Multidisciplinary System Design Optimization (MSDO)

Approximation Methods

Lecture 19

16 April 2004

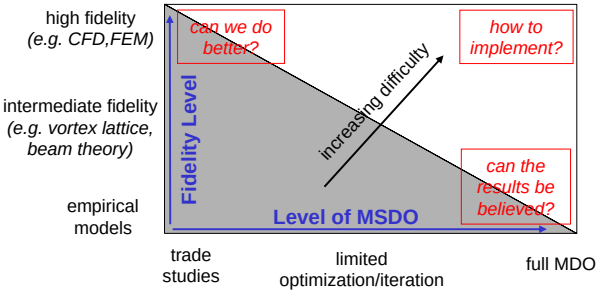
Karen Willcox

Today's Topics

- Design variable linking
- Reduced-Basis Methods
- Response Surface Approximations
- Kriging
- Variable-Fidelity Models

Why Approximation Methods?

We have seen throughout the course the constant trade-off between **computational cost** and **fidelity.**



from Giesing, 1998

Approximation methods provide a way to get high-fidelity model information throughout the optimization without the computational expense.

Approximation Methods

- Recall that the analysis or simcode must be invoked each time the optimizer selects a new design vector to try
- Typically, hundreds (thousands) of design vectors will be analyzed throughout an optimization run
- Can use **Approximation Models** (**Surrogate Models**) for objective functions and constraints
- If approximate models are inexpensive to evaluate, can analyze many more design vector options without worrying about computational resources
- Concept first introduced in structural optimization by Barthelemy and Haftka, 1993

Approximation Methods Overview

Design variable linking Reduced-basis methods	• reduce the number of design variables in the optimization code • simcode analysis full order
Response surface methods Kriging	• same number of design variables • simcode analysis simplified
Variable-fidelity methods	• combine high-fidelity and approximation models

Design Variable Linking

- Not all design variables may be independent
- For example, symmetry may exist in the problem

$$\mathbf{y} = \mathbf{T}\mathbf{x} + \mathbf{y}^C$$

optimizer uses **y**, but provides **x** to simcode for analysis

Define:

$$\underset{r\times 1}{\begin{bmatrix}\mathbf{y}\end{bmatrix}} = \underset{r\times n}{\begin{bmatrix}\mathbf{T}\end{bmatrix}} \underset{n\times 1}{\begin{bmatrix}\mathbf{x}\end{bmatrix}} + \underset{r\times 1}{\begin{bmatrix}\mathbf{y}^C\end{bmatrix}}$$

Reduced-Basis Methods

Consider r feasible design vectors: $\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^r$

We could consider the desired design to be a linear combination of these basis vectors:

$$\mathbf{x}^* = \sum_{i=1}^{r} \alpha_i \mathbf{x}^i + \mathbf{x}^C$$

scalar coefficient — basis vector — added for generality

Reduced-Basis Methods

We can now optimize $J(\mathbf{x})$ by finding the optimal values for the coefficients α_i.

dimension n → dimension r

- do one full-order evaluation of resulting answer
- approach is efficient if $r << n$
- will give the true optimum only if $\mathbf{x}^*$ lies in the span of $\{\mathbf{x}^i\}$
- basis vectors could be
  - previous designs
  - solutions over a particular range (DoE)
  - derived in some other way (POD)

Reduced-Basis Example

Example using a reduced-basis approach (van der Plaats Fig 7-2): airfoil design for a unique application.

- many airfoil shapes with known performance are available
- design variables are (x,y) coordinates at chordwise locations (n~100)
- use four basis airfoil shapes (low-speed airfoils) which contain the n geometry points
- plus two basis shapes which allow trailing edge thickness to vary
- r=6 (r<<n)
- optimize for high speed, maximum lift with a constraint on drag

Reduced-Basis Example

Vanderplaats, G. N. *Numerical Optimization Techniques for Engineering Design.* Vanderplaats R&D, 1999. Figure 7-2.

Vanderplaats, G. N. *Numerical Optimization Techniques for Engineering Design.* Vanderplaats R&D, 1999. Figure 7-3.

