Multidisciplinary System Design Optimization (MSDO)

Scaling & Approximation Methods

Recitation 8

Andrew March

Today's Topics

- Convergence Rates
  - Steepest Descent
  - Conjugate Gradient
  - Quasi-Newton
  - Newton
- Scaling
- Approximation Methods
  - Quadratic Response Surface
  - Kriging
- More on trust regions

Quadratic Functions

- The analysis to be presented only applies to quadratic functions:

$$f(x) = \frac{1}{2} x^T Q x + c^T x$$

- It assumes the line-search is exact:

$$x_{k+1} = x_k - \alpha_k D_k \nabla f\left(x_k\right)$$

$$\alpha_k = \arg\min_{\alpha} f\left(x_k - \alpha_k D_k \nabla f\left(x_k\right)\right)$$

- It also provides only a worst case upper bound, but is generally good in practice.

Steepest Descent

- $f(x) = \frac{1}{2} x^T Q x + c^T x$
- Exact line search solution:

$$\alpha_k = \frac{\nabla f(x_k)^T \nabla f(x_k)}{\nabla f(x_k)^T H(x_k) \nabla f(x_k)}$$

- Convergence rate:

$$f(x_{k+1}) \le \left(\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} \right)^2 f(x_k) \text{ or } f(x_{k+1}) \le \left(\frac{\lambda_{\max}/\lambda_{\min} - 1}{\lambda_{\max}/\lambda_{\min} + 1} \right)^2 f(x_k)$$

- Where $0 \le \lambda_1, \lambda_2, \ldots, \lambda_{n-1}, \lambda_n$ are eigenvalues of Q
  - $\lambda_1 = \lambda_{min}$, and $\lambda_n = \lambda_{max}$

Conjugate Gradient

- $f(x) = \frac{1}{2} x^T Q x + c^T x$
- Where $0 \leq \lambda_1, \lambda_2, \ldots, \lambda_{n-1}, \lambda_n$ are eigenvalues of Q
- $\|x - x^*\|_A^2 \equiv (x - x^*)^T A (x - x^*)$
- Convergence rate:

$$\|x_{k+1} - x^*\|_Q^2 \leq \left(\frac{\lambda_{n-k} - \lambda_1}{\lambda_{n-k} + \lambda_1} \right)^2 \|x_0 - x^*\|_Q^2$$

- Less tight bound:

$$\|x_{k+1} - x^*\|_Q \leq 2 \left(\frac{\sqrt{\lambda_n / \lambda_1} - 1}{\sqrt{\lambda_n / \lambda_1} + 1} \right)^k \|x_0 - x^*\|_Q$$

- Maximum number of iterations?

Quasi-Newton (Broyden Class)

- $f(x) = \frac{1}{2} x^T Q x + c^T x$
- Where $0 \le \lambda_1, \lambda_2, \ldots, \lambda_{n-1}, \lambda_n$ are eigenvalues of Q
- $\|x - x^*\|_A^2 \equiv (x - x^*)^T A (x - x^*)$
- Convergence rate:

$$\|x_{k+1} - x^*\|_Q^2 \le \left(\frac{\lambda_{n-k} - \lambda_1}{\lambda_{n-k} + \lambda_1} \right)^2 \|x_0 - x^*\|_Q^2$$

- Less tight bound:

$$\|x_{k+1} - x^*\|_Q \le 2 \left(\frac{\sqrt{\lambda_n / \lambda_1} - 1}{\sqrt{\lambda_n / \lambda_1} + 1} \right)^k \|x_0 - x^*\|_Q$$

- Note for the Broyden class:
  - If the objective function is quadratic,
  - the initial Hessian estimate is identity,
  - and the line-search is exact,
- Then the iterates are the same as the conjugate gradient method

Newton's Method

- Convergence bound?

$$f\left(x_{k+1}\right) \leq 0 \cdot f\left(x_k\right)$$

Why Scaling?

- $f(x) = \frac{1}{2} x^T Q x + c^T x$
- For a method using: $x_{k+1} = x_k - \alpha_k D_k \nabla f(x_k)$
- Convergence rate:

$$f(x_{k+1}) \leq \left(\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} \right)^2 f(x_k) \text{ or } f(x_{k+1}) \leq \left(\frac{\lambda_{\max}/\lambda_{\min} - 1}{\lambda_{\max}/\lambda_{\min} + 1} \right)^2 f(x_k)$$

