Automatic Learning of Algebraic Models for Optimization

Alison Cozad[1,2], Nick Sahinidis[1,2], and David Miller[1]

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

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

PROBLEM STATEMENT

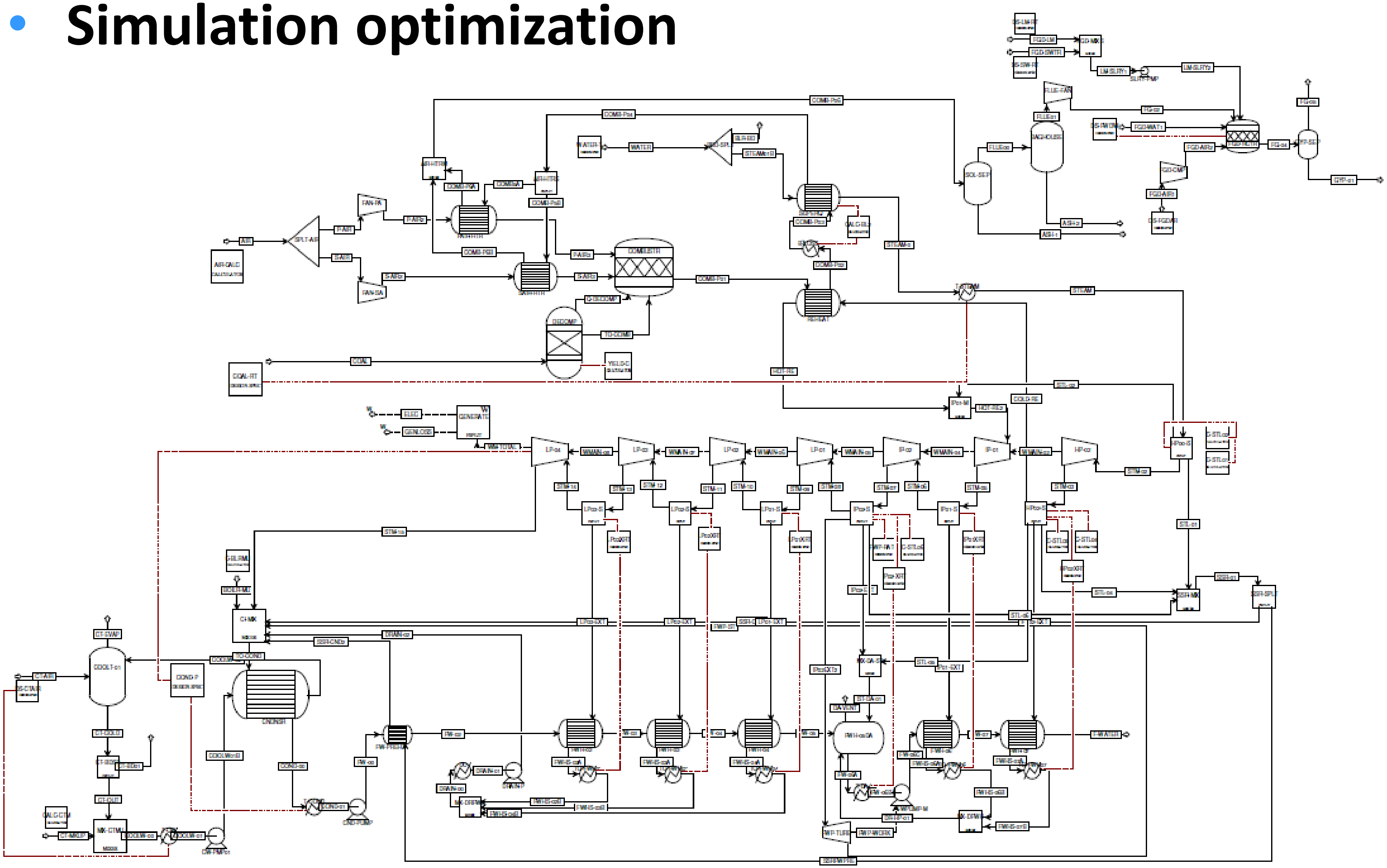
$$
\begin{aligned}
\min \quad & f(x) \\
\text{s.t.} \quad & g(x) \leq 0 \\
& x \in A \subset \mathbb{R}^n
\end{aligned}
$$

$x \rightarrow$ **Input-output black box** $\rightarrow$ $f(x)$, $g(x)$

where $f(x)$ is an algebraic or black-box cost function
$g(x)$ is a set of algebraic or black-box constraints
A is a set of box constraints on x

MOTIVATION

- **Simulation optimization**

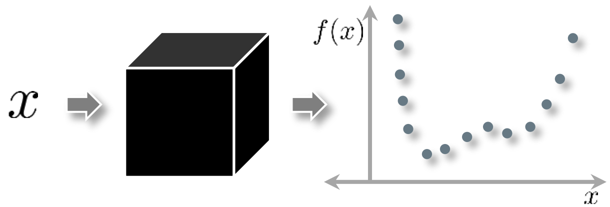


Pulverized coal plant Aspen Plus® simulation provided by the National Energy Technology Laboratory

CHALLENGES

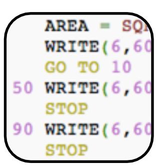
SOURCE: Simulator 1

No algebraic model



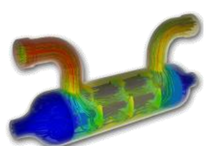
SOURCE : Optimizer 2

Costly simulations



seconds

minutes



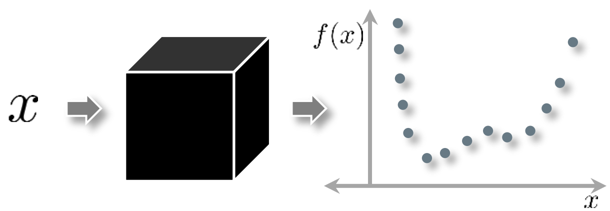
hours

✗ **Gradient-based methods**

CHALLENGES

SOURCE: Simulator — 1

No algebraic model



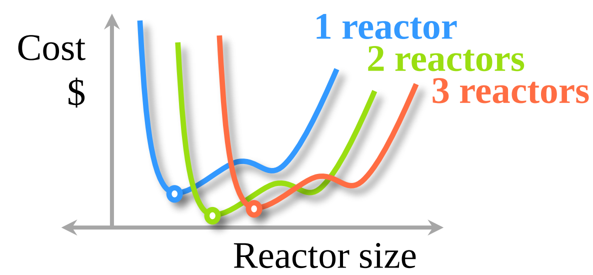
SOURCE : Optimizer — 2

Costly simulations

✗ Gradient-based methods

SOURCE : Simulator — 3

Complex process alternatives

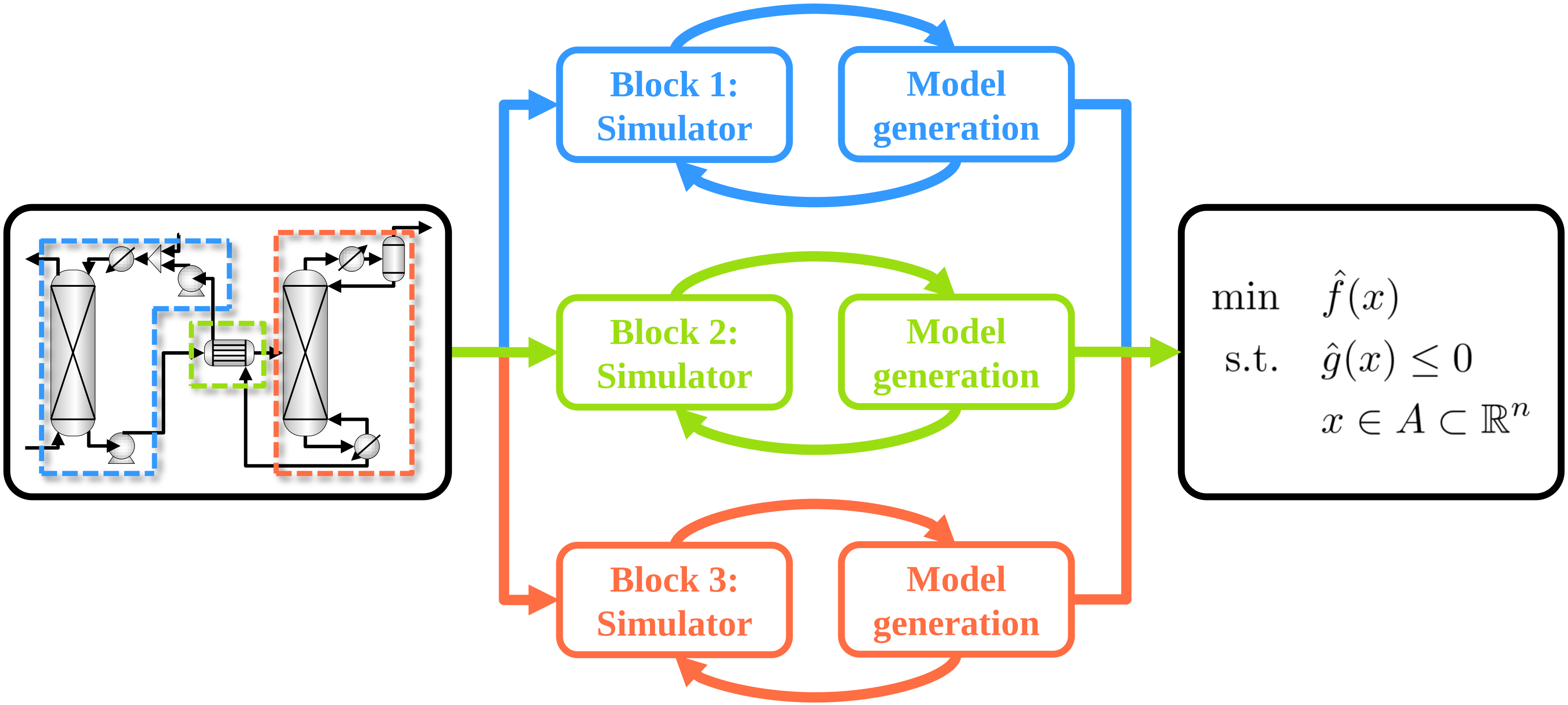


SOURCE : Optimizer — 4

Scarcity of fully robust simulations

✗ Derivative-free methods

SOLUTION STRATEGY



Process Simulation

Disaggregate process into process blocks

Surrogate Models

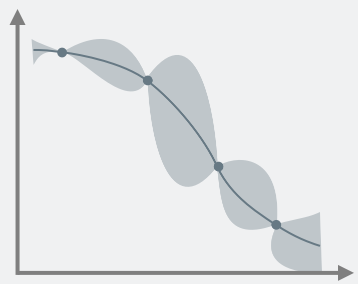
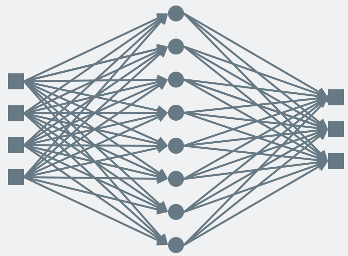
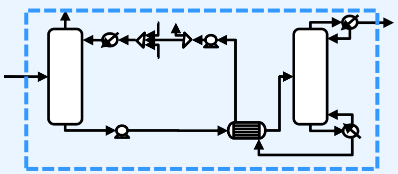
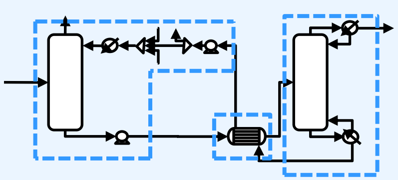
Build simple and accurate models with a functional form tailored for an optimization framework

Optimization Model

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

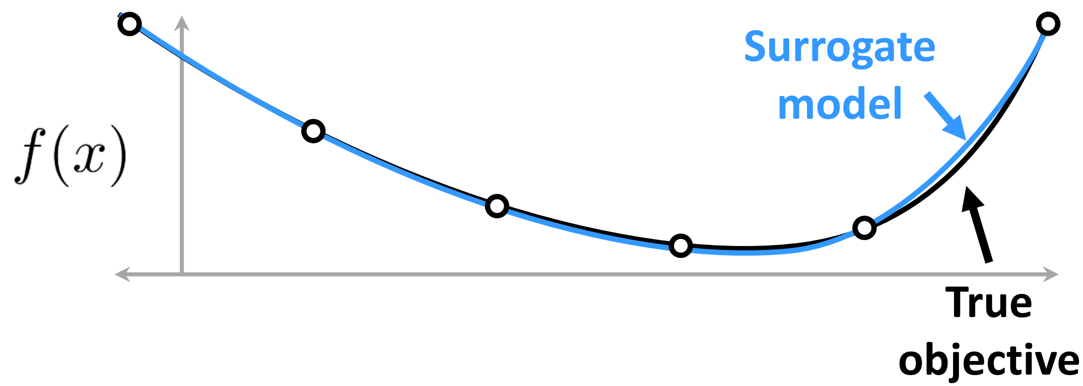
RECENT WORK IN CHEMICAL ENG

Modeling Methods Used

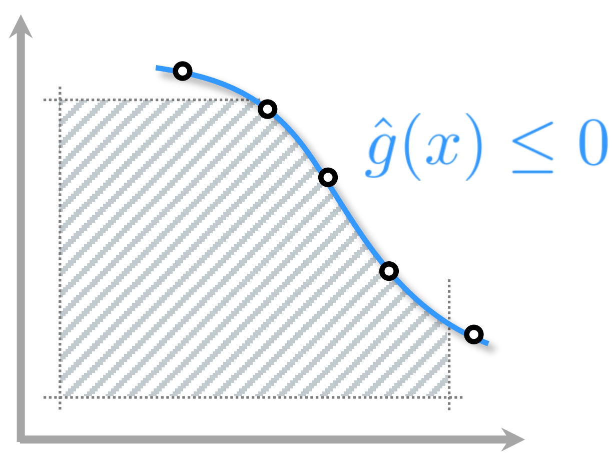
	Kriging	Neural nets	Polynomial-based
			$\hat{z}(x) = \sum_{j=1,2,\ldots} \beta_j x^j$
Full process 	▪ Palmer and Realff, 2002 ▪ Huang et al., 2006 ▪ Davis and Ierapetritou, 2012	▪ Michalopoulos et al., 2001	▪ Palmer and Realff, 2002
Disaggregated 	▪ Caballero and Grossmann, 2008	▪ Henao and Maravelias, 2011	

USE SURROGATE MODELS

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



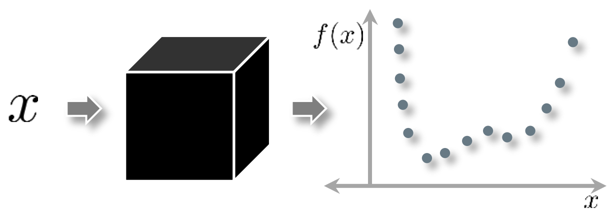
- **To replace black-box constraints**
  - Define the problem space
  - Generate equality or inequality constraints



CHALLENGES

SOURCE: Simulator (1)

No algebraic model



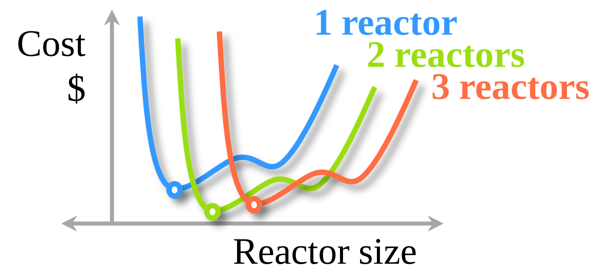
SOURCE : Optimizer (2)

Costly simulations

✗ **Gradient-based methods**

SOURCE : Simulator (3)

Complex process alternatives



SOURCE : Optimizer (4)

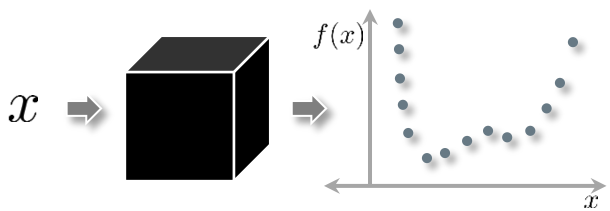
Scarcity of fully robust simulations

✗ **Derivative-free methods**

CHALLENGES

SOURCE: Simulator — 1

No algebraic model

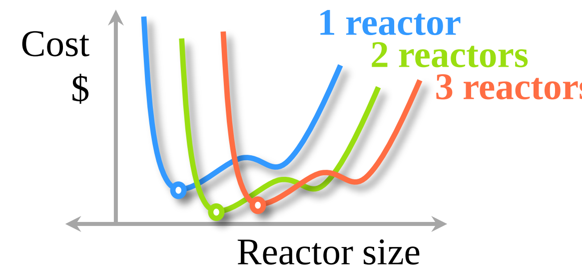


SOURCE : Optimizer — 2

Costly simulations

SOURCE : Simulator — 3

Complex process alternatives



SOURCE : Optimizer — 4

Scarcity of fully robust simulations

✗ **Gradient-based methods**

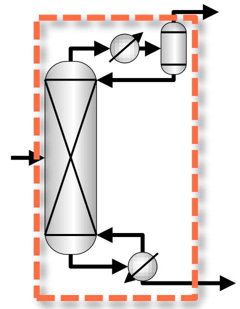
✗ **Derivative-free methods**

LEARNING PROBLEM STATEMENT

- **Model building problem:**
  - Build a model of output variables z as a function of input variables x over a specified interval

$$x \in \mathbb{R}^D \qquad x^l \le x \le x^u \qquad \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \\ \vdots \\ x_D \end{pmatrix}$$

Process simulation or experiment



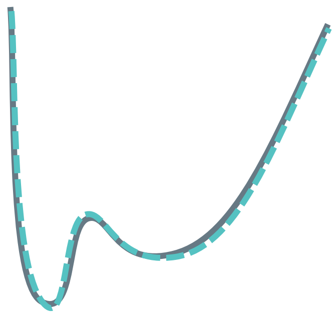
$$\begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_k \\ \vdots \\ z_K \end{pmatrix} \qquad z \in \mathbb{R}^K \qquad z = f(x)$$

Independent variables:
Operating conditions, inlet flow properties, unit geometry

Dependent variables:
Efficiency, outlet flow conditions, conversions, heat flow, etc.