Proper Orthogonal Decomposition

Also known as principal components analysis, Kahunen-Loève decomposition, singular value decomposition

$$\mathbf{x}^* = \sum_{i=1}^{r} \alpha_i \varphi^i$$

- The r basis vectors φ_i are orthogonal
- They are computed from M empirical solutions $\{\mathbf{x}^1, \mathbf{x}^2, \ldots \mathbf{x}^M\}$
- They are optimal in the sense that they minimize the error between the original and the projected data

$$\max_{\varphi} \frac{\left\langle \left| (\mathbf{x}, \varphi) \right|^2 \right\rangle}{\left\| \varphi \right\|^2}$$

Proper Orthogonal Decomposition

These optimal basis functions can be calculated by:

1. Evaluating the correlation matrix:

$$R_{ij} = \mathbf{x}^{i\mathrm{T}} \mathbf{x}^j$$

2. Solving the $M \times M$ eigenvalue problem:

$$R\mathbf{v}_i = \lambda_i \mathbf{v}_i$$

3. Constructing the basis vectors:

$$\varphi_j = \sum_{i=1}^{M} v_j^i \mathbf{x}^i$$

Use components of the jth eigenvector to calculate the jth POD basis vector. The jth eigenvalue tells us how important is the jth basis vector.

Approximation Methods Overview

Design variable linking Reduced-basis methods	• reduce the number of design variables in the optimization code • simcode analysis full order
Response surface methods Kriging	• same number of design variables • simcode analysis simplified
Variable-fidelity methods	• combine high-fidelity and approximation models

Response Surface Methodology

- Keep the same number of design variables, but simplify the simcode analysis
- Create approximating functions to objective and constraints
- Optimize using the approximations
- Update approximations using current optimal solution guess and repeat
- Response surfaces are smooth even if design space is noisy
- Polynomial-based modeling technique
- Provide compact, explicit functional relationship between response and design variables
- Least squares is computationally inexpensive and easy to implement

Local Approximations

Most common are Taylor series expansions:

$$J(\mathbf{x}) = J(\mathbf{x}^0) + \left[\nabla J(\mathbf{x}^0)\right]^T \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T \mathbf{H}(\mathbf{x}^0)\delta\mathbf{x} + \cdots$$

$$\delta\mathbf{x} = \mathbf{x} - \mathbf{x}^0$$

- Could use the first two terms $\Rightarrow$ linear approximation
- Solve, reanalyze and repeat = sequential linear programming
- Could also include quadratic term:

$$J(\mathbf{x}) = J(\mathbf{x}^0) + \left[\nabla J(\mathbf{x}^0)\right]^T \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T \mathbf{H}(\mathbf{x}^0)\delta\mathbf{x} + \cdots$$

update requires: 1 function evaluation; n function evaluations; $n(n+1)/2$ function evaluations

Local Approximations

- It is expensive to update the gradient vector and Hessian matrix
- One approach:
  - perform several approximation cycles updating only the constant term
  - then update the linear term and repeat
  - finally, update the Hessian only when no other progress can be made
- If the design space is highly nonlinear, there is no guarantee that this approach will work

Response Surface Methodology

- Another approach: use what information is available to create the approximation
- Use this approximation to make a small move in the design variables
- Analyze the result precisely $\Rightarrow$ new function evaluation
- Use the new function evaluation to improve the approximation to the design space

$\Rightarrow$ **Fit a response surface**

- Can use a quadratic or higher order surface
- Might choose to use only some of the function evaluations (*e.g.* those in a local neighborhood)

RSM

e.g. define $\Delta J = J(\mathbf{x}) - J(\mathbf{x}^0)$

quadratic approximation:

$$\begin{aligned}\Delta J &= \nabla J^{\mathrm{T}}\delta\mathbf{x} + \delta\mathbf{x}^{\mathrm{T}}\mathbf{H}\delta\mathbf{x} \\ &= \frac{\partial J}{\partial x_1}\delta x_1 + \frac{\partial J}{\partial x_2}\delta x_2 + \ldots + \frac{\partial J}{\partial x_n}\delta x_n \\ &+ \frac{1}{2}\left(H_{11}\delta x_1^2 + H_{22}\delta x_2^2 + \ldots + H_{nn}\delta x_n^2\right) \\ &+ H_{12}\delta x_1\delta x_2 + H_{13}\delta x_1\delta x_3 + \ldots + H_{1n}\delta x_1\delta x_n \\ &+ H_{23}\delta x_2\delta x_3 + \ldots + H_{n-1,n}\delta x_{n-1}\delta x_n\end{aligned} \qquad (*)$$

(all derivatives and entries of **H** are evaluated at $\mathbf{x}^0$)

RSM

Assume we have evaluated the baseline plus q designs:

$$\mathbf{x}^0, \mathbf{x}^1, \ldots, \mathbf{x}^q \rightarrow J^0, J^1, \ldots J^q$$

