LECTURE 3b

Formulation Of The Redesign Problem For Solution By Sequential Linear Programming

(Hughes, Mistre, Zanic)

The elements of a mathematical statement of an optimization or redesign are design variables: constraints and an objective function.

If **X** represents the vector of design variables,

- *N* the number of design variables
- *N1* the number of inequality constraints,
- *NC* the total number of constraints,
- $g_j(\mathbf{X})$ the i-th constraint function,

the redesign problem can be written as follows:

Sequential Linear Programming

Find the design variables **X** which satisfy the constraints:

$$\begin{aligned} g_i(\mathbf{X}) &\geqslant 0; \qquad i = 1, \ldots, N1 \\ g_i(\mathbf{X}) &= 0; \qquad i = N1 + 1, \ldots, NC \end{aligned} \tag{1}$$

such that the objective function:

$$\sum_{j=1}^{N} V(K_j, \mathbf{X})$$

is **optimized**.

In the preceding, K_j represents a cost coefficients.

The constraints can vary from being linear to highly nonlinear.

Sequential Linear Programming

The constraint function *g(X)* is normalized in the form

$$g(\mathbf{X}) = \frac{r(\mathbf{X}) - 1}{r(\mathbf{X}) + 1}$$

where *r(X)* is the structural feasibility ratio defined as

$$r(\mathbf{X}) = \frac{C(\mathbf{X})}{LD(\mathbf{X})}$$

where *L* is the load factor specified by the designer, D(**X**) the demand and C(**X**) the capability of the structure.

For example, in the case of a stress response the structural feasibility ratio *r(X)* is

$$r(\mathbf{X}) = \frac{\sigma_{\text{limit}}(\mathbf{X})}{L\sigma(\mathbf{Q},\mathbf{X})}$$

where the terms are $C = \sigma_{\text{limit}}$ and $D = \sigma\,(\mathbf{Q}, \mathbf{X})$.

Linearization using first-order terms

In the simplest form of Sequential Linear Programming, the objective function and the nonlinear constraints are linearized by using the first-order terms of a Taylor series expansion.
This first-order expansion for the constraint function g(**X**) about a design point $\mathbf{X}^0$ is given by

$$g(\mathbf{X}) \equiv g(X^0) + (X_1 - X_1^0)\left(\frac{\partial g}{\partial X_1}\right)_0 + (X_2 - X_2^0)\left(\frac{\partial g}{\partial X_2}\right)_0 + \ldots\ldots\ldots\ldots + (X_N - X_N^0)\left(\frac{\partial g}{\partial X_N}\right)_0$$

It follows that the linear approximation, using first-order terms only, to the constraint g(**X**) ~ 0 for the design point $\mathbf{X}^0$ is

$$\sum_{j=1}^{N}\left(\frac{\partial g}{\partial X_j}\right)_0 X_j \geqslant -g(\mathbf{X}^0) + \sum_{j=1}^{N}\left(\frac{\partial g}{\partial X_j}\right)_0 X_j^0$$

Sequential Linear Programming

For a total of *NC* constraints the preceding is written as

$$\sum_{j=1}^{N} p_{i,j} X_j \geqslant q_i, \ i = 1, \ldots, NC$$

where

$$p_{i,j} = \left(\frac{\partial g_i}{\partial X_j}\right)_0$$

and

$$q_i = -g_i(\mathbf{X}^0) + \sum_{j=1}^{N} \left(\frac{\partial g_i}{\partial X_j}\right)_0 X_j^0$$

Sequential Linear Programming

Similarly, the linearized form of the objective function is

$$V(X) \equiv \sum_{j=1}^{N} \left(\frac{\partial V}{\partial X_j}\right)_0 X_j + V(\mathbf{X}^0) - \sum_{j=1}^{N} \left(\frac{\partial V}{\partial X_j}\right)_0 X_j^0$$

If

$$m_j = \left(\frac{\partial V}{\partial X_j}\right)_0 \quad \text{and} \quad n = V(\mathbf{X}^0) - \sum_{j=1}^{N} \left(\frac{\partial V}{\partial X_j}\right)_0 X_j^0$$

then

$$V(\mathbf{X}) = \sum_{j=1}^{N} m_j X_j + n$$

- The first term of the preceding equation represents the increase in the objective function if the design is changed, and is referred to as the incremental cost.
- The second term represents the base cost of the current design. The base cost does not affect the solution of the redesign problem because it is constant and therefore need not be considered in formulating the redesign problem.

