# A sequential approximation method using neural network for engineering design optimization

元智大學機研所博士生　王 樹 根

指導教授：徐業良博士

# Agenda

- Introduction
- A review on non-linear discrete optimization
- The sequential neural network approximation method
- Design examples using the SNA method

  (1) Verify quality of solution

  (2) Test in large discrete design spaces (up to 10 design variables)
- Engineering optimization problems using the SNA method

  (1) Demonstrate the practicality of the SNA method in different applications

  (2) Zooming strategy
- Conclusions and Discussion

# Introduction

# Introduction (1)

❖ **Characteristics of engineering optimization problems**

- Non-linear
- Discrete physical quantities of design variables using standard sizes
- Subject to implicit constraint
- Critical constraint in "pass-fail" binary type

❖ **Mathematic Model**

$$\min.\ f(\mathbf{x})$$

$$\text{s.t.}\quad \mathbf{g_e}(\mathbf{x}) \le 0$$

$$\mathbf{g_i}(\mathbf{x}) \le 0$$

$$\mathbf{g_b}(\mathbf{x}) = 1$$

$$x_i \in X_i = \{d_{i1}, d_{i2}, \ldots, d_{il}\}, i \in I_d$$

# Introduction (2)

- This dissertation presents a sequential approximation method using neural network (the SNA method) specifically for engineering optimization problems with discrete design variables, implicit constraints, in particular “pass-fail” type constraints.

# A review on non-linear discrete optimization

# A review on non-linear discrete optimization (1)

- Arora et al. [1994] classified numerical methods for mixed-discrete non-linear programming into 6 types:

  (1) Branch and Bound [Sandgren, 1990]

  (2) Simulated Annealing [Balling, 1991]

  (3) Sequential Linearization [Loh and Papalambros,1991]

  (4) Penalty Function Approach [Shin, 1990]

  (5) Lagrangian Relaxation [Johnson and Larsson,1990]

  (6) Other methods: Rounding Approach [Arora,1989], Genetic Algorithm [Lin and Hajela,1992].

# A review on non-linear discrete optimization (2)

- Among these methods, the branch and bound method, simulated annealing, and genetic algorithm are suitable implementations for problem with non-differentiable functions.

- But these methods require many function evaluations, which may not be suitable for engineering optimization problems with implicit constraints.

# Sequential neural network approximation (SNA) method

❖**Mathematic Model ($M_{real}$)**

$$\min.\ f(\mathbf{x})$$

$$\text{s.t.}\quad \mathbf{g}_e(\mathbf{x}) \le 0$$

$$\mathbf{g}_i(\mathbf{x}) \le 0$$

$$\mathbf{g}_b(\mathbf{x}) = 1$$

$$x_i \in X_i = \{d_{i1}, d_{i2}, \ldots, d_{il}\}, i \in I_d$$

❖**Approximated Model ($M_{NN}$)**

$$\min.\ f(\mathbf{x})$$

$$\text{s.t.}\quad \mathbf{g}_e(\mathbf{x}) \le 0$$

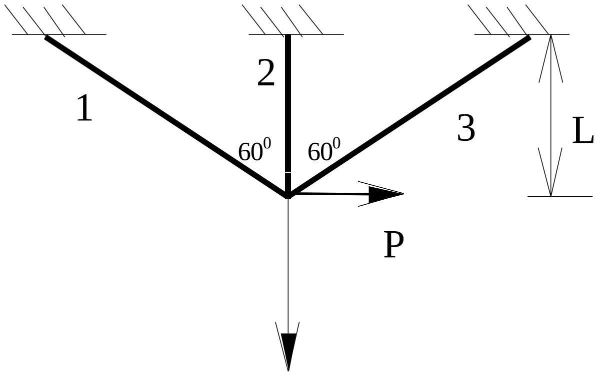
$$NN(\mathbf{x}) = 1$$

# Using Neural Network (NN) in optimization

- Hopfield and Tank [1985] simulates problem of minimizing an energy function formulated by design restraints.
- Watta [1996] proposed energy function with an objective function plus a penalty function to enforce the restraints by two interacting recurrent network.
- Lee and Chen [1991] used NN to simulate the constraint functions
- Renaud et al. [1994] use NN to approximate discrete system space and repeatedly evaluate the network.
- Instead of simulating the function values of objective functions and constraints, the SNA method developed in this research only simulates the feasible domain of the optimization problem (0-1).

# The sequential neural network approximation method

# A 3-bar-truss example to describe the SNA Algorithm




Haftka and Gurdal [1992]

min. $W = 7.85(4A_1 + A_2)$ (minimize weight)

s.t. $g_1 : \dfrac{200}{(A_1 + 4A_2)} + \dfrac{25\sqrt{3}}{3A_1} \le 1$

(both max. stress limit)

$g_2 : \dfrac{800}{(A_1 + 4A_2)} \le 1$ , $i = 1, 2$

$A_i \in \{1,4,9,\ldots,289\}$

# Design point representation and initial training data set

•A back-propagation neural network is trained to simulated a rough map of the feasible domain formed by the constraints using a few representative training data.

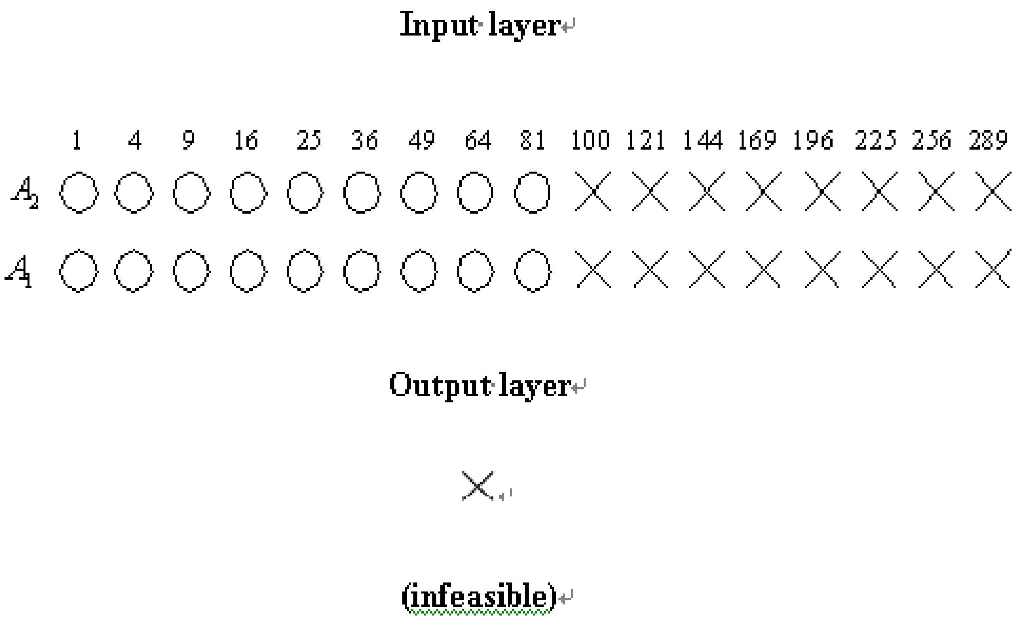
•A training data consists of a discrete design point and whether this design point is feasible or infeasible. Function values of the constraints are not required.

