Computational and Experimental Aeroelasticity Multidisciplinary Design and Optimization

Prabhat Hajela
Mechanical, Aerospace and Nuclear Engineering
Rensselaer Polytechnic Institute, Troy, New York USA

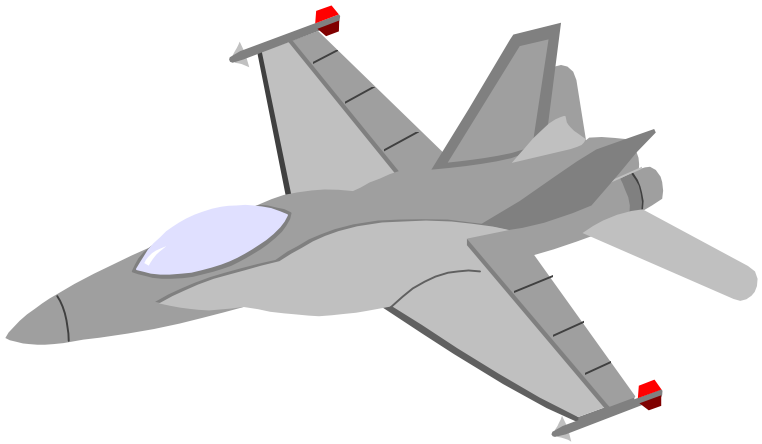
Sponsored by

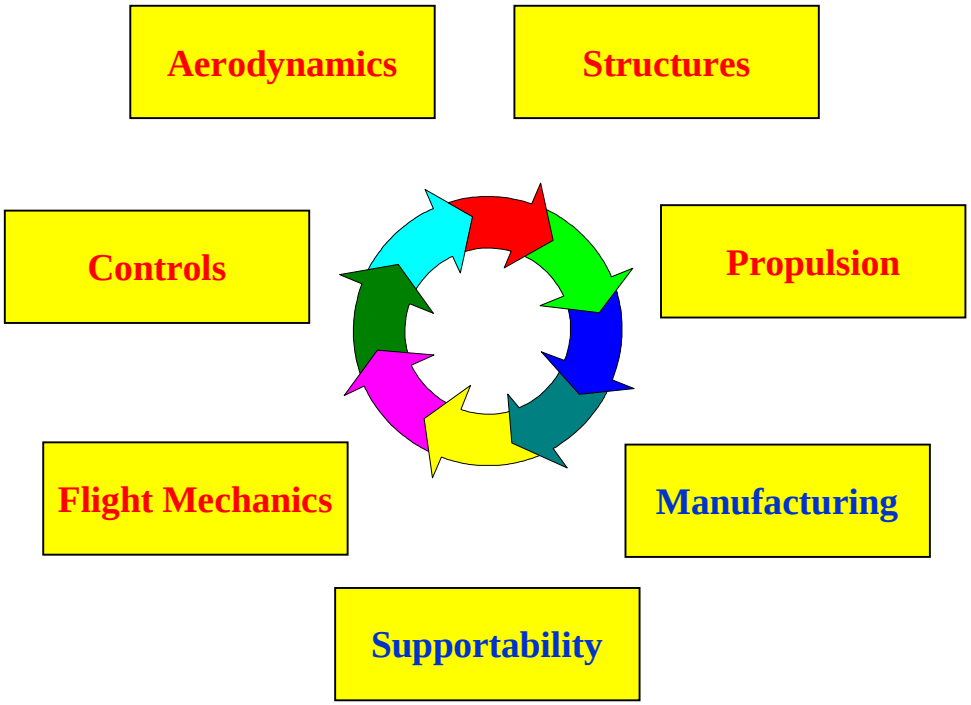
RTA-NATO
METU, Ankara, Turkey

Outline of MDO Lectures

- The multidisciplinary design problem
  - components, challenges, and opportunities
- Review of numerical optimization methods
- Methods for sensitivity analysis
- Decomposition methods in design
- Applications of soft computing in multidisciplinary optimization

The MDAO Problem





MDAO Problem - General Characteristics

- Final system or product is generally an integration of several distinct subsystems - the desired objectives of the system are met by varying parameters within each subsystem
- Collectively, for all subsystems, the number of design parameters and constraints can be quite large
- Design decisions/parameter changes within a particular subsystem cannot be generally affected without a benificial/adverse effect in another subsystem
- Analysis techniques within a subsystem develop over time to be uniquely well suited to a particular discipline

The MDAO Problem

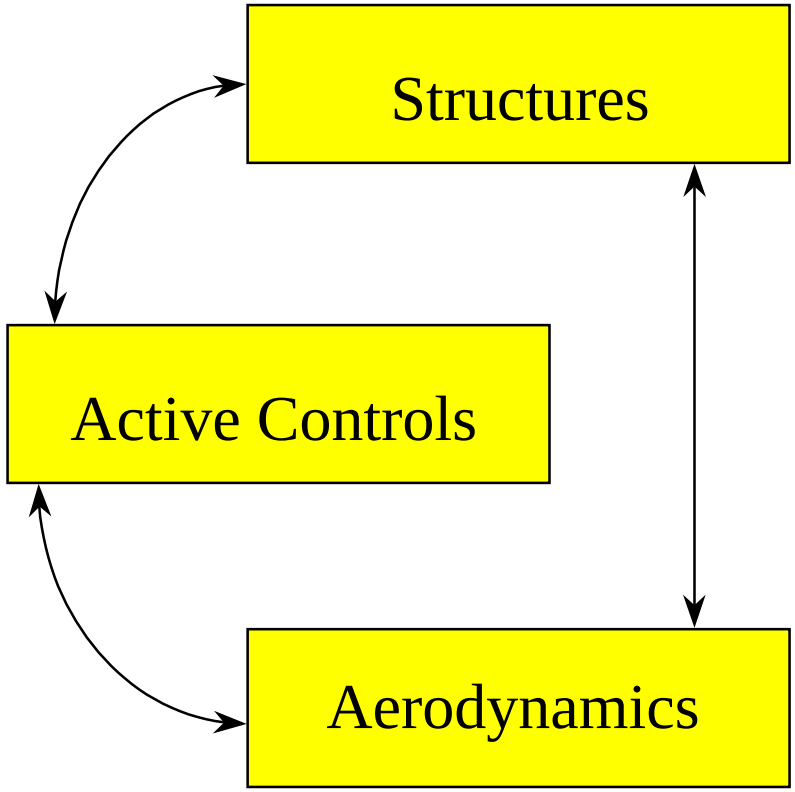
- System analysis
  - analysis for design
    - analytical or numerical modeling
    - approximate analysis
- Design problem formulation
- Design optimization
  - strategies, methods
  - sensitivity
- Interpretation of results
- Data-handling