Sequential Linear Programming

The linearized redesign problem may now be stated as follows:

Find the design variables **X** which minimize the incremental cost .

$$\sum_{j=1}^{N} m_j X_j$$

and **satisfy the linear constraints**

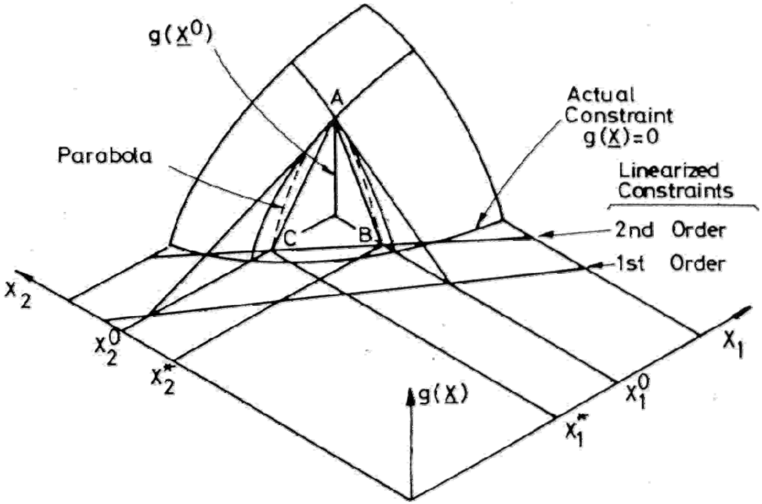
$$\sum_{j=1}^{N} p_{i,j} X_j \geqslant q_i, \; i = 1, \ldots, N1$$

and

$$\sum_{j=1}^{N} p_{i,j} X_j = q_i, \; i = N1 + 1, \ldots, NC$$

Sequential Linear Programming

Figure_1



Linearization using second-order terms

A well-known disadvantage of using a first-order approximation is that if

- ❑ all the derivatives $(\partial g/ \partial X)_o$ are small,

or

- ❑ the constraint function $g(\boldsymbol{X})$ has a high degree of curvature (large values of $\partial^2 g/ \partial X^2$ near $\mathbf{X}^O$),

→ the tangent plane is a poor representation of $g(\boldsymbol{X})$ and

→ the resulting linearized constraints of are far removed from the actual constraints $g_j(\boldsymbol{X}) = 0$.

Sequential Linear Programming

$$g(\mathbf{X}) = g(\mathbf{X}^0) + \sum_{j=1}^{N} (X_j - X_j^0) \left(\frac{\partial g}{\partial X_j}\right)_0 + \frac{1}{2} \sum_{j=1}^{N} \sum_{k=1}^{N} (X_j - X_j^0)(X_k - X_k^0) \left(\frac{\partial^2 g}{\partial X_j \partial X_k}\right)_0$$

Better results would be obtained by retaining the second-order terms of the Taylor series expansion. This, however, introduces many new terms.

- for **first-order linearization** of a constraint which is a function of N design variables, $\boldsymbol{N+1}$ terms need to be computed,
- for a **second-order linearization** $\boldsymbol{N^2}$ additional terms need to be evaluated

Sequential Linear Programming

It has been found quite adequate to retain only the uncoupled, diagonally dominant second-derivative terms. The preceding equation then reduces to

$$g(\mathbf{X}) \equiv g(\mathbf{X}^0) + \sum_{j=1}^{N} (X_j - X_j^0) \left(\frac{\partial g}{\partial X_j}\right)_0 + \frac{1}{2} \sum_{j=1}^{N} (X_j - X_j^0)^2 \left(\frac{\partial^2 g}{\partial X_j{}^2}\right)0 \quad (3)$$

In this case, a **total of *2N* + 1 terms have to be evaluated**.

Retaining only the diagonally dominant second-order terms is equivalent to representing *g(X)* by a parabola in each design variable direction.

Hence, the intercept of the parabola, X_j^*;, constitutes one point of the hyper-plane which approximates *g(X)*.

Sequential Linear Programming

In each direction the gradient of this hyper-plane is the slope of the secant line (for example, AB in Fig. 1). The gradient is therefore

$$\left(\frac{\partial g}{\partial X_j}\right)_0^* = \frac{-g(\mathbf{X}^0)}{(X_j^* - X_j^0)} \qquad (4)$$

From equation (3), for the *j-th* variable direction, the quadratic to be solved in order to determine $X_j - X_j^0$ is

$$\frac{1}{2}\left(\frac{\partial^2 g}{\partial X_j^2}\right)_0 (X_j - X_j^0)^2 + \left(\frac{\partial g}{\partial X_j}\right)(X_j - X_j^0) + g(\mathbf{X}^0) = 0$$

When the quadratic has real roots, the correct root is the one which gives the intercept X_j^* closest to X_j^0 and this always requires that the positive sign be selected in the quadratic formula.

