Optimization Using Surrogates for Engineering Design

LtCol Mark Abramson, AFIT

Charles Audet & Gilles Couture
École Polytechnique de Montréal

John Dennis, Rice University

Andrew Booker, Evin Cramer, Paul Frank,
Joerg Gablonsky, Martin Meckesheimer, Boeing Co.

Winter 2005

Background: Introduction

- Target problems
- The strawman surrogate algorithm
- Closed and Open Constraints

The surrogate management framework

Selected numerical results

Software and Conclusions

Our aim is to provide usable algorithms for expensive simulation-driven design. We use surrogates, but we solve the simulation-driven problem

Target Class of Problems

$$(NLP) \qquad \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega, \end{array}$$

where $f : \Re^n \to \Re \cup \{\infty\}$ may be discontinuous and:

Target Class of Problems

$$(NLP) \qquad \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega, \end{array}$$

where $f : \Re^n \rightarrow \Re \cup \{\infty\}$ may be discontinuous and:

- the functions are expensive black boxes - secs, mins, days, weeks

Target Class of Problems

$$(NLP) \qquad \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega, \end{array}$$

where $f : \Re^n \to \Re \cup \{\infty\}$ may be discontinuous and:

- the functions are expensive black boxes - secs, mins, days, weeks
- the functions provide few correct digits and may fail unexpectedly even for $x \in \Omega$

Target Class of Problems

$$(NLP) \qquad \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega, \end{array}$$

where $f : \Re^n \to \Re \cup \{\infty\}$ may be discontinuous and:

- the functions are expensive black boxes - secs, mins, days, weeks
- the functions provide few correct digits and may fail unexpectedly even for $x \in \Omega$
- the constraints are nonlinear, nonconvex, and may be yes/no

Target Class of Problems

$$(NLP) \qquad \begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega, \end{array}$$

where $f : \Re^n \to \Re \cup \{\infty\}$ may be discontinuous and:

- the functions are expensive black boxes - secs, mins, days, weeks
- the functions provide few correct digits and may fail unexpectedly even for $x \in \Omega$
- the constraints are nonlinear, nonconvex, and may be yes/no
- accurate approximation of derivatives is problematic

Relation of target problems to design

The design problem is:

$$\begin{aligned} \text{minimize} \quad & f_p(x) \\ \text{subject to} \quad & x \in X \cap \{x \in \Re^n : C_p(x) \leq 0\}, \end{aligned}$$

Relation of target problems to design

The design problem is:

$$\begin{aligned} \text{minimize} \quad & f_p(x) \\ \text{subject to} \quad & x \in X \cap \{x \in \Re^n : C_p(x) \leq 0\}, \end{aligned}$$

There are parameters and variables:

- Contextual parameters p, supposedly fixed, and

Relation of target problems to design

The design problem is:

$$\begin{aligned} \text{minimize} \quad & f_p(x) \\ \text{subject to} \quad & x \in X \cap \{x \in \Re^n : C_p(x) \leq 0\}, \end{aligned}$$

There are parameters and variables:

- Contextual parameters p, supposedly fixed, and
- Optimization or design or control variables x

Properties of the design problem

- p is fixed in the simulation, but it is subject to uncertainty because of incomplete knowledge (e.g. material properties) or because the product is used differently (e.g., the altimeter is off by 5%)

Properties of the design problem

- p is fixed in the simulation, but it is subject to uncertainty because of incomplete knowledge (e.g. material properties) or because the product is used differently (e.g., the altimeter is off by 5%)
- $x \in X$ must be satisfied for the underlying simulations for f_p, C_p to be called.

Properties of the design problem

- p is fixed in the simulation, but it is subject to uncertainty because of incomplete knowledge (e.g. material properties) or because the product is used differently (e.g., the altimeter is off by 5%)
- $x \in X$ must be satisfied for the underlying simulations for f_p, C_p to be called.
- $C_p(x) \leq 0$ only has to hold at the solution

Properties of variables and constraints

- Function evaluations typically involve numerically linking legacy PDE solvers.

Properties of variables and constraints

- Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly

Properties of variables and constraints

- Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly
- There are few correct digits.

Properties of variables and constraints

- Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly
- There are few correct digits. Accurate approximation of derivatives is problematic

Properties of variables and constraints

- Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly
- There are few correct digits. Accurate approximation of derivatives is problematic
- Some variables are categorical, i.e., the simulations only run with certain discrete choices, e.g., choice and sequencing of manufacturing processes. We rigorously treat categorical optimization variables in our algorithms

Uncertainty in design variables

Instead of specified design x_*, the delivered product might be the design specified by a nearby x,

Uncertainty in design variables

Instead of specified design x_*, the delivered product might be the design specified by a nearby x, and so every design near the one we find must be acceptable

Uncertainty in design variables

Instead of specified design x_*, the delivered product might be the design specified by a nearby x, and so every design near the one we find must be acceptable

This is a very common difficulty, but it is tough to get good data on the distribution

Uncertainty in design variables

Instead of specified design x_*, the delivered product might be the design specified by a nearby x, and so every design near the one we find must be acceptable

This is a very common difficulty, but it is tough to get good data on the distribution

Robustify against such "variability" by giving up some optimality at a given point in return for optimality over nearby designs

Uncertainty in context variables

Products are designed for optimal performance under specified operating conditions, loads, etc

Uncertainty in context variables

Products are designed for optimal performance under specified operating conditions, loads, etc

We would be willing to give up some performance at the exact conditions p in return for optimal performance over all likely nearby conditions q - or we can constrain probability of failure for the nominal p

Seems to be "chance constrained" stochastic programming - nothing new

Hubris or Sanguinity?

Both types of uncertainty can be modeled for optimization by incorporation into f, C

Hubris or Sanguinity?

Both types of uncertainty can be modeled for optimization by incorporation into f, C

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x) \\ \text{subject to} \quad & x \in \hat{X} \cap \{x \in \Re^n : \hat{C}(x) \leq 0\}, \end{aligned}$$

where $\hat{f}(x), \hat{c}_j(x)$ incorporate uncertainty modeled by the user's scheme of choice

Hubris or Sanguinity?

Both types of uncertainty can be modeled for optimization by incorporation into f, C

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x) \\ \text{subject to} \quad & x \in \hat{X} \cap \{x \in \Re^n : \hat{C}(x) \leq 0\}, \end{aligned}$$

where $\hat{f}(x), \hat{c}_j(x)$ incorporate uncertainty modeled by the user's scheme of choice

Still belongs to target class, Bring 'em on

Hubris or Sanguinity?

Both types of uncertainty can be modeled for optimization by incorporation into f, C

$$\begin{aligned} \text{minimize} \quad & \hat{f}(x) \\ \text{subject to} \quad & x \in \hat{X} \cap \{x \in \Re^n : \hat{C}(x) \leq 0\}, \end{aligned}$$

where $\hat{f}(x), \hat{c}_j(x)$ incorporate uncertainty modeled by the user's scheme of choice

Still belongs to target class, Bring 'em on for now, but seek tighter coupling

A confidence builder

Meckesheimer, Booker, and Torng report very good performance of our approach on some structural engineering problems in dealing with probability of failure by constraints or as the objective

Appeared in OTPE special issue: "Reliability based design optimization using Design Explorer" (Design Explorer is Boeing's implementation of the surrogate management framework - SMF)

A confidence builder

Meckesheimer, Booker, and Torng report very good performance of our approach on some structural engineering problems in dealing with probability of failure by constraints or as the objective

Appeared in OTPE special issue: "Reliability based design optimization using Design Explorer" (Design Explorer is Boeing's implementation of the surrogate management framework - SMF)

From this point on, assume the formulation incorporates uncertainty

A strawman surrogate approach

1. Choose surrogates s_f and S_C based on either:
   1.1 simplified physical models; or
   1.2 surfaces obtained from f, C values at selected sites

A strawman surrogate approach

1. Choose surrogates s_f and S_C based on either:
   1.1 simplified physical models; or
   1.2 surfaces obtained from f, C values at selected sites
2. Minimize $s_f(x)$ for $S_C(x) \leq 0$ to obtain x_s.
   Every user has their favorite approach for this part