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*



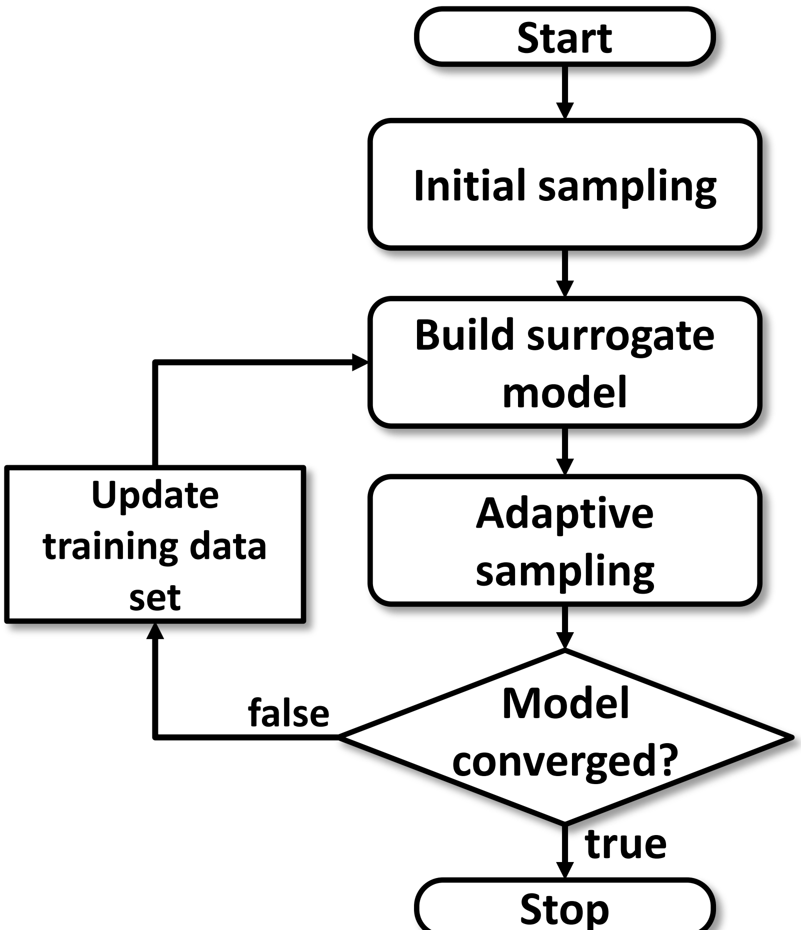
$$\hat{f}(x) = \sum_{i=1}^{n} \gamma_i \exp\left(\frac{\|x\|}{\sigma^2}\right) + \beta_0 + \beta_1 x + \dots$$

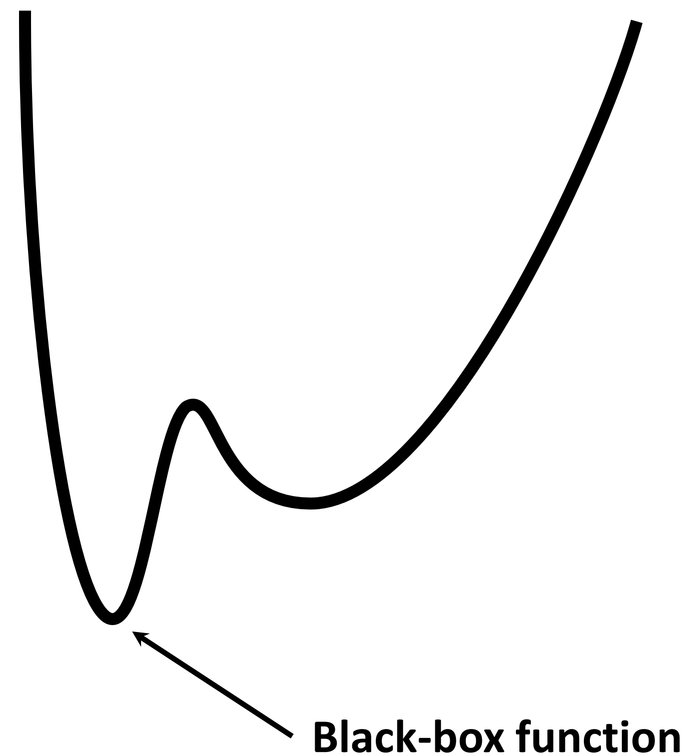
$$\hat{f}(x) = \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 e^x$$

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

ALAMO

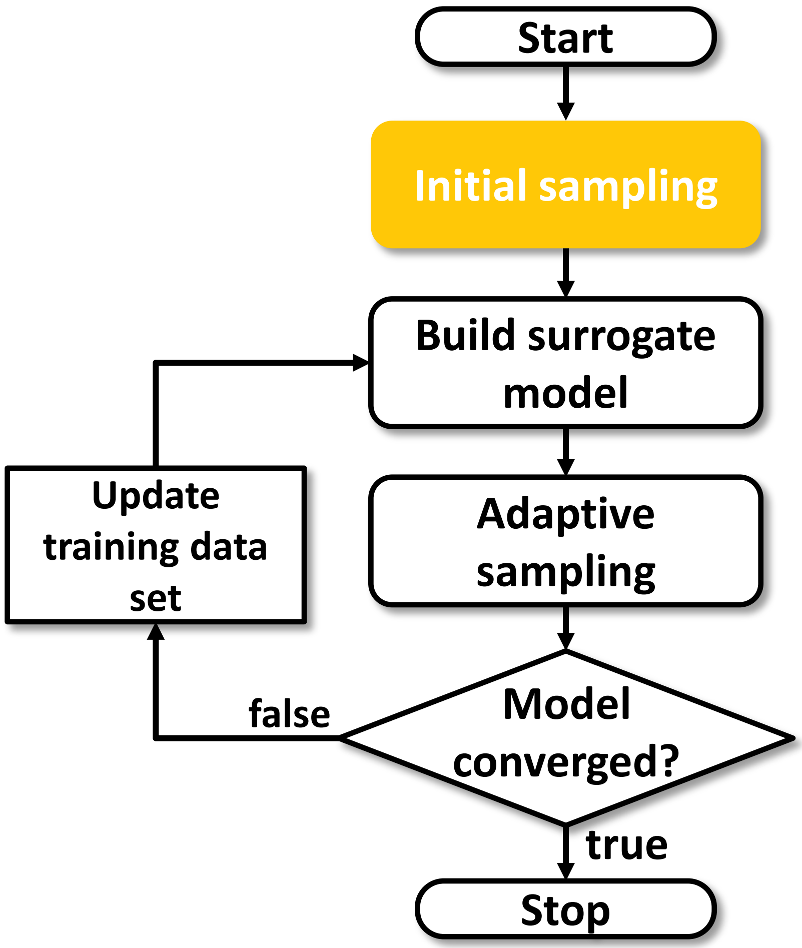
Automated Learning of Algebraic Models for Optimization

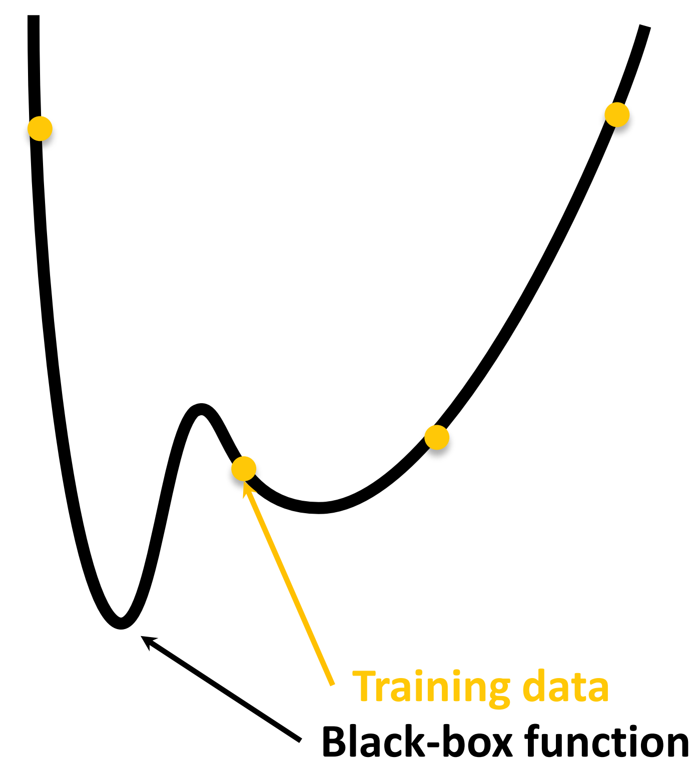




ALAMO

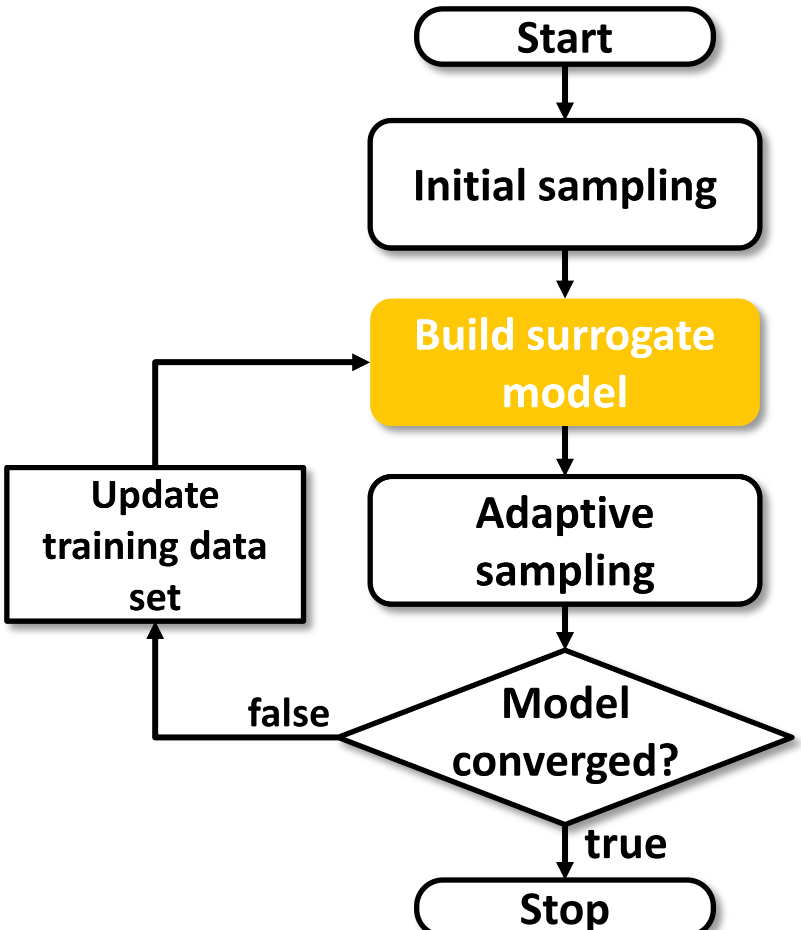
Automated Learning of Algebraic Models for Optimization

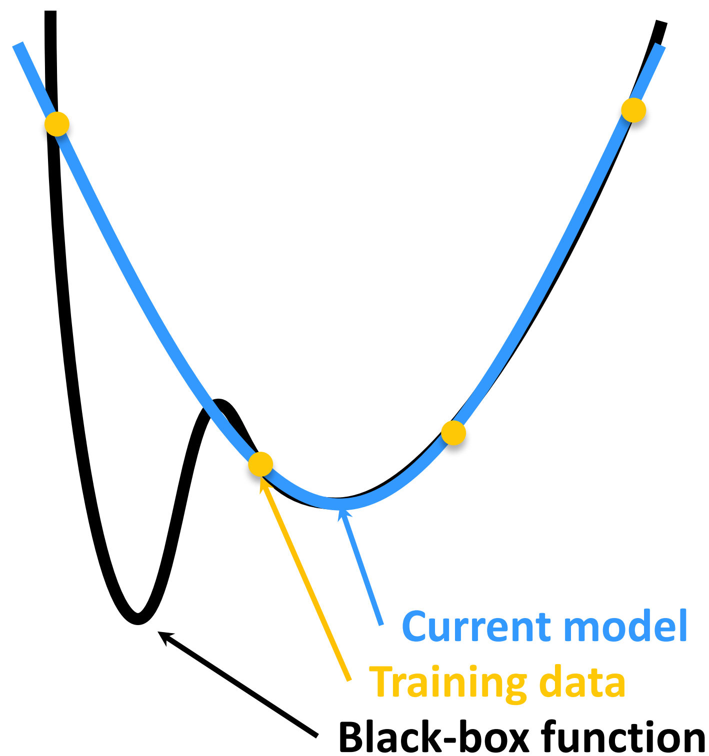




ALAMO

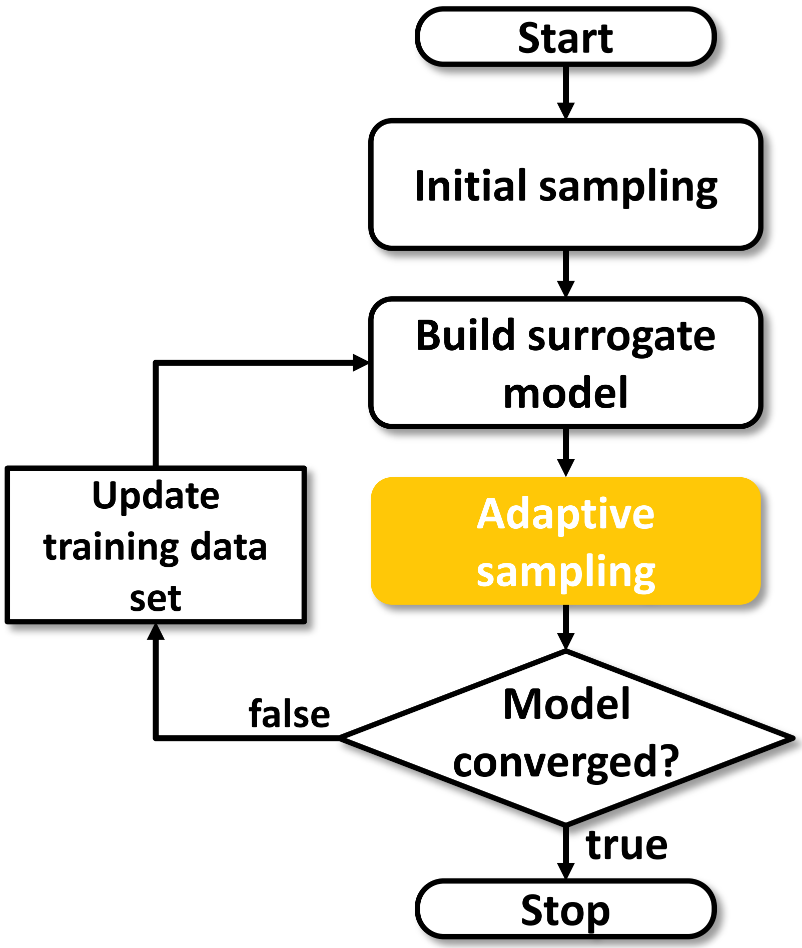
Automated Learning of Algebraic Models for Optimization

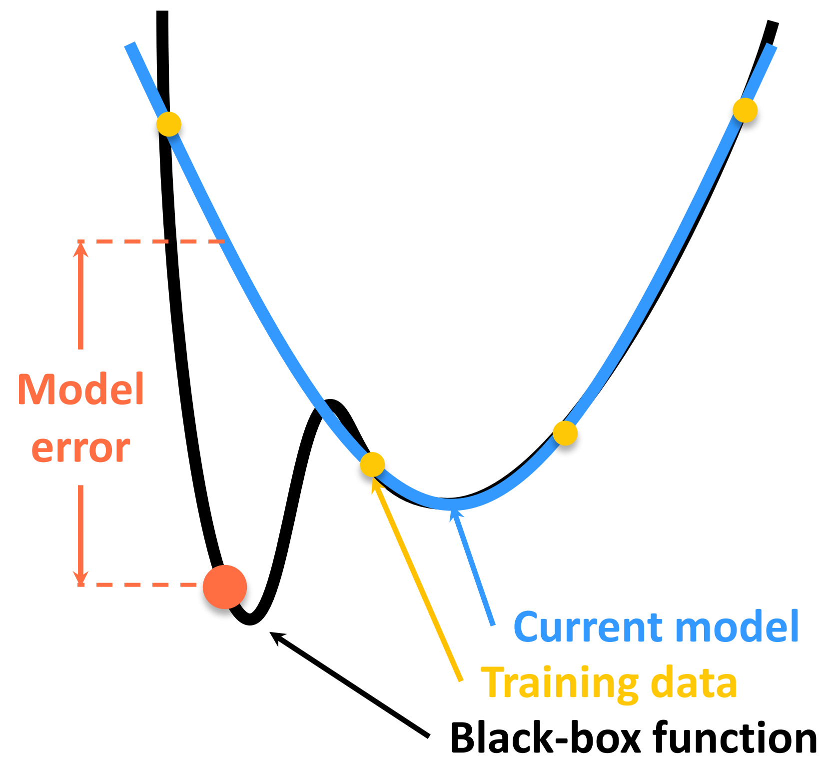




ALAMO

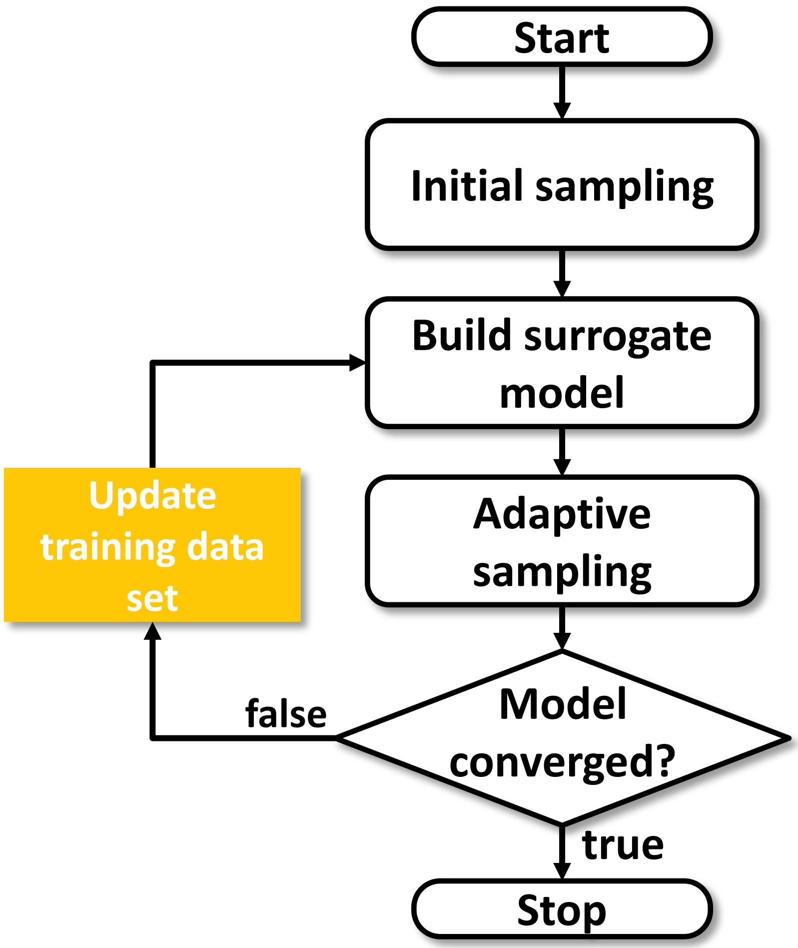
Automated **L**earning of **A**lgebraic **M**odels for **O**ptimization