Sequential Linear Programming

Hence

$$X_j^* - X_j^0 = \frac{-\left(\frac{\partial g}{\partial X_j}\right)_0 + \sqrt{\left(\frac{\partial g}{\partial X_j}\right)^2 - 2\left(\frac{\partial^2 g}{\partial X_j^2}\right)_0 g(\mathbf{X}^0)}}{\left(\frac{\partial^2 g}{\partial X_j^2}\right)_0} \qquad (5)$$

From equations (4) and (5) the gradient is given by

$$\left(\frac{\partial g}{\partial X_j}\right)_0^* = \frac{g(X^0)\left(\frac{\partial^2 g}{\partial X_j^2}\right)_0}{\left(\frac{\partial g}{\partial X_j}\right)_0 - \sqrt{\left(\frac{\partial g}{\partial X_j}\right)_0^2 - 2\left(\frac{\partial^2 g}{\partial X_j^2}\right)_0 g(\mathbf{X}^0)}} \qquad (6)$$

When there are no real roots, the constraint is approximated by the tangent at X_j^* so that

$$\left(\frac{\partial g}{\partial X_j}\right)_0^* = \left(\frac{\partial g}{\partial X_j}\right)_0 \qquad (7)$$

Sequential Linear Programming

The absence of a real root in the X_i-direction for a two-variable problem is illustrated in Fig. 2.

For a point $\mathbf{X}^0$ there is only one pair of real roots in the g - X_2 plane.

To circumvent this difficulty, the first-order intercept, Point B, in the g - X_1 plane is used. In the g - X_2 plane, Point C is the intercept determined using the second-order terms. The linearized constraint function

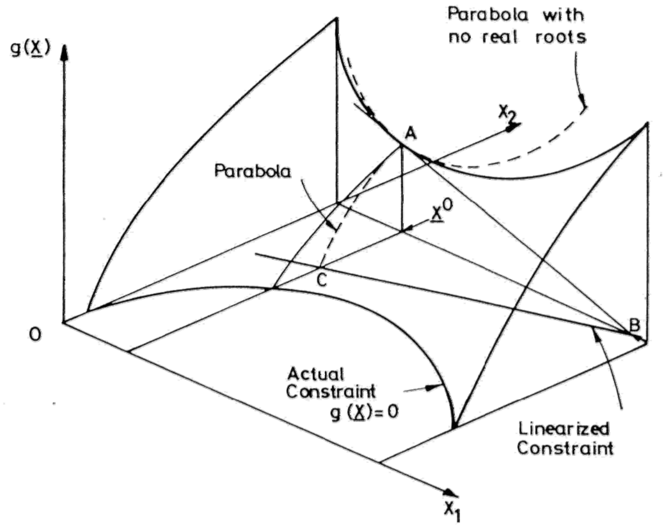


Figure 2

g(X) is in the plane ABC, and in the (X_1,X_2) design space the linearized constraint equation is the line BC.

Sequential Linear Programming

Using the method outlined, each of the nonlinear constraints can be replaced by linear approximations which use second-order terms. For a total of *NC* constraints, the resulting system of linear constraints can reads:

$$\left.\begin{array}{l}\sum_{j=1}^{N}\left(\frac{\partial g_i}{\partial X_j}\right)_0^* X_j > q_i,\ i = 1, \ldots, N1 \\ \sum_{j=1}^{N}\left(\frac{\partial g_i}{\partial X_j}\right)_0^* X_j = q_i,\ i = N1+1, \ldots, NC\end{array}\right\} \quad (8)$$

where

$$q_i = \sum_{j=1}^{N}\left(\frac{\partial g_i}{\partial X_j}\right)_0^* X_j^0 - g_i(\mathbf{X}^0)$$

and $(\partial g/\partial Xj\ )_0{}^*$ is given by equations (6) or (7).

Sequential Linear Programming

- The constraints of equation (8) replace those in the first-order formulation of the linearized optimization problem, given in equation (2).

- The linearized problem is then solved using the standard Dual Revised Simplex Method.

  - Dual : ng >> nv (inversion of nv x nv matrix)
  - Revised : memory efficient (only B^{-1} kept in memory)
  - Dantzig's Simplex algorithm extremely efficient

- The constraints and the objective function are then relinearized and the entire process is repeated until convergence is achieved for the particular strake under consideration.

❑ *Treatment of multi-strake constraints*

- ❑ A strake is an important part of the hull girder, such that collapse of a strake could, for all practical purposes, be considered as collapse of the structure.
- ❑ The error in this assumption is small and it lies on the conservative side.
- ❑ However, the separate treatment of the strakes also requires that the stresses in a particular strake are affected mainly by design changes within that strake.
- ❑ This is true for all secondary stresses and hence any constraints which involve these stresses can be satisfied by making changes within that strake.
- ❑ This, however, is not the case with the hull girder bending stress, $\sigma 1$.

Treatment of multi-strake constraints

Since this **stress depends on the overall section modulus**, and hence on all strakes in differing degrees, there are two problems which must be overcome when using a strake-by-strake approach:

(i) A **change of design variables in a particular strake** will influence σ1 in that strake.
(If this effect is not accounted for during the strake redesign, but is delayed until the next finite-element analysis, then convergence will be slow).

(ii) If a constraint is violated because **σ1 is too large**, the redesign should **not be restricted to only that strake** because this would cause disproportionate member sizes and would probably prevent convergence. The optimal solution requires changes to several strakes, particularly the strakes in the strength deck and bottom.