A strawman surrogate approach

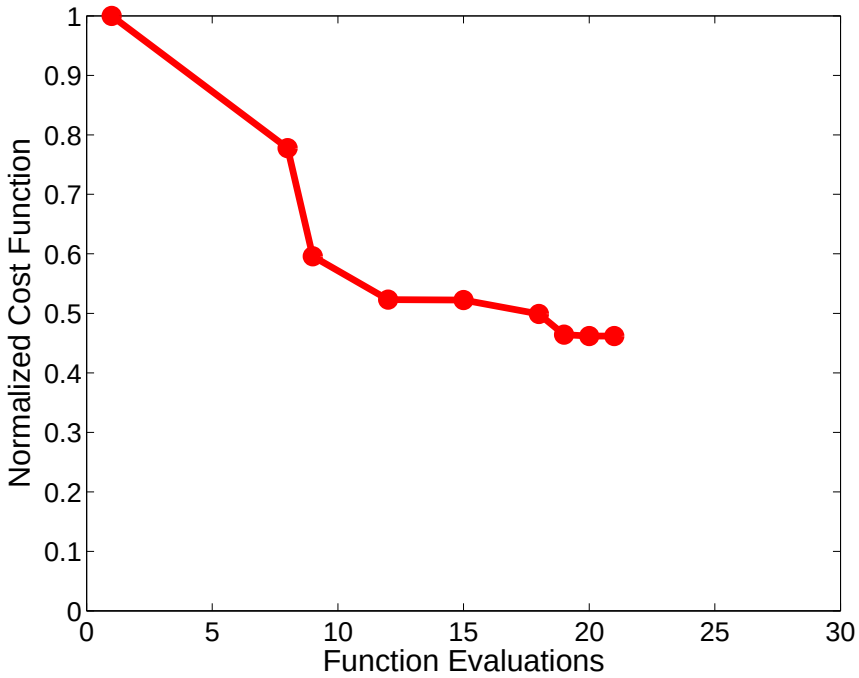
1. Choose surrogates s_f and S_C based on either:
   1.1 simplified physical models; or
   1.2 surfaces obtained from f, C values at selected sites
2. Minimize $s_f(x)$ for $S_C(x) \leq 0$ to obtain x_s.
   Every user has their favorite approach for this part
3. Compute $f(x_s), C(x_s)$ to determine if improvement has been made over the best x found to date

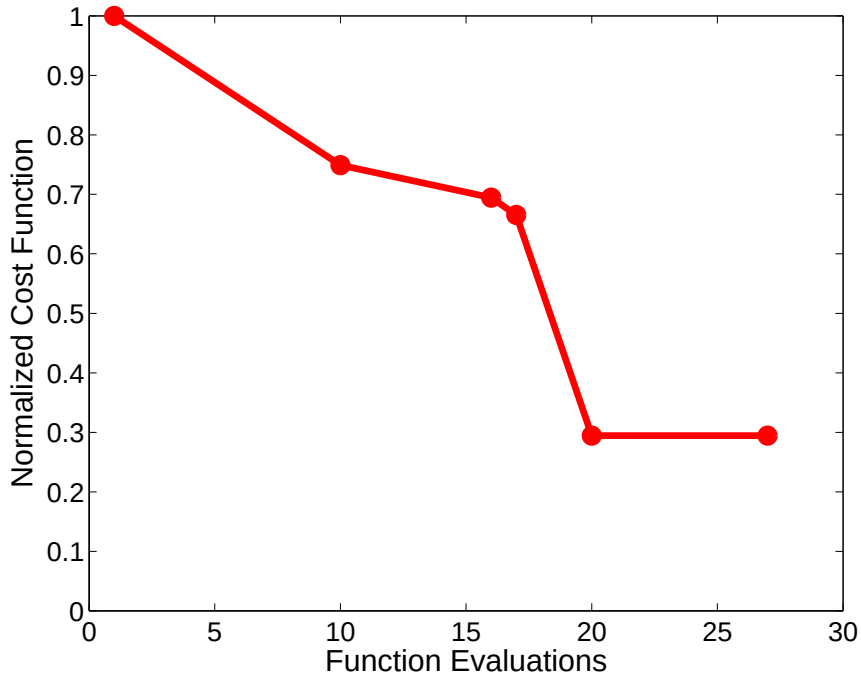
A strawman surrogate approach

1. Choose surrogates s_f and S_C based on either:
   1.1 simplified physical models; or
   1.2 surfaces obtained from f, C values at selected sites
2. Minimize $s_f(x)$ for $S_C(x) \leq 0$ to obtain x_s.
   Every user has their favorite approach for this part
3. Compute $f(x_s), C(x_s)$ to determine if improvement has been made over the best x found to date
   Stalls when surrogate stagnates

Trailing edge design for unsteady flow

Optimize shape for total acoustic noise reduction. Work of Stanford fluids grad student Alison Marsden, strawman left and SMF right





Closed constraints - a barrier approach

To enforce X constraints, replace f by a barrier objective

$$f_X(x) := \begin{cases} f(x) & \text{if } x \in X, \\ +\infty & \text{otherwise.} \end{cases}$$

Then apply the unconstrained algorithm to f_X.

Closed constraints - a barrier approach

To enforce X constraints, replace f by a barrier objective

$$f_X(x) := \begin{cases} f(x) & \text{if } x \in X, \\ +\infty & \text{otherwise.} \end{cases}$$

Then apply the unconstrained algorithm to f_X.

The quality of the solution found by the algorithm depends the local smoothness of f, not of f_X.

Closed constraints - a barrier approach

To enforce X constraints, replace f by a barrier objective

$$f_X(x) := \begin{cases} f(x) & \text{if } x \in X, \\ +\infty & \text{otherwise.} \end{cases}$$

Then apply the unconstrained algorithm to f_X.

The quality of the solution found by the algorithm depends the local smoothness of f, not of f_X.

Our theory is powerful for this approach,

Closed constraints - a barrier approach

To enforce X constraints, replace f by a barrier objective

$$f_X(x) := \begin{cases} f(x) & \text{if } x \in X, \\ +\infty & \text{otherwise.} \end{cases}$$

Then apply the unconstrained algorithm to f_X.

The quality of the solution found by the algorithm depends the local smoothness of f, not of f_X.

Our theory is powerful for this approach, which saves LES runs in Alison Marsden's application to trailing edge design in fully turbulent flow.

Open constraints

Open constraints need not be satisfied at every iteration

Open constraints must be satisfied at convergence

Open constraints

Open constraints need not be satisfied at every iteration
Open constraints must be satisfied at convergence

Let $h(x)$ be the aggregate open constraint violation at x. We use:

$$h(x) = \sum_j \max(0, c_j(x))^2$$

h inherits smoothness from C

Open constraints

Open constraints need not be satisfied at every iteration
Open constraints must be satisfied at convergence

Let $h(x)$ be the aggregate open constraint violation at x. We use:

$$h(x) = \sum_j \max(0, c_j(x))^2$$

h inherits smoothness from C

Note that $h_X(x) = 0$ iff x is feasible for both open and closed constraints

Ways to handle open constraints

- *Choose ρ large enough* and minimize the augmented Lagrangian $f_X(x) + \Lambda C(x) + \rho h_X(x)$

Ways to handle open constraints

- *Choose ρ large enough* and minimize the augmented Lagrangian $f_X(x) + \Lambda C(x) + \rho h_X(x)$
- *Penalty methods may require ρ to become very large and robustness is an issue. Good estimates of Λ require derivatives*

Ways to handle open constraints

- *Choose ρ large enough* and minimize the augmented Lagrangian $f_X(x) + \Lambda C(x) + \rho h_X(x)$
- *Penalty methods may require ρ to become very large and robustness is an issue. Good estimates of Λ require derivatives*
- Use a filter method and forget about ρ and Λ

Ways to handle open constraints

- *Choose ρ large enough* and minimize the augmented Lagrangian $f_X(x) + \Lambda C(x) + \rho h_X(x)$
- *Penalty methods may require ρ to become very large and robustness is an issue. Good estimates of Λ require derivatives*
- Use a filter method and forget about ρ and Λ

We use the filter. (originally idea from Fletcher & Leyffer)

We admit that our filter theory is currently weaker than for other approaches.