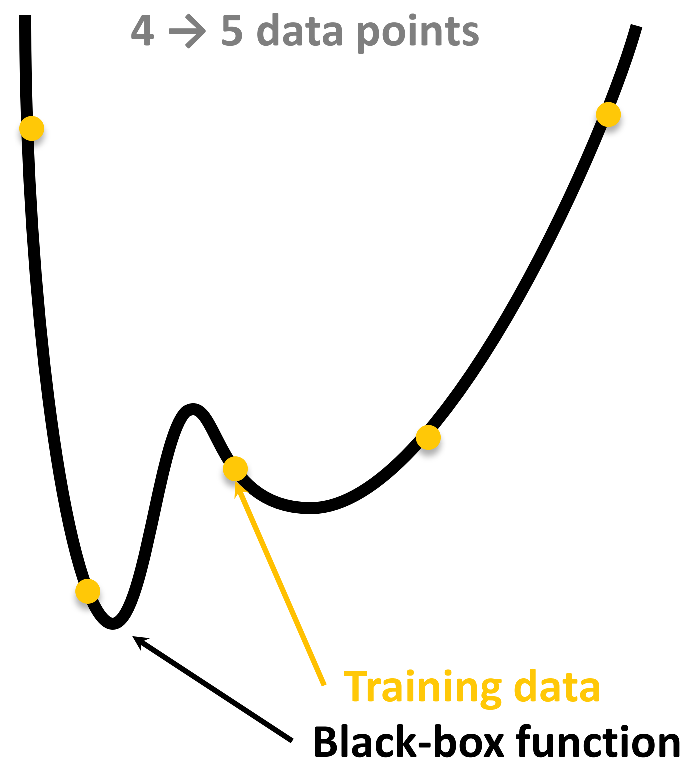




ALAMO

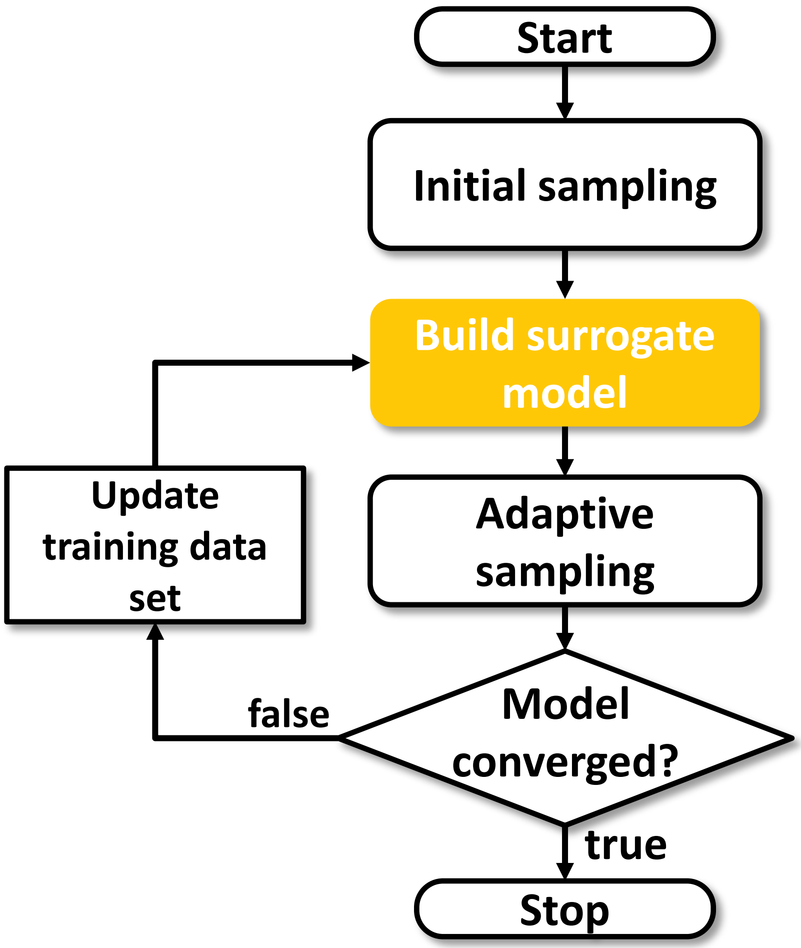
Automated Learning of Algebraic Models for Optimization

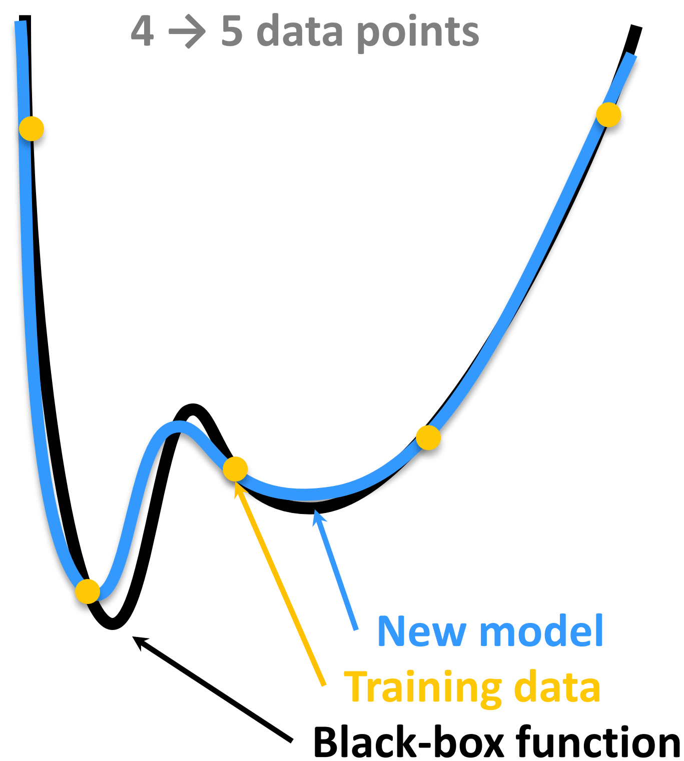




ALAMO

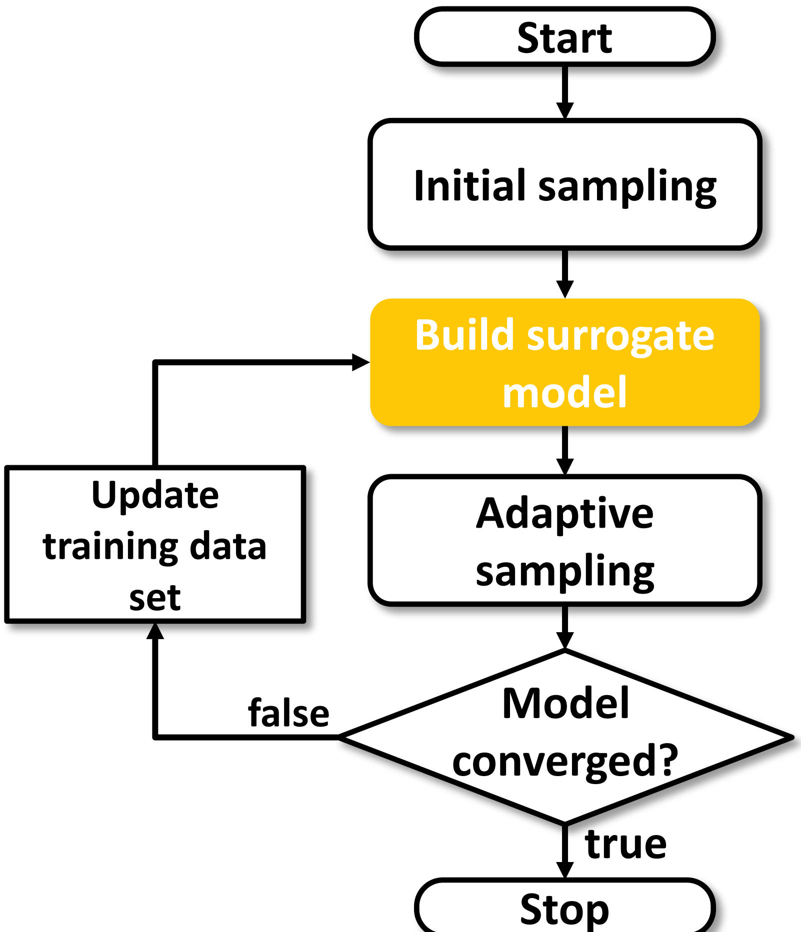
Automated Learning of Algebraic Models for Optimization

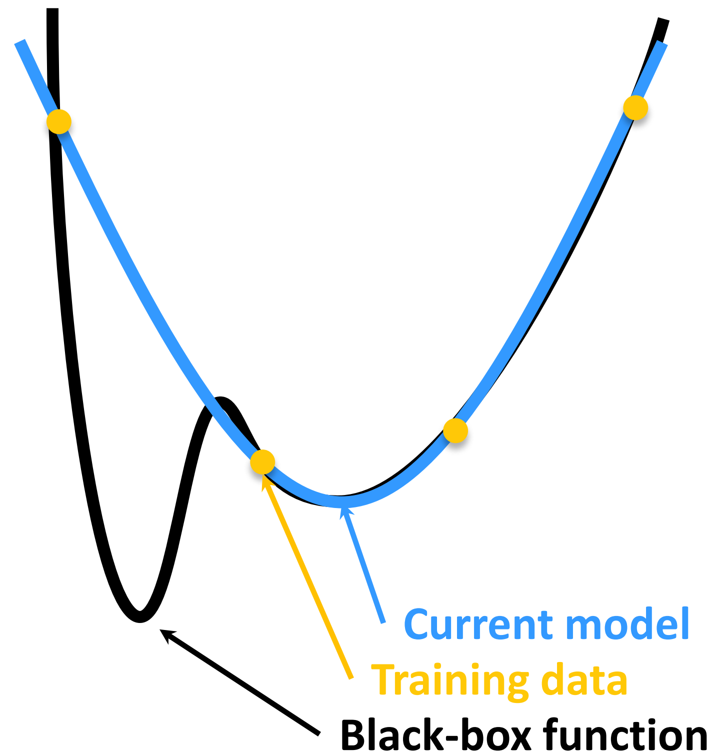




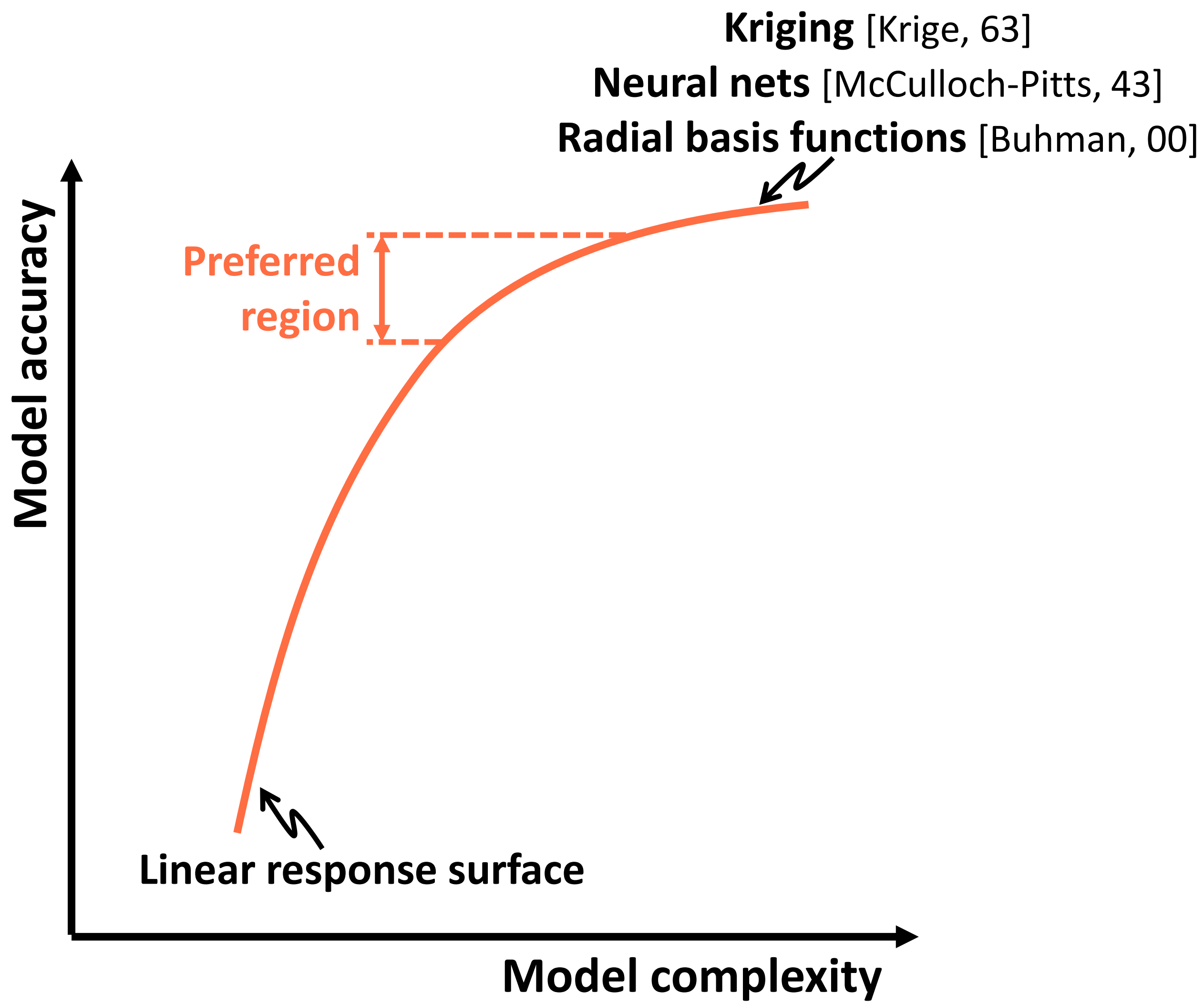
ALAMO: ADAPTIVE SAMPLING

Identifying simple, accurate models





MODEL COMPLEXITY TRADEOFF



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.

