model fidelity

$$\begin{aligned} \min \quad & f(x,y) \\ \text{s.t.} \quad & g(x,y) \leq 0 \\ & h(x,y) = 0 \end{aligned}$$

- **Surrogate model identification**
  - Simple model identification – MILP formulation
  - Error Maximization – More information found per each simulated data point



- **Enforcing systems insights**
  - Iteratively use implicit regression constraints to ensure limits on models