The Analysis Problem

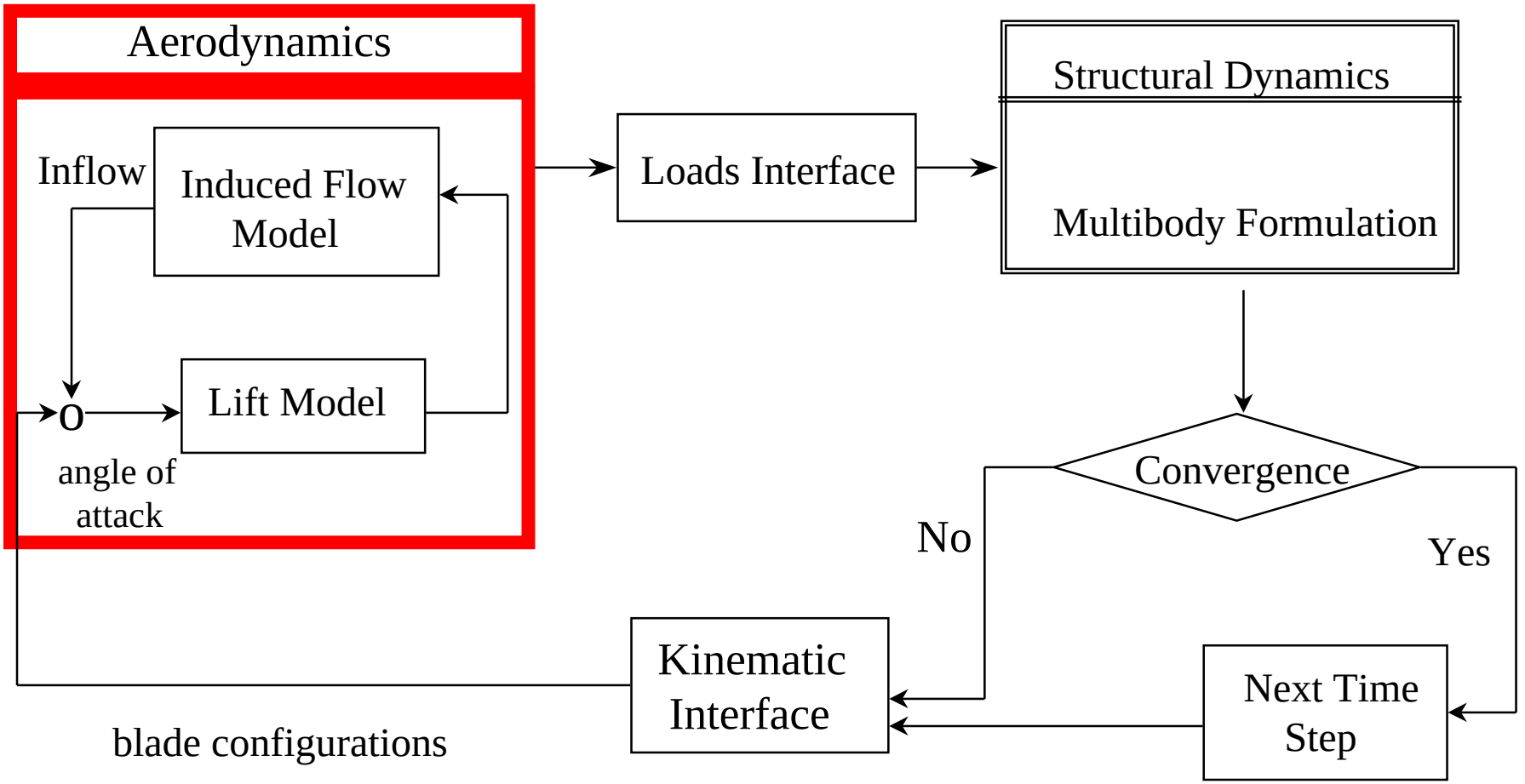
- Complex-coupled analysis
  - computationally demanding
- Difficulty in computing sensitivities
- Lack of analytical or numerical models
  - manufacturing
  - supportability
  - cost

Rotorblade Design

- Design of rotorblade
  - structural design
  - aerodynamic planform
  - airfoil selection
  - control system gains
- Complex-coupled nonlinear analysis
  - computationally intense
  - incompletely defined



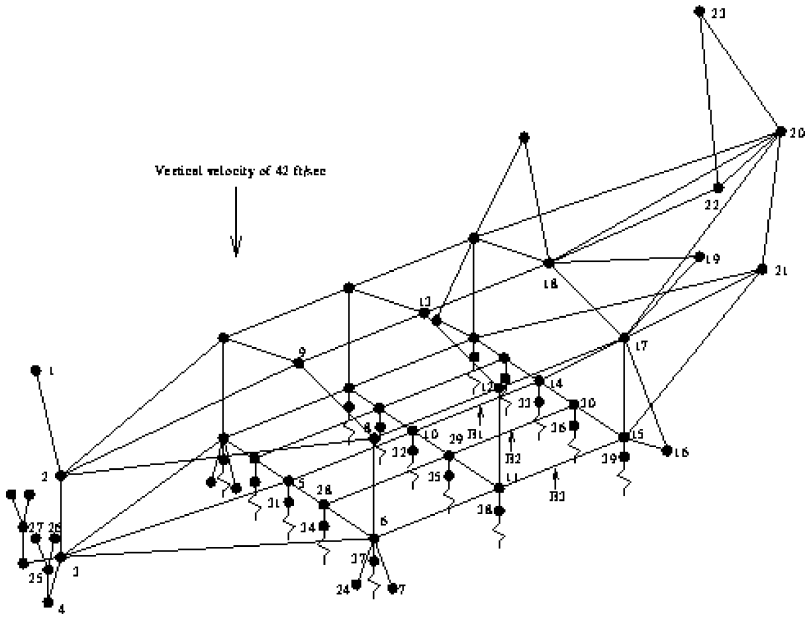
Typical Coupled Analysis



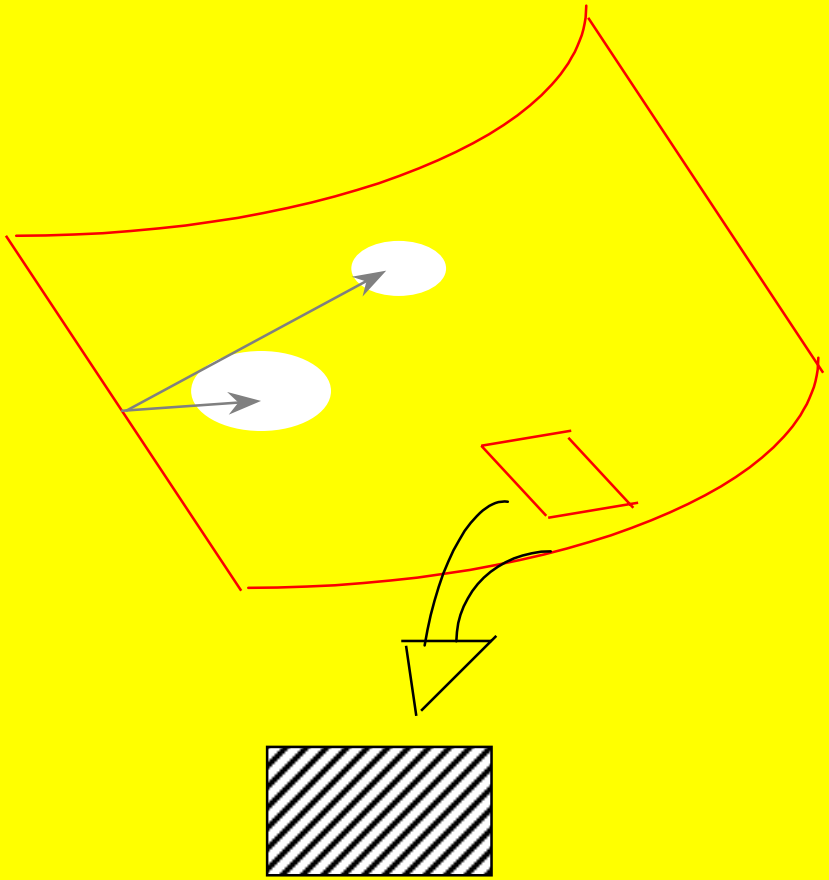
Crashworthy Design of Rotorcraft Structures

- Topological design of grillage-type subfloor structures - enhanced crashworthiness
  - placement of energy absorbing components
  - optimal load-deflection characteristics of energy absorbing components
  - selection of geometry of energy absorbing components

CH-47 KRASH model



Process Modeling for Design



Fiber Layup Machine

> Panel Thickness
> Panel Geometry
> Panel Curvature
> Location of Cutouts

Goals

> Layup time minimization
> Material cost minimized
> Structural Integrity
> Maintainability

The Design Problem

- The problem formulation
- Design space is a mix of continuous, discrete, or integer variables - high dimensionality
- Design space may be nonconvex or disjointed
- Non- crisp design information