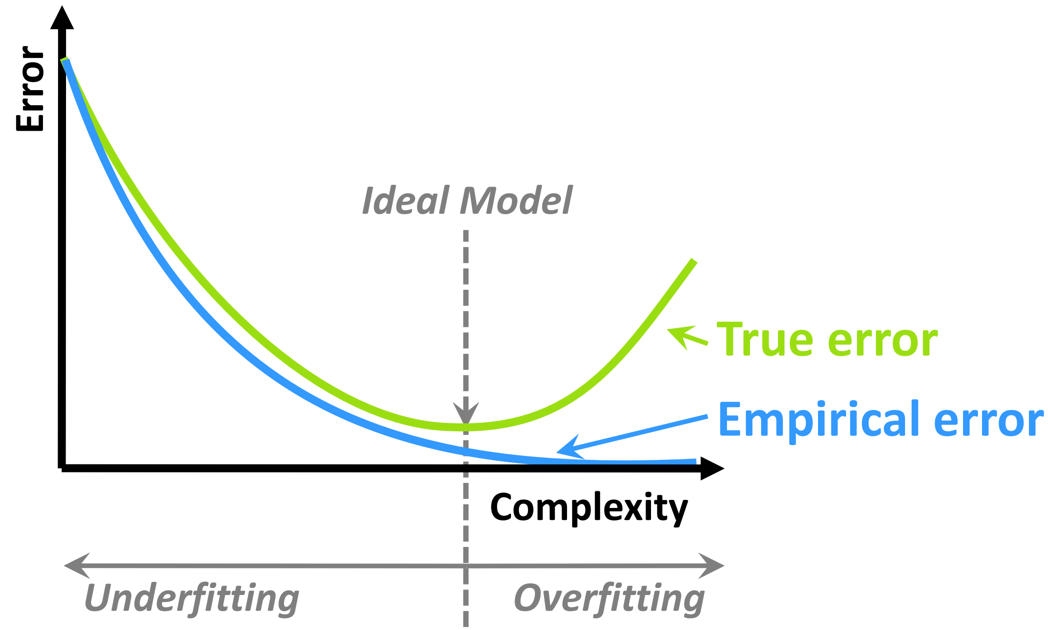
OVERFITTING AND TRUE ERROR

Step 1: *Define a large set of potential basis functions*

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

Step 2: *Model reduction*

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



MODEL REDUCTION TECHNIQUES

- **Qualitative tradeoffs of model reduction methods**

CPU modeling cost

Efficacy

Best subset methods
- *Enumerate all possible subsets*

Regularized regression techniques
- *Penalize the least squares objective using the magnitude of the regressors*

Stepwise regression [Efroymson, 60]

Backward elimination [Oosterhof, 63]

Forward selection [Hamaker, 62]

MODEL REDUCTION TECHNIQUES

- **Qualitative tradeoffs of model reduction methods**

To solve large problems we
- Use optimization rather than enumeration
- Decouple the model identification into
  1. Model size
  2. Term selection

Best subset methods
- *Enumerate all possible subsets*

Regularized regression techniques
- *Penalize the least squares objective using the magnitude of the regressors*

CPU modeling cost

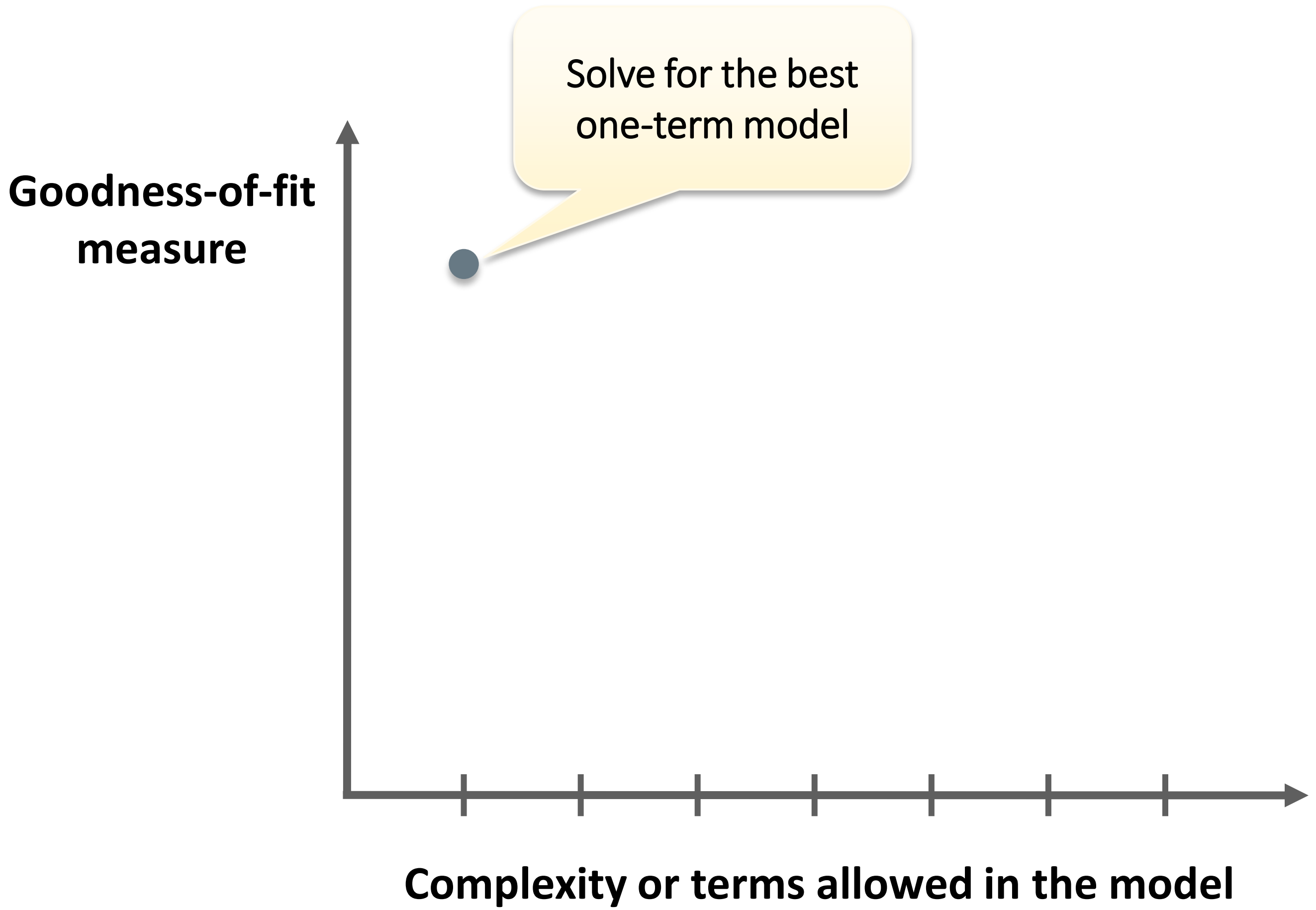
Efficacy

Stepwise regression [Efroymson, 60]

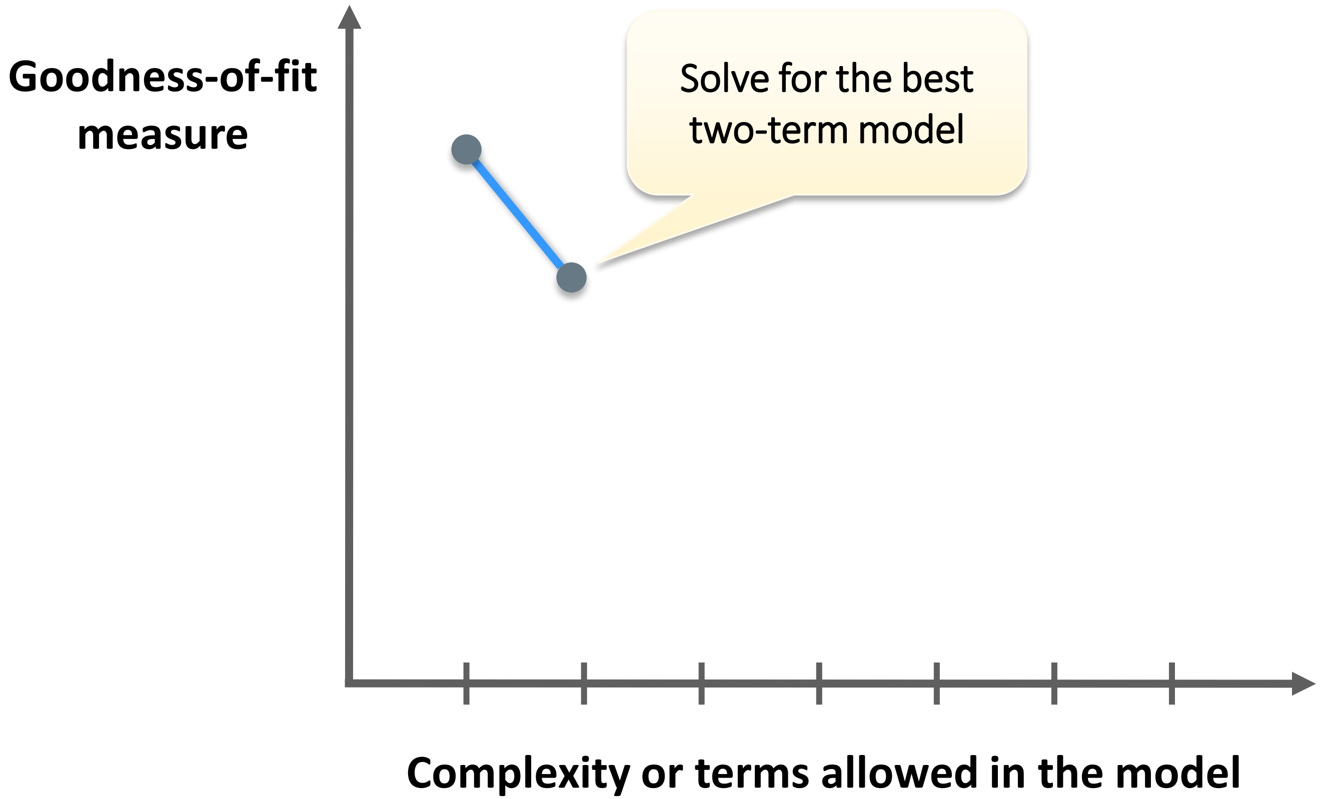
Backward elimination [Oosterhof, 63]

Forward selection [Hamaker, 62]

MODEL SIZING



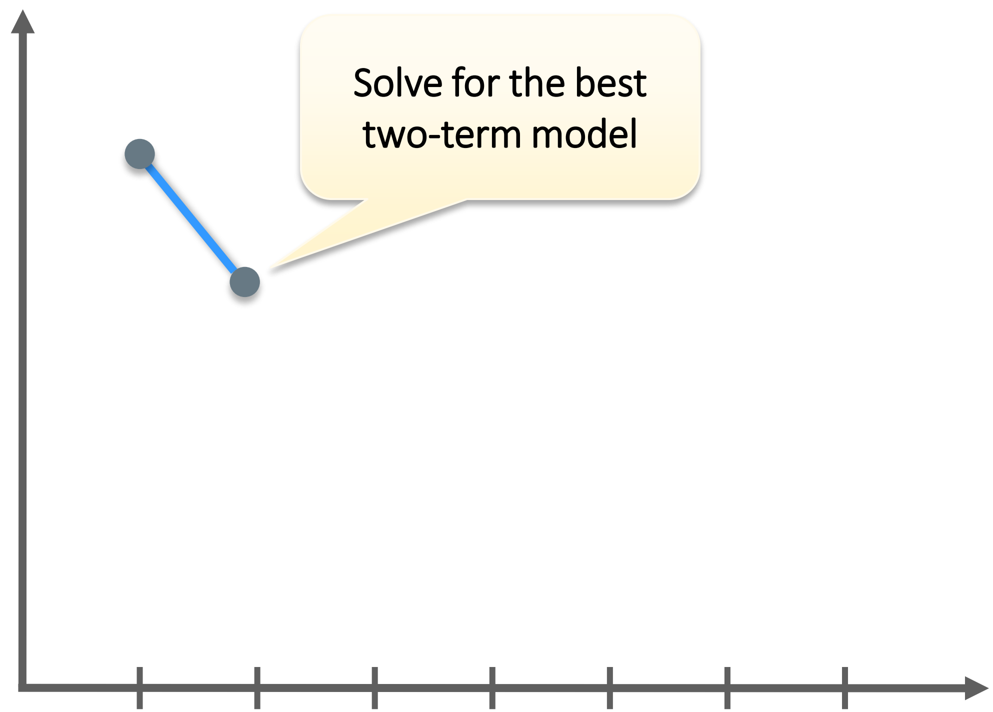
MODEL SIZING



MODEL SIZING

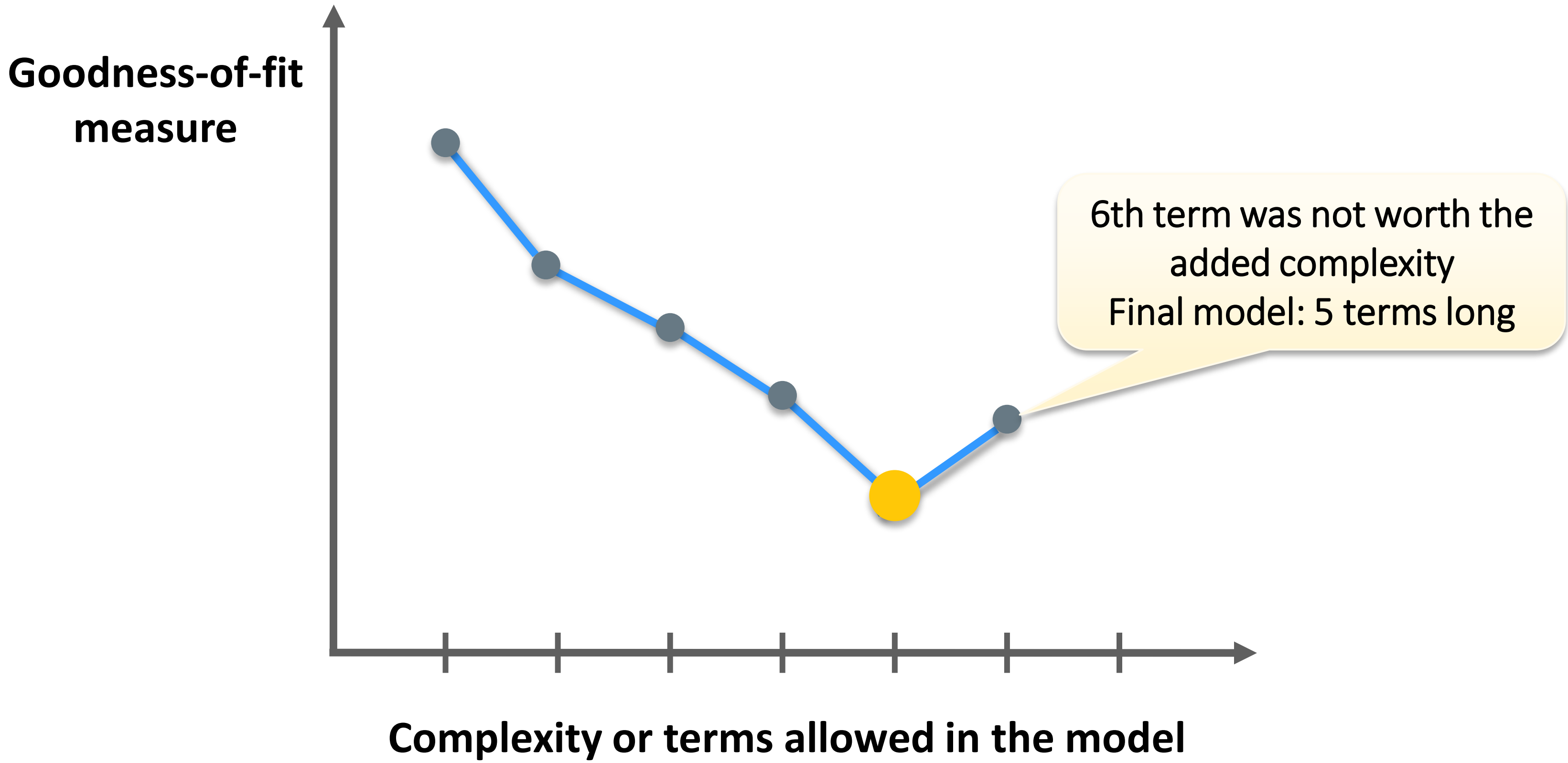
Goodness-of-fit measure

Some measure of error that is sensitive to overfitting (AICc)



Complexity or terms allowed in the model

MODEL SIZING



BASIS FUNCTION SELECTION

$$
\begin{aligned}
\min \quad & SE = \sum_{i=1}^{N} \left| z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right| \\
\text{s.t.} \quad & \sum_{j \in \mathcal{B}} y_j = T \\
& -U(1-y_j) \leq \sum_{i=1}^{N} X_{ij} \left(z^i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right) \leq U(1-y_j) && j \in \mathcal{B} \\
& \beta^l y_j \leq \beta_j \leq \beta^u y_j && j \in \mathcal{B} \\
& y_j = \{0, 1\} && j \in \mathcal{B}
\end{aligned}
$$

BASIS FUNCTION SELECTION

$$\min \quad SE = \sum_{i=1}^{N} \left| z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right|$$

Find the model with the least error

$$\text{s.t.} \quad \sum_{j \in \mathcal{B}} y_j = T$$

$$-U(1-y_j) \leq \sum_{i=1}^{N} X_{ij} \left(z^i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right) \leq U(1-y_j) \qquad j \in \mathcal{B}$$

$$\beta^l y_j \leq \beta_j \leq \beta^u y_j \qquad j \in \mathcal{B}$$

$$y_j = \{0, 1\} \qquad j \in \mathcal{B}$$

BASIS FUNCTION SELECTION

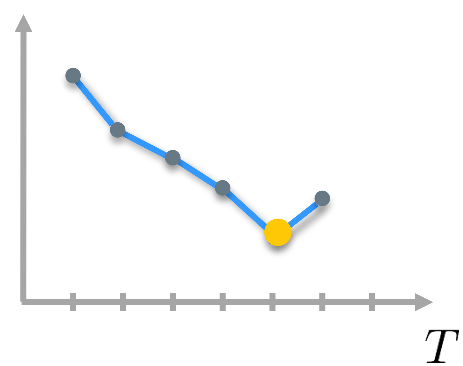
$$\min \quad SE = \sum_{i=1}^{N} \left| z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right|$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{B}} y_j = T$$

$$-U(1-y_j) \leq \sum_{i=1}^{N} X_{ij} \left(z^i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right) \leq U(1-y_j) \qquad j \in \mathcal{B}$$

$$\beta^l y_j \leq \beta_j \leq \beta^u y_j$$

$$y_j = \{0, 1\}$$



We will solve this model for increasing T until we determine a model

BASIS FUNCTION SELECTION

$$\min \quad SE = \sum_{i=1}^{N} \left| z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right|$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{B}} y_j = T$$

$$-U(1-y_j) \leq \sum_{i=1}^{N} X_{ij} \left(z^i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right) \leq U(1-y_j) \qquad j \in \mathcal{B}$$

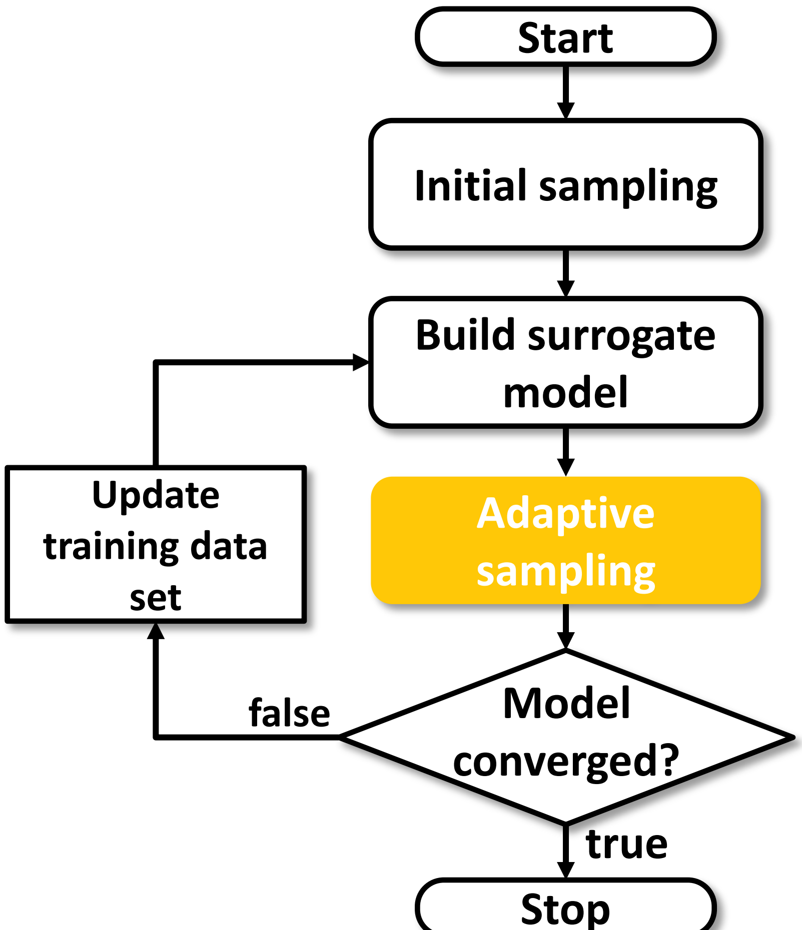
$$\beta^l y_j \leq \beta_j \leq \beta^u y_j \qquad j \in \mathcal{B}$$