We could write q equations of the form $(*)$ using

$$\left(\mathbf{x}^1 - \mathbf{x}^0, J^1 - J^0\right), \left(\mathbf{x}^2 - \mathbf{x}^0, J^2 - J^0\right), \cdots, \left(\mathbf{x}^q - \mathbf{x}^0, J^q - J^0\right)$$

There is a total of $N=n+n(n+1)/2$ unknowns:

$$\frac{\partial J}{\partial x_1}, \frac{\partial J}{\partial x_2}, \ldots, \frac{\partial J}{\partial x_n}, H_{11}, H_{12}, \ldots, H_{nn}$$

RSM

- q equations, N unknowns
- If $q<N$, only some coefficients can be calculated
- If $q>N$, use weighted least squares
- Weight designs closer to current $\mathbf{x}^q$ more heavily
- In general, use $q \geq n+1$ initial designs so an initial linear approximation can be provided
- If have baseline plus n designs, can fit a linear approximation in each direction (*i.e.* a hyperplane)
- If have more solutions, can fit a quadratic or higher order surface

RSM

- In other words, fit objective function with a polynomial
- *e.g.* quadratic approximation:

$$J(\mathbf{x}) = \mathbf{a_0} + \sum_i b_i x_i + \sum_i c_{ii} x_i^2 + \sum_{i,j<i} c_{ij} x_i x_j$$

- Update model by including a new function evaluation then doing least squares fit to compute the new coefficients

RSM

Estimation problem: $\mathbf{J} \approx \mathbf{X}\,\mathbf{c}$

$$\mathbf{J} = \begin{bmatrix} J^1 & J^2 & \cdots & J^M \end{bmatrix}^{\mathrm{T}}$$

J=vector containing *M* responses

$$\mathbf{x} = \begin{bmatrix} 1 & x_1^1 & x_2^1 & \cdots & x_1^1 x_1^1 & \cdots \\ 1 & x_1^2 & & & & \\ \vdots & \vdots & & & & \\ & & & & & \\ 1 & x_1^M & x_2^M & & x_1^M x_1^M & \end{bmatrix}$$

$$\mathbf{c} = \begin{bmatrix} c_0 \\ c_1 \\ \\ c_{p-1} \end{bmatrix}$$

c=vector containing *p* coefficients

x=*M*×*p* matrix containing *M* design vector inputs as rows. Columns depend on approximation.

Least squares solution: $\mathbf{c} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{J}$

Kriging

- RSM tends to create a global model of the design space (especially if all points weighted equally in the LS)
- May be of limited accuracy when multiple extrema exist (especially quadratic polynomial models)
- Kriging : combine global model of design space with a model for local deviations which interpolates sample points
- Developed in fields of spatial statistics and geostatistics
- Original statistical techniques developed by mining engineer D.G. Krige
- Interpolation-based modeling technique
- Computationally more expensive and not as simple to implement as RSM

Kriging

Unknown function to be modeled expressed as:

$$J(\mathbf{x}) = f(\mathbf{x}) + Z(\mathbf{x})$$

- known function
- "global" model for design space
- *e.g.* use RSM to fit $f(\mathbf{x})$ using *M* observations

- Gaussian random function
- zero mean
- variance σ^2
- "localized" deviation from global model

$$\mathrm{cov}\left[Z(\mathbf{x}^i), Z(\mathbf{x}^j)\right] = \sigma^2 \mathbf{R}\left[R(\mathbf{x}^i, \mathbf{x}^j)\right]$$

covariance matrix of $Z(\mathbf{x})$ — correlation matrix — correlation function

Approximation Methods Overview

Design variable linking
Reduced-basis methods

- reduce the number of design variables in the optimization code
- simcode analysis full order

Response surface methods
Kriging

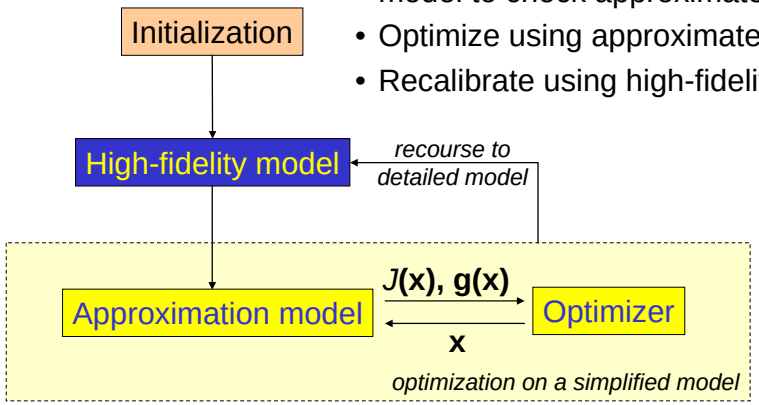
- same number of design variables
- simcode analysis simplified