Design Problem Statement

- Objective can be a scalar or vector function
- Both maximization and minimization are admissible
- Constraints may be linear or nonlinear - equality and/or inequality
- Objective criterion may be treated as constraints
- Problem formulation is critical to the efficiency of the numerical search process

$$\textit{Minimize} \quad F(X) = \{ f_1, f_2, \ldots f_p \}$$

$$\textit{Subject to}:$$

$$g_j(X) \le 0 \quad j = 1, m$$

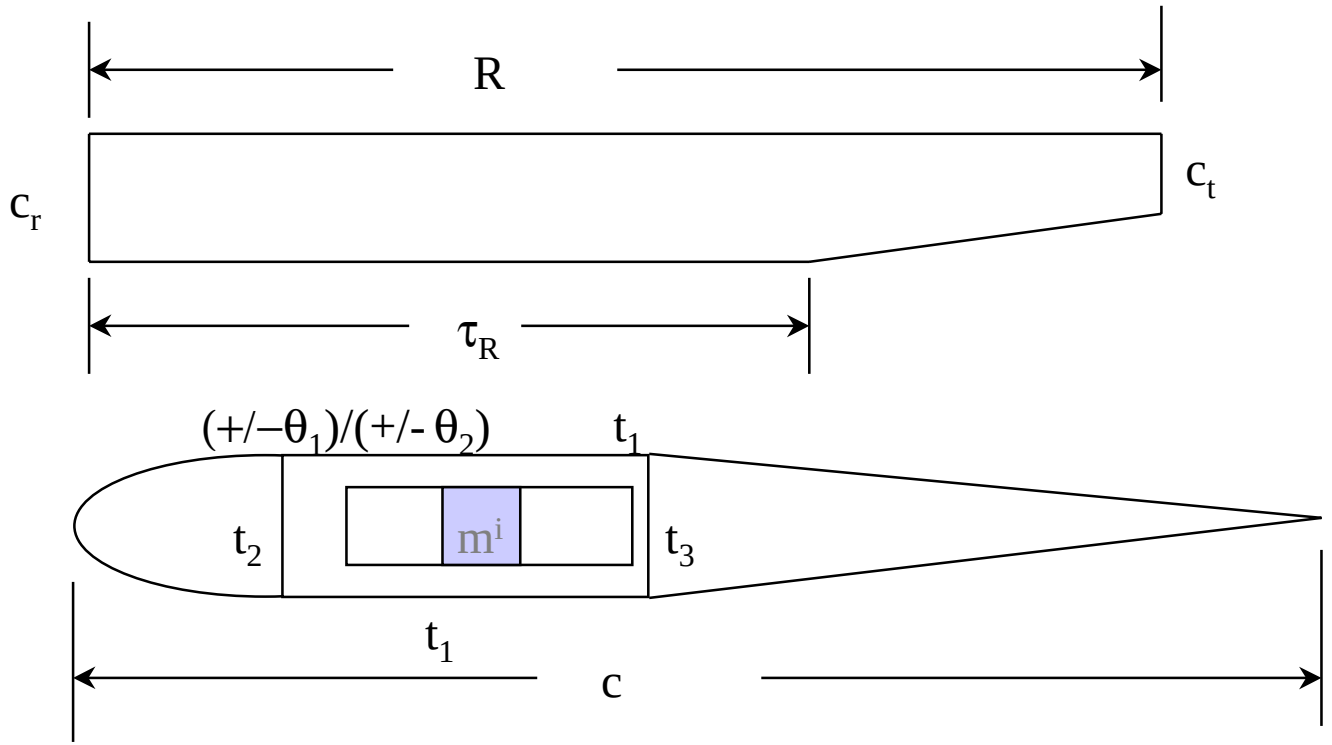
$$h_k(X) = 0 \quad k = 1, l$$

$$X_i^L \le X_i \le X_i^U \quad i = 1, n$$

Challenge of Dimensionality

- High-Speed Civil Transport - Wing Structure (Ref. Sobieski'97)
  - Analysis : 2694 nodes, 16152 EDOF's, 3634 quad elements
  - Optimization : 7918 design variables, 10 constraints per element, 60 different load cases, 2,200,000 design constraints
- Large dimensionality over-extends current capabilities of traditional search techniques
- Compromised ability to interpret progress of design optimization
- Storage and subsequent use of information
  - matrix of constraint derivatives (ndvxncon=17Gwords)

Mixed Variable Design Space
Rotor Blade Design



Mixed Variable Space

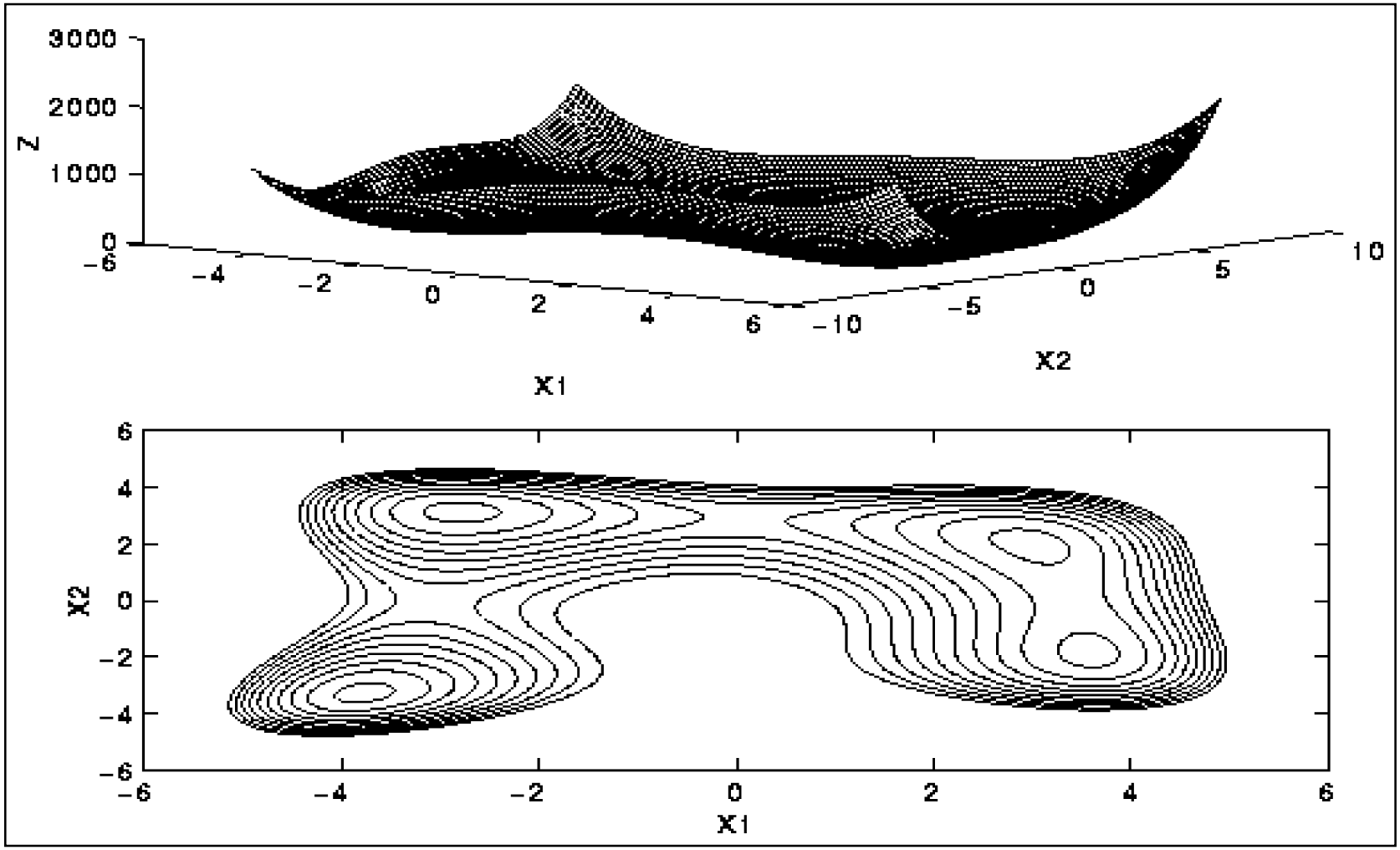
Design Variable	*ID*	*Type*
horizontal flange thickness	t_1^i	continuous
tuning mass	m^i	continuous
vertical web thickness	t_2^i, t_3^i	continuous
blade twist	θ_t	continuous
twist shape parameter	δ	continuous
taper inception point	τ_R	continuous
chord ratio	λ_c	continuous
rotational speed	Ω	integer
composite layup angle	θ_1,θ_2	discrete