Ways to handle open constraints

- *Choose ρ large enough* and minimize the augmented Lagrangian $f_X(x) + \Lambda C(x) + \rho h_X(x)$
- *Penalty methods may require ρ to become very large and robustness is an issue. Good estimates of Λ require derivatives*
- Use a filter method and forget about ρ and Λ

We use the filter. (originally idea from Fletcher & Leyffer)

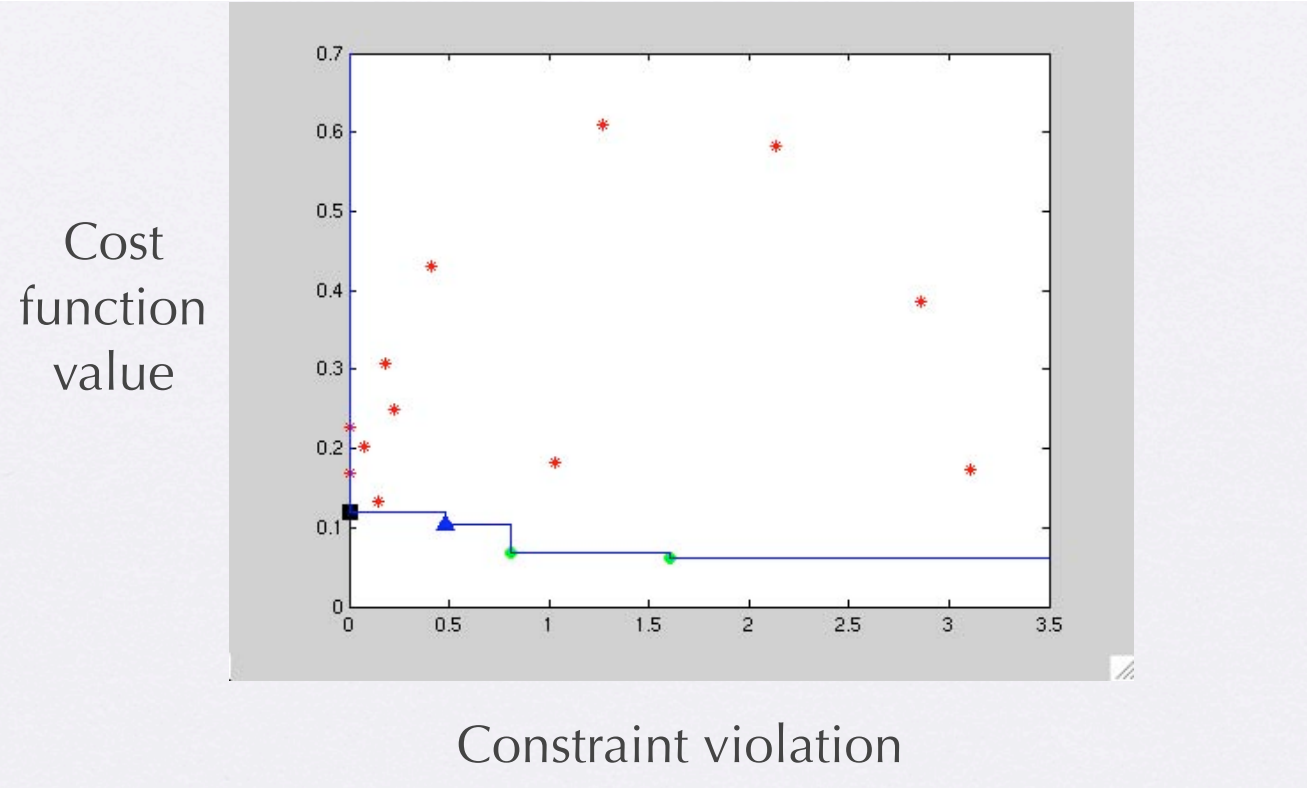
We admit that our filter theory is currently weaker than for other approaches. Charles Audet and I are working on it.

Final filter for RANS trailing edge design

A trial point x is *unfiltered* if $h_X(x) < h_{max}$ and no earlier point is at least as optimal and as feasible

SUCCESSFUL iterations find unfiltered points

UNSUCCESSFUL iterations don't



The SMF pseudo code

Given initial surrogates s_f, s_C and $p_0 \in M_0$, a mesh on X, let $P_0 \subset M_0$ be p_0 and the points of M_0 adjacent to x_0

The SMF pseudo code

Given initial surrogates s_f, s_C and $p_0 \in M_0$, a mesh on X, let $P_0 \subset M_0$ be p_0 and the points of M_0 adjacent to x_0
For $k = 0, 1, \cdots,$ do

The SMF pseudo code

Given initial surrogates s_f, s_C and $p_0 \in M_0$, a mesh on X,
let $P_0 \subset M_0$ be p_0 and the points of M_0 adjacent to x_0
For $k = 0, 1, \cdots,$ do

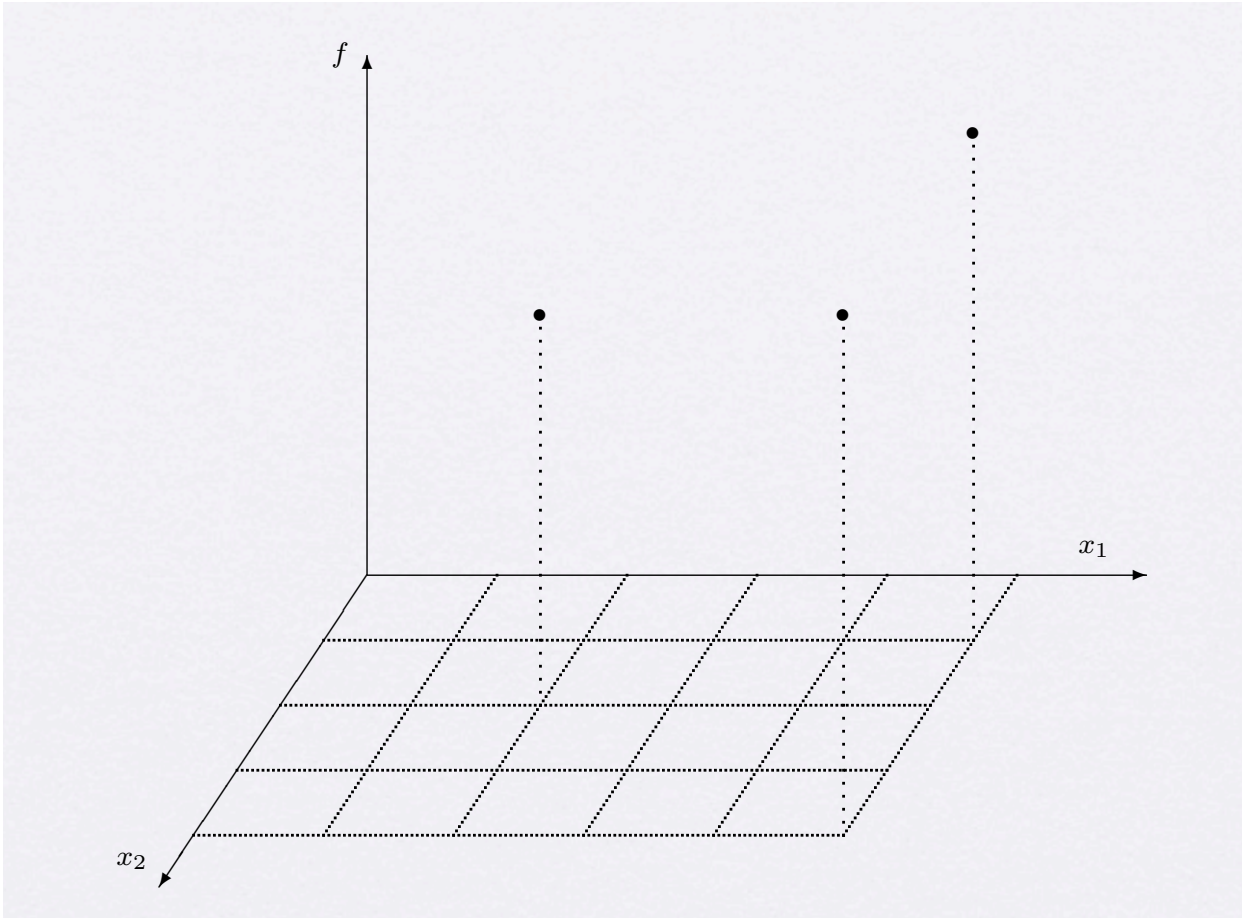
1. (Strawman)Search on s_f, s_C to find an unfiltered $x_{k+1} \in M_k$.
   If found, then set $M_{k+1} = M_k$ and update the surrogates;