Inclusion of stress derivative term for σ1

The effect which a change of scantlings of a particular strake has on the hull girder stress in that strake is accounted for by including the derivative of σ1 with respect to the design variables in the formulation of the linearized constraint functions.

Optimization method obtains the linearized form of a general nonlinear constraint function $g(\mathbf{X})$ in terms of the derivative $(\partial g/\partial X_j)_0$*.

The computation of this derivative uses the current values of **X** and of σ1, thus treating σ1 as if it were constant.

The interaction between σ1 and **X** is then accounted for by adding a term containing *(∂σ1 /∂X_j)$_0$*:*

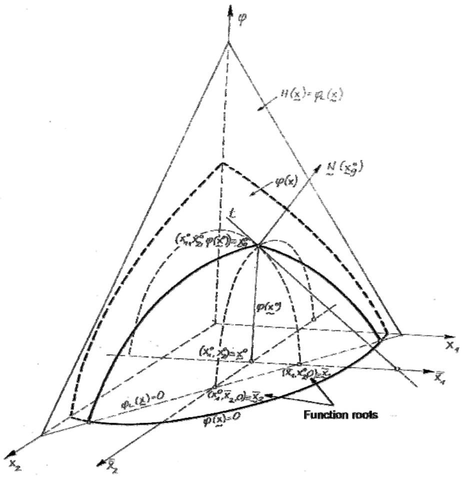
$$\left(\frac{\partial g_i}{\partial X_j}\right)_0^* = \left(\frac{\partial g_i}{\partial X_j}\right)_{0,\sigma_1}^* + \frac{\partial g_i}{\partial \sigma_1}\frac{\partial \sigma_1}{\partial X_j}$$

A rapid and simple algorithm for calculating the stress derivative *(∂σ1/∂X_j)$_0$**is used.

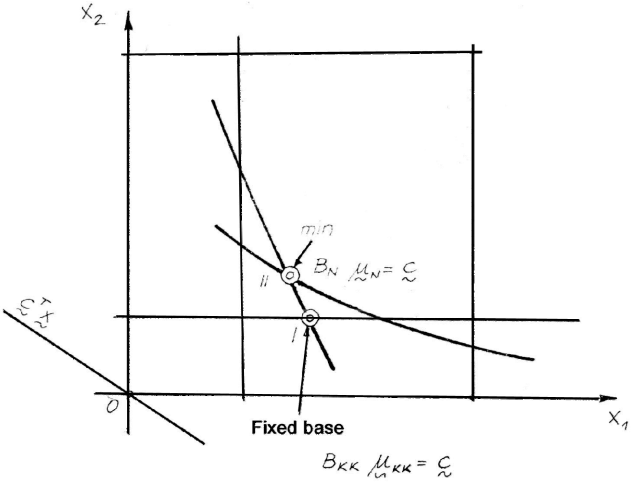
An expression for *(∂g/∂X_j)$_0$** for each constraint may be obtained by simply differentiating the expression which defines *g(**X**)* for that constraint, treating all terms other than σ1 as constant.

Practical considerations in calculation

(a) Parabolization



(b) Dual logical base



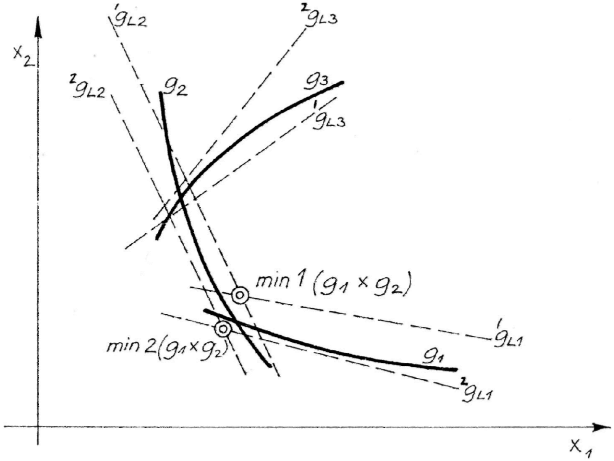
x_2
x_1
0
$\underset{\sim}{c}^T \underset{\sim}{x}$
min
$B_N \ \underset{\sim}{\mu}_N = \underset{\sim}{c}$
II
I
Fixed base
$B_{KK} \ \underset{\sim}{\mu}_{KK} = \underset{\sim}{c}$

Matrix update : $B_N^{-1} = P^1 \ldots P^N B_S^{-1}$;

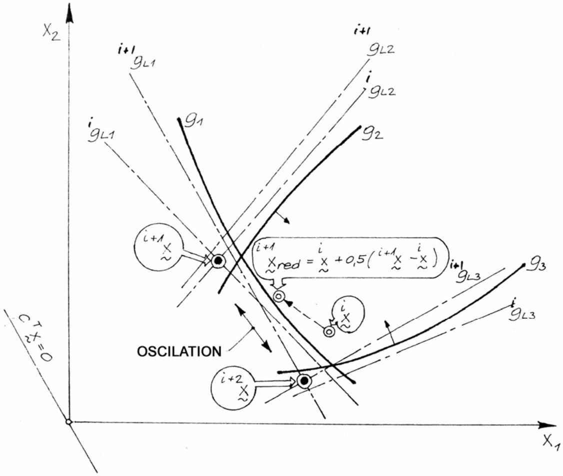
(c) Convergence of the procedure

- Convergence is tested for feasibility of all constraints.
- Convergence of design variables is tested wrt. specified tolerances.
- Objective function is tested for increase/decrease in value or convergence.