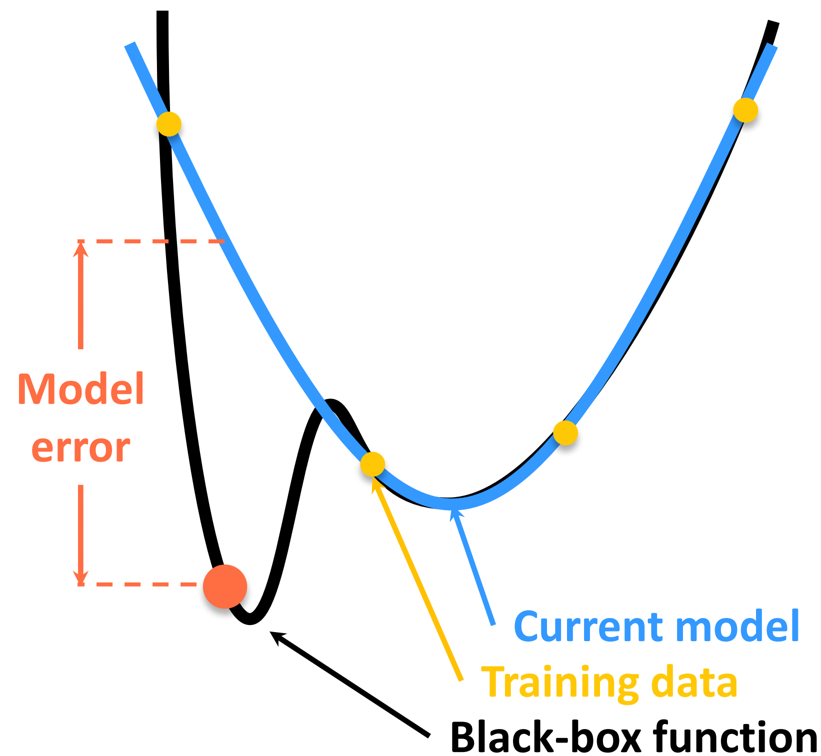
$$y_j = \{0,1\} \qquad j \in \mathcal{B}$$

$$\left(\begin{array}{c} \mathbf{y_j = 1} \\ \textbf{Basis function used in the model} \\ \beta_j \text{ is chosen to satisfy a least} \\ \text{squares regression} \\ (\textit{assumes loose bounds on } \beta_j) \end{array} \right) \bigvee \left(\begin{array}{c} \mathbf{y_j = 0} \\ \textbf{Basis function NOT used} \\ \textbf{in the model} \\ \beta_j = 0 \end{array} \right)$$

ALAMO: ADAPTIVE SAMPLING

Choosing new data points to sample





ERROR MAXIMIZATION SAMPLING

- **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**

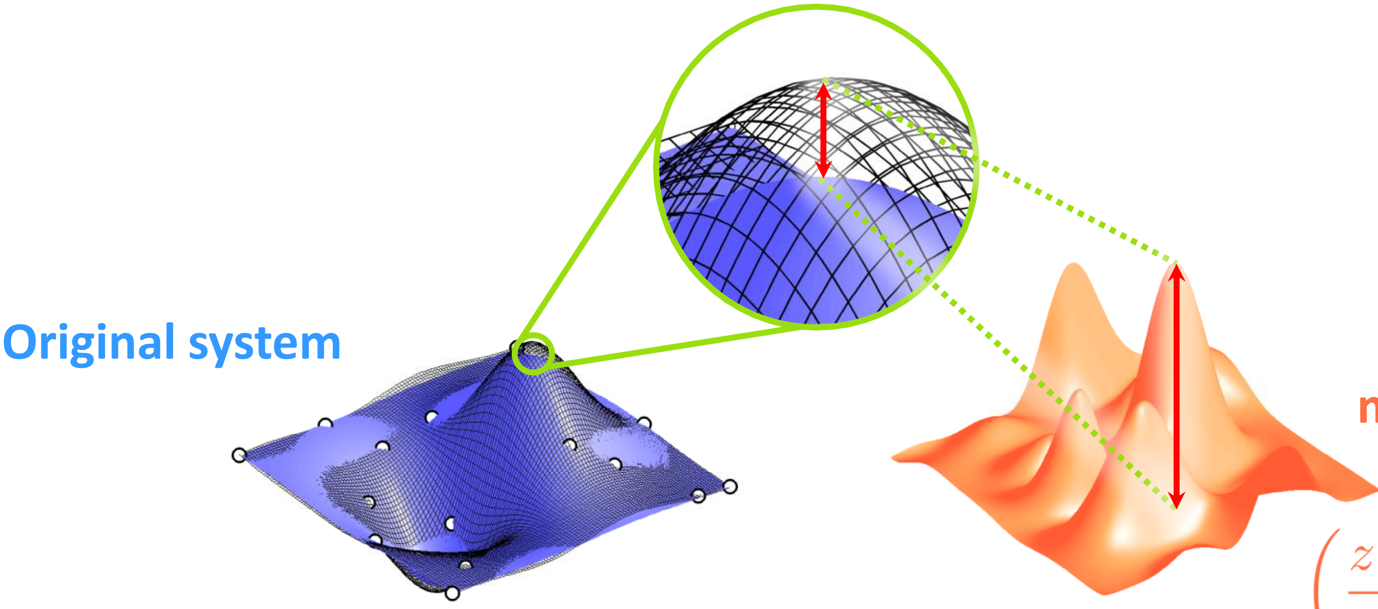
Surrogate model

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

Black-box value

  - **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



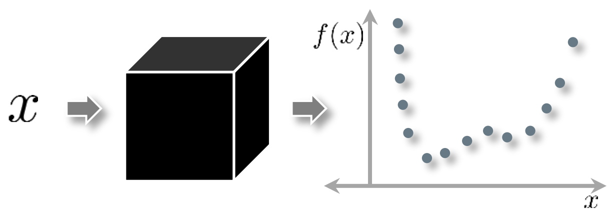
$$\left(\frac{z(x) - \hat{z}(x)}{z(x)}\right)^2$$

- **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*

CHALLENGES

SOURCE: Simulator ✓ 1

No algebraic model



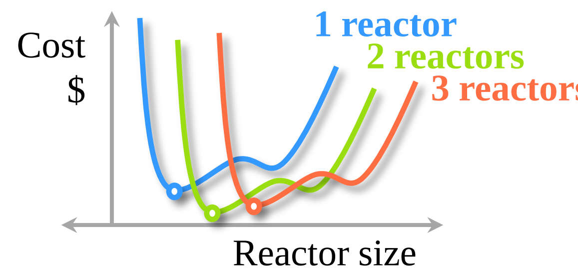
SOURCE : Optimizer ✓ 2

Costly simulations

✗ Gradient-based methods

SOURCE : Simulator ✓ 3

Complex process alternatives



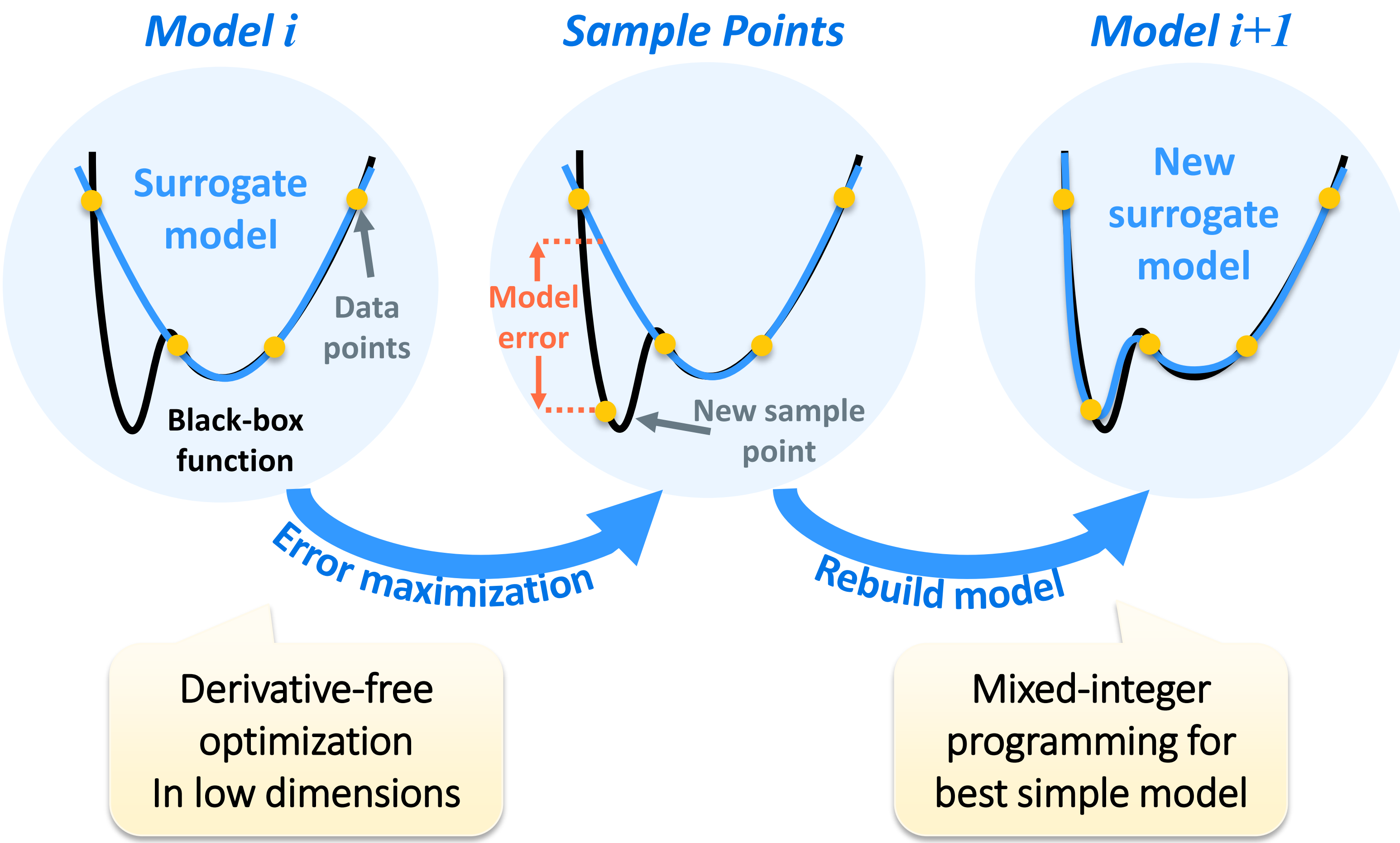
SOURCE : Optimizer ✓ 4

Scarcity of fully robust simulations

✗ Derivative-free methods

SYNOPSIS

- **Leverage accurate, simple, efficiently build surrogate models to expand the scope of MINLPs**



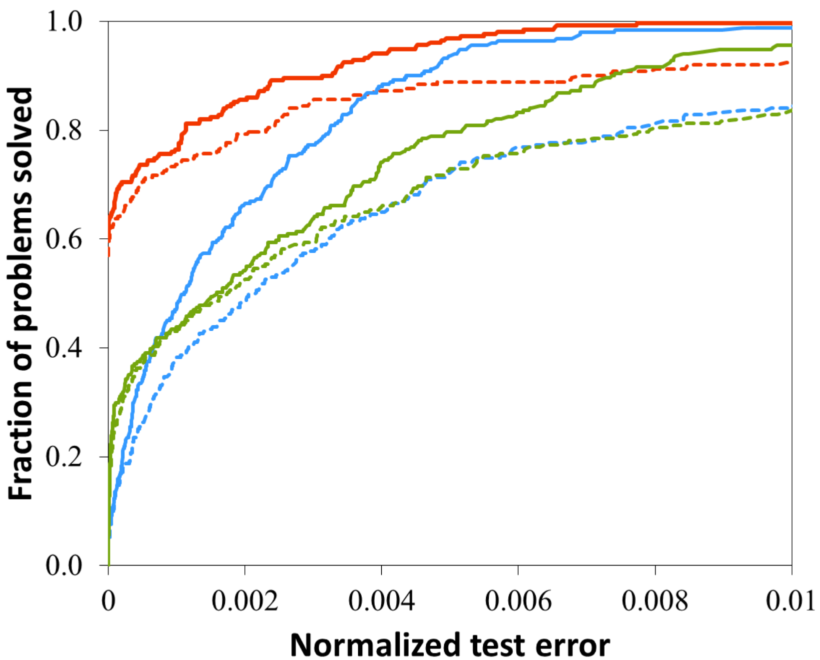
ACCURATE, SIMPLE, AND EFFICIENT

Computational experiments to validate ALAMO

- **Goal - Test the accuracy, efficiency, and model simplicity**
- **Modeling methods compared**
  - MIP – Proposed methodology
  - LASSO – The lasso regularization
  - OLR – Ordinary least-squares regression
- **Sampling methods compared**
  - EMS – Proposed error maximization technique
  - SLH – Single Latin hypercube (no feedback)
- **Two test sets**
  - Test set A – Bases available to ALAMO
  - Test set B – Functions with forms not available to ALAMO