where, i denotes the blade spanwise segment

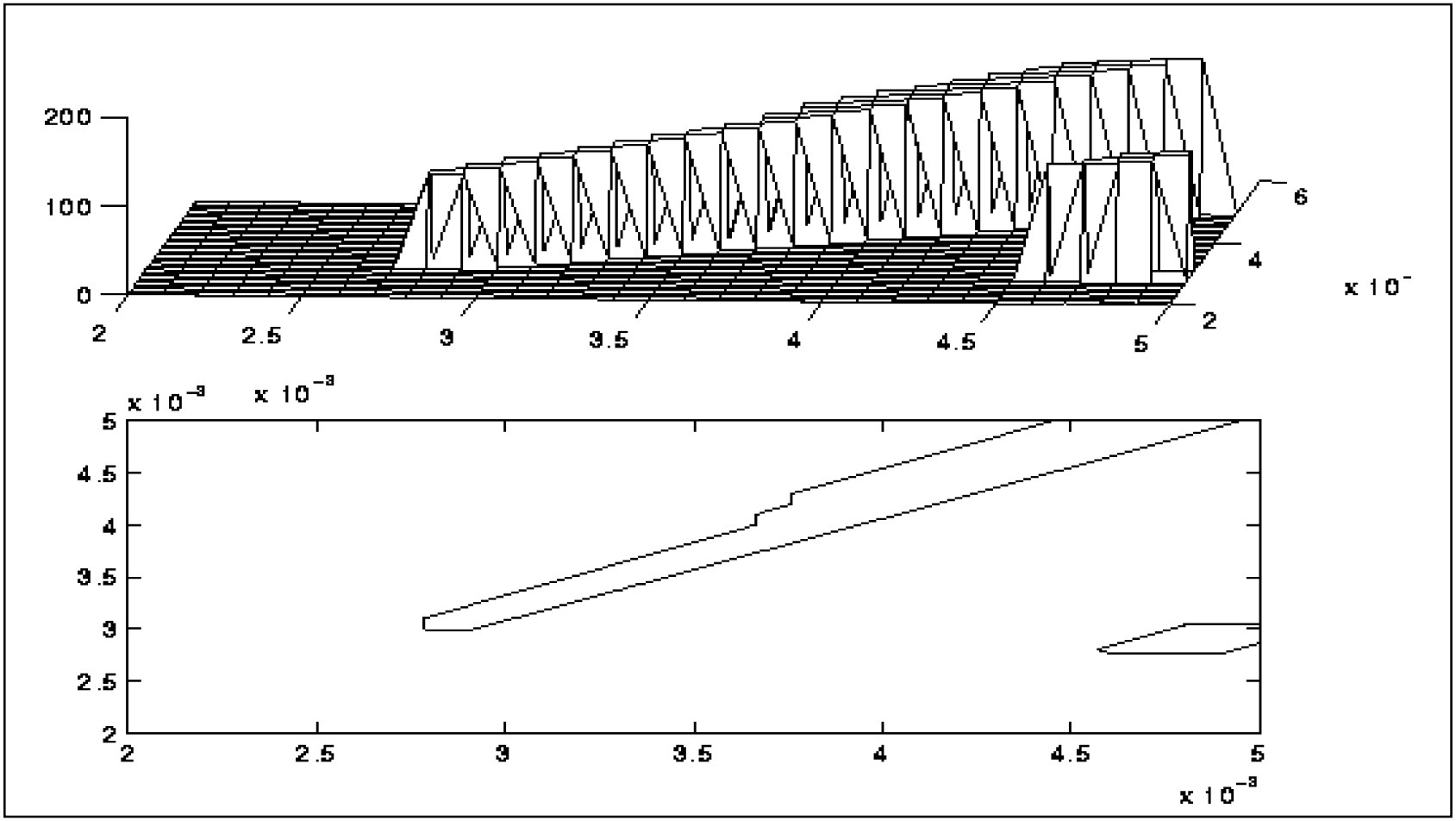
Mixed Variable Design Space

- Mix of discrete, integer and continuous design variables
- Traditional optimization methods ill-equipped to handle this mixed-variable space
  - rounding strategies
  - branch-and-bound techniques
- Discontinuity in gradient information

Nonconvex Design Space



Disjoint Design Space



Non-Crisp Design Problem Statement

- Design requirements may vary over the design cycle
- Imprecise information related to process models
- Uncertainties in manufacturing processes and/or availability of materials
- Market forces
- Uncertainties in life-cycle requirements

Overview of Numerical Optimization

Design Optimization

- General statement of optimization problem

$$\begin{aligned}
&\textit{Minimize} \quad F(X) \\
&\textit{Subject to} \\
&g_j(X) \le 0 \qquad j = 1, m \quad \text{inequality constraint} \\
&h_k(X) = 0 \qquad k = 1, p \quad \text{equality constraint} \\
&X_i^L \le X_i \le X_i^U \qquad i = 1, n \quad \text{side constraint}
\end{aligned}$$

- Example would be to minimize the weight of a wing structure so that stresses in all elements be below permissible levels and first natural frequency be equal to some stipulated value

Mathematical Optimization

- The objective and constraint functions may be linear, nonlinear, explicit, or implicit
- Objective function and all constraints linear - linear optimization problem or linear programming problem
- Both objective and constraint functions are nonlinear - nonlinear programming problem
- Some function minimization or maximization may involve no constraints - unconstrained optimization problem

- *Important issue to consider up front is that the principal computational cost in numerical optimization is due to the repetitive analysis that must be performed*

General Approach in Optimization

- Given a starting point X^0
- At the q-th iteration, update the design vector as follows

$$X^q = X^{q-1} + \alpha S^q$$

  - S^q search direction vector and α is the step size
- Calculation of the search direction and step size differ in the different strategies for nonlinear programming based design optimization methods
  - step size calculation is a one-dimensional search
  - search direction calculation may require sensitivity information

Some Definitions

- Design variables - parameters that can be varied to improve the design
- Constraints - conditions that must be satisfied for the design to be acceptable (inequality - one sided, equality - precisely, side - bounds on the design variables)
- Objective function - function of design variables that must be minimized or maximized
- Design parameters - fixed parameters for which an optimal design is obtained
- Feasible design - all constraints satisfied
- Infeasible design - one or more constraint is violated
- Active constraint - design is on a constraint boundary