Variable-fidelity methods

- combine high-fidelity and approximation models

Variable Fidelity Models

- Use information from high-fidelity model to check approximate designs
- Optimize using approximate model
- Recalibrate using high-fidelity model



From: Fig. 1, Alexandrov et al.

Variable Fidelity Models

Questions:

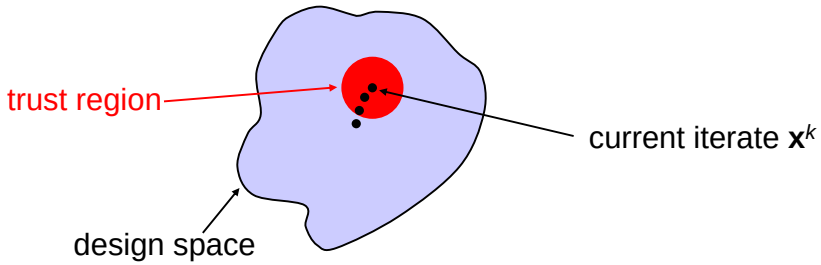
- What do we do when the design derived from the low-fidelity optimization does not provide an improvement in the true objective?
- How can we use information about the predictive capability of the approximation model to decide when to go back to the high-fidelity model?
- How do we decide when to update the approximate model?

Variable Fidelity Models

When optimization with approximation model is unsuccessful, there are two possible approaches:

1. Improve the fidelity of the approximate model
2. Do “less” optimization

One option: use a **trust region** approach



Trust Region Approach

Classic approach:

- Regulate the length of the steps taken by the iterative optimization algorithm
- Regulate based on quality of current approximation model

e.g. Quadratic Taylor series model:

$$J(\mathbf{x}^k + \mathbf{s}) \approx q^k(\mathbf{x}^k + \mathbf{s}) = J(\mathbf{x}^k) + \left[\nabla J(\mathbf{x}^k)\right]^T \mathbf{s} + \frac{1}{2}\mathbf{s}^T \mathbf{B}^k \mathbf{s}$$

where $\mathbf{s}$ is the prospective step in the design variables, $\Delta\mathbf{x}$, and $\mathbf{B}^k$ is a model of the Hessian matrix at $\mathbf{x}^k$.

Trust Region Approach

- Restrict the step size to a region in which we trust the quadratic model to approximate J well
- Done by adding a constraint on the step

$\Rightarrow$ trust region subproblem

$$\begin{aligned} &\min q^k(\mathbf{x}^k + \mathbf{s}) \\ &\text{s.t.} \quad \|\mathbf{s}\| \le \delta^k \end{aligned}$$

- In practice, variables are scaled to improve performance (step size can vary in different directions)
- Then decide whether to accept the step:

$$\mathbf{x}^{k+1} = \begin{cases} \mathbf{x}^k + \mathbf{s} & \text{if } J(\mathbf{x}^k + \mathbf{s}) < J(\mathbf{x}^k) \\ \mathbf{x}^k & \text{otherwise} \end{cases}$$

Trust Region Update

- Trust radius, δ^k, is updated adaptively
- Update depends on predictive quality of quadratic model:
  - ➢ if model did a good job of predicting J, or if there was more improvement than predicted, then increase δ^k
  - ➢ if model did a bad job of predicting J (J increased or decrease was much lower than predicted), then decrease δ^k
  - ➢ if model did an acceptable job of predicting J, then do not change δ^k

Trust Region Update

- Numerically, compare actual and predicted decrease:

$$r = \frac{J(\mathbf{x}^k) - J(\mathbf{x}^k + \mathbf{s})}{J(\mathbf{x}^k) - q(\mathbf{x}^k + \mathbf{s})}$$

- Define constants r_1, r_2 ($r_1 < r_2$) and apply the rules:
  - if $r < r_1$, decrease the trust radius
  - if $r > r_2$, increase the trust radius
- Typical values are $r_1 = 0.1$, $r_2 = 0.75$
- Note that prediction of ascent/descent is more important than prediction of actual value