Convergence of the procedure



(d) Oscilation


X_2
X_1
$\overset{i+1}{g}_{L1}$
$\overset{i}{g}_{L1}$
$\overset{i+1}{g}_{L2}$
$\overset{i}{g}_{L2}$
g_1
g_2
g_3
$\overset{i+1}{g}_{L3}$
$\overset{i}{g}_{L3}$
$\overset{i+1}{\underset{\sim}{X}}$
$\overset{i+1}{\underset{\sim}{X}}_{red} = \overset{i}{\underset{\sim}{x}} + 0{,}5(\overset{i+1}{\underset{\sim}{x}} - \overset{i}{\underset{\sim}{x}})$
$\overset{i}{\underset{\sim}{x}}$
OSCILATION
$\overset{i+2}{\underset{\sim}{X}}$
$\underset{\sim}{c}^T \underset{\sim}{x} = 0$

Reduced move approach to oscillation

$$x_i^{red} = R_{ii}\,(x_{i+1}-x_i) + x_i$$

where $R_{ii} = 0.5 - 0.7$ and $R_{ij} = 0$.

Method of reduced move :

- slows down the process (no. of iterations increases)
- blocks large jumps in design space based upon bad linearization
- necessary only in final cycles of the process.

(e) Linearization problems

Five unacceptable cases in the final linearized form

$$\mathbf{g}_L(\mathbf{x}) \geq 0 \rightarrow \mathbf{a}^T \mathbf{x} \geq \mathbf{b}$$

are:

1. $\mathbf{a} < \varepsilon \equiv \{(10^{-7})\}$ which yields $0 \geq b$
   - constraint unnecessary, if $b < 0$;
   - constraint is infeasible if $b > 0$.

2. if every a_i is of the same sign and $b = 0$, due to $\mathbf{x} \geq \mathbf{0}$:
   - constraint is unnecessary for $\mathbf{a} \geq \mathbf{0}$;
   - constraint is infeasible for $\mathbf{a} < \mathbf{0}$.

(e) Linearization problems

3. if every $a_i \leq 0$ and $b > 0$
   - constraint is infeasible.

4. if every $a_i \geq 0$ and $b < 0$
   - constraint is removed.

5. if constraint contour $g^{\alpha}_{jk}(\mathbf{x}^{\alpha}) = 0$ lies outside of the domain restricted by the **min-max constraints S_1**:

   - constraint is unfeasible if the domain S_1 doesn't lie within the S_3 half-space defined by the constraint $g^{\alpha}_{jk}(\mathbf{x}^{\alpha}) \geq 0$:

   $$S_1 \cap S_3 = 0 \text{ (empty set)};$$

   - constraint is unnecessary if S_1 is subset of S_3.

B Decomposition of structure on substructures Multilevel Systems and Decomposition

Successful design of **large, complex systems** invariably involves **decomposition of the system** into a number of smaller subsystems each with its own goals and constraints.

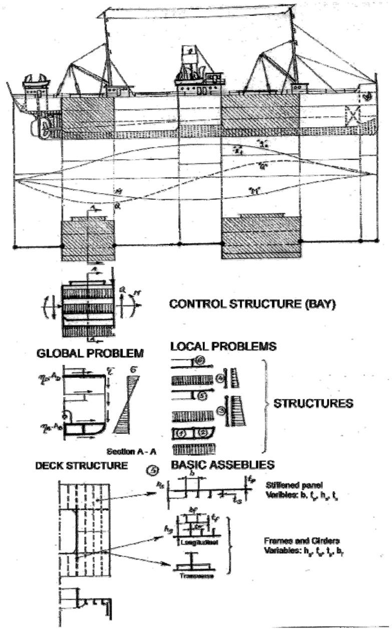
The resulting interconnection of subsystems may take on many forms, but one of the most common is the **hierarchical form** in which a given-level **unit controls or coordinates the units** on the level below it and in turn is controlled by the units on the level above it.