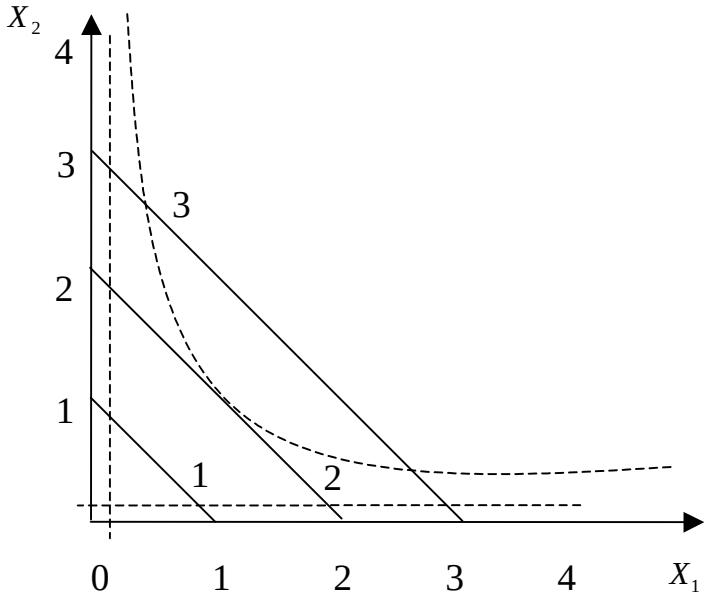
Example

Minimize $F(X) = X_1 + X_2$

Subject to

$g(X) = \frac{1}{X_1} + \frac{1}{X_2} - 2 \le 0$ *inequality*

$X_1 \ge 0.2 \quad X_2 \ge 0.2$ *side*



Necessary Conditions for Optimality Kuhn-Tucker Conditions

- If X^* is feasible then the following conditions hold

$$\lambda_j g_j(X^*) = 0$$

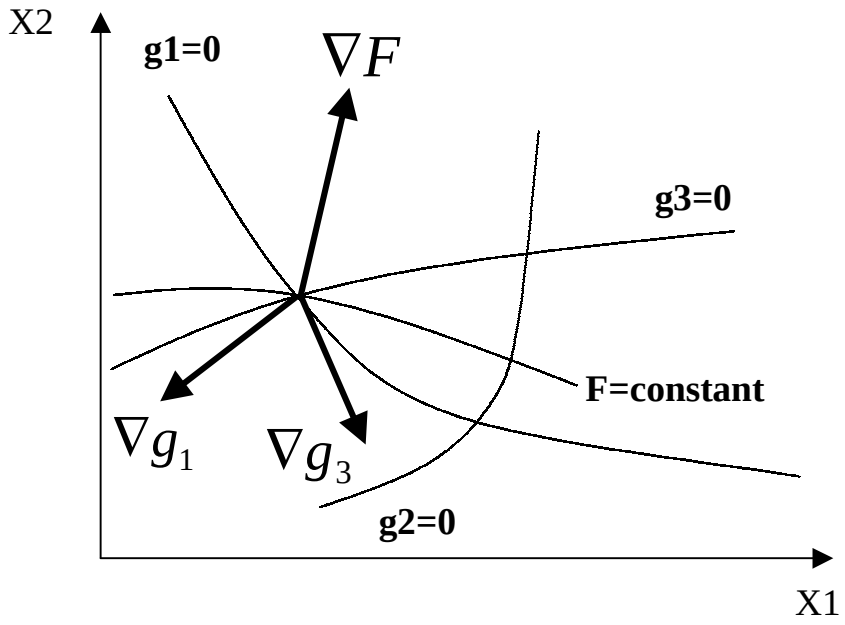
$$\nabla F(X^*) + \sum_{j=1}^{m} \lambda_j \nabla g_j(X^*) + \sum_{k=m+1}^{m+p} \lambda_k \nabla h_{k-m}(X^*) = 0$$

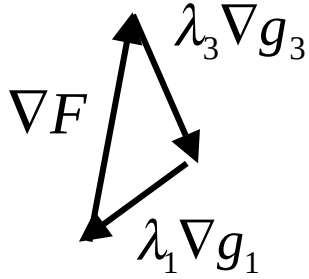
$$\lambda_j \geq 0, \quad j = 1, m$$

λ_k unrestricted in sign for equality constraints

- Vector sum of the objective gradients and scaled gradients of the active constraints must add to zero at the optimum

Necessary Conditions for Optimality
Kuhn-Tucker Conditions





Unconstrained Optimization

- Find X to minimize F(X) and where the Kuhn-Tucker condition reduces to $\nabla F(X) = 0$

- Many different algorithms
  - Powell's method (zero order), steepest descent, Fletcher-Reeves, Davidon-Fletcher-Powell, Broydon-Fletcher-Goldfard-Shanno (first order), Newton's method (second order)
- Repetitive process in which a search direction is found along which an optimal step size is determined (repeat from finding a new search direction)

The One-Dimensional Search

- Recall that at the q-th iteration, the design variable vector is obtained as $X^{q} = X^{q-1} + \alpha S^{q}$
- In the above relation, α is the unknown and hence the objective function F(X) becomes a function of α as F(α)

$$F(\alpha)=F(X^{q-1} + \alpha S^{q})$$

- Two widely used methods
  - polynomial interpolation
  - golden section search

Polynomial Interpolation

- Select an order for the polynomial to fit, let's say a second order polynomial

$$F(\alpha) = a + b\alpha + c\alpha^2$$

- Further, let us assume that we have computed F(α) for α=0,1,2 as F(α=0)=10, F(α=1)=6, and F(α=2)=8
- We have 3 equations in 3 unknowns a, b and c

$$a + b(0) + c(0) = 10$$
$$a + b(1) + c(1) = 6$$
$$a + b(2) + c(4) = 8$$

- From which we solve a=10, b=-7, c=3

Polynomial Interpolation

- The function F(α) the assumes the form

$$F(\alpha) = 10 - 7\alpha + 3\alpha^2$$

- The value of α yielding a minimum of F(α) is obtained from

$$\frac{\partial F}{\partial \alpha} = -7 + 6\alpha = 0$$

$$\alpha^* = 7/6$$

  - this is the optimal step size from polynomial interpolation

Golden-Section Search

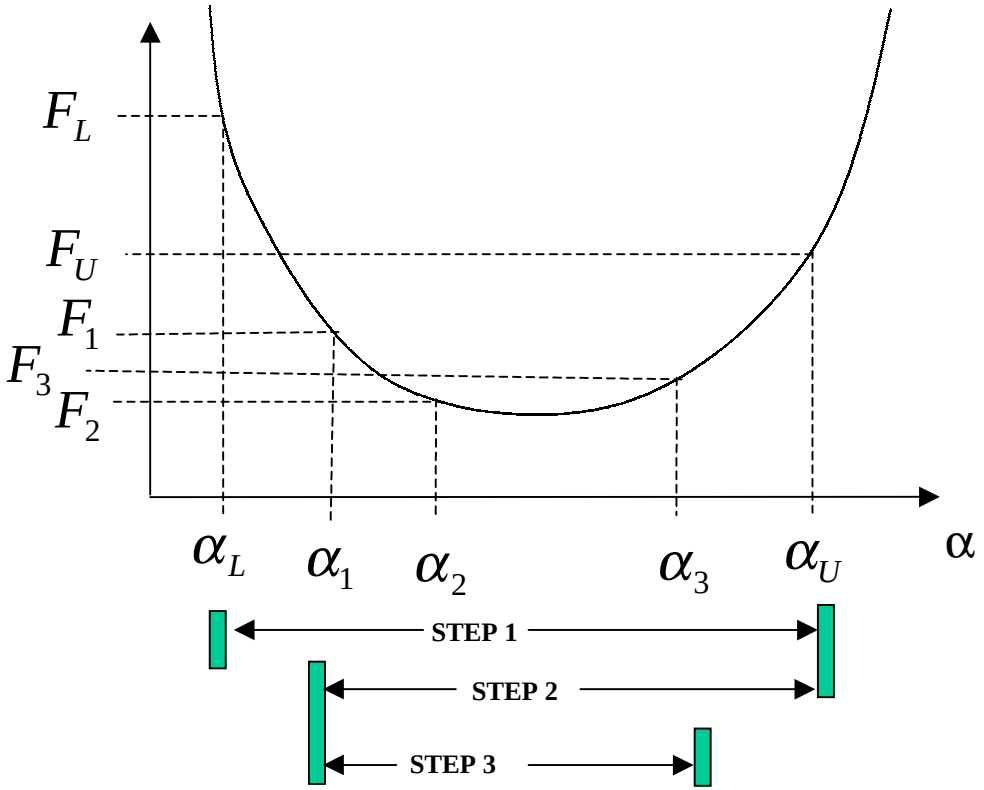
- Based on the golden-section ratio that has been encountered in nature
  - first find bounds on the minimum of a function
  - objective then is to pick two points within the bounded interval so as to shrink the bounds as rapidly as possible about the minimum point

$$\alpha_1 = \alpha_L + \theta(\alpha_U - \alpha_L)$$

$$\alpha_2 = \alpha_U - \theta(\alpha_U - \alpha_L)$$

$$\theta^{-1} = 1.618034 \text{ is the golden section ratio}$$

Golden-Section Search
Graphical Interpretation



Golden Section Search - Example

Ref. Haftka and Gurdal, Structural Optimization

- Find x by golden section that minimizes f(x)=x(x-3) on the interval of search $0 \le x \le 2$

$$x_1 = 0 + 0.382(2) = 0.764 \qquad f(x_1) = -1.708$$

$$x_2 = 2 - 0.382(2) = 1.236 \qquad f(x_2) = -2.180$$

- Since $f(x_2) < f(x_1)$ the new interval for search is $(x_1, 2)$ and the next point is located at

$$x_3 = 2 - 0.382(2 - 0.764) = 1.5278 \qquad f(x_3) = -2.249$$

- Since $f(x_3) < f(x_2)$ we reject the interval (x_1, x_2). The new interval for search is $(x_2, 2)$ and the next point is located at

$$x_4 = 2 - 0.382(2 - 1.236) = 1.7082 \qquad f(x_4) = -2.207$$

Golden Section Search - Example

- Since $f(x_4) < f(x_2) < f(2)$ we reject the interval $(x_4, 2)$. The new interval for search is (x_2, x_4) and the next point is located at

$$x_5 = 1.236 + 0.382(1.7082 - 1.236) = 1.4164 \qquad f(x_5) = -2.243$$

- Show to yourself that the interval to be retained is (x_2, x_4) and the next point is located at

$$x_6 = 1.5967 \qquad f(x_6) = -2.241$$

- The exact solution is located at x*=1.5 and f(x*)=-2.25

Summary - One-Dimensional Search

- Golden section search is easy to implement but is not as efficient
- Polynomial interpolation is efficient - care is required in its implementation
  - work with no higher than cubic polynomial
- implementation of one-dimensional search is perhaps the most difficult aspect of coding an optimization algorithm