The SMF pseudo code

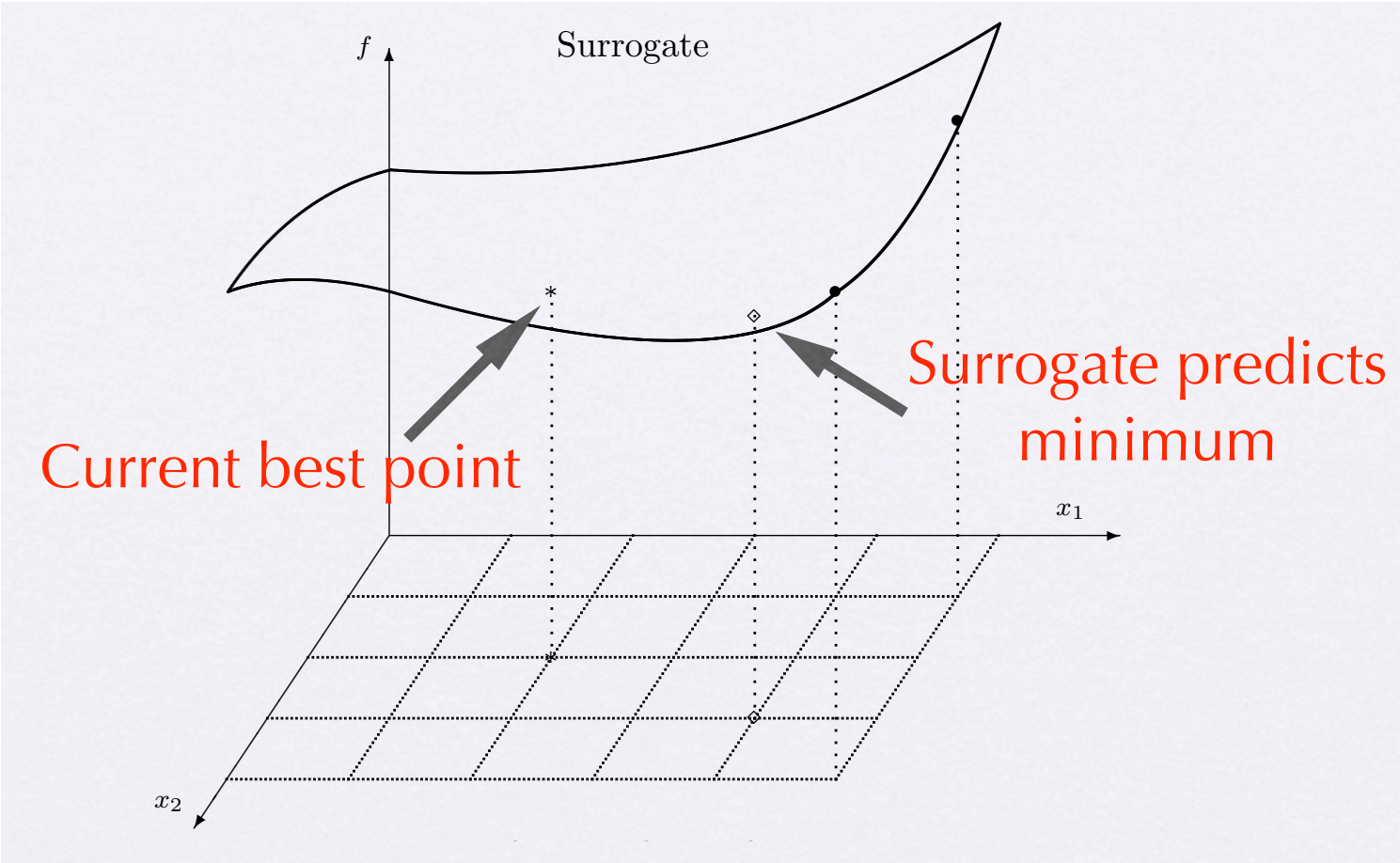
Given initial surrogates s_f, s_C and $p_0 \in M_0$, a mesh on X,
let $P_0 \subset M_0$ be p_0 and the points of M_0 adjacent to x_0
For $k = 0, 1, \cdots,$ do

1. (Strawman)Search on s_f, s_C to find an unfiltered $x_{k+1} \in M_k$.
   If found, then set $M_{k+1} = M_k$ and update the surrogates;
2. Else if p_k is the only unfiltered point in P_k; Then set
   $x_{k+1} = x_k$ and $M_{k+1} = M_k/4$ and update the surrogates;
   Else return to 1