- Where $\lambda_1 = \lambda_{min}$, and $\lambda_n = \lambda_{max}$ of the matrix:

$$(D_k)^{1/2} Q (D_k)^{1/2}$$

Scaling-Practice

- $$\min_{x \in \Re^n} f(x) = \frac{1}{2} x^T Q x + c^T x$$
- $$Q = \begin{bmatrix} 4 & 0 \\ 0 & 100 \end{bmatrix}, \quad c = \begin{bmatrix} 6 \\ 200 \end{bmatrix}$$
- What is P, such that performing the optimization of f(x) using $\tilde{x} = Px$ requires the fewest number of iterations possible?
  – How many iterations will be required for:
    - Newton
    - CG/Quasi-Newton
    - Steepest Descent

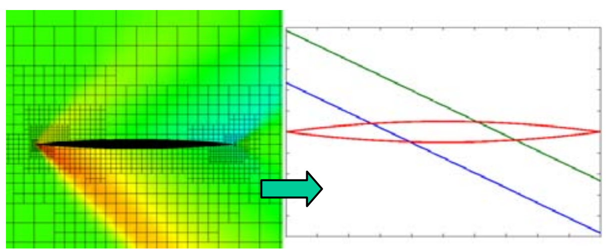
Approximation Methods

Multifidelity Surrogates

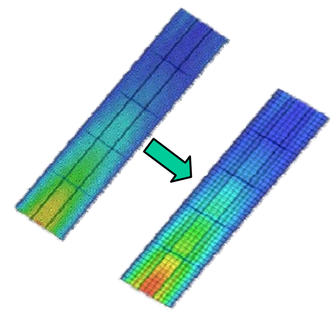
- Definition: *High-Fidelity*
  - The best model of reality that is available and affordable, the analysis that is used to validate the design.
- Definition: *Low(er)-Fidelity*
  - A method with unknown accuracy that estimates metrics of interest but requires lesser resources than the high-fidelity analysis.

Hierarchical Models

Reduced Physics

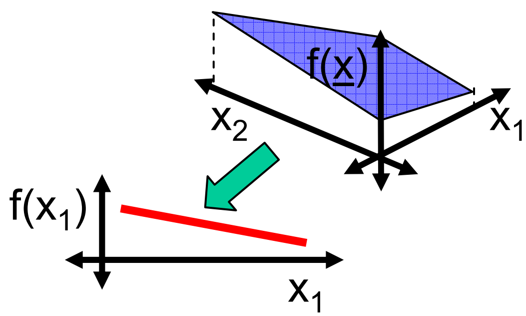


Coarsened Mesh

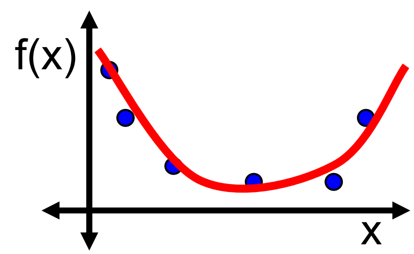


Approximation Models

Reduced Order Model



Regression Model



Gradient-Based: Response Surface

- Generate a response surface:
- x_{ij} i=dimension
j=sample point #
- Sample at a collection of x_i

$$X = \begin{bmatrix} 1 & x_{11} & x_{21} & x_{11}x_{21} & x_{12}^2 & x_{21}^2 \\ 1 & x_{12} & x_{22} & x_{12}x_{22} & x_{12}^2 & x_{22}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{1n} & x_{2n} & x_{1n}x_{2n} & x_{1n}^2 & x_{2n}^2 \end{bmatrix}$$

$$\beta = [\beta_1 \quad \beta_2 \quad \beta_3 \quad \beta_4 \quad \beta_5 \quad \beta_6]^T$$

$$F = [f(x_{11}, x_{21}) \quad f(x_{12}, x_{22}) \quad \cdots \quad f(x_{1n}, x_{2n})]^T$$

- Solve for β: $X\beta = F$
- Or least-squares solution: $X^T X\beta = X^T F$

Kriging Methods

- Recommendation:
  - DACE toolbox for Matlab: http://www2.imm.dtu.dk/~hbn/dace/
- Glutton's for punishment:
  - Gaussian Processes for Machine Learning (Book-available online)

    http://www.gaussianprocess.org/gpml/
  - Simplest version on pg 19.