COMPUTATIONAL EXPERIMENTS

Model accuracy

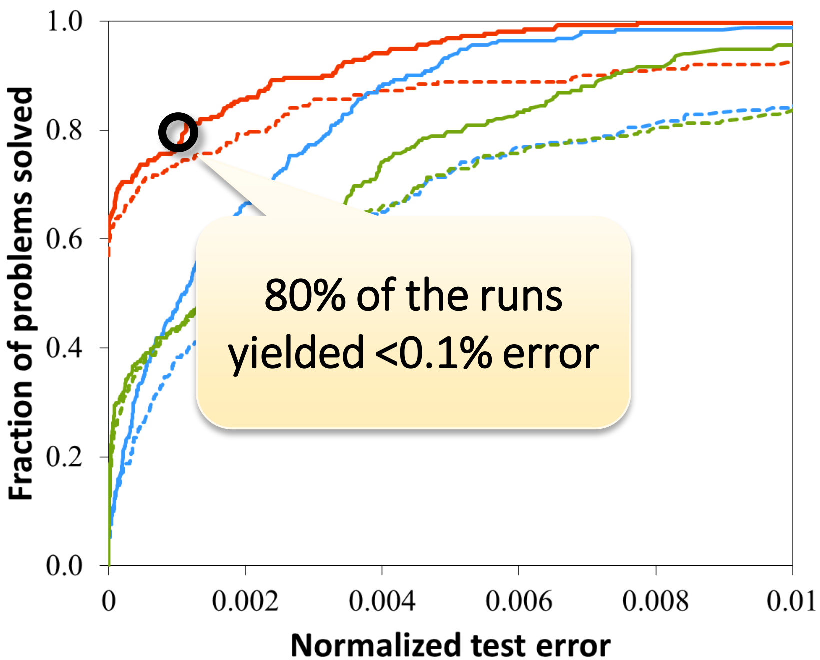


Modeling methods

Our method | LASSO | Least squares

COMPUTATIONAL EXPERIMENTS

Model accuracy

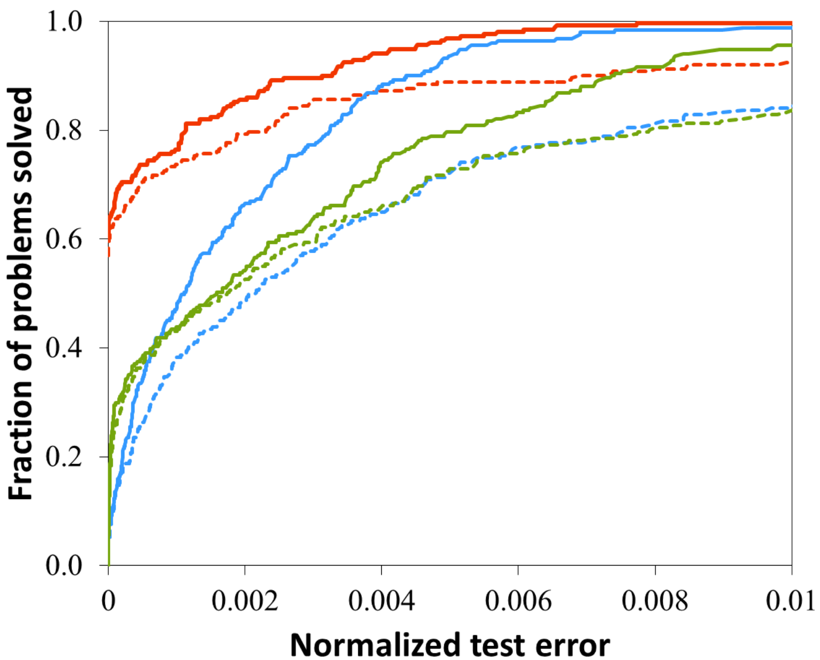


Modeling methods

Our method	LASSO	Least squares

COMPUTATIONAL EXPERIMENTS

Model accuracy

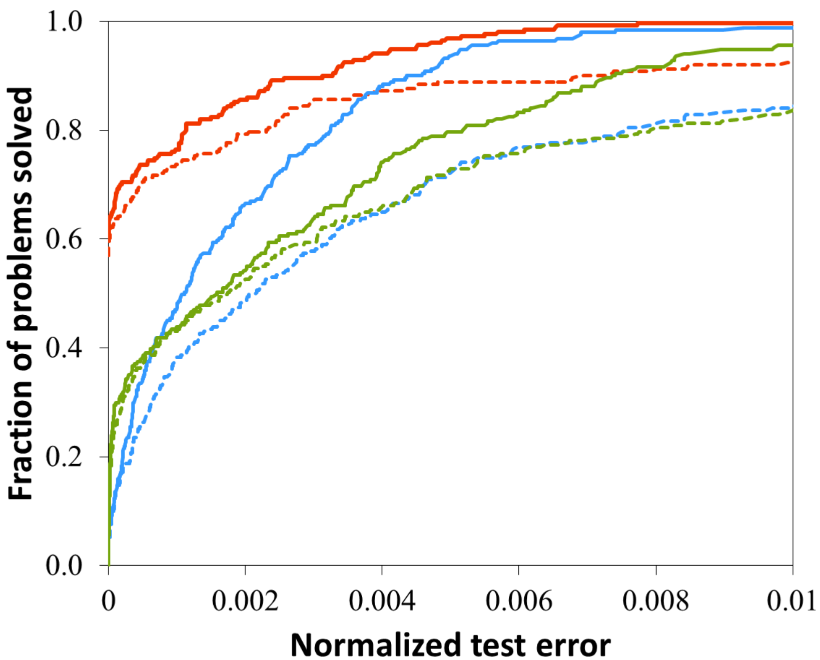


Modeling methods

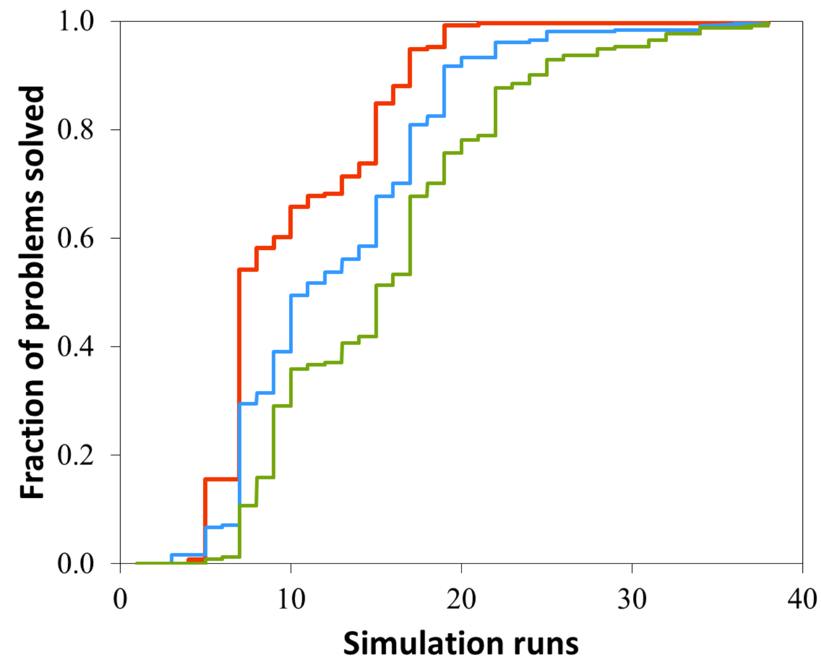
Our method | **LASSO** | **Least squares**

COMPUTATIONAL EXPERIMENTS

Model accuracy



Modeling efficiency



Modeling methods

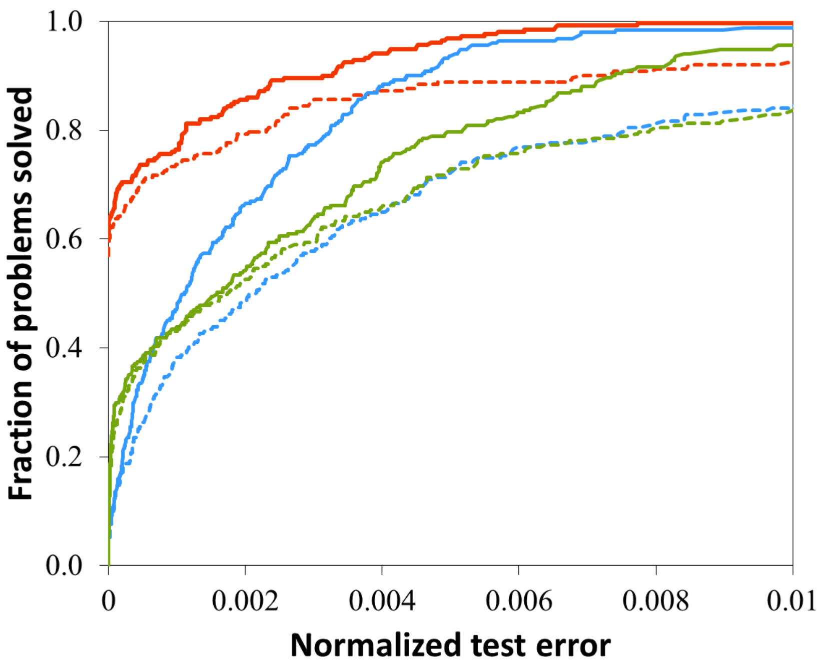
Our method | LASSO | Least squares

Sampling methods

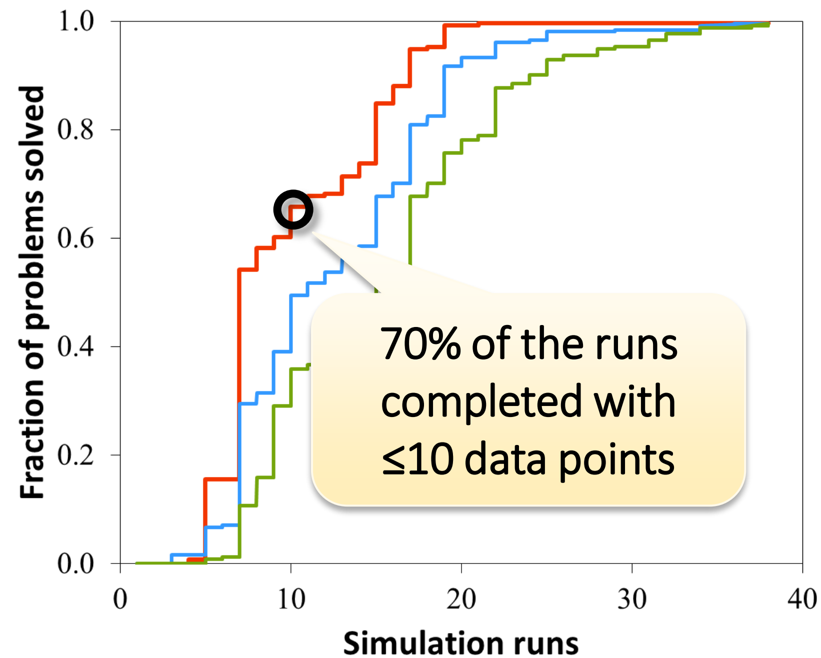
Error maximization | Single Latin hypercube

COMPUTATIONAL EXPERIMENTS

Model accuracy



Modeling efficiency



Modeling methods

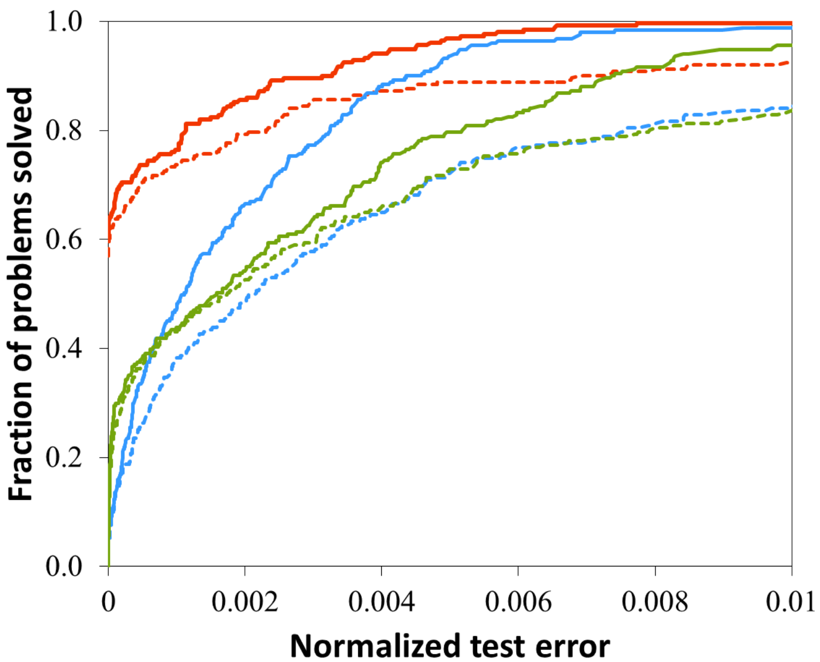
Our method | **LASSO** | **Least squares**

Sampling methods

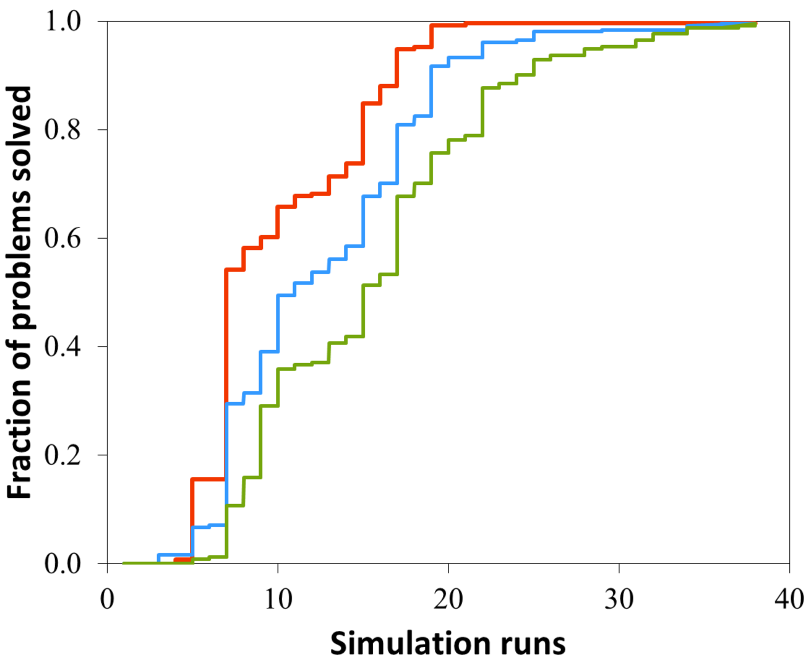
Error maximization | **Single Latin hypercube**

COMPUTATIONAL EXPERIMENTS

Model accuracy



Modeling efficiency



Modeling methods

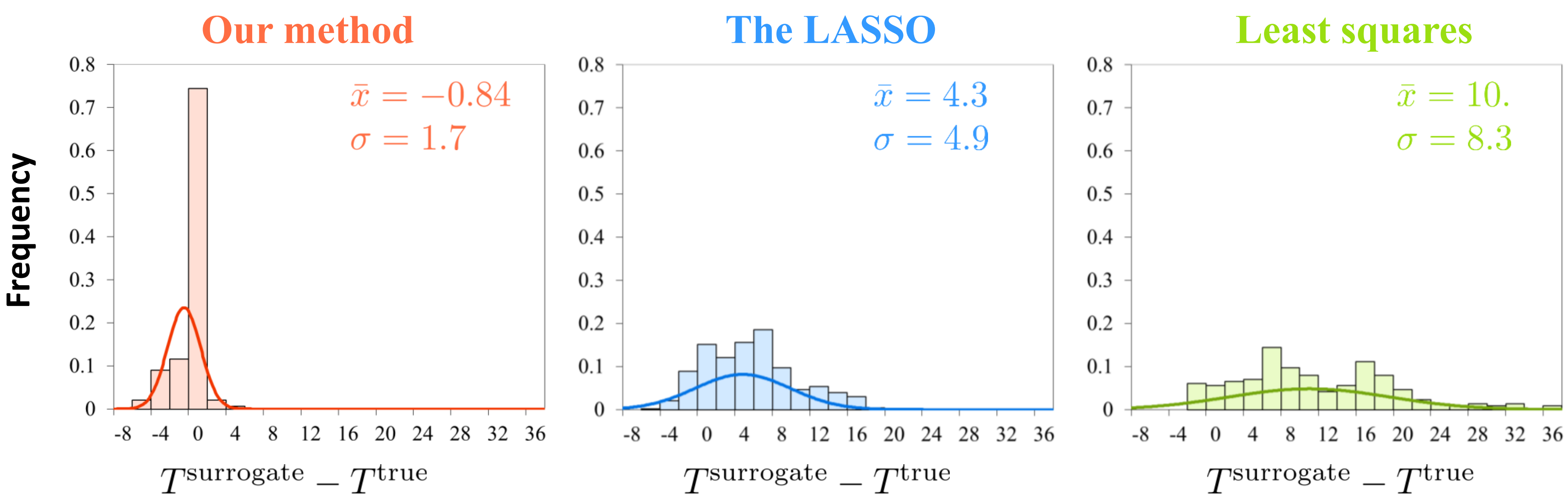
Our method | LASSO | Least squares

Sampling methods

Error maximization | Single Latin hypercube

MODEL SIZING RESULTS

$$\begin{bmatrix} \text{No. of terms in the} \\ \text{surrogate model} \end{bmatrix} - \begin{bmatrix} \text{No. of terms in} \\ \text{the true function} \end{bmatrix}$$



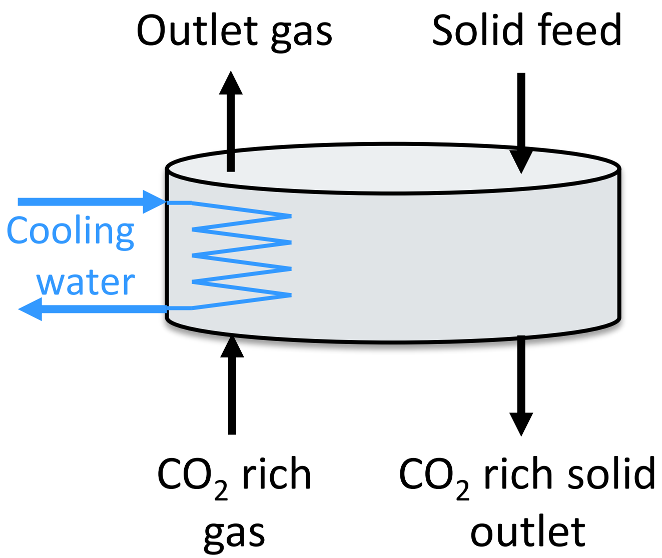
45 problems with 2-10 available bases, 5 repeats

REMARKS

- **Model building**
  - The ALAMO model building method shows the highest accuracy, using the fewest data points, while giving the most simple models
- **Experimental design**
  - The Error Maximization Sampling method used provides more information per data point sampled resulting in more accurate models with a given data set size
- **ALAMO availability**
  - Licensed through the National Energy Technology Laboratory (Department of Energy Lab) to several industrial companies

ILLUSTRATIVE EXAMPLE

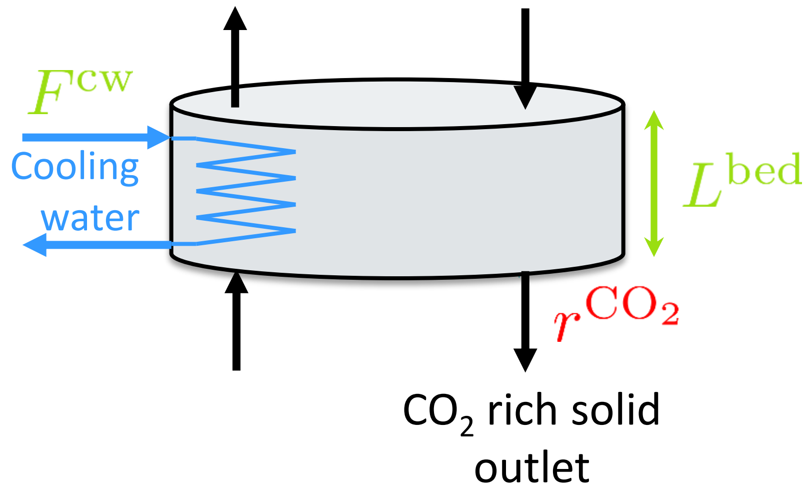
Bubbling fluidized bed adsorber



- **Goal: Optimize a bubbling fluidized bed reactor by**
  - Minimizing the cost of electricity
  - Maximizing CO2 removal