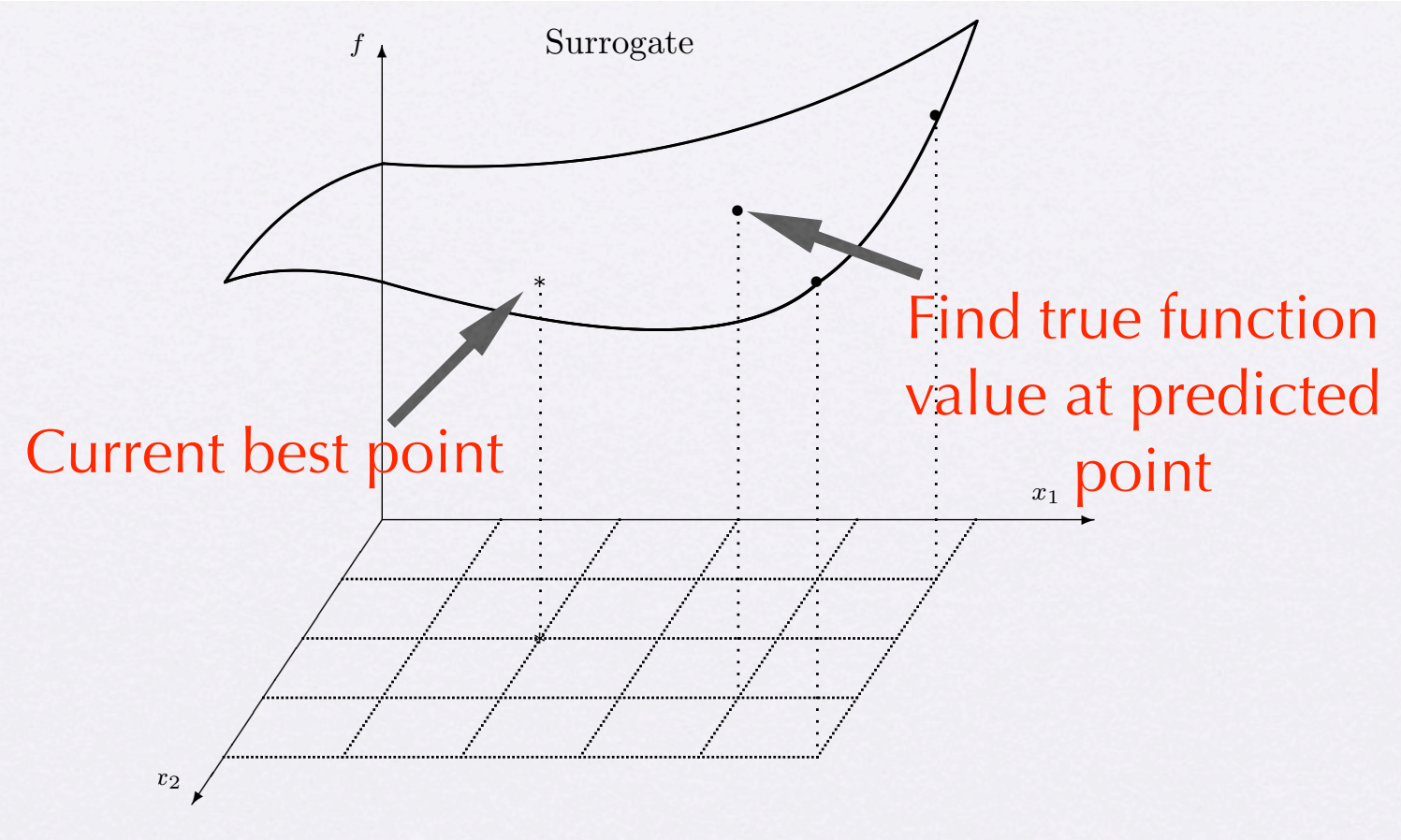
Gather surrogate training set



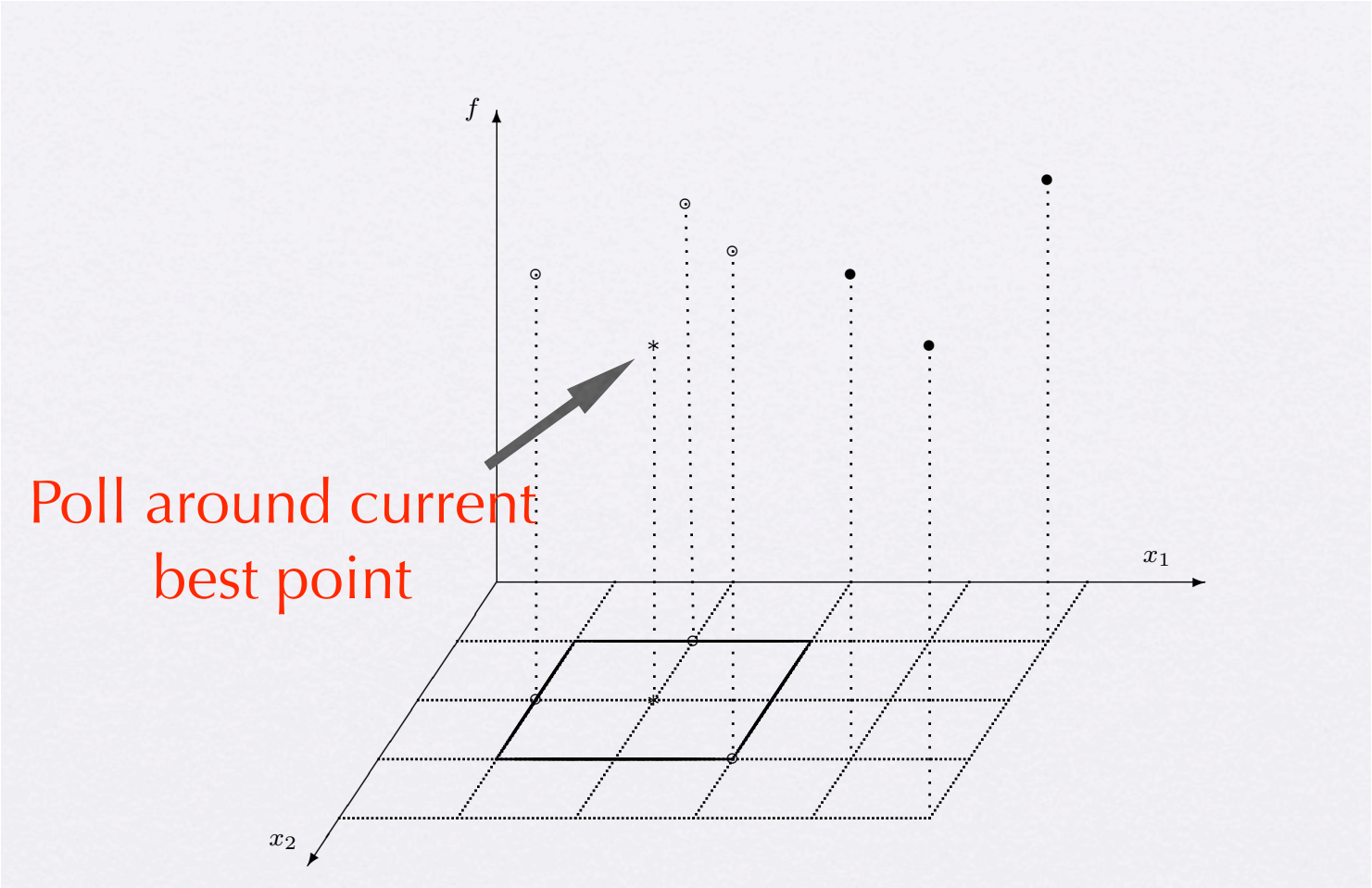
Use surrogate to predict better design



Check for improvement



If surrogate fails, then POLL



The POLL step

- The POLL step is the key to the proof that the SMF converges to a minimizer of the true expensive function.

The POLL step

- The POLL step is the key to the proof that the SMF converges to a minimizer of the true expensive function.
- The POLL step consists of evaluating the actual function on on neighboring mesh points in a positive spanning set of directions until an unfiltered design is found.

The POLL step

- The POLL step is the key to the proof that the SMF converges to a minimizer of the true expensive function.
- The POLL step consists of evaluating the actual function on on neighboring mesh points in a positive spanning set of directions until an unfiltered design is found.
- If no unfiltered point is found in the POLL, the mesh is refined and we return to use the surrogate on the finer mesh.

The POLL step

- The POLL step is the key to the proof that the SMF converges to a minimizer of the true expensive function.
- The POLL step consists of evaluating the actual function on on neighboring mesh points in a positive spanning set of directions until an unfiltered design is found.
- If no unfiltered point is found in the POLL, the mesh is refined and we return to use the surrogate on the finer mesh.

There are NO assumptions on the surrogate, but better surrogates provide better predictions and hence fewer expensive function calls.

How good is the SMF?

It finds designs overnight as good or much better than designs that took months to find the old way - if they could be found

How good is the SMF?

It finds designs overnight as good or much better than designs that took months to find the old way - if they could be found

Is the SMF that good?

How good is the SMF?

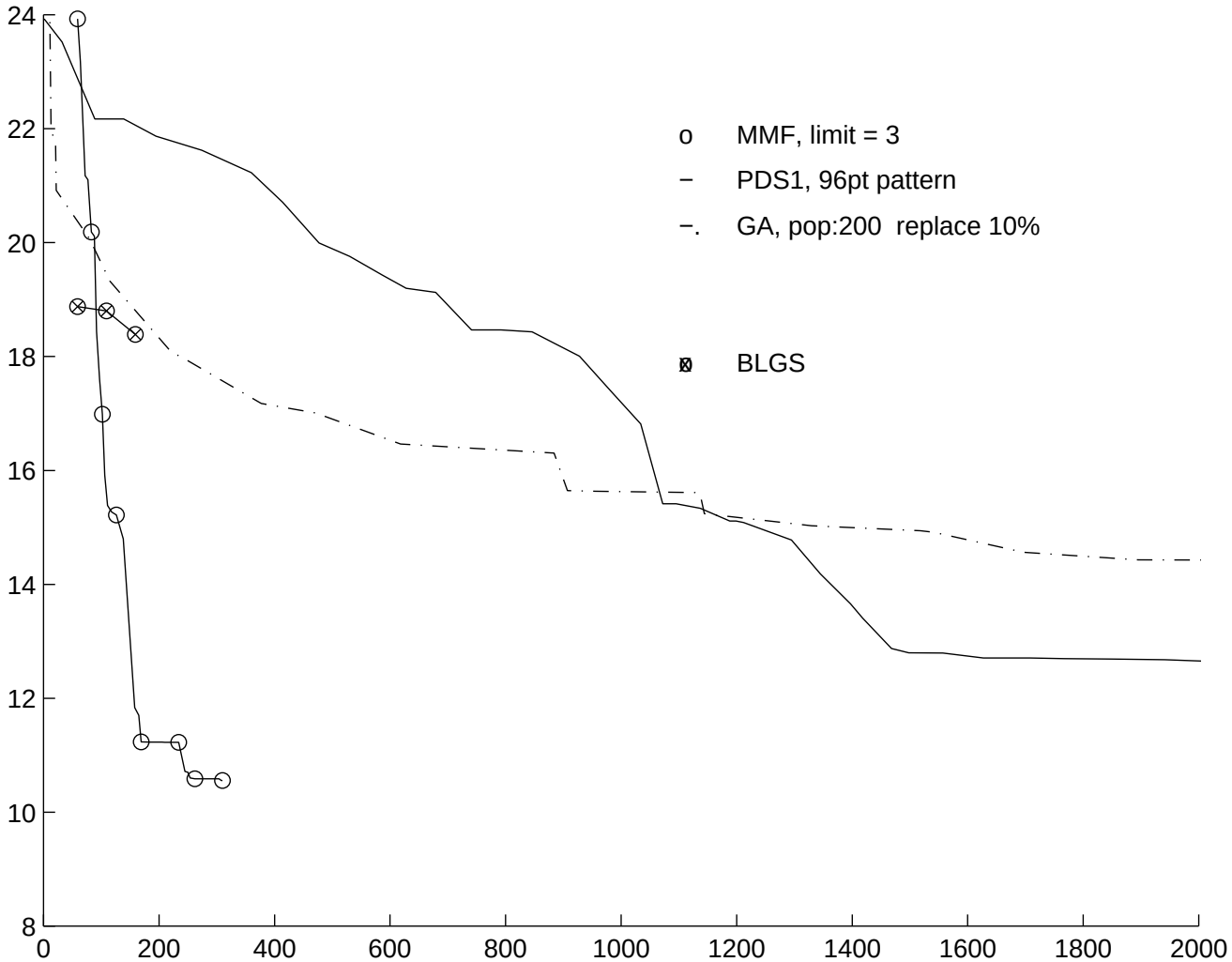
It finds designs overnight as good or much better than designs that took months to find the old way - if they could be found