Search Direction
Steepest Descent

- Search direction S is taken to be the gradient of the objective function

$$S = -\nabla F(X)$$

- Compute step size in direction S and update design variable vector as

$$X^{q} = X^{q-1} + \alpha S^{q}$$

- Repeat until convergence

- METHOD IS VERY INEFFICIENT AND NOT RECOMMENDED FOR ANY PROBLEM OF PRACTICAL INTEREST

Search Direction
Fletcher-Reeves Conjugate Direction

- For the first step q=1, set the search direction $S = -\nabla F(X)$
- Else define a quantity β as

$$\beta = \frac{\left|\nabla F(X^{q})\right|^{2}}{\left|\nabla F(X^{q-1})\right|^{2}}$$

- Set $S = -\nabla F(X^{q}) + \beta S^{q-1}$ and search in that direction.
- Repeat by updating β and computing new search direction

Fletcher-Reeves Conjugate Direction

- Requires function values and gradients, is easy to implement and requires little storage
- Converges in N or fewer iterations for quadratic problems in N variables - restart with steepest descent every N+1 iterations if progress in objective function decrease slows down
- Simple modification of the steepest descent method speeds up the performance considerably
- Variable metric methods like the DFP and BFGS have similar performance - use gradients to create approximations to the Hessian matrix (quasi-Newton methods)
  - higher storage requirements

Newton's Method

- The oldest second order method for minimizing a n-dimensional function
- The general update equation required is of the form

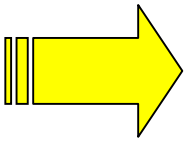
$$X^{q} = X^{q-1} - \alpha Q_{q-1}^{-1} \nabla F(X^{q-1})$$

- Here Q is the Hessian of the objective function, and α is obtained by a 1-D search along the Newton direction
- Q=I is the steepest descent solution
- For a quadratic function, it can be shown that the update relation reaches the optimum solution in one step with α=1

$$X^{*} = X^{0} - [Q(X^{0})]^{-1} \nabla F(X^{0})$$

Constrained Function Minimization

- Optimization algorithms
  - linear programming, feasible useable search directions, generalized reduced gradients
- Optimization strategies
  - sequential unconstrained minimization, sequential linear programming, sequential quadratic programming



Changes in One-Dimensional Search

- Objective and all constraints are approximated as polynomials
  - for objective, step size is chosen to minimize F
  - for constraints, seek step size so that $g_j(\alpha) = o$
  - for all possible step sizes computed in this manner, choose the smallest one
- Several possibilities
  - initially feasible but no active constraints
  - initially feasible and active constraints
  - initially infeasible, active and violated constraints
  - initially infeasible, no feasible solution

Changes in One-Dimensional Search

- Estimating initial step size
  - too large, quality of polynomial fit will be poor
  - too small, many steps required to bracket solution
- Use maximum amount of information
  - if bounds are found with first step, use a linear fit to constraints to estimate step size that will overcome violations in constraints or hit new constraints and use quadratic fit on objective to estimate minimum
  - increase polynomial order as more information is accumulated

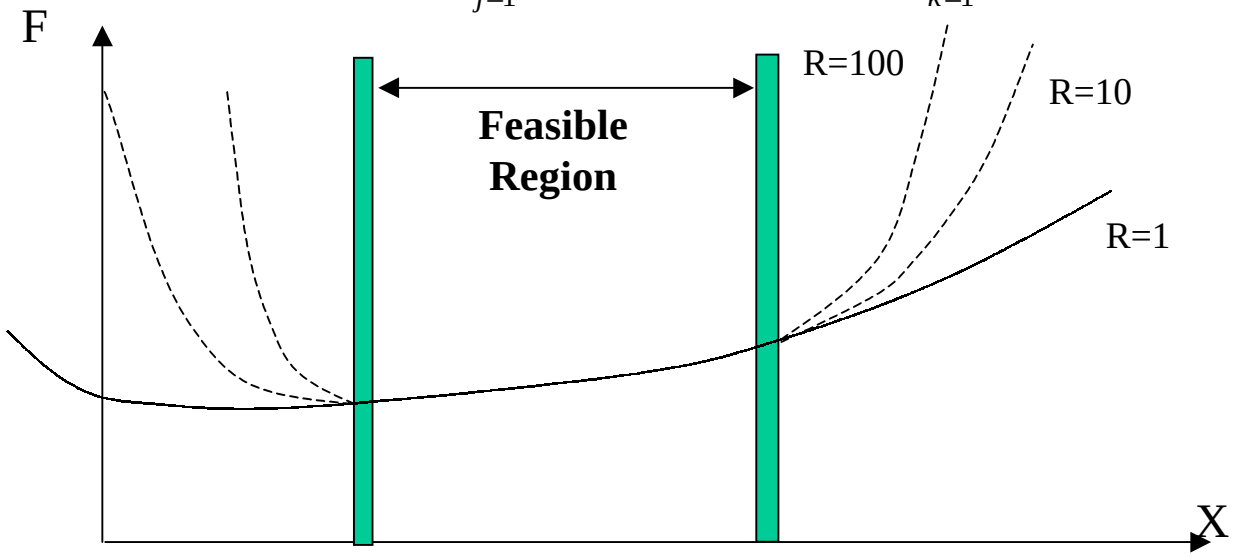
Penalty Function Methods

- Create a set of pseudo-objective functions that will penalize constraint violations
- Use well established unconstrained minimization techniques to minimize the pseudo-objective
  - exterior penalty function
  - interior penalty function and extended interior penalty
  - augmented Lagrange multiplier

Exterior Penalty Function
Single Variable

- Pseudo-objective

$$\overline{F} = F + R\sum_{j=1}^{m} \max[0, g_j(X)]^2 + R\sum_{k=1}^{p} [h_k(X)]^2$$