ILLUSTRATIVE EXAMPLE

Bubbling fluidized bed adsorber



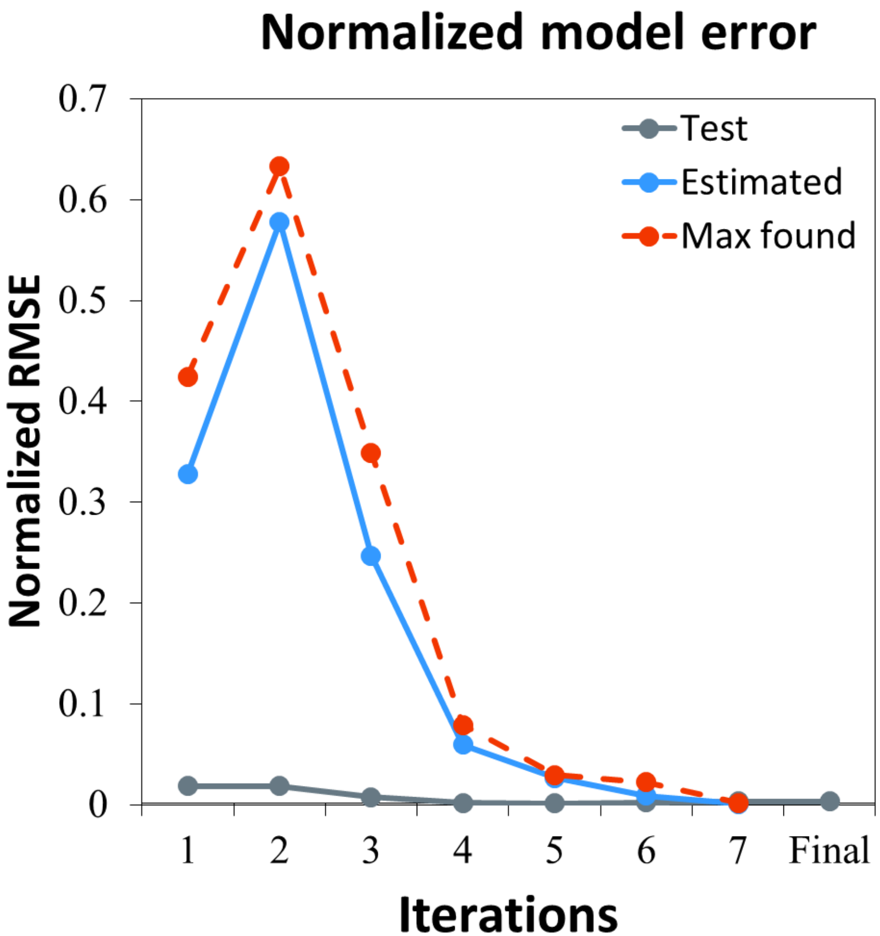
- **Generate model of % CO2 removal:**

$$r^{\mathrm{CO_2}}(L^{\mathrm{bed}}, F^{\mathrm{cw}}) = f_1(L^{\mathrm{bed}}, F^{\mathrm{cw}})$$

- **Problem space:**

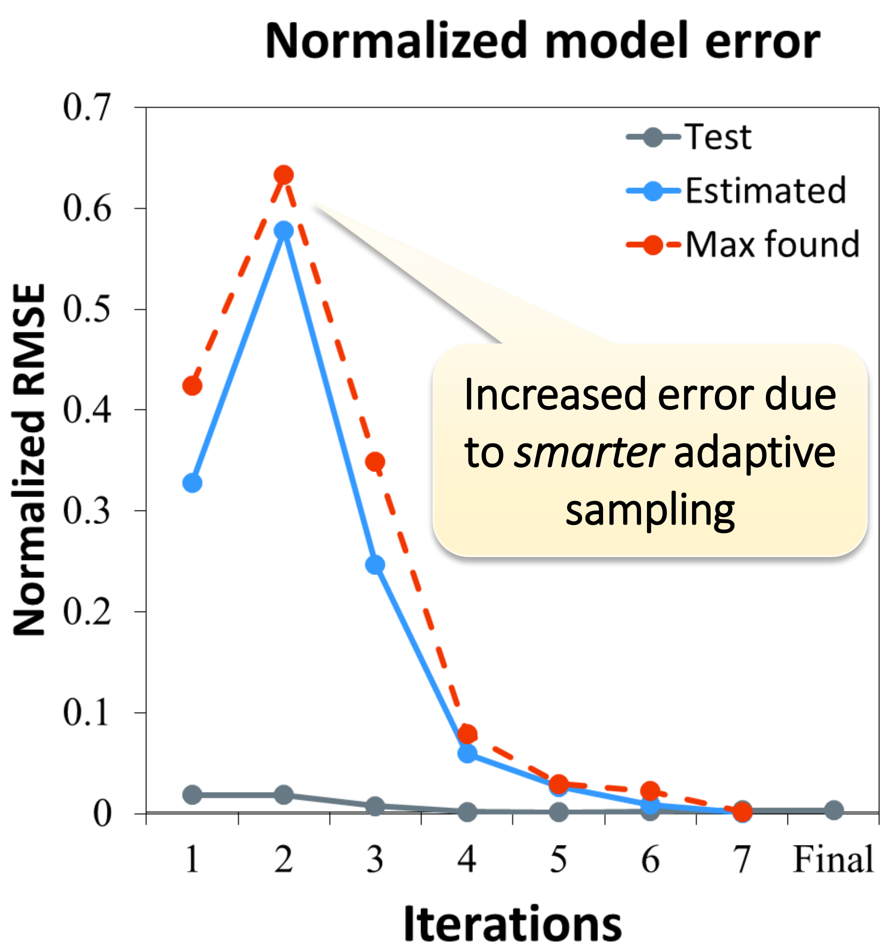
$$\begin{aligned} 1\ \mathrm{m} &\leq L^{\mathrm{bed}} &&\leq 10\ \mathrm{m} \\ 1 \cdot 10^4\ \frac{\mathrm{kmol}}{\mathrm{h}} &\leq F^{\mathrm{cw}} &&\leq 20 \cdot 10^4\ \frac{\mathrm{kmol}}{\mathrm{h}} \end{aligned}$$

ALGORITHM PROGRESS



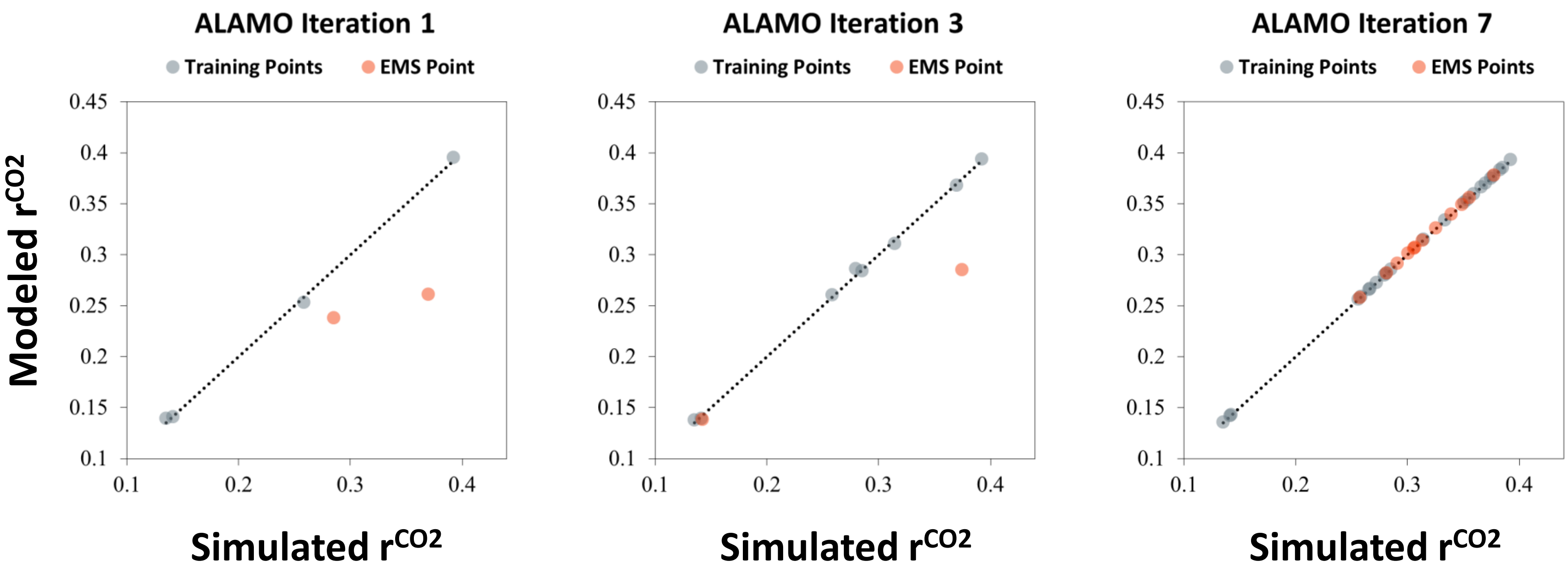
Iteration	Training points
1	4
2	6
3	8
4	10
5	12
6	14
7	23
Final	35

ALGORITHM PROGRESS



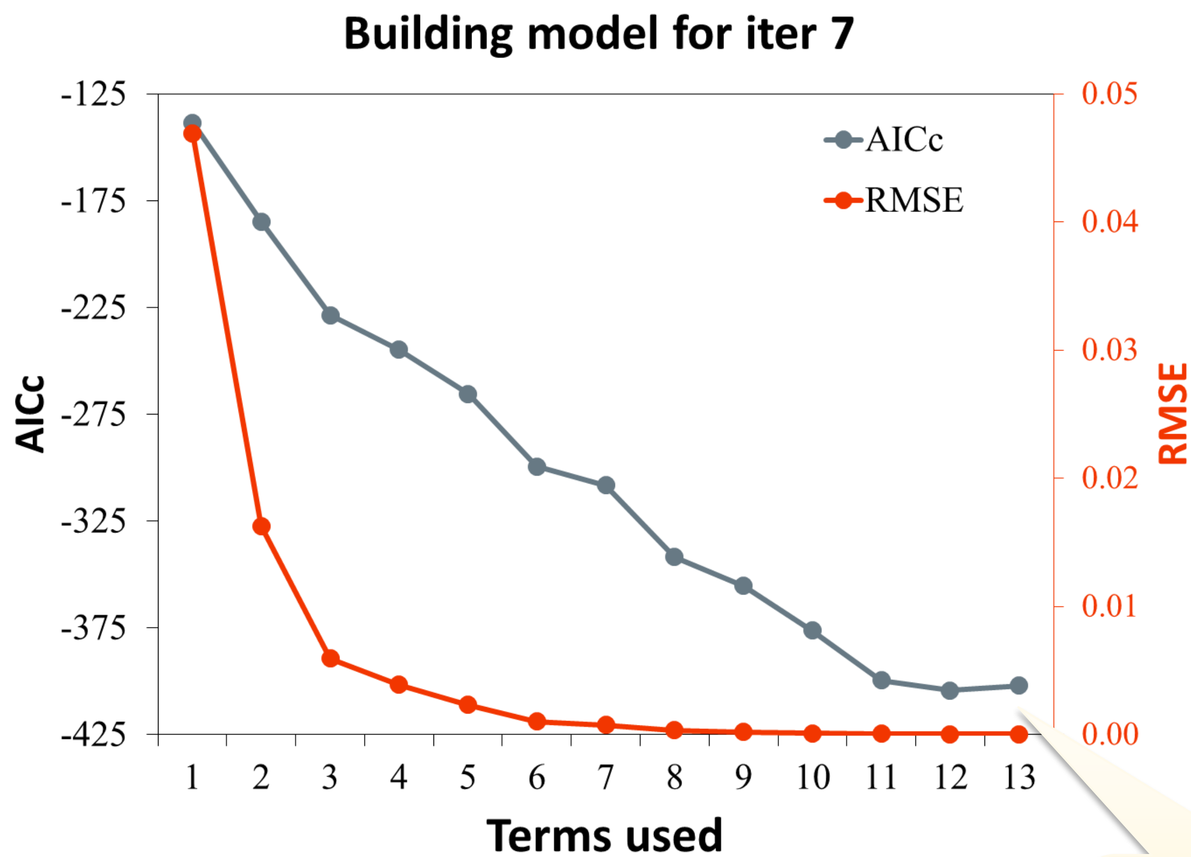
Iteration	Training points
1	4
2	6
3	8
4	10
5	12
6	14
7	23
Final	35

ITERATION SNAPSHOTS



Iteration	Terms (max 67)	Model
1	2	$\left(8.1\cdot 10^{-7}\right)\, F\,\sqrt{L} + 0.14$
3	3	$0.014\,\sqrt[3]{F} - \left(4.4\,F\,\sqrt{L} + 1.1\,F\,L\right)\cdot 10^{-6}$
7	12	$-\frac{4.7\,L}{\sqrt{F}} + \frac{0.39}{\sqrt[3]{L}} + 0.15\,\sqrt[3]{L} + \frac{8.7\,L^2}{F} + \left(\frac{3300}{F\,L}\right)^2 - \left(\frac{2500}{F\,L}\right)^4 + \left(-\frac{0.01\,F}{\sqrt{L}} - 5.5\,\sqrt{F}\,L + 5.6\,\sqrt{F} + 41\,L^2\right)\cdot 10^{-5}$

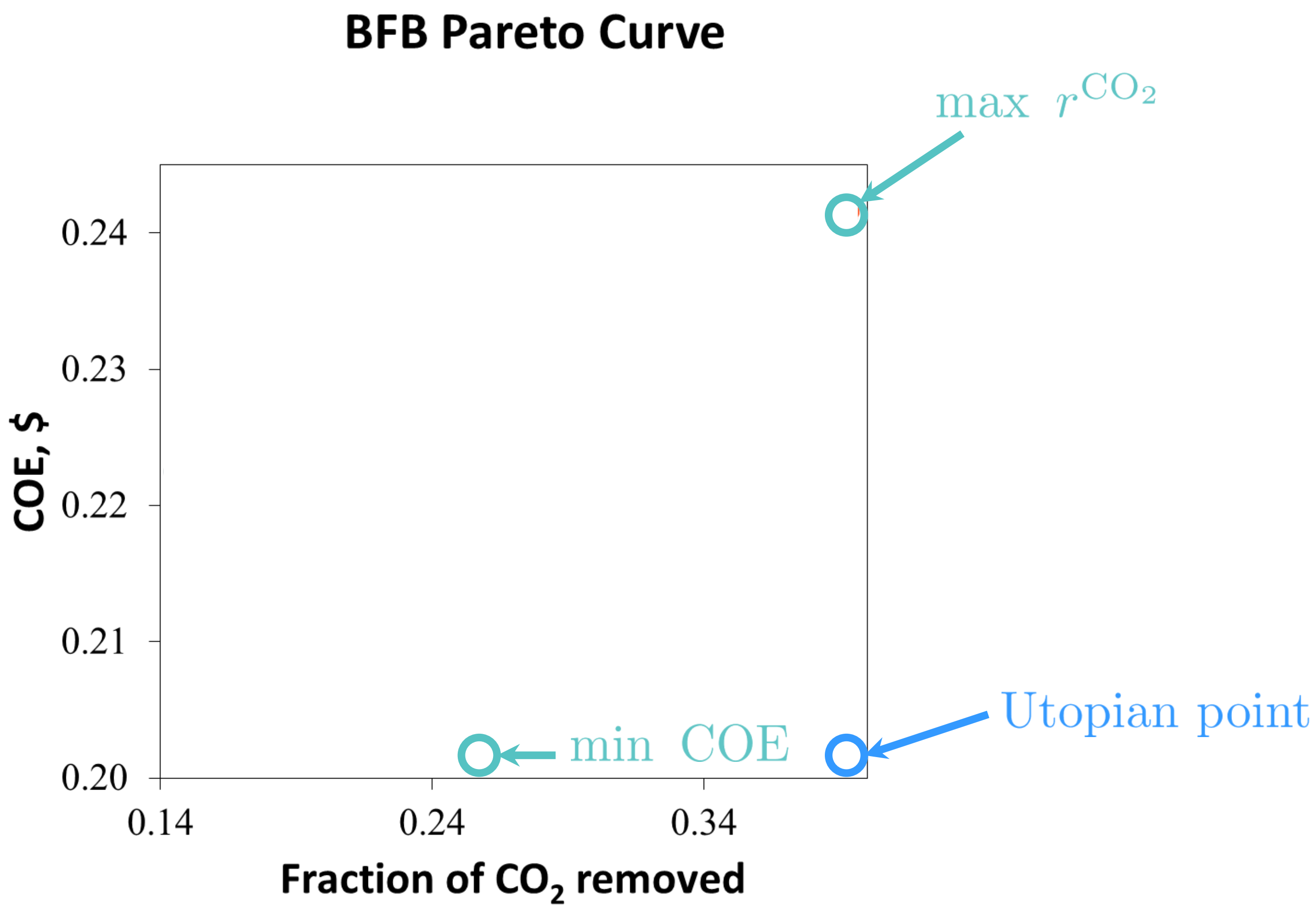
FINAL ITERATION – MODEL BUILD



13th term is not worth the added complexity

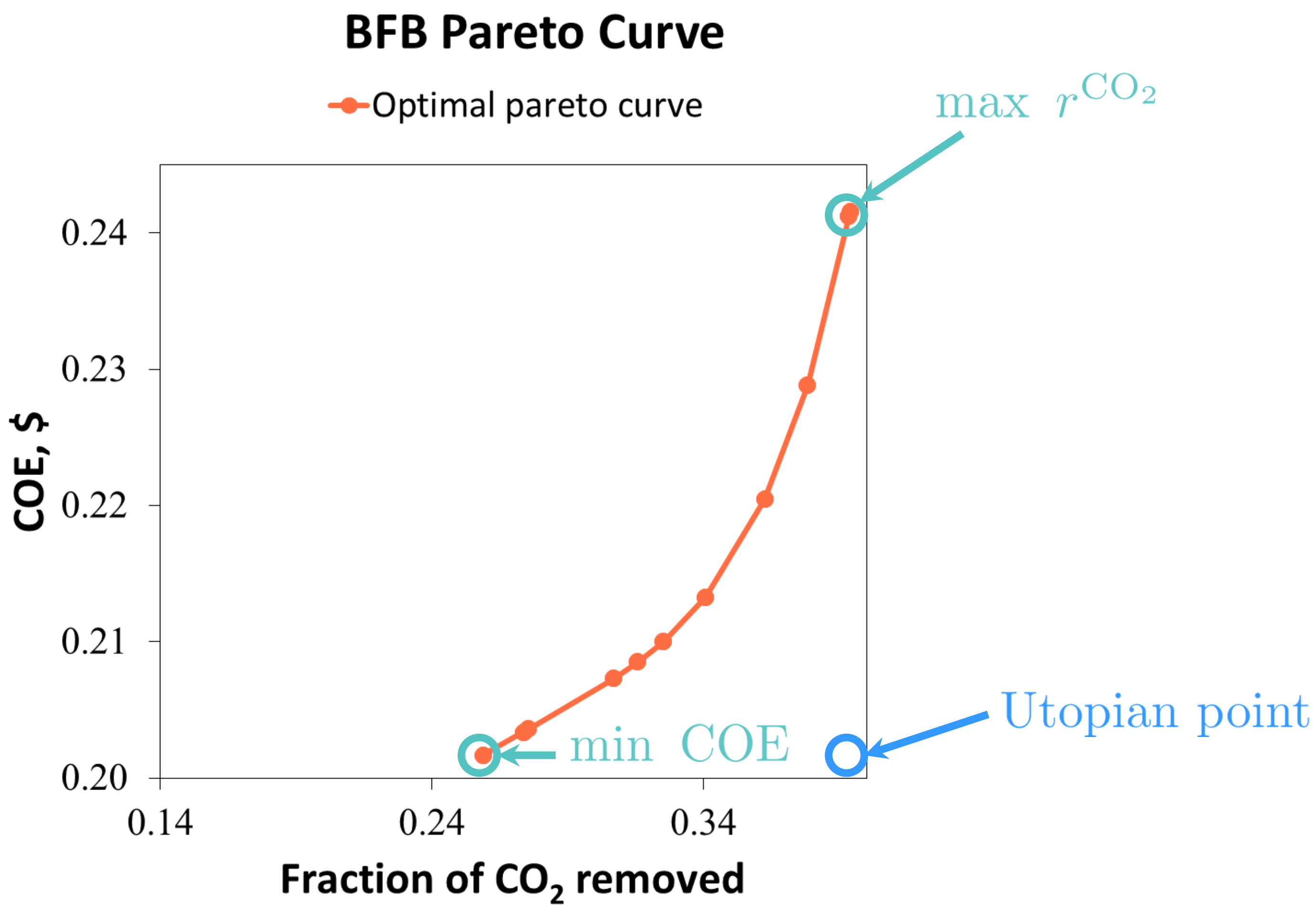
OPTIMAL PARETO CURVE

- **Once a set surrogate models are built, many optimization problem can be efficiently solved**