Is the SMF that good?

Presently it is the only realistic possibility for really expensive problems

Here are some results:

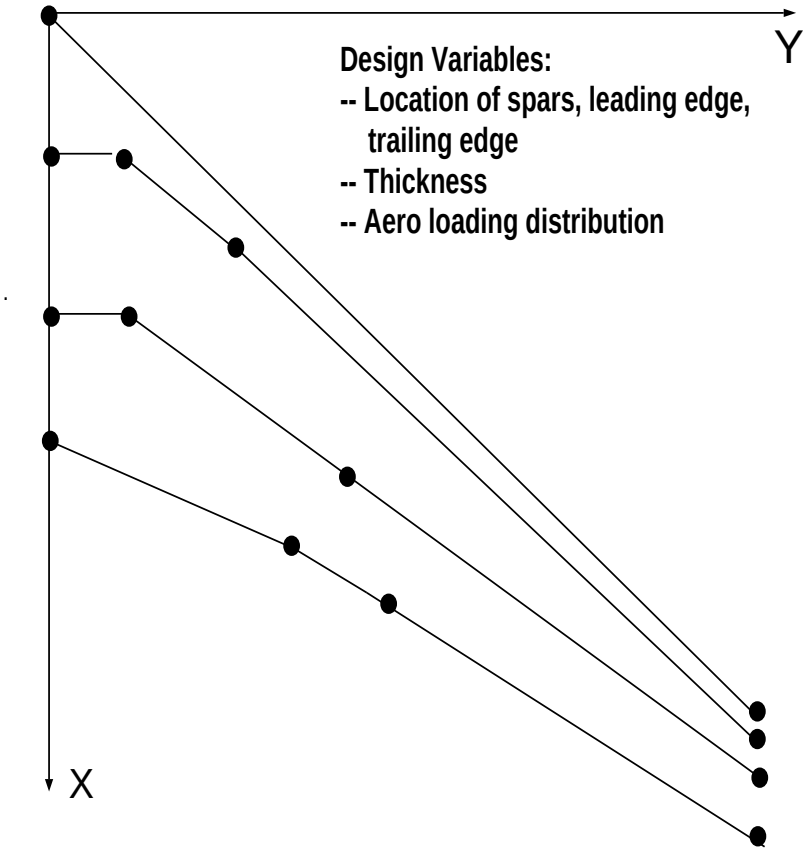
Comparisons on 31d helicopter example



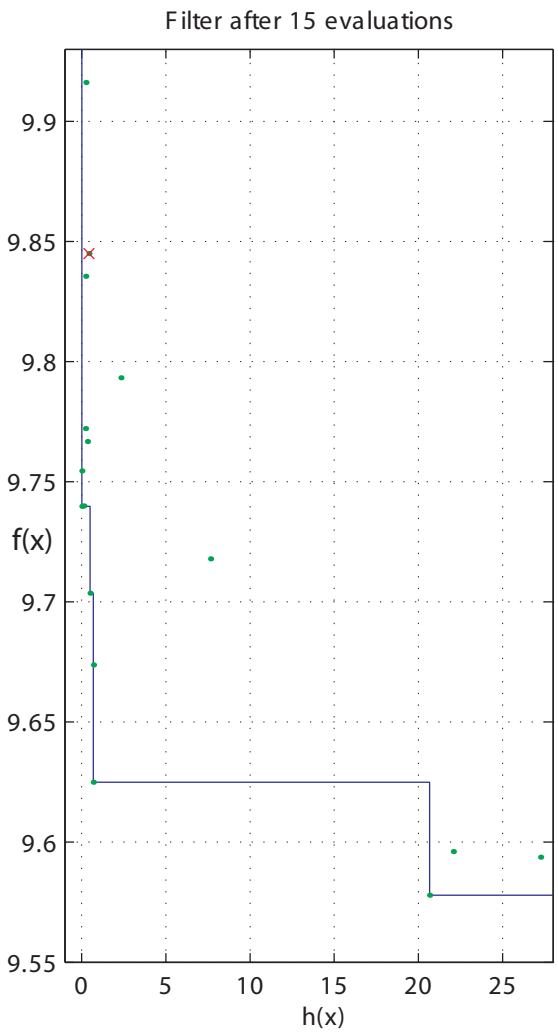
Boeing wing planform design - infeasible baseline