Multilevel Systems and Decomposition

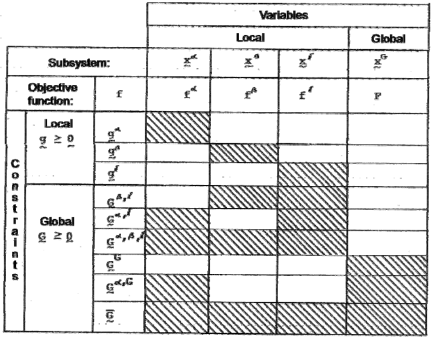
Of particular importance is the consideration of whether or not such an arrangement is capable of

acting in an overall system-optimal manner

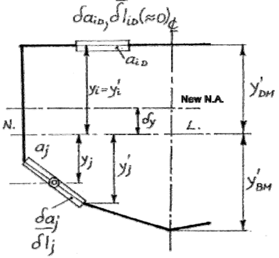
and if so, how to ensure that all units, acting according to their own goals, will somehow achieve this overall goal**.**


CONTROL STRUCTURE (BAY)
GLOBAL PROBLEM
LOCAL PROBLEMS
STRUCTURES
Section A - A
DECK STRUCTURE
BASIC ASSEBLIES
Stiffened panel
Varibles: b, t_p, h_s, t_s
Longitudinal
Transverse
Frames and Girders
Variables: h_g, t_w, t_f, b_f

Multilevel Systems and Decomposition



			Variables			
			Local			Global
Subsystem:			$\underset{\sim}{x}^{\alpha}$	$\underset{\sim}{x}^{\beta}$	$\underset{\sim}{x}^{\gamma}$	$\underset{\sim}{x}^{G}$
Objective function:		f	f^{α}	f^{β}	f^{γ}	F
Constraints	Local $\underset{\sim}{g} \geq \underset{\sim}{0}$	$\underset{\sim}{g}^{\alpha}$	▨			
		$\underset{\sim}{g}^{\beta}$		▨		
		$\underset{\sim}{g}^{\gamma}$			▨	
	Global $\underset{\sim}{G} \geq \underset{\sim}{0}$	$\underset{\sim}{G}^{\beta,\gamma}$		▨	▨	
		$\underset{\sim}{G}^{\alpha,\gamma}$	▨		▨	
		$\underset{\sim}{G}^{\alpha,\beta,\gamma}$	▨	▨	▨	
		$\underset{\sim}{G}^{G}$				▨
		$\underset{\sim}{G}^{\alpha,G}$	▨			▨
		$\underset{\sim}{\overline{G}}$	▨	▨	▨	▨



Multilevel Systems and Decomposition

Unfortunately, it is clear that an **integrated optimization problem** involving many subsystems and variables **cannot be decomposed** or partitioned into **independent sub-problems** which can be independently optimized.

Optimizing a single subsystem in a large system **without regard to the effects of interactions** can lead to such degraded performance in other subsystems that **overall process performance is worse than without any optimization.**

Multilevel Systems and Decomposition

The objective is (J. D. Schoeffler) to introduce **multilevel systems which do permit decomposition** of static optimization problems into **independent sub-problems** each of which when solved independently yields the overall system optimum:

- ❑ ensure that certain variables or parameters in the sub-problems called ***coordinating variables*** are free to be manipulated by a second-level controller

- ❑ choose the coordinating variables in such a way that the independent first-level systems are forced to choose solutions which in fact correspond to an overall system optimum.

Multilevel Systems and Decomposition

The decentralization of the static optimization problem is carried *out* in a two-step sequence.

- First, the integrated problem (objective function and constraints) is converted to a two-level or multilevel form with separate and distinct tasks assigned to the levels.

- Second, those parts of the first-level task or problem which do not interact with other parts are split apart, forming a decomposition or partition of the first-level problem.

Multilevel Systems and Decomposition

.

Transformation of a given constrained optimization problem into a multilevel problem. Two different approaches:

→ ***model-coordination*** *method* (or the *feasible method)*

→ ***goal-coordination*** *method* (or the *dual-feasible method)*.

The Model-coordination Method

Consider the simple integrated optimization problem associated with the system with two coupled subsystems. The variables in the system are defined below:

$\mathbf{m}$	vector of free variables
$\mathbf{m}^i$	vector of free variables for subsystem i
$\mathbf{x}^1$	vector of interaction variables from subsystem 1 to subsystem 2
$\mathbf{x}^2$	vector of interaction variables from subsystem 2 to subsystem 1

The Model-coordination Method

Let the system constraints set be

$$\mathbf{G(m, x)} > \mathbf{0}$$

where the vector **G** contains the system constraints for each subsystem:

$$\mathbf{G}_i(\mathbf{m}^i, \mathbf{x}^1, \mathbf{x}^2) > 0 \qquad i = 1,2$$

Let the objective function (performance function) f ≡ P, which is to be minimized, reads:

$$P(\boldsymbol{m}, \mathbf{x}) = P_1(\mathbf{m}^1, \mathbf{x}^1) + P_2(\boldsymbol{m}^2, \mathbf{x}^2)$$

The Model-coordination Method

The minimization is to take place over all allowable **m**, and **x** which satisfy the system equations:

$$\min_{\mathbf{m},\,\mathbf{x}} P(\mathbf{m}, \mathbf{x})$$

$$\text{subject to } \mathbf{G}(\mathbf{m}, \mathbf{x}) > 0$$

Depending on the problem, the set of allowable variables may be finite or infinite, real or integer, constrained or unconstrained.