- Start with small R and increase after each unconstrained minimization

Interior Penalty Function

- Reciprocal function

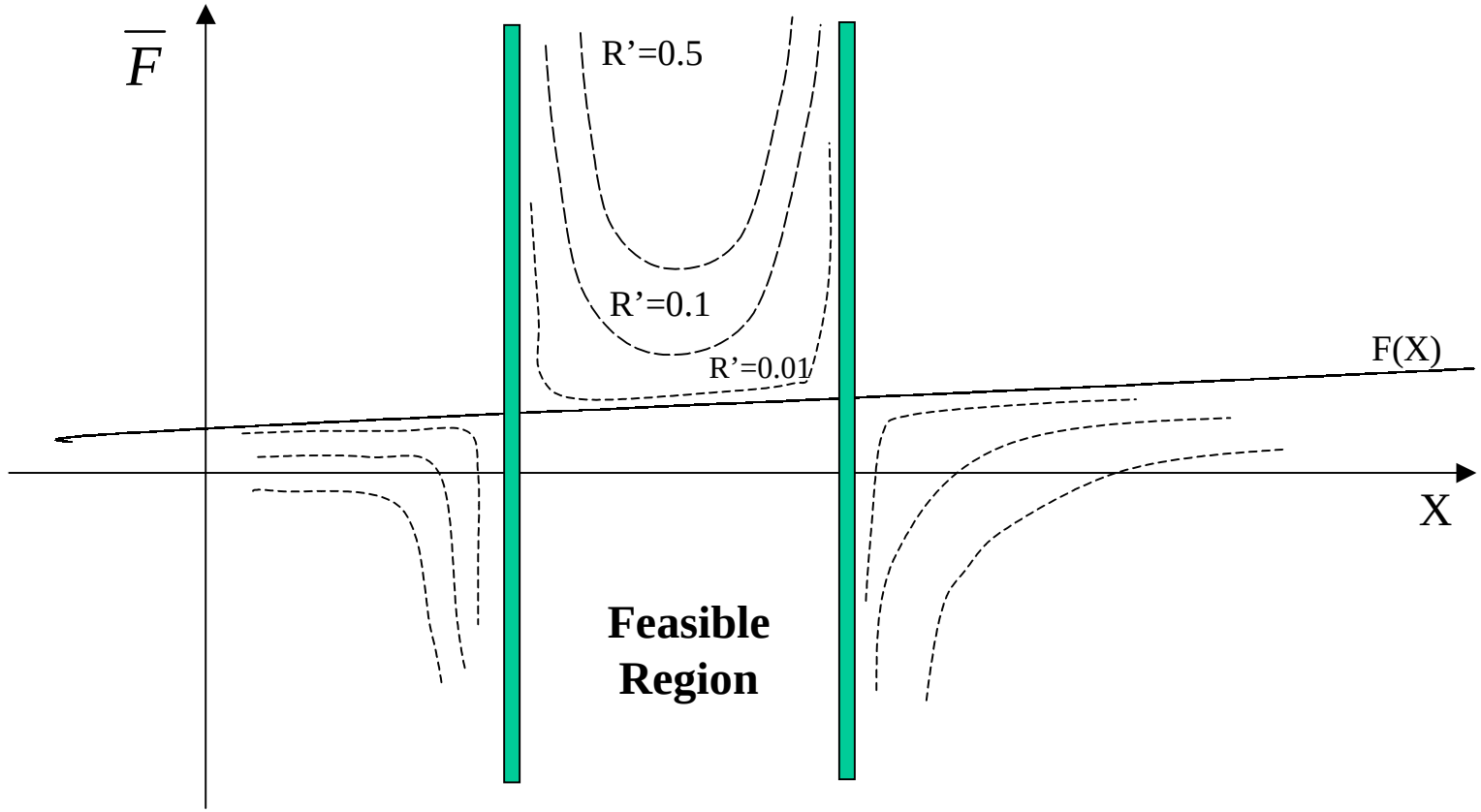
$$\overline{F} = F + R' \sum_{j=1}^{m} \frac{-1}{g_j(X)} + R \sum_{k=1}^{p} [h_k(X)]^2$$

- Log function

$$\overline{F} = F + R' \sum_{j=1}^{m} -Log[-g_j(X)] + R \sum_{k=1}^{p} [h_k(X)]^2$$

Interior Penalty Function

Start with small R' and decrease after each minimization



Summary of SUMT

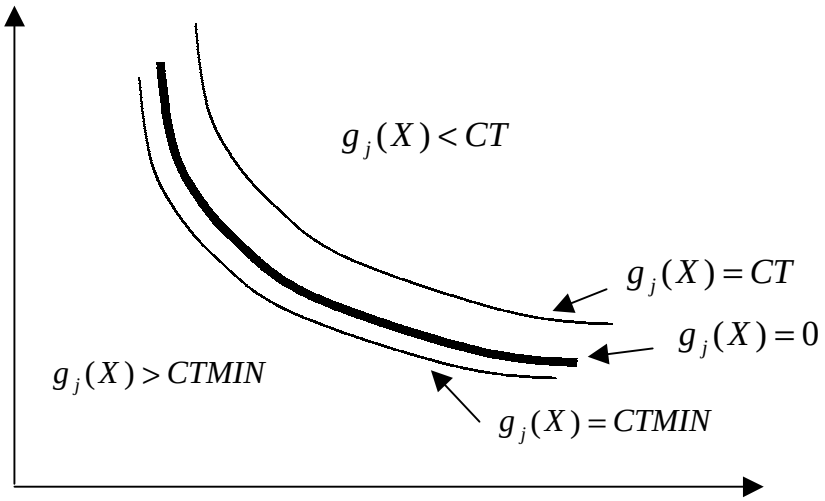
- Large number of function evaluations - computational expense
- Exterior method has best chance of locating true optimum in the presence of multiple relative optima
- Although these were popular 35-40 years ago, are now receiving increased attention in large scale problems
  - with use of approximation methods

Feasible-Useable Search

- Developed in 1960 by Zoutendijk and is coded in two widely used optimizers - CONMIN and ADS
- Is very effective in rapidly finding a near optimal design
- Used only for inequality constrained problems although the modified version does have provision for handling equality constraints
- Method finds a direction that is both feasible and useful, and then does a 1-D search in that direction
  - central to the search direction determination strategy is the concept of determining the active constraints

Active Constraint Set

- Constraint $g_j(X)$ is considered active if $g_j(X) \geq CT$
  - initially CT=-0.05 to “trap” the almost active constraints
    - CT is reduced during optimization until CT= - CTMIN
- $g_j(X)$ is considered violated if



Feasible-Useable Search Search Direction

- If no constraints are active or violated, use the steepest descent direction at the first step and the Fletcher-Reeves conjugate direction thereafter
  - restart with steepest descent every N iterations or whenever progress is slow
- In the presence of active constraints, a sub-problem is solved as follows

$$\textit{Maximize } \beta$$

$$\textit{Subject to}:$$

$$\nabla F(X)^T S^q + \beta \le 0 \quad \text{(useable direction)}$$

$$\nabla g_j(X)^T S^q + \theta_j \beta \le 0 \quad (\ j \in J \text{ - feasible direction})$$

Feasible-Useable Search Search Direction

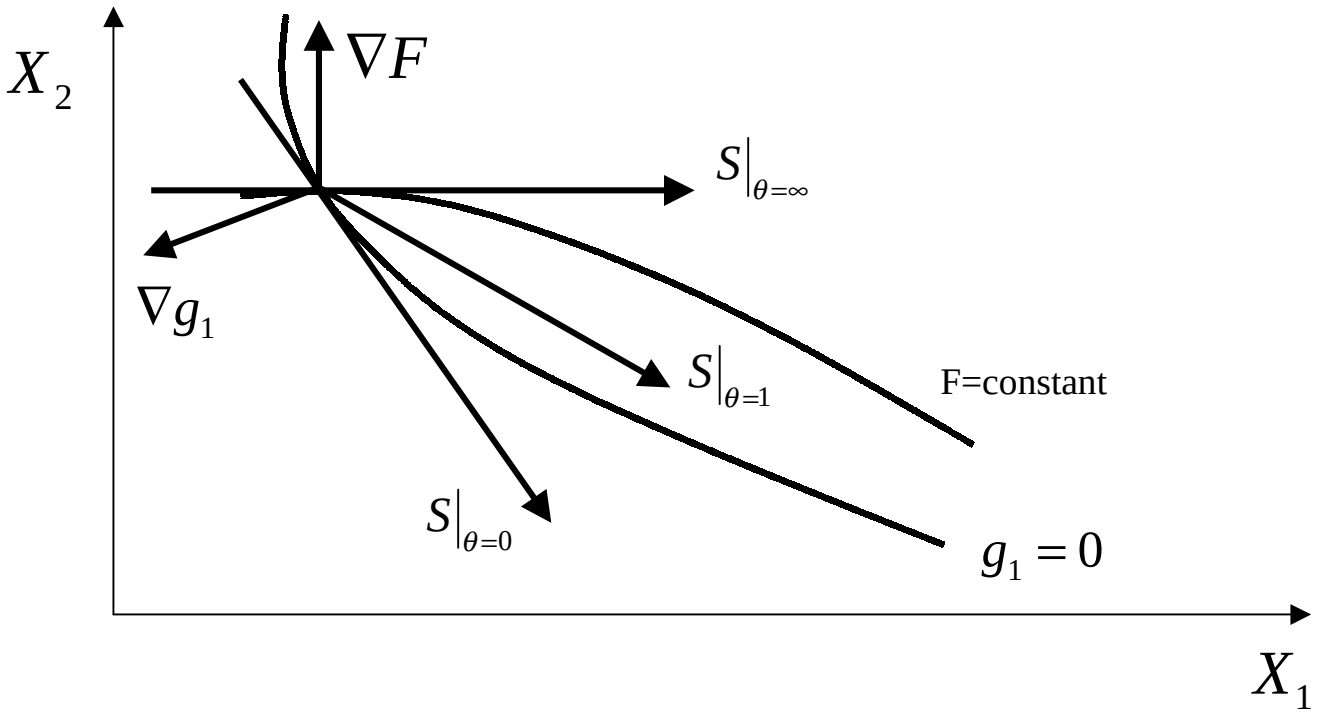
- J is the set of active constraints
- θ_j is the push-off factor and is computed as follows

$$\theta_j = \left[1 - \frac{g_j(X^{q-1})}{CT}\right]^2 \quad \text{for} \quad g_j(X^{q-1}) > CT$$