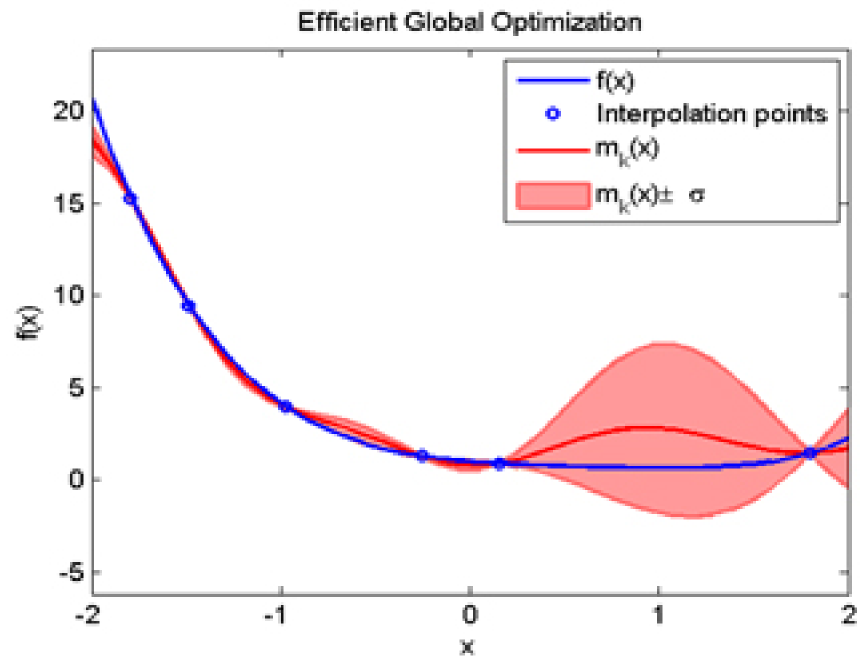
Efficient Global Optimization

- Started by Jones 1998
- Based on probability theory
  - Assumes:

$$f(\mathbf{x}) \approx \beta^T \mathbf{x} + N\left(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\right)$$

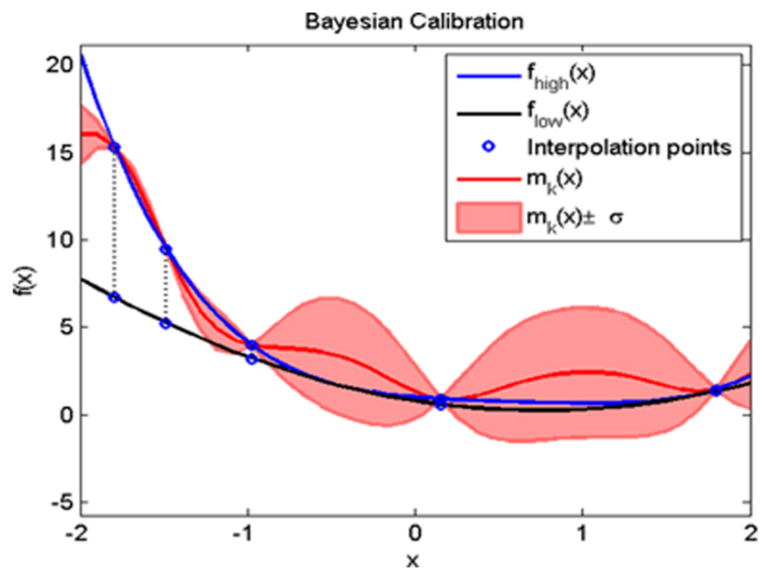
- $\beta^T \mathbf{x}$, true behavior, regression
- $N\left(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\right)$, error from true behavior is normally distributed, with mean $\mu(\mathbf{x})$, and variance $\sigma^2(\mathbf{x})$

- Estimate function values with a Kriging model (radial basis functions)
  - Predicts mean and variance
  - Probabilistic way to find optima
- Evaluate function at “maximum expected improvement location(s)” and update model



Bayesian Model Calibration

- $f_{high}(\mathbf{x}) \approx m_k(\mathbf{x}) = f_{low}(\mathbf{x}) + \varepsilon_k(\mathbf{x})$
- Model the error between a high- and low-fidelity function
  - Bayesian approach
- If the low-fidelity function is “good”:
  - Converges faster
  - Lower variance
- Global calibration procedure



Kriging Demo

Trust-Region Algorithm Summary

- Solve the trust-region subproblem to determine a candidate step, $\mathbf{s}_k$:

$$\min_{\mathbf{s}_k \in \Re^n} m_k(\mathbf{x}_k + \mathbf{s}_k)$$

$$s.t. \quad \|\mathbf{s}_k\| \leq \Delta_k$$

- Evaluate f_{high} at the candidate point and compute the ratio of actual to predicted reduction:

$$\rho_k = \frac{f_{high}(\mathbf{x}_k) - f_{high}(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}$$

- Accept/reject iterate: $\mathbf{x}_{k+1} = \begin{cases} \mathbf{x}_k + \mathbf{s}_k & \rho_k > 0 \\ \mathbf{x}_k & \text{otherwise} \end{cases}$

- Update trust region size: $\Delta_{k+1} = \begin{cases} \min\{2\Delta_k, \Delta_{\max}\} & \rho_k \geq 0.75 \\ 0.5\Delta_k & \rho_k < 0.25 \end{cases}$

- Perform convergence check: $\|\nabla f_{high}(\mathbf{x}_k)\| \leq \varepsilon_1$