The Model-coordination Method

Although the performance function may be separated into two non-interacting functions, one for each subsystem, there is interaction because of the interaction variables x which affect both subsystems.

The **model-coordination method** converts this integrated optimization problem into a two-level problem by ***fixing the interaction variables.*** That is, fix the interaction variables at some value, say **z**:

constrain $\mathbf{x} = \mathbf{z}$

Then under these conditions, the integrated prob1em may be split into a first-level problem and a second-level problem.

The Model-coordination Method

FIRST-LEVEL PROBLEM:

$$\text{determine } H(\mathbf{z}) = \min_{\mathbf{m}} P(\mathbf{m}, \mathbf{z})$$

$$\text{subject to } \mathbf{G}(\mathbf{m}, \mathbf{z}) > 0$$

SECOND-LEVEL PROBLEM:

$$\min_{\mathbf{z}} H(\mathbf{z})$$

The Model-coordination Method

If

$$S_0 = \{\mathbf{m} \mid \mathbf{G}(\mathbf{m}, \mathbf{z}) > 0\}$$

then the minimization of P is over the set S_0 . Notice that this set may be empty in some problems for some choices of **z**. Define

$$S_1 = \{\mathbf{z} \mid H(\mathbf{z}) \text{ exists}\}$$

That is, S_1 is the set of all **z** such that the system equations may be satisfied for this value of **z** and the minimum of the objective function exists (is finite).

If the original integrated optimization problem has a solution, then S_1 is not empty for it contains (at least) the point $\mathbf{z} = \mathbf{x}_{opt}$, that is, the optimal values for the interaction variables.

The Model-coordination Method

The second level produces an estimate $\mathbf{z}$ of $\mathbf{x}_{opt}$, the optimal values for the interaction variables, and transmits this estimate to the first-level unit which determines the optimal values for $\mathbf{m}$ assuming that $\mathbf{x} = \mathbf{z}$.

The first-level unit transmits the values for $\mathbf{m}$ to the second-level unit which then produces a better estimate for the interaction variables, etc.

Thus the solution of the optimization problem proceeds in an **iterative fashion** with the effort divided between the first- and second-level units.

The Model-coordination Method

Multilevel form of the problem leads directly to a method of decomposition of the first-level problem because the first-level problem partitions immediately into two independent problems.

First*-level problem for subsystem *i (i = 1, 2):

$$H(\mathbf{z}) = \min_{\mathbf{m}^{\mathbf{i}}} P_i(\mathbf{m}^{\mathbf{i}}, \mathbf{z}^{\mathbf{i}})$$

$$\text{subject to } \mathbf{G}_i\,(\mathbf{m}^{\mathbf{i}}, \mathbf{z}^{\mathbf{l}}, \mathbf{z}^{\mathbf{2}}) > 0$$

The second-level problem is to choose **z** so as to

$$\text{minimize } H(\mathbf{z}) = H_1(\mathbf{z}) + H_2(\mathbf{z}).$$

The Model-coordination Method

The set of variables which are fixed are termed the ***coordinating variables***.

The ***term*** **model coordination** arises from the fact that decomposition is made possible by **adding a constraint** to the mathematical models of the system, namely, that certain internal interacting variables be fixed.

The alternate name, the **feasible method**, arises *from* the fact that throughout the iterative calculation, **all intermediate values *for* the variables m and x are feasible or allowed**.

The Goal-coordination Method

In goal-coordination method literally removes the interactions by "cutting" all links between subsystems.

The outputs of the subsystem which are inputs to another subsystem are labeled $\mathbf{x}^i$ as before, but the corresponding inputs to the subsystems are now labeled $\mathbf{z}^i$ and the meaning of the word "cut" is literal; that is, $\mathbf{z}^i$ and $\mathbf{x}^i$ need not be equal.

Moreover, the $\mathbf{z}^{\mathrm{i}}$ now act like arbitrary variables and must be selected, like **m** and **x**, by the optimizing subsystems.

This of course decouples the *two* subsystems completely and since the objective function was already decoupled, it is clear that there is no interaction in the system at all.