# Point Representation with Neurons in NN

- Architecture of a 3-layer NN: input, hidden, and output layer.

- Each neuron with a circle represents “1” while a cross represents “0 ”.

- Neurons of input layer depend on no. of variables and no. of discrete values of each variable. 12 neurons are used in the hidden layer. A single neuron in output layer is used to represent whether this design point is feasible or infeasible.




Representation of an infeasible training point {(81, 81), infeasible}.

# Initial Training Data Set

❖ Orthogonal arrays are adopted to define reasonably the set of initial training data because they are geometrically balanced in coverage of the whole design domain.

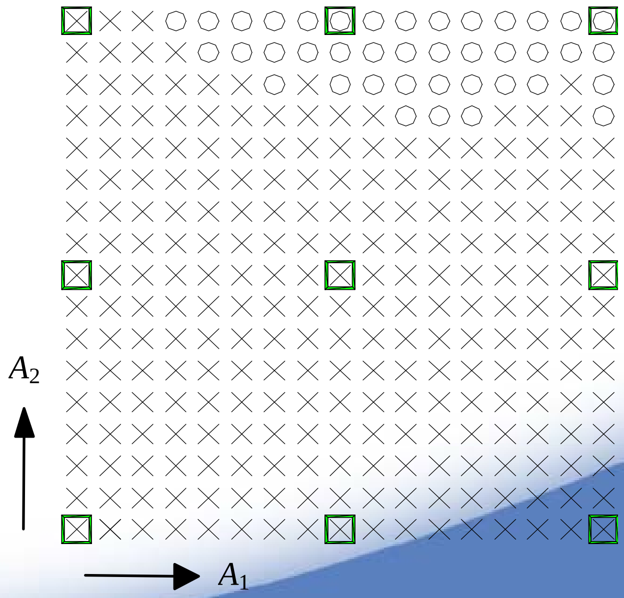
❖ Array type depends on number of variables, such as $L_9(3^4)$ is useful for no more than 4 variables, and 9 data points are need in a 3-level array.

| $L_9(3^4)$ | 1 | 2 | $A_1$ | $A_2$ | Feasible |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | N |
| 2 | 1 | 2 | 1 | 81 | N |
| 3 | 1 | 3 | 1 | 289 | N |
| 4 | 2 | 1 | 81 | 1 | N |
| 5 | 2 | 2 | 81 | 81 | N |
| 6 | 2 | 3 | 81 | 289 | Y |
| 7 | 3 | 1 | 289 | 1 | N |
| 8 | 3 | 2 | 289 | 81 | N |
| 9 | 3 | 3 | 289 | 289 | Y |

Initial start

$W$=4.81

$W$=11.34



# Search, Check, Update

**Search**

A search algorithm then searches for the “optimal point” in the feasible domain simulated by NN.

**Check**

This new design point is checked against the true constraints to see whether it is feasible, and is then added to the training set.

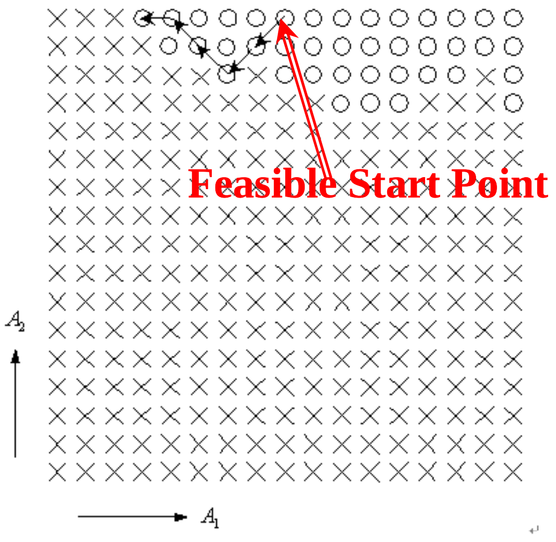
**Update**

The NN is trained again with this added information, in the hope that the network will better simulated the boundary of the feasible domain of the true optimization problem.

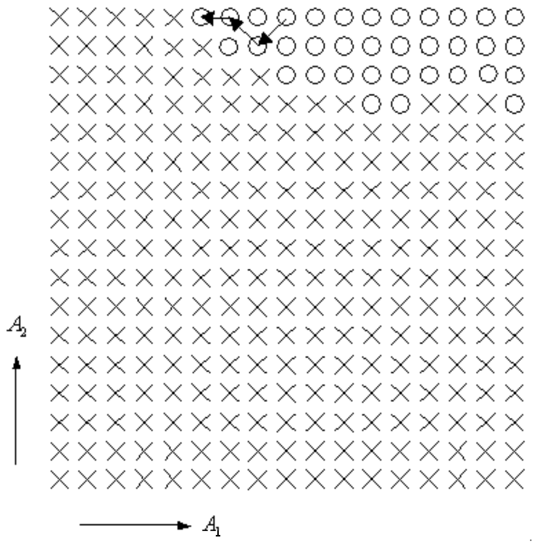
**Terminate**

Then we search for the “optimal point” in this new approximated feasible domain again. This process continues in an iterative manner until the approximate model insists the same “ optimal point” in consecutive iterations.