Convergence Requirements

- First-order consistency:

$$f_{high}(\mathbf{x}_k) = m_k(\mathbf{x}_k)$$
$$\nabla f_{high}(\mathbf{x}_k) = \nabla m_k(\mathbf{x}_k)$$

- Simplest trust-region model:

$$m_k(\mathbf{x}_k) = f_{high}(\mathbf{x}_k) + \nabla f_{high}(\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k) + \frac{1}{2}(\mathbf{x}-\mathbf{x}_k)^T \nabla^2 f_{high}(\mathbf{x}_k)(\mathbf{x}-\mathbf{x}_k)$$

- For a general low-fidelity function:

$$\beta = \frac{f_{high}(\mathbf{x})}{f_{low}(\mathbf{x})}$$
$$\beta_c = \beta(\mathbf{x}_k) + \nabla\beta(\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k)$$
$$m_k(\mathbf{x}) = \beta_c(\mathbf{x})\, f_{low}(\mathbf{x})$$

$$a(\mathbf{x}) = f_{high}(\mathbf{x}) - f_{low}(\mathbf{x})$$
$$\nabla a(\mathbf{x}) = \nabla f_{high}(\mathbf{x}) - \nabla f_{low}(\mathbf{x})$$
$$m_k(\mathbf{x}) = f_{low}(\mathbf{x}) + a(\mathbf{x}_k) + \nabla a(\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k)$$

First-Order Consistent Trust Region

- Trust region approach
  [Alexandrov1997, 1999]
- Requires:

$$f_{high}(\mathbf{x}_k) = m_k(\mathbf{x}_k)$$

$$\nabla f_{high}(\mathbf{x}_k) = \nabla m_k(\mathbf{x}_k)$$

- β-Correlation

$$\beta = \frac{f_{high}(\mathbf{x})}{f_{low}(\mathbf{x})}$$

$$\beta_c = \beta(\mathbf{x}_k) + \nabla\beta(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k)$$

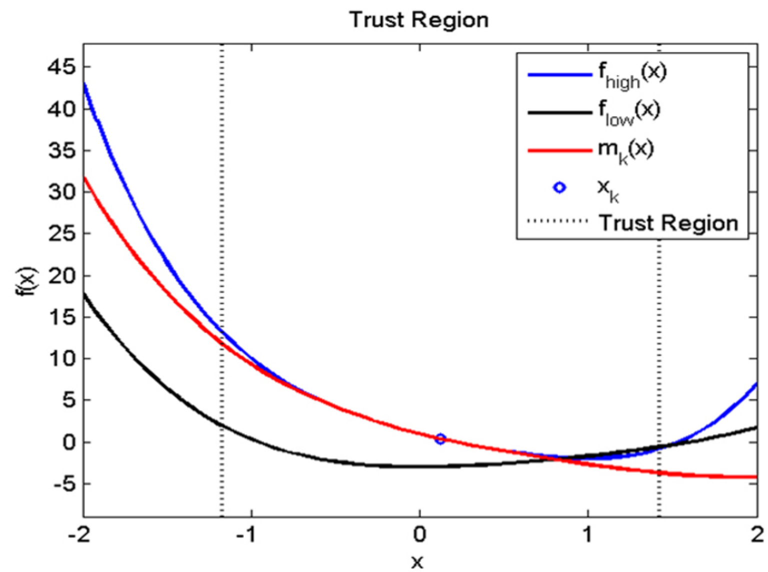
$$m_k(\mathbf{x}) = \beta_c(\mathbf{x})\, f_{low}(\mathbf{x})$$

- Additive-Correction

$$a(\mathbf{x}) = f_{high}(\mathbf{x}) - f_{low}(\mathbf{x})$$

$$\nabla a(\mathbf{x}) = \nabla f_{high}(\mathbf{x}) - \nabla f_{low}(\mathbf{x})$$

$$m_k(\mathbf{x}) = f_{low}(\mathbf{x}) + a(\mathbf{x}_k) + \nabla a(\mathbf{x}_k)^T(\mathbf{x} - \mathbf{x}_k)$$



$$\rho_k = \frac{f_{high}(\mathbf{x}_k) - f_{high}(\mathbf{x}_k + \mathbf{s}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{s}_k)}$$

Trust Region Demo

Summary

- Scaling can be really important
  - Demonstrated theory
  - Surprising importance in practice
- Approximation methods
  - Use only when necessary
  - Can save a lot of time
  - Do your best to choose the right one, exploit the aspects of your problem that you can.
    - Gradients available/Finite-difference reliable?
    - Constrained?
    - Physical behavior similar to a lower-fidelity model?

ESD.77 / 16.888 Multidisciplinary System Design Optimization
Spring 2010