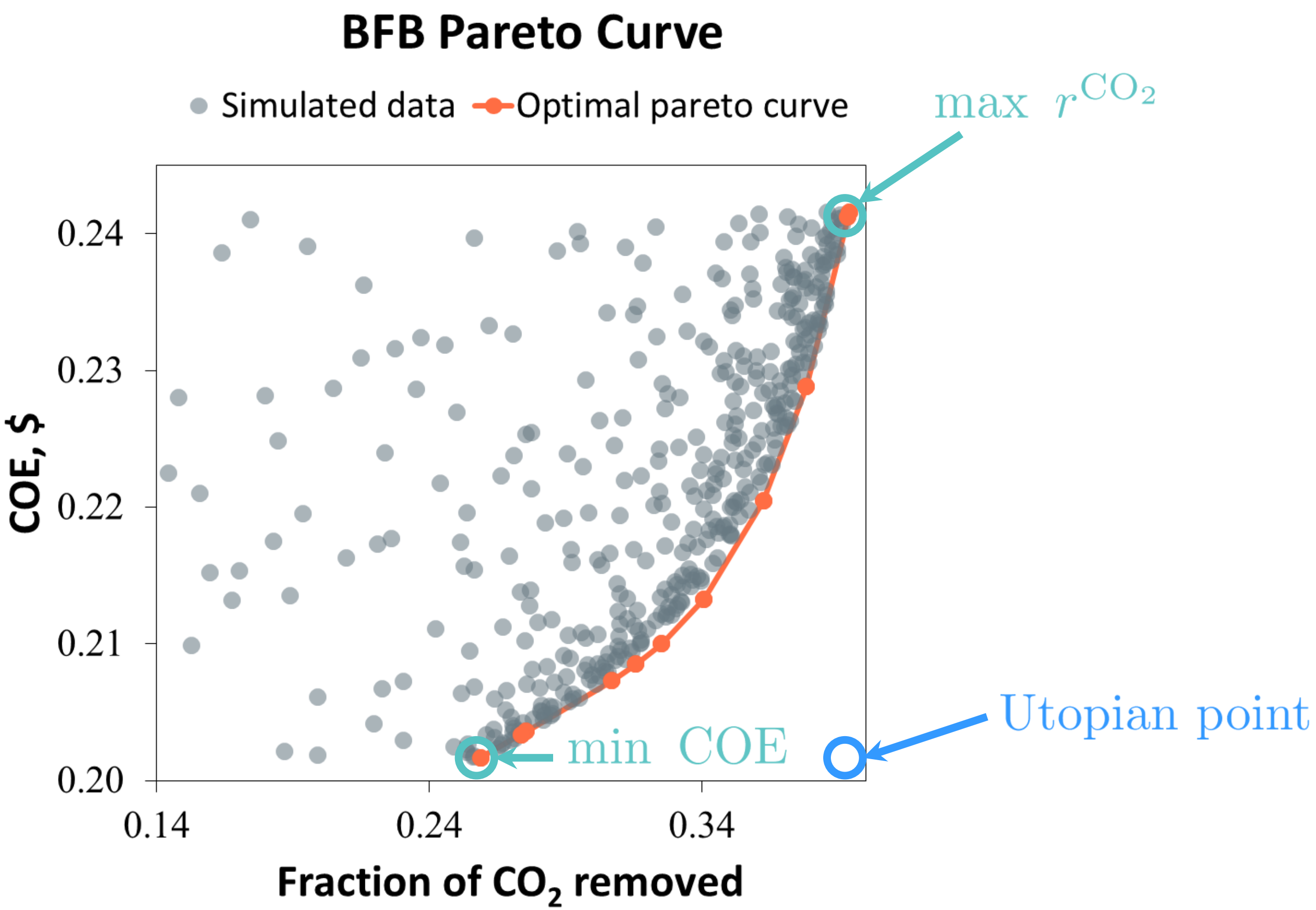
OPTIMAL PARETO CURVE

- **Once a set surrogate models are built, many optimization problem can be efficiently solved**



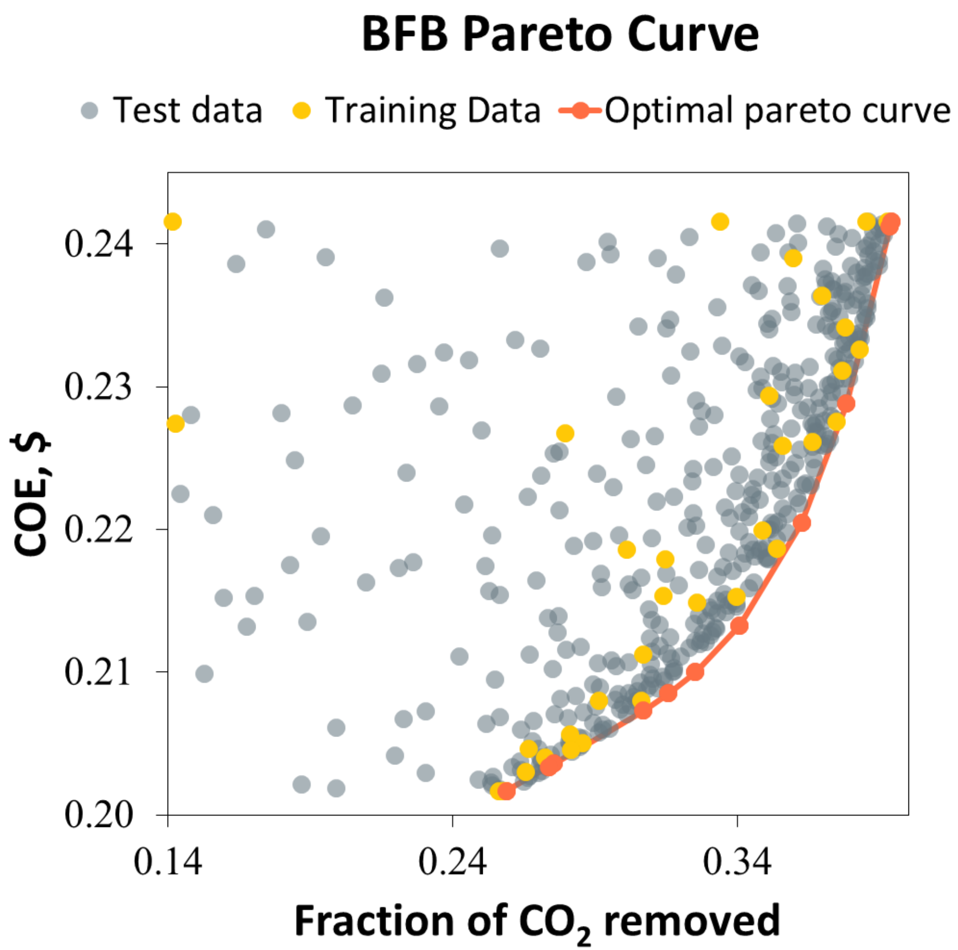
OPTIMAL PARETO CURVE

- **Once a set surrogate models are built, many optimization problem can be efficiently solved**



OPTIMAL PARETO CURVE

- **Once a set surrogate models are built, many optimization problem can be efficiently solved**

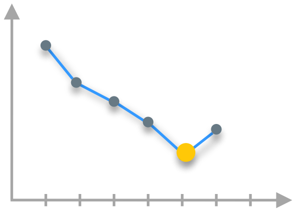


FINAL REMARKS

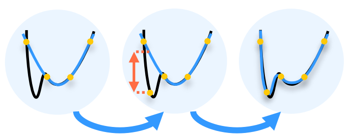
- **Expanding the scope of MINLPs**
  - Using low-complexity surrogate models to strike a balance between optimal decision-making and model fidelity

$$\begin{aligned} \min \quad & \hat{f}(x) \\ \text{s.t.} \quad & \hat{g}(x) \leq 0 \\ & x \in A \subset \mathbb{R}^n \end{aligned}$$

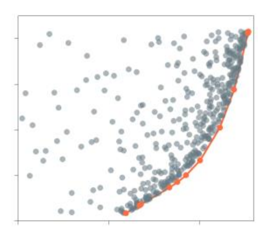
- **Surrogate model identification**
  - Simple, accurate model identification – MILP formulation



- **Error Maximization**
  - More information found per each simulated data point



- **Surrogates used to replace black-boxes**
  - Efficiently solve numerous and/or complex optimization problems



BEST SUBSET METHOD

- **Generalized best subset problem:**

$$\begin{aligned} \min_{\mathcal{S},\beta} \quad & \Phi(\mathcal{S},\beta) \\ \text{s.t.} \quad & \mathcal{S} \subseteq \mathcal{B} \end{aligned}$$

where $\Phi(\mathcal{S},\beta)$ is a goodness of fit measure for the subset of basis function, $\mathcal{S}$, and regression coefficients, β.

BEST SUBSET METHOD

- **Surrogate subset model:**

$$\hat{z}(x) = \sum_{j \in \mathcal{S}} \beta_j X_j(x)$$

- **Mixed-integer surrogate subset model:**

$$\hat{z}(x) = \sum_{j \in \mathcal{B}} (y_j \beta_j)\, X_j(x) \quad \text{such that} \quad \begin{array}{ll} y_j = 1 & j \in \mathcal{S} \\ y_j = 0 & j \notin \mathcal{S} \end{array}$$

- **Generalized best subset problem mixed-integer formulation:**

Very tough to solve

$$\begin{aligned} \min_{\beta, y} \quad & \Phi(\beta, y) \\ \text{s.t.} \quad & y_j = \{0, 1\} \end{aligned}$$

MIXED-INTEGER AICC

- **Corrected Akaike information criterion (AICc)** [Hurvich and Tsai, 93]

$$AICc(\mathcal{S},\beta) = N\log\left(\frac{1}{N}\sum_{i=1}^{N}\left(z_i - \sum_{j\in\mathcal{S}}\beta_j X_{ij}\right)^2\right) + 2|\mathcal{S}| + \frac{2|\mathcal{S}|\left(|\mathcal{S}|+1\right)}{N-|\mathcal{S}|-1}$$

- **Substituting the mixed integer surrogate form into AICc:**

$$AICc(\beta,y_j) = N\log\left(\frac{1}{N}\sum_{i=1}^{N}\left(z_i - \sum_{j\in\mathcal{B}}\left(y_j\beta_j\right)X_{ij}\right)^2\right) + 2\sum_j y_j + \frac{2\sum_j y_j\left(\sum_j y_j+1\right)}{N-\sum_j y_j-1}$$

$$\textbf{OR} \text{ if } \sum_j y_j = T$$

$$AICc(\beta,y_j) = N\log\left(\frac{1}{N}\sum_{i=1}^{N}\left(z_i - \sum_{j\in\mathcal{B}}\left(y_j\beta_j\right)X_{ij}\right)^2\right) + 2T + \frac{2T\left(T+1\right)}{N-T-1}$$

MIXED-INTEGER PROBLEM

$$
\begin{aligned}
\min_{\beta,T,y} \quad & AICc(\beta,T,y) = N \log\left(\frac{1}{N}\sum_{i=1}^{N}\left(z_i - \sum_{j\in\mathcal{B}}(y_j\beta_j)\,X_{ij}\right)^2\right) + 2T + \frac{2T\,(T+1)}{N-T-1} \\
\text{s.t.} \quad & \sum_{j\in\mathcal{B}} y_j = T \\
& y_j = \{0,1\} \qquad j \in \mathcal{B}
\end{aligned}
$$

MIXED-INTEGER PROBLEM

- **Further reformulation**
  - Replace bilinear terms with big-M constraints

$$y_j\beta_j \quad \longrightarrow \quad \beta_j^l y_j \leq \beta_j \leq \beta_j^u y_j$$

  - Decouple objective into two problems

$$\text{General:} \qquad \min_{\beta,T,y} \Phi(\beta,T,y) = \min_{T} \left\{ \overbrace{\underbrace{\min_{\beta,y} \left[\Phi_{\beta,y}(\beta,y)|_T\right]}_{\text{b) basis and coefficient selection}} + \Phi_T(T)}^{\text{a) model sizing}} \right\}$$

$$AICc(\beta,T): \qquad AICc_{\beta,y}(\beta,y)|_T = N \log\left(\frac{1}{N}\sum_{i=1}^{N}\left(z_i - \sum_{j\in\mathcal{B}} (y_j\beta_j)\, X_{ij}\right)^2 \right)$$

$$AICc_T(T) = 2T + \frac{2T\,(T+1)}{N-T-1}$$

  - Inner minimization objective reformulation

NESTED MIXED-INTEGER PROBLEM

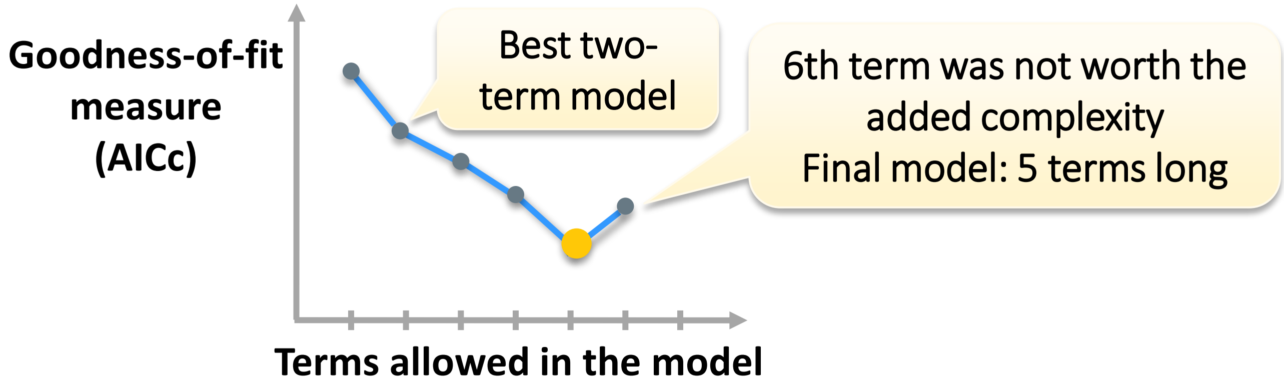
$$
\begin{aligned}
\min_{T \in \{1,\ldots,T^u\}} \quad & N \log \left(\frac{1}{N} \sum_{i=1}^{N} \left(z_i - \sum_{j \in \mathcal{B}} (y_j \beta_j)\, X_{ij} \right)^2 \right) + 2T + \frac{2T\,(T+1)}{N-T-1} \\
\text{s.t.} \quad & \min_{\beta, y} \quad \sum_{i=1}^{N} \left(z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right)^2 \\
& \text{s.t.} \quad \sum_{j \in \mathcal{B}} y_j = T \\
& \qquad \beta^l y_j \le \beta_j \le \beta^u y_j \qquad j \in \mathcal{B} \\
& \qquad y_j = \{0, 1\} \qquad j \in \mathcal{B}
\end{aligned}
$$

a) **Model sizing**

b) **Basis and coefficient selection**

PROBLEM SIMPLIFICATIONS

- **Outer problem**
  - The outer problem is parameterized by T and a local minima is found



- **Inner problem**
  - Stationarity condition used to solve for continuous variables

$$\frac{d}{d\beta_j}\sum_{i=1}^{N}\left(z_i - \sum_{j\in\mathcal{S}}\beta_j X_{ij}\right)^2 \propto \sum_{i=1}^{N} X_{ij}\left(z_i - \sum_{j\in\mathcal{S}}\beta_j X_{ij}\right) = 0, \quad j\in\mathcal{S}$$

  - Linear objective used to solved for integer variables

$$\text{Objective:} \quad \sum_{i=1}^{N}\left|z_i - \sum_{j\in\mathcal{S}}\beta_j X_{ij}\right|$$

FINAL BEST SUBSET MODEL

$$
\begin{aligned}
\min \quad & SE = \sum_{i=1}^{N} \left| z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right| \\
\text{s.t.} \quad & \sum_{j \in \mathcal{B}} y_j = T \\
& -U(1 - y_j) \leq \sum_{i=1}^{N} X_{ij} \left(z_i - \sum_{j \in \mathcal{B}} \beta_j X_{ij} \right) \leq U(1 - y_j) && j \in \mathcal{B} \\
& \beta^l y_j \leq \beta_j \leq \beta^u y_j && j \in \mathcal{B} \\
& y_j \in \{0, 1\} && j \in \mathcal{B} \\
& \beta_j \in \left[\beta_j^l, \beta_j^u\right] && j \in \mathcal{B}
\end{aligned}
$$

- **This model is solved for increasing values of *T* until the *AICc* worsens**