However, in order to ensure that the independent subsystem problems yield the overall system optimality, it is necessary to ensure that the *interaction principle* be satisfied, namely, that the **independently selected $\mathbf{x}^i$ and $\mathbf{z}^i$ actually be equal.** Consider the addition of a penalty term which penalizes the performance of the system if interactions do not balance:

$$P(\mathbf{m}, \mathbf{x}, \mathbf{z}, \lambda) = P_1(\mathbf{m}^1, \mathbf{x}^1) + P_2(\mathbf{m}^2, \mathbf{x}^2) + \lambda^T(\mathbf{x} - \mathbf{z})$$

where λ is a vector of weighting parameters (positive or negative as needed) which causes any interaction unbalance **$(\mathbf{x} - \mathbf{z})$** to affect performance.

$$\mathbf{G}_1(\mathbf{m}^1, \mathbf{x}^1, \mathbf{z}^2) > 0$$
$$\mathbf{G}_2(\mathbf{m}^2, \mathbf{x}^2, \mathbf{z}^1) > 0$$

or

$$S_0 = \{(\mathbf{m}, \mathbf{x}, \mathbf{z}) \mid \mathbf{G}_1 > 0; \mathbf{G}_2 > 0\}$$

The Goal-coordination Method

Minimizing the objective function (with penalty *term)* over the set of allowable system variables results in a function of λ:

$$H(\lambda) = \min_{(\mathbf{m},\mathbf{x},\mathbf{z})\ \lambda \text{ e } S} P(\boldsymbol{m}, \mathbf{x}, \mathbf{z}, \lambda)$$

Define the set S_1 as the domain of $H(\lambda)$:

$$S_1 = \{\lambda \mid H(\lambda) \text{ exists}\}$$

That is, S_1 is the set of all λ such that the minimum of P exists. Assuming there exists some vector λ such that solving the above optimization problem with the penalty term results in the constraints are satisfied, the set S_1 is not empty.

First expand the penalty term into

$$\boldsymbol{\lambda}^{\mathbf{T}}(\mathbf{x} - \mathbf{z}) = \boldsymbol{\lambda}^{\mathbf{T}}(\mathbf{x}^1 - \mathbf{z}^1) + \boldsymbol{\lambda}^{\mathbf{T}}(\mathbf{x}^2 - \mathbf{z}^2)$$

The Goal-coordination Method

Then the first-level problems separate into
SUBSYSTEM 1:

$$\min_{\mathbf{m}^1,\, \mathbf{x}^1,\, \mathbf{z}^2} P_1(\mathbf{m}^1, \mathbf{x}^1,) + \boldsymbol{\lambda}_1^T\mathbf{x}^1 - \boldsymbol{\lambda}_2^T\mathbf{z}^2$$

$$\text{subject to } \mathbf{G_1}(\mathbf{m}^1, \mathbf{x}^1, \mathbf{z}^2) > 0$$

SUBSYSTEM 2:

$$\min_{\mathbf{m}^2,\, \mathbf{x}^2,\, \mathbf{z}^1} P_2(\mathbf{m}^2, \mathbf{x}^2,) + \boldsymbol{\lambda}_1^T\mathbf{z}^1 - \boldsymbol{\lambda}_2^T\mathbf{x}^2$$

$$\text{subject to } \mathbf{G_2}(\mathbf{m}^2, \mathbf{x}^2, \mathbf{z}^1) > 0$$

Notice that the goals of the individual subsystems have been modified in that the coordinating variables λ enter into each subsystem goa1.

The Goal-coordination Method

Since the objective of the second-level unit is to choose λ, it is clear that the coordination of the independent first-level problems is taking place by manipulating the first-level problem objectives, thus the term goal coordination.

The alternate term, dual-feasible method, arises from further consideration of the function $H(\lambda)$. Whether the coordinating parameter λ is considered to be a penalty weight or a Lagrange multiplier is immateria1.

The Goal-coordination Method

FIRST-LEVEL PROBLEM:

Determine $H(\lambda)$ by minimizing $P(\boldsymbol{m}, \mathbf{x}, \lambda)$ over the set S_0 .

SECOND-LEVEL PROBLEM:

Choose λ such that solution to the first-level problem results in satisfaction of the interaction principle; that is, $\mathbf{x} = \mathbf{z}$.

From a Lagrange multiplier point of view, the penalty term may be viewed as a constraint (that the interactions be in balance) and the conditions under which a λ exists which solves the integrated optimization problem may be determined.

Multilevel Formulation of Optimization Problems

The integrated optimization problem is formulated by including the constraints through the use of either Lagrange multipliers or penalty terms, conversion to a multilevel form is then done by

1. Choosing a set of coordinating variables (actual variables, Lagrange multipliers, penalty weights, or any combination of these) and assuming these to be *fixed,* thereby producing a first-level optimization problem with certain fixed variables
2. Assigning to the second level the task of determining the optimal values of the coordinating variables
3. Deriving an algorithm by which the second-level unit may successively **improve its estimate of the optimal interaction** variables

Multilevel Formulation of Optimization Problems

The choice of coordinating variables is made on the basis of decomposition of the first-level problem.

That is, a **set of coordinating variables must be selected** so that the resulting first-level problem, which is the original problem with the coordinating variables assumed fixed, **may be decomposed into independent sub-problems.**

From an efficiency point of view, it is desirable that the solutions to the independent **first-level problems be very simple** since in the iterative solution they are solved many times.