Classic Trust Region Algorithm

Choose $\mathbf{x}^0 \in \mathbb{R}^n$ and $\delta^0 > 0$

For k=0,1,... until convergence do {

Find an approximate solution $\mathbf{s}_k$ to the subproblem:

$$\min q^k(\mathbf{x}^k + \mathbf{s})$$
$$\text{s.t.} \quad \|\mathbf{s}\| \le \delta^k$$

Compare the actual and predicted decrease:

$$r = \frac{J(\mathbf{x}^k) - J(\mathbf{x}^k + \mathbf{s})}{J(\mathbf{x}^k) - q(\mathbf{x}^k + \mathbf{s})}$$

Update $\mathbf{x}^k$ and δ^k

}

From Fig. 2, Alexandrov et al.

β-Correlation Method

- Basic idea is to take low-fidelity approximate model, J_a, and correct it by scaling
- Define the scale factor:

$$\beta = \frac{J(\mathbf{x})}{J_a(\mathbf{x})}$$

Given the current design $\mathbf{x}^k$, build a first order model of β about $\mathbf{x}^k$:

$$\beta^k(\mathbf{x}) = \beta(\mathbf{x}^k) + \nabla\beta(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k)$$

Optimize with approximate model and use local model of β to scale result :

$$J(\mathbf{x}) \approx \beta^k J_a(\mathbf{x})$$

Variable Complexity Model

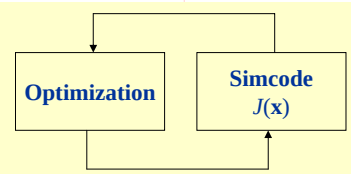
- iSight uses a Variable Complexity Model (VCM)
- Use two simcodes for the same physical process:
  (1) more accurate, longer running (exact) $J(\mathbf{x})$
  (2) less accurate, shorter running (approx.) $J_a(\mathbf{x})$
- Compute a multiplicative or additive correction factor, σ:

$$\sigma = \frac{J(\mathbf{x}^0)}{J_a(\mathbf{x}^0)}$$
$$J(\mathbf{x}) = \sigma J_a(\mathbf{x})$$

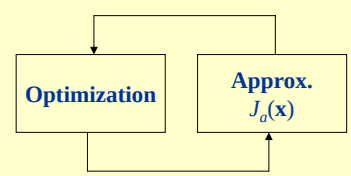
or

$$\sigma = J(\mathbf{x}^0) - J_a(\mathbf{x}^0)$$
$$J(\mathbf{x}) = \sigma + J_a(\mathbf{x})$$

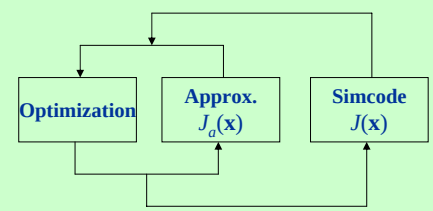
VCM



Conventional optimization with simcode



Conventional optimization with approximate model



Optimize with approximate model, update with simcode:

1. Run both models, calculate σ^0
2. Optimize using approximate model and σ^0
3. Update correction factor using simcode and repeat

Lecture Summary

- Approximation methods are one way to capture high-fidelity responses without the computational cost
- Design variable linking if physical problem allows
- Reduced-basis methods use low-order representation of the design vector
  - use previous designs, DoE or POD
- Response surface methodology
  - use polynomial models
  - weighted least squares
- Kriging
  - interpolation models
  - global + local behavior
- Variable-fidelity models
  - Trust region approach
  - β-correlation

References

Barthelemy, J-F. M. and Haftka, R.T., “Approximation concepts for optimum structural design – a review”, *Structural Optimization*, 5:129-144, 1993.

Giunta, A.A. and Watson, L.T.,”A comparison of approximation modeling techniques: polynomial versus interpolating models”, AIAA Paper 98-4758, 1998.

LeGresley, P.A. and Alonso, J.J., “Airfoil design optimization using reduced order models based on proper orthogonal decomposition”, AIAA Paper 2000-2545.

Alexandrov, N., Dennis, J.E., Lewis, R.M. and Torczon, V., “A trust region framework for managing the use of approximation models in optimization”, NASA CR-201745, ICASE Report No. 97-50, October 1997.

Practical Optimization, P.E. Gill, W. Murray and M.H. Wright, Academic Press, 1986.

Numerical Optimization Techniques for Engineering Design, G.N. Vanderplaats, Vanderplaats R&D, 1999.