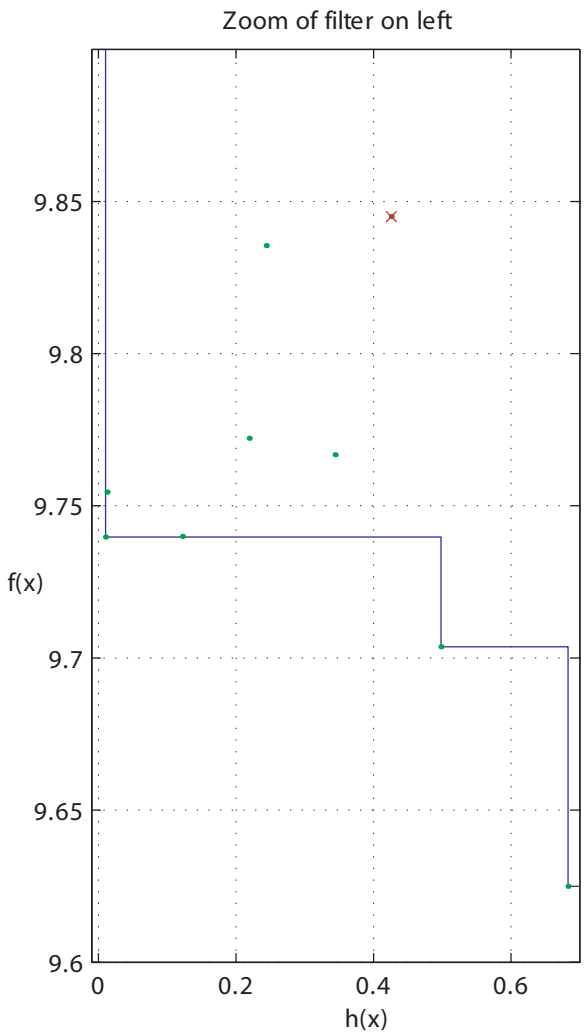
Infeasible baseline design

	n	# of ctrs	# of fevals
A	15	11	304
B	15	11	292

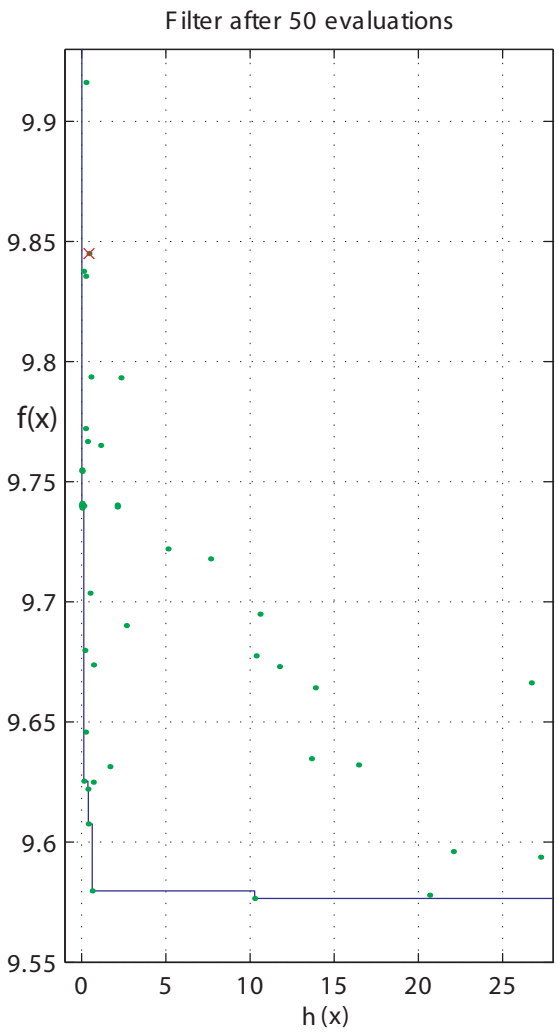


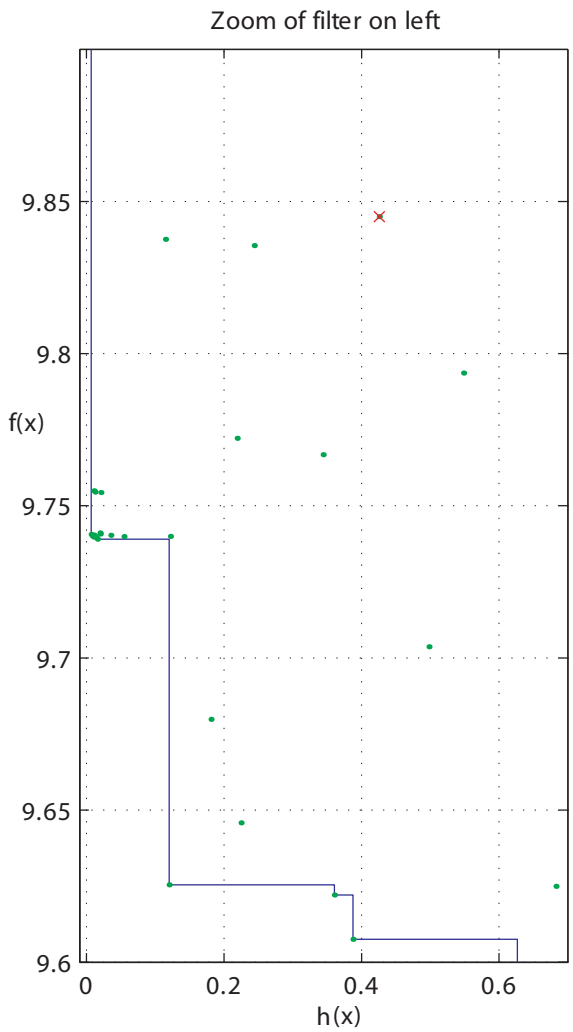
Boeing SonicCruiser planform initial filter



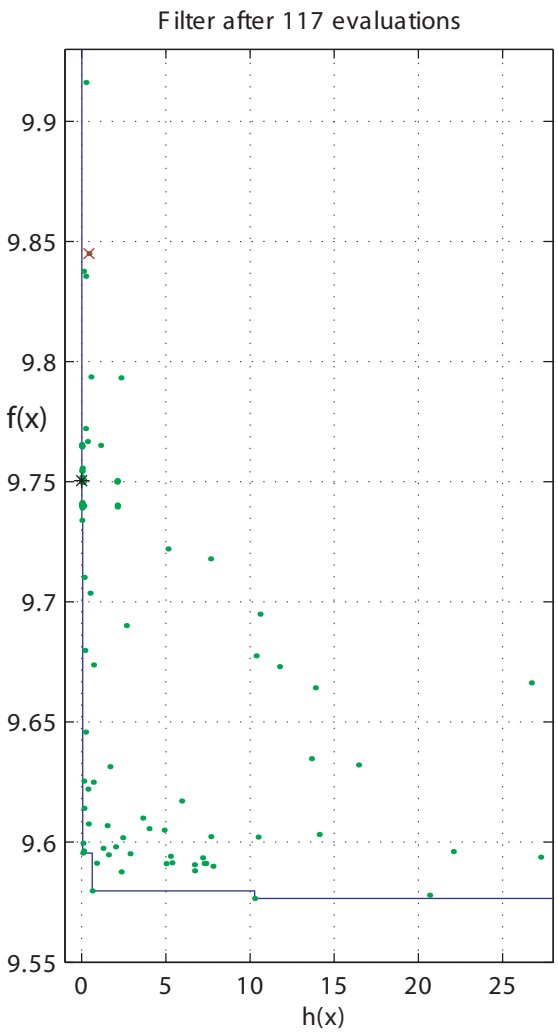


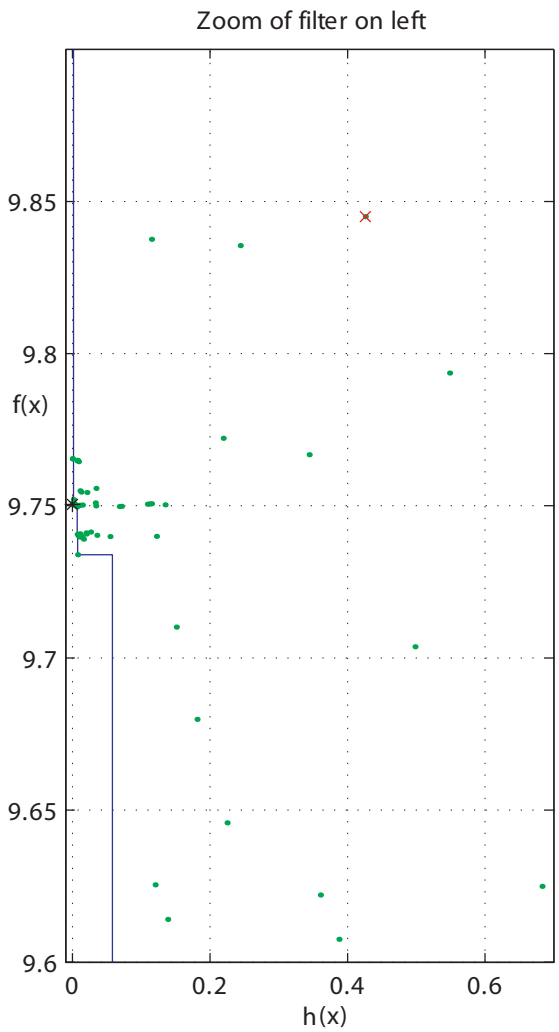
A planform filter after 50 evaluations



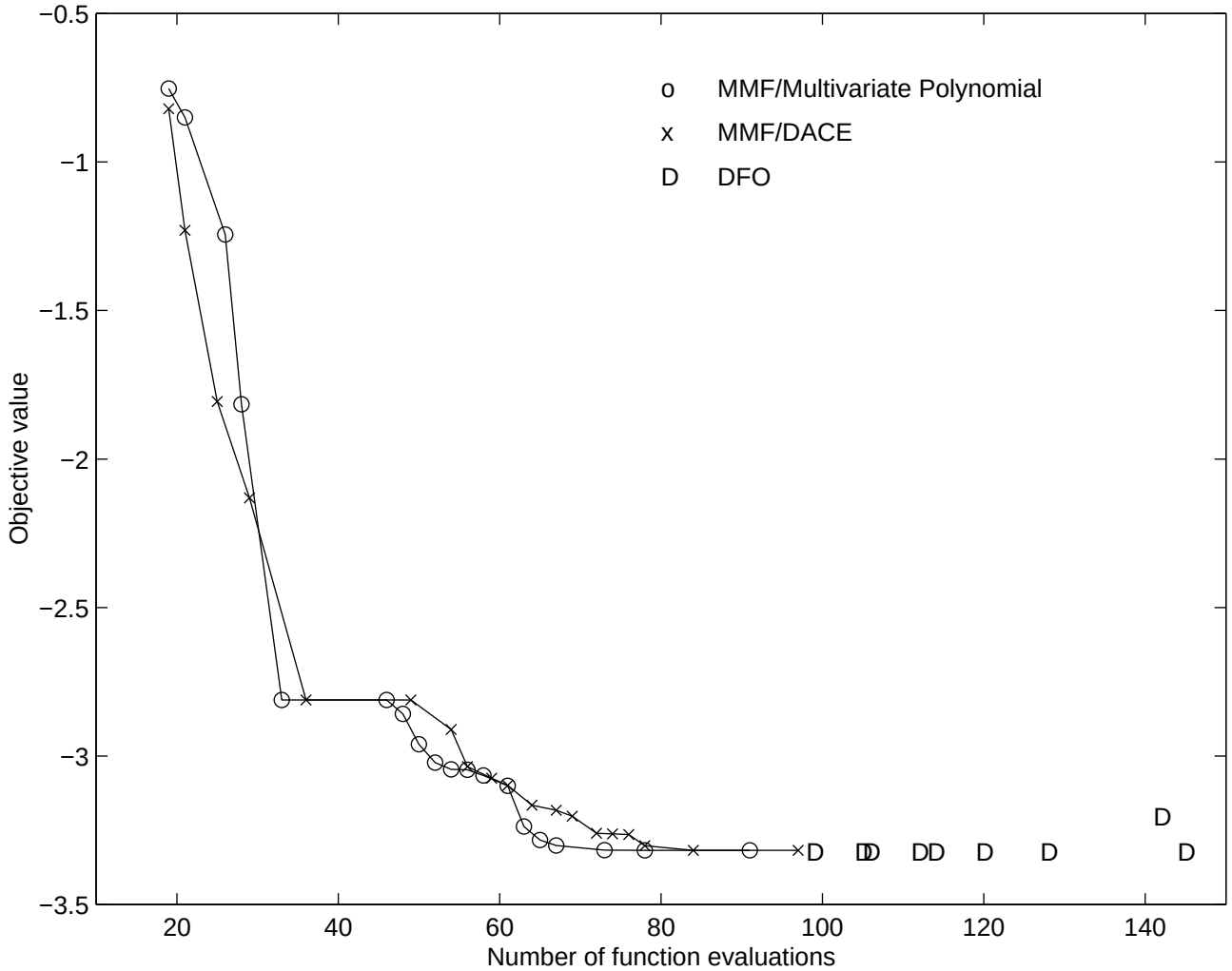


Feasibility after 117 evaluations

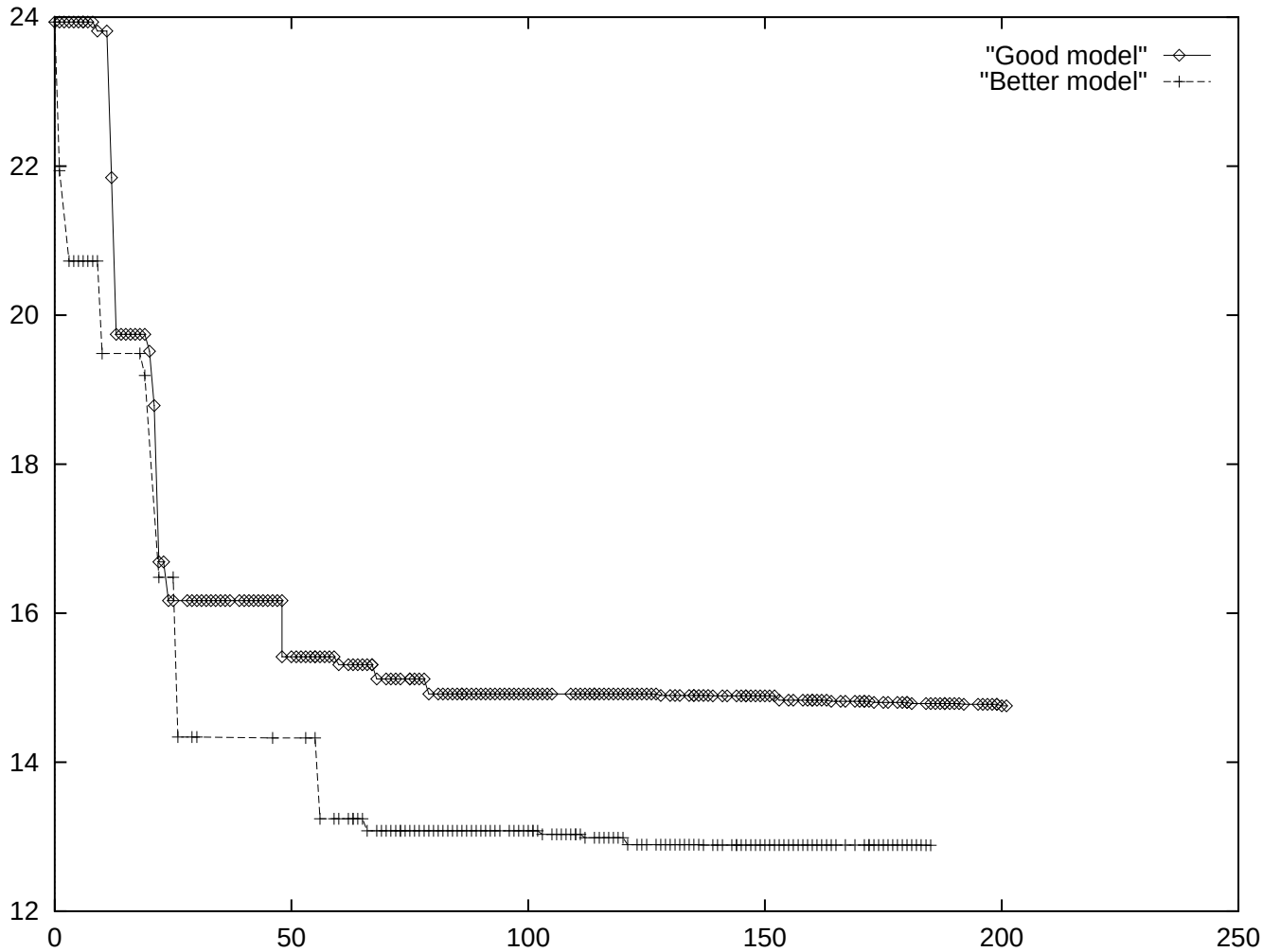




Polynomial vs DACE surrogates



Recovering from a bad surrogate



Implementations

*=freely available. Red=under active development.

- NOMAD C++*: MADS with barrier and filter
  - in use in company-wide tool at ExxonMobil
- NOMADm*: Our MatLab version with everything
  - in use at Siemens
- Design Explorer: Boeing's implementation available to nonBoeing users from Phoenix Integration
  - includes DACE surrogates, used on the Boeing 777 and in use on the 7e7
- Alison Marsden's Matlab code Used for trailing edge design
  - she has completed one iteration with her SMF implementation on LES turbulent flow trailing edge design with a 54% reduction in noise - 3 weeks per simulation

Conclusions and Claims

- The SMF is a completely rigorous tool for engineering design governed by expensive simulations. It solves the expensive problem - not the surrogate problem
- The SMF is based on our derivative-free MADS algorithms
- The quality of the surrogate as an approximation is not an assumption, but a good surrogate saves expensive simulations
- The ANOVA implementation in Boeing's Design Explorer suggests which are the important design variables
- The work to find an approximation based on a large training set is not justified for its use in surrogate optimization, but it may be justified if the approximation has other purposes - Phillips Research NL does this