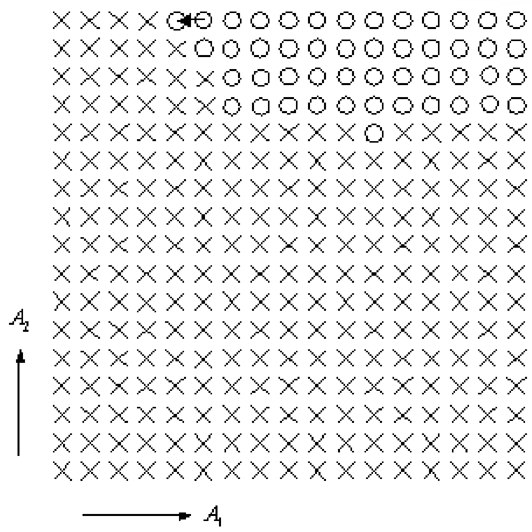
# Search, Check, Update (1)




**Iteration 1**




**Iteration 2**

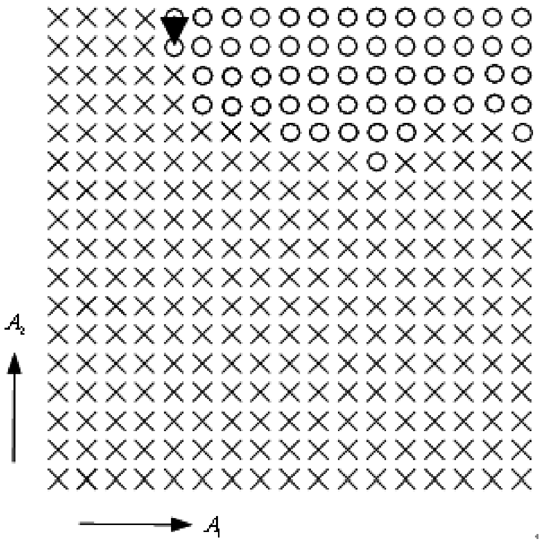



**Iteration 3**

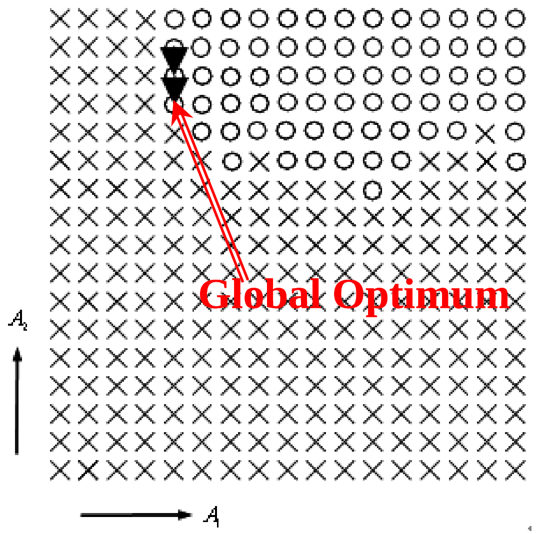
The path of search from feasible point (81, 289), to point (16, 289).

Checked with $M_{real}$, design point (16, 289) is infeasible. The feasible domain is updated and searched again from point (81, 289) to a new design point (36, 289).

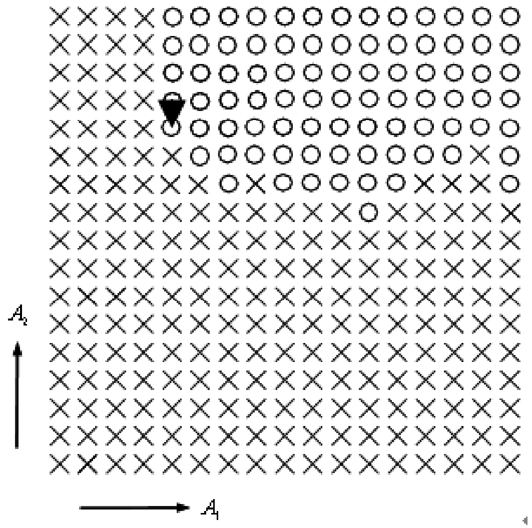
# Search, Check, Update (2)



Iteration 4

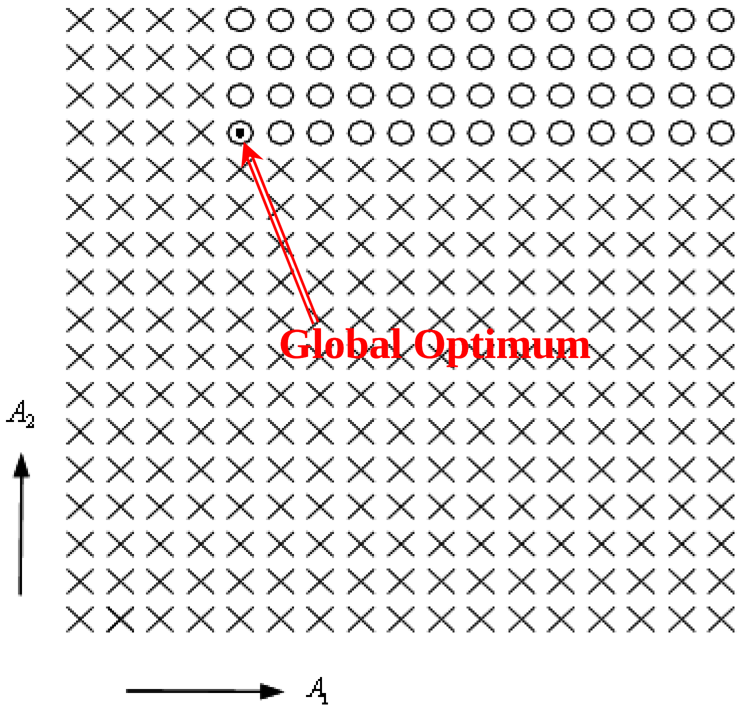



Iteration 5



Iteration 6

# Search, Check, Update (3)

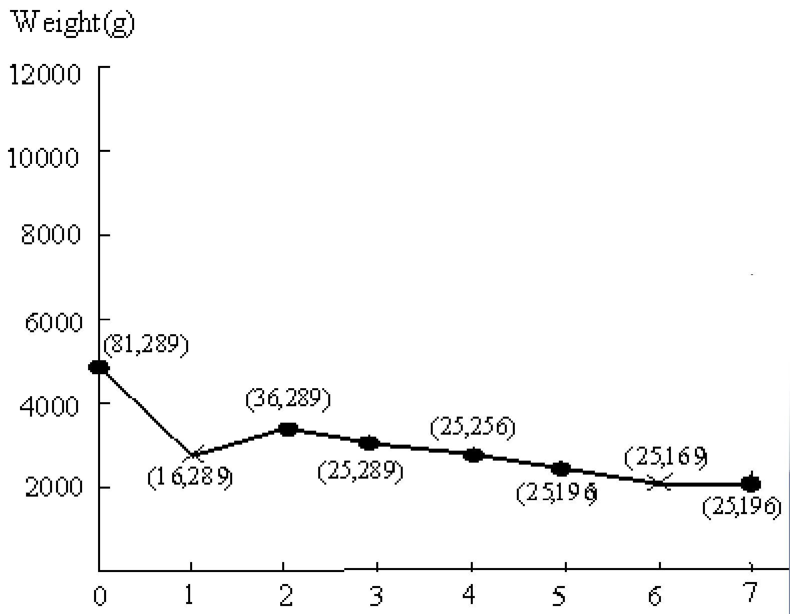



**Iteration 7**

❖Design point (25, 196) was obtained repeatedly after iteration 7, process stops.

❖This design point is also the global optimum, checked by exhaust search.

❖The objective function value at this point is 2.32kg, or 48.3% of that4.81kg of initial design point (81, 289).



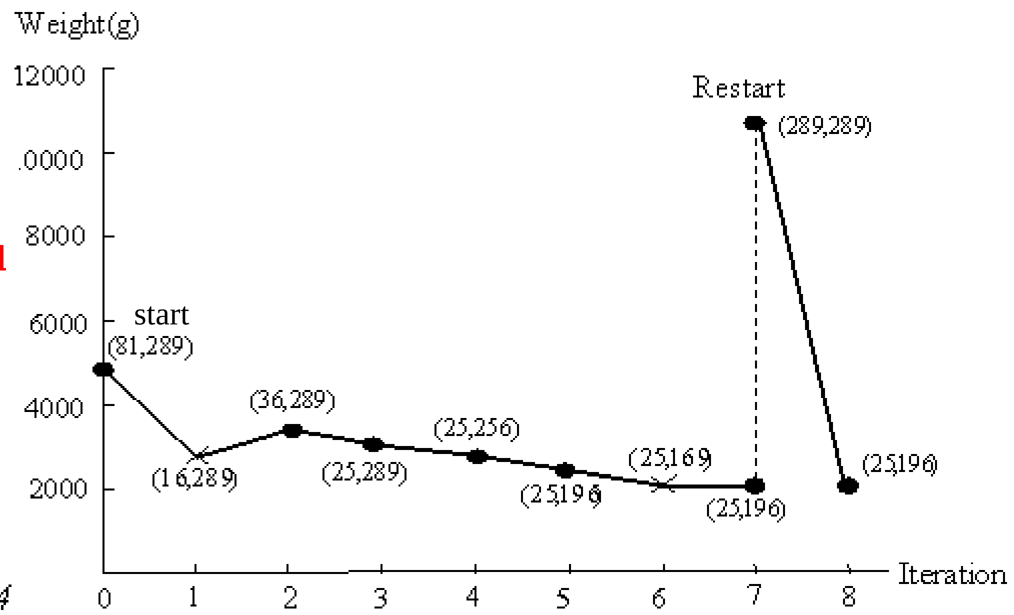
# Restart Strategy :

- Purpose : to ensure a better chance of finding a global optimum.
- Restart the searching process from another feasible point in set of initial training data.

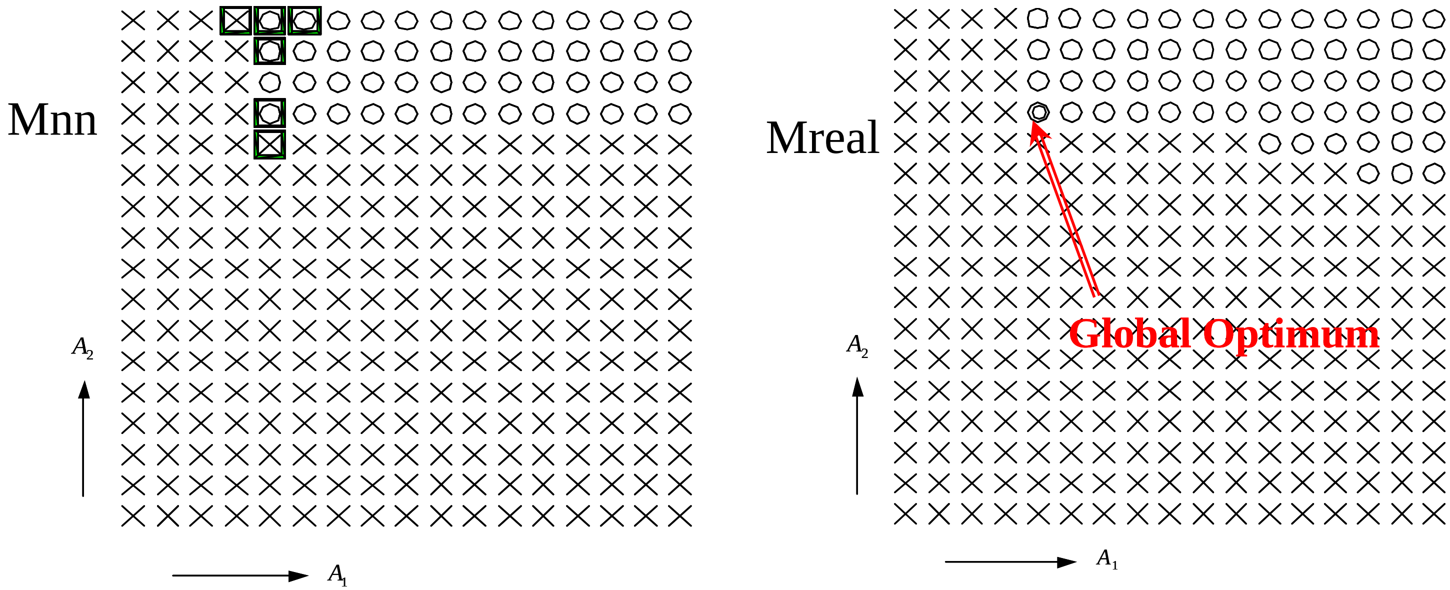
| $L_9(3^4)$ | 1 | 2 | $A_1$ | $A_2$ | Feasible |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | N |
| 2 | 1 | 2 | 1 | 81 | N |
| 3 | 1 | 3 | 1 | 289 | N |
| 4 | 2 | 1 | 81 | 1 | N |
| 5 | 2 | 2 | 81 | 81 | N |
| 6 | 2 | 3 | 81 | 289 | Y |
| 7 | 3 | 1 | 289 | 1 | N |
| 8 | 3 | 2 | 289 | 81 | N |
| 9 | 3 | 3 | 289 | 289 | Y |

Initial start — $W=4.81$

restart — $W=11.34$



# Feasible Boundary Comparison:simulated and real



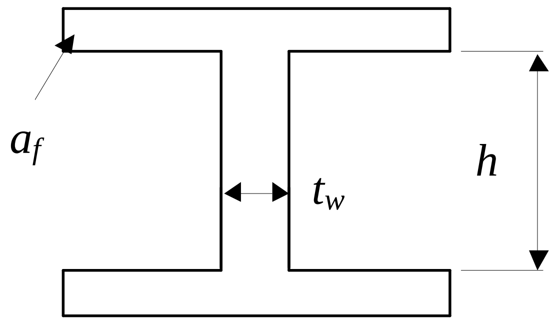

❖ 6 more discrete points are evaluated (nodes enclosed by square brackets) beside 9 initial training points in simulated domain Mnn.

❖ simulated feasible boundary of Mnn in neighborhood of the optimum point is exactly the same as that of real (Mreal).

❖ Total of 5.5% (16 ) points out of 289 possible combinatorial combinations are evaluated to obtain this optimum point.

# Design examples using the SNA method

# **Verification Examples**: I-beam design (1)




Discussed by Farkas [1984],

Yuan *et al*. [1990],

Farkas and Jarmai [1997].

**Opt. Model:** **3** design variables with explicit constraints

min. $f = ht + 2a_f$ (function of cross-section area)

s.t. $g_1 = 1 - \dfrac{\sigma_M + \sigma_N}{R_u} \geq 0$ (material ultimate stress)

$$g_2 = 145\sqrt[4]{\frac{(1+\sigma_N/\sigma_M)^2}{1+173(\sigma_N/\sigma_M)^2}} - \frac{h}{t_w} \geq 0 \quad \text{(buckling stress)}$$

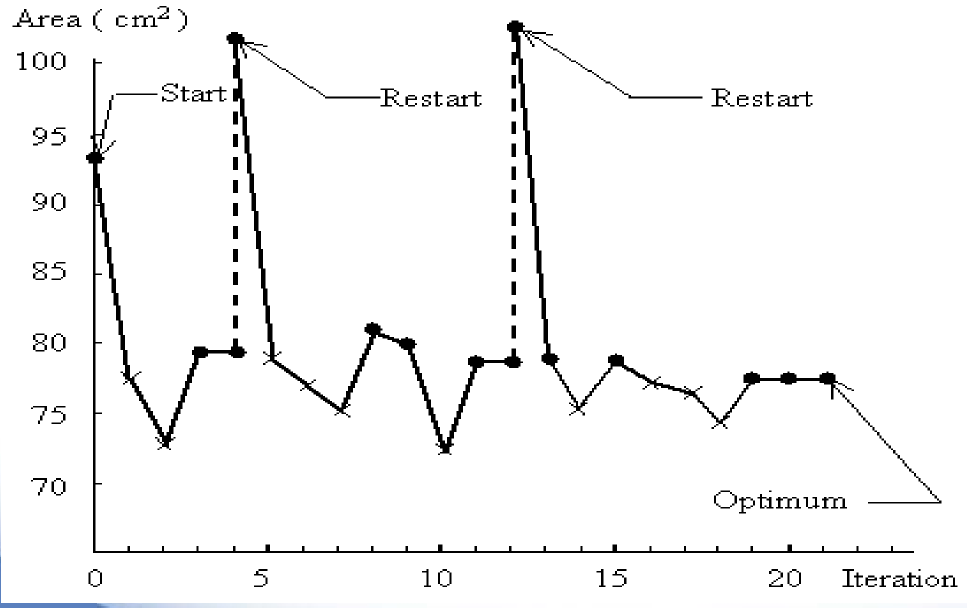
where $\sigma_M = \dfrac{M}{w_x}$, $\sigma_N = \dfrac{N}{a_{total}}$

bending load $M$=320kNm, comp. load $N$=128kN,

and the section modulus $w_x \cong h(a_f + ht_w/6)$

# **Verification Examples:** I-beam design (2)

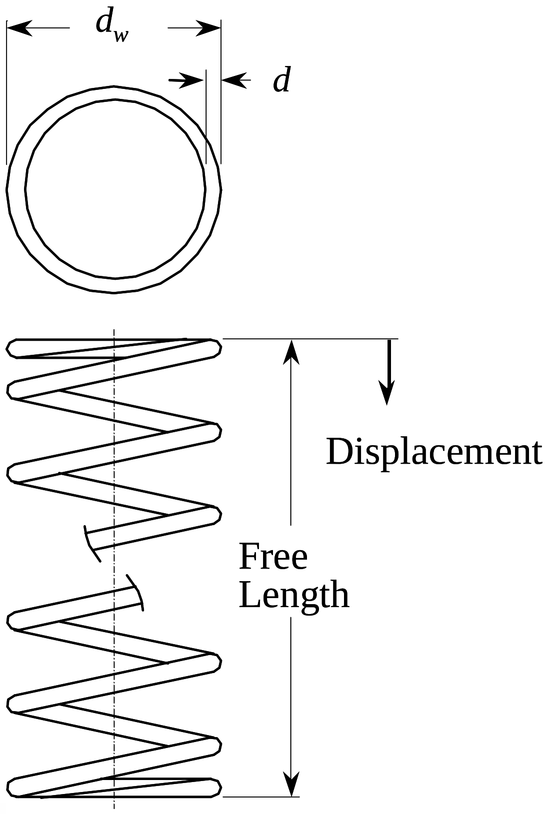
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| *h* | 66 | 68 | 70 | 72 | 74 | | | | |
| *t* | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 | | | | |
| *a* | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |




❖ Verify global solution by exhaust search, final optimal point (70, 0.6, 18) with 78 $cm^2$ (83.9% of initial starting) area was found at the 2nd restart.

❖ Total of 13.3% points out of 225 combinatorial combinations evaluated .

## **Verification Examples**: Helical Compression Spring Design (1)




Sandgren [1990]

**Opt. Model:** **3** design variables, explicit constraints

min. $F = \pi d_w d^2(n+2)/4$ (function of total volume)

s.t. 7 constraints due to design limitations:

(1) max. allowable shear stress189000psi

(2) min. wire diameter 0.2 inch

(3) max. outside coil diameter 3 inch

(4) min. inner coil diameter 3 times of wire dia.

(5) max. deflection under preload 6 inch

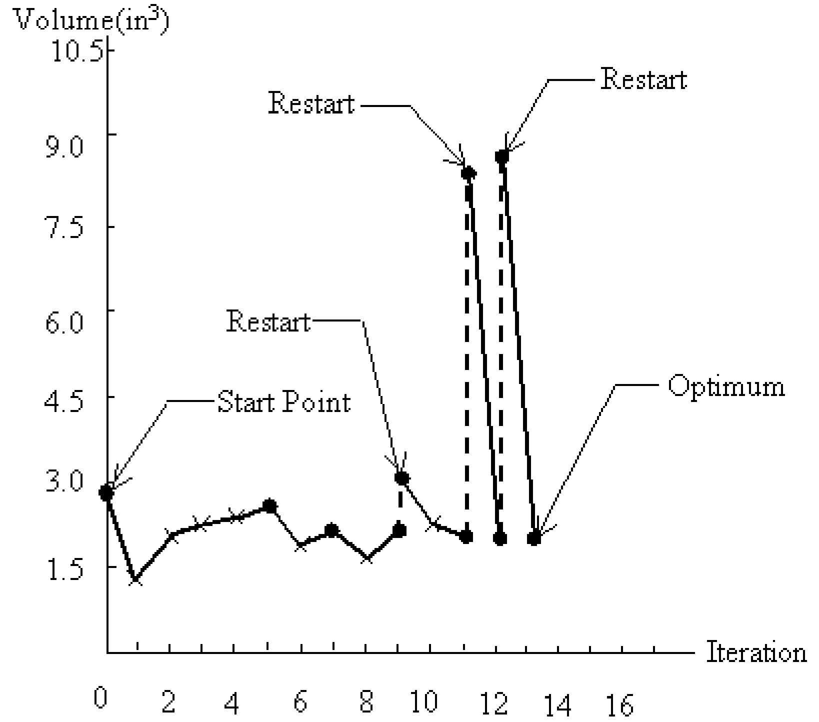
(6) total def. consists with coil $L_{free}$1.25 inch

(7) min. def. from preload to max. load 14 inch

# **Verification Example**: Helical Compression Spring Design (2)

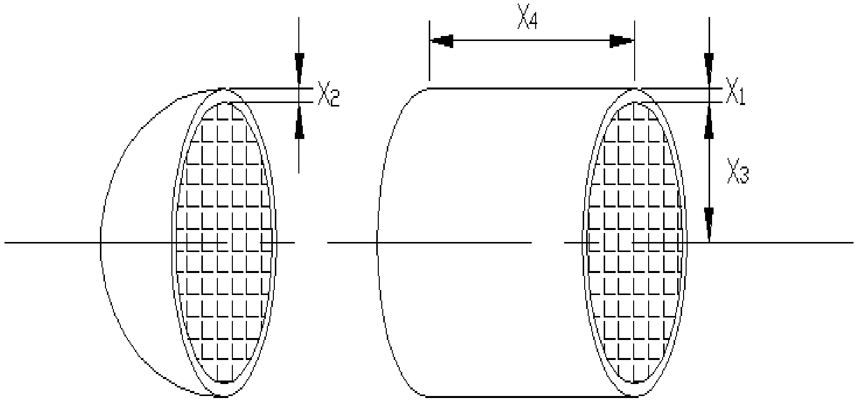
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *n* (no.) | 7 | 8 | 9 | 10 | 11 | 12 | 13 | | | | |
| *d* (in) | 0.207 | 0.225 | 0.244 | 0.263 | 0.283 | 0.307 | 0.331 | 0.362 | 0.375 | 0.394 | 0.438 |
| $d_w$ (in) | 1.13 | 1.18 | 1.23 | 1.28 | 1.33 | 1.38 | 1.43 | 1.53 | 1.68 | 1.85 | 2.000 |

| Design variables | SNA method | Lagrange Multiplier Kannan & Kramer [1994] | Branch & Bound by Sandgren [1990] | Siddle [1982] |
|---|---|---|---|---|
| *n* (no.) | 7 | 7 | 10 | 16(16.2roundoff) |
| *d* (in) | 0.283 | 0.283 | 0.283 | 0.263 |
| $d_w$ (in) | 1.13 | 1.329 | 1.18 | 1.173 |
| $f(in^3)$ | 2.010 | 2.365 | 2.799 | 3.604 |




•SNA method gets global optimum at every (re)start search, 2.6% points out of 847 combinations are evaluated.

## **Example demonstrates practicality**: Pressure vessel design (1)




Opt. Model [Sandgren,1990;Hsu,2001,2003]:

$$\text{min. } f = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

$$\text{s.t. } g_1 = -x_1 + 0.0193x_3 \le 0$$

$$g_2 = -x_2 + 0.00954x_3 \le 0$$

$$g_3 = (-\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3)/1296000 + 1 \le 0$$

$$g_4 = x_4 - 240 \le 0$$

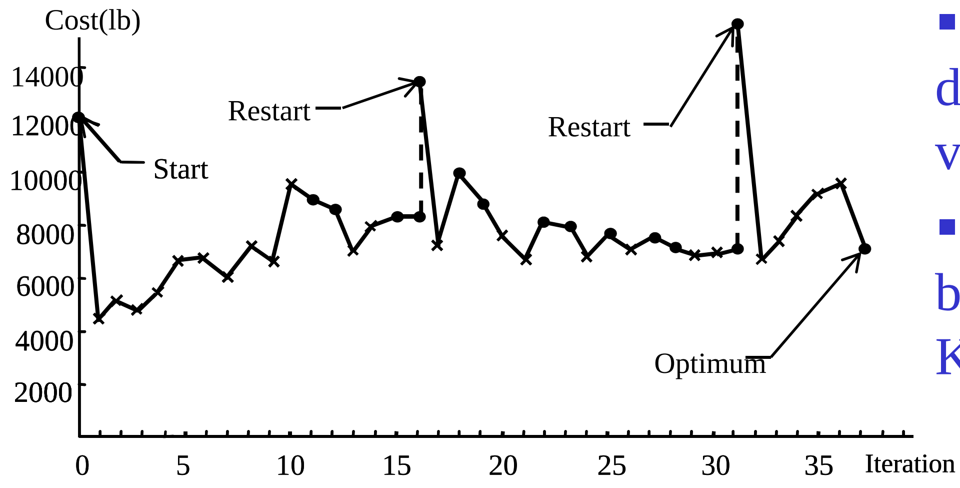
•Min. total cost s.t. min. shell thickness ,750ft$^3$volume above, 3000psi working

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$(in.) | 1.125 | 1.1875 | 1.25 | 1.3125 | 1.375 | 1.4375 | 1.5 | 1.5625 | 1.625 | 1.6875 | 1.75 | 1.8125 |
| $x_2$(in.) | 0.625 | 0.6875 | 0.75 | 0.8125 | 0.875 | 0.9375 | 1 | 1.0625 | 1.125 | 1.1875 | 1.25 | 1.3125 |
| $x_3$(in.) | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 |
| $x_4$(in.) | 40 | 45 | 50 | 55 | 60 | 65 | 70 | 75 | 80 | 85 | 90 | 95 |

| | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$(in.) | 1.875 | 1.9375 | 2 | | | | | | | | |
| $x_2$(in.) | 1.375 | 1.4375 | 1.5 | 1.5625 | 1.625 | 1.6875 | 1.75 | 1.8125 | 1.875 | 1.9375 | 2 |
| $x_3$(in.) | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | | |
| $x_4$(in.) | 100 | 105 | 110 | 115 | 120 | | | | | | |

Test in large discrete design spaces.

# **Example demonstrates practicality**: Pressure vessel design (2)

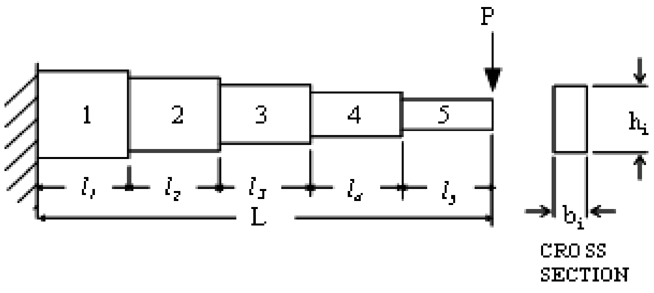



- Sandgren, Kannan & Kramer defied $x_3$ and $x_4$ as continuous variables.
- Sandgren used branch and bound method, while Kannan & Kramer used Lagrange method.

| | Continuous solution | Discrete solution SNA method | Mixed discrete solution Kannan & Kramer [1994] | Discrete solution Sandgren [1990] |
|---|---|---|---|---|
| $x_1$ | 1.125 | 1.1875 | 1.125 | 1.125 |
| $x_2$ | 0.625 | 0.625 | 0.625 | 0.625 |
| $x_3$ | 58.290 | 59 | 58.291 | 48.97 |
| $x_4$ | 43.693 | 40 | 43.69 | 106.72 |
| $g_1$ | -0.000 | -0.049 | +0.000 | -0.180 |
| $g_2$ | -0.069 | -0.062 | -0.069 | -0.158 |
| $g_3$ | +0.000 | -0.001 | -0.000 | +0.000 |
| $g_4$ | -196.307 | -200 | -196.31 | -133.28 |
| $f$ | 7198.01 | 7442.02 | 7198.04 | 8129.14 |

Restarts only search the (same) optimum with 62.21% cost of the initial start point, and 0.037% points out of 123,165 combinations are evaluated.

# **Example demonstrates practicality**: 5-step cantilever beam (1)




Thanedar and Vanderplaats [1995]

$$\text{min. } V = D(b_1h_1l_1 + b_2h_2l_2 + b_3h_3l_3 + b_4h_4l_4 + b_5h_5l_5)$$

$$\text{s.t. } g_1 = \frac{6Pl_5}{b_5h_5^2} - 14000 \le 0$$

$$g_2 = \frac{6P(l_5 + l_4)}{b_4h_4^2} - 14000 \le 0$$

$$g_3 = \frac{6P(l_5 + l_4 + l_3)}{b_3h_3^2} - 14000 \le 0$$

$$g_4 = \frac{6P(l_5 + l_4 + l_3 + l_2)}{b_2h_2^2} - 14000 \le 0$$

$$g_5 = \frac{6P(l_5 + l_4 + l_3 + l_2 + l_1)}{b_1h_1^2} - 14000 \le 0$$

$$g_6 = \frac{Pl^3}{3E}\left(\frac{1}{I_5} + \frac{7}{I_4} + \frac{19}{I_3} + \frac{37}{I_2} + \frac{61}{I_1}\right) - 2.7 \le 0$$

$$g_7 = \frac{h_5}{b_5} - 20 \le 0$$

$$g_8 = \frac{h_4}{b_4} - 20 \le 0$$

$$g_9 = \frac{h_3}{b_3} - 20 \le 0$$

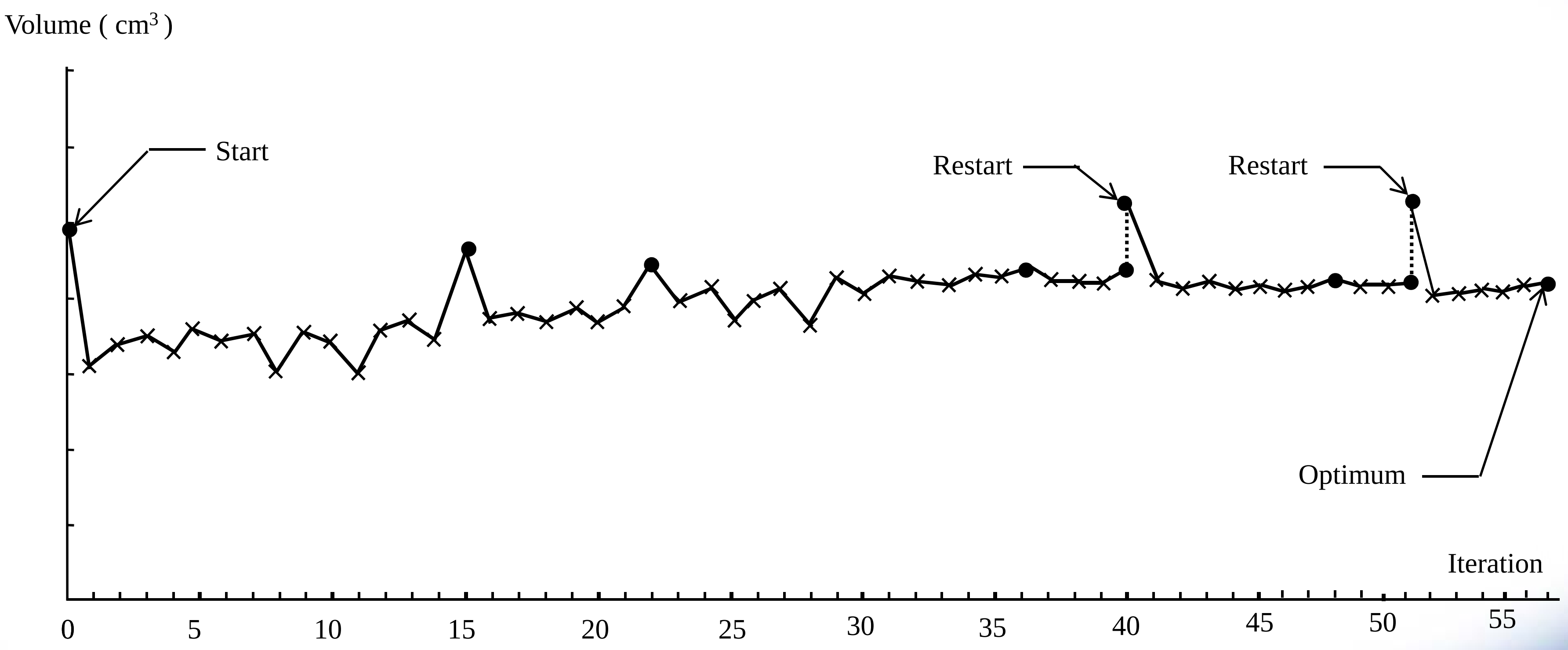
$$g_{10} = \frac{h_2}{b_2} - 20 \le 0$$

$$g_{11} = \frac{h_1}{b_1} - 20 \le 0$$

| | | | | | |
|---|---|---|---|---|---|
| $b_1$ | 3.0 | 3.2 | 3.4 | 3.6 | 3.8 |
| $b_2$ | 3.0 | 3.2 | 3.4 | 3.6 | 3.8 |
| $b_3$ | 2.2 | 2.4 | 2.6 | 2.8 | 3.0 |
| $b_4$ | 2.2 | 2.4 | 2.6 | 2.8 | 3.0 |
| $b_5$ | 1.6 | 1.7 | 1.8 | 1.9 | 2.0 |
| $h_1$ | 58 | 59 | 60 | 61 | 62 |
| $h_2$ | 54 | 55 | 56 | 57 | 58 |
| $h_3$ | 48 | 49 | 50 | 51 | 52 |
| $h_4$ | 42 | 43 | 44 | 45 | 46 |
| $h_5$ | 33 | 34 | 35 | 36 | 37 |

Explore the search computational effort up to 10 design variables

# **Example demonstrates practicality**: 5-step cantilever beam (2)




Test the search computational effort up to 10 design variables

# Engineering optimization problems using the SNA method

# **Real design with implicit constraint**: Reinforced Cylinder (1)

**Opt. Model:**

minimize volume

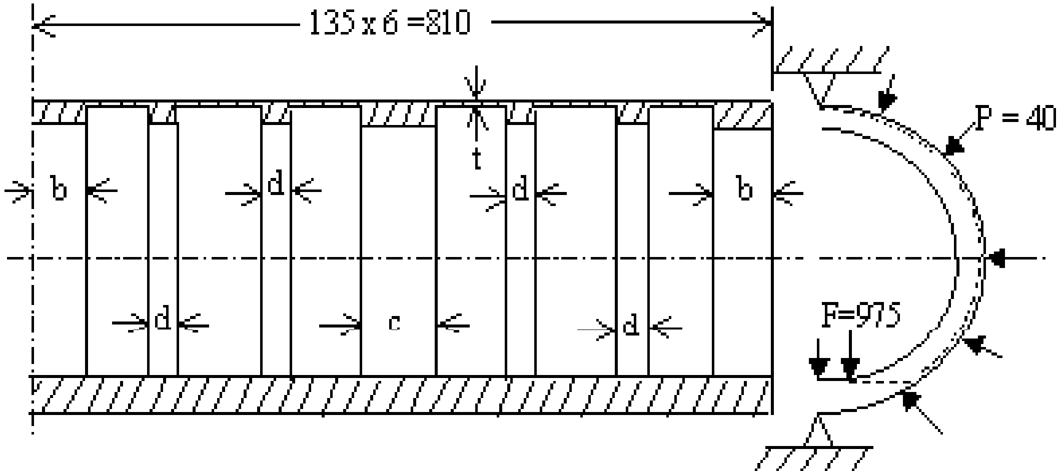
s.t. stress & disp. Limit

$$\min.\quad f(b,c,d,t) = 3.14159\{[535^2-(535-2t)^2](405)+(b+c+2d)[(535-2t)^2-482^2]\}$$

$$\text{s.t.}\quad s(b,c,d,t) < s_{allow}$$

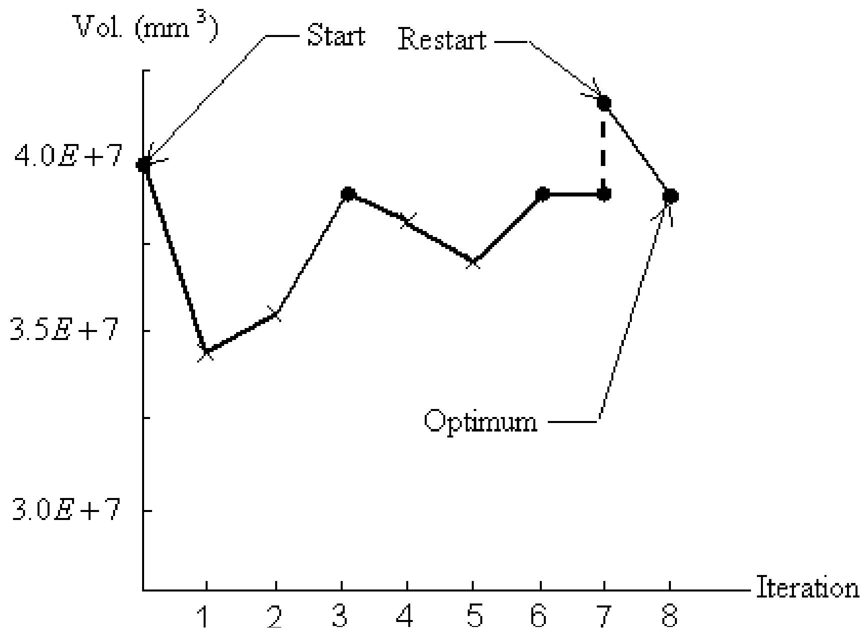
$$disp(b,c,d,t) < t$$

•Purpose: demonstrate the practicality of the SNA method in sizes of structure applications.




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| *b*(mm) | 41 | 43 | 45 | 47 | 49 | 51 | 53 |
| *c*(mm) | 33 | 35 | 37 | 39 | 41 | 43 | 45 |
| *d*(mm) | 16 | 18 | 20 | 22 | 24 | 26 | 28 |
| *t*(mm) | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

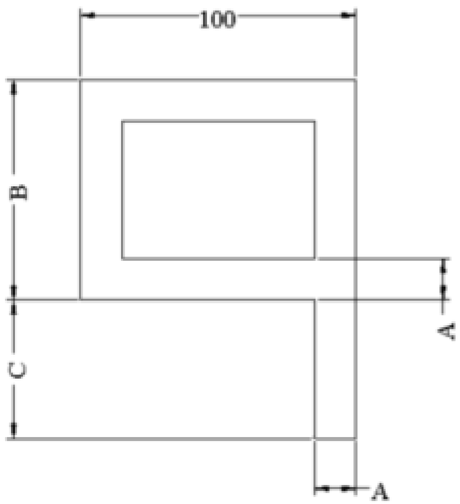
# **Real design with implicit constraint**: Reinforced Cylinder (2)