- The search direction must also be bounded

$$\{S^q\}^T \{S^q\} \leq 1$$

Geometric Interpretation



Sequential Linear Programming

- Linearize both the objective and constraint functions at the current design point
  - impose move limits on the design variables to preserve the integrity of the linear approximations
  - solve the resulting linear programming problem
  - repeat above steps, successively reducing the move limits in the later stages of the optimization process
- Method is very widely used although not favored by “theoreticians”

Sequential Linear Programming

- The linearized equations

$$\overline{F} = F(X_0) + \nabla F(X_0)^T \Delta X$$

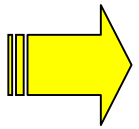
$$\overline{g_j} = g_j(X_0) + \nabla g_j(X_0)^T \Delta X \qquad j = 1, m$$

$$\Delta X = X - X_0$$

- Move limits of +/- 20% are widely used

Sequential Quadratic Programming

- Has become quite popular over the past few years
  - create a quadratic approximation to the Lagrangian
  - create linear approximations to the constraints
  - solve the quadratic problem for the search direction
  - perform one-dimensional search with penalty functions to avoid constraint violations
  - update the approximations
  - cycle to convergence



Sequential Quadratic Programming

- The search direction S is computed from the subproblem

$$Minimize \;\; Q(S) = F(X) + \nabla F(X)^T S + \frac{1}{2} S^T B S$$

$$Subject\ to:$$

$$\nabla g_j(X)^T S + \gamma g_j(X) \le 0$$

$$\nabla h_k(X)^T S + \gamma h_k(X) = 0$$

- γ=0.9 when constraint is violated and zero otherwise - it is used to overcome constraint violations

Sequential Quadratic Programming

- One-dimensional search
  - minimize the exterior penalty function

$$\overline{F} = F + R\sum_{j=1}^{m} \lambda_j \max[0, g_j(X)]^2 + R\sum_{k=1}^{p} \lambda_{k+m} [h_k(X)]^2$$

  - where λ_j are the Lagrange multipliers from the quadratic sub-problem and R is a large penalty multiplier

Discrete Variable Optimization

- Available methods include
  - rounding
    - suboptimal designs or even infeasible designs
  - dual methods
    - limited applications
  - branch and bound
    - correct approach for convex problems
    - very expensive for large scale problems

Branch and Bound Algorithm

- Example problem

$$Minimize \quad F(X) = X_1^{\,2} + X_2^{\,2}$$

$$Subject \;\; to$$

$$g(X) = \frac{1}{X_1} + \frac{1}{X_2} - 2 \le 0 \qquad inequality$$

$$X_1 \ge 0.2 \quad X_2 \ge 0.2 \qquad side$$

$$X_1 \in (0.3, 0.7, 0.9, 1.2, 1.5, 1.8)$$

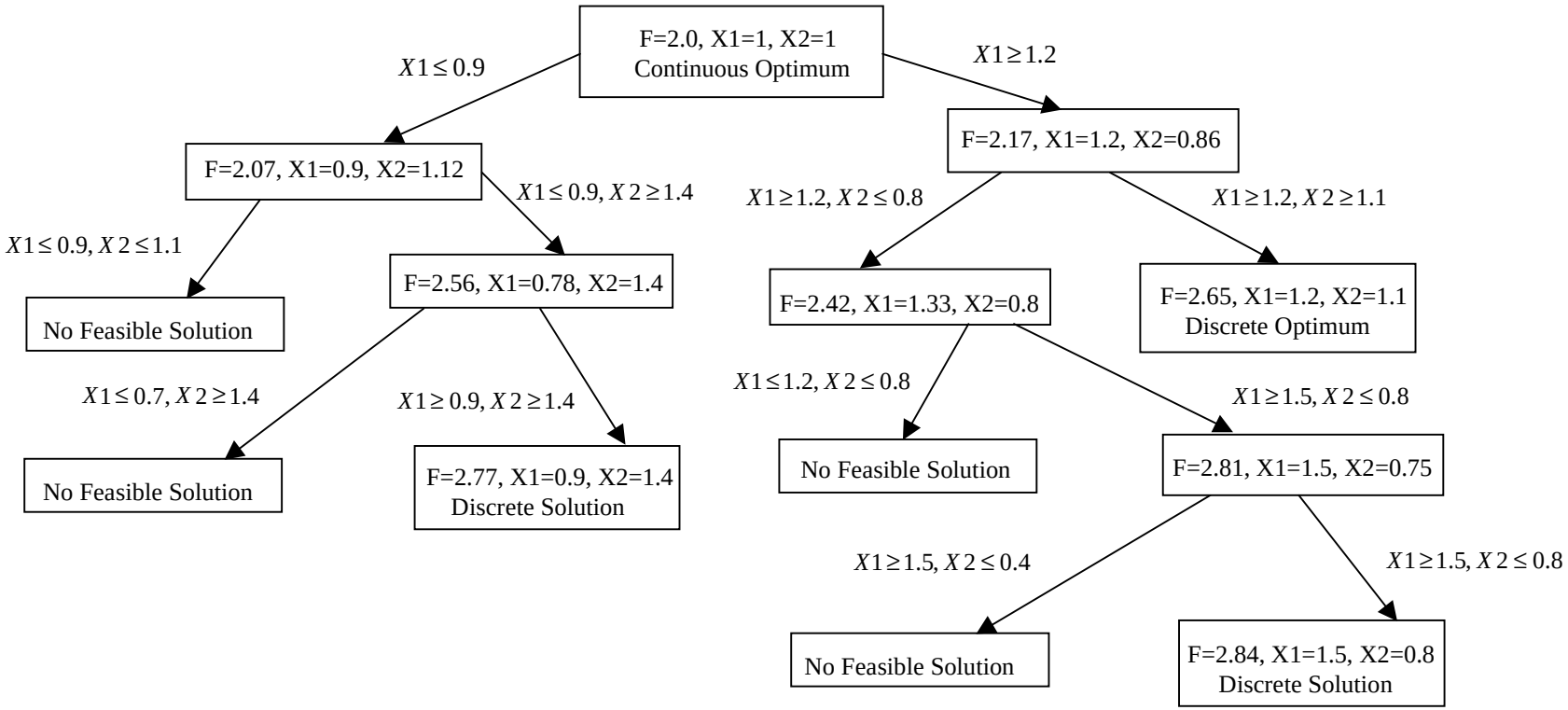
$$X_2 \in (0.4, 0.8, 1.1, 1.4, 1.6)$$

- The continuous solution is F(X)=2.0, X_1=1.0, X_2=1.0

Branch and Bound Algorithm

- Design space may contain a mix of discrete, continuous, or integer variables
- Starting from a continuous solution, start to branch on one variable at a time
  - the continuous solution is a lower bound on the mixed variable problem
  - each branched problem is a continuous optimization problem
  - solution of each branch continues unless no feasible solution is found or the objective function value indicates a “fathomed” path
- Illustrated by solution of previously defined problem

Branch and Bound - Solution



Branch and Bound Solution

- If function values are inexpensive, use actual function evaluations
- For costly function evaluations, use approximations that are of high quality
- Poor quality approximations will yield
  - infeasible discrete solutions
  - non-optimal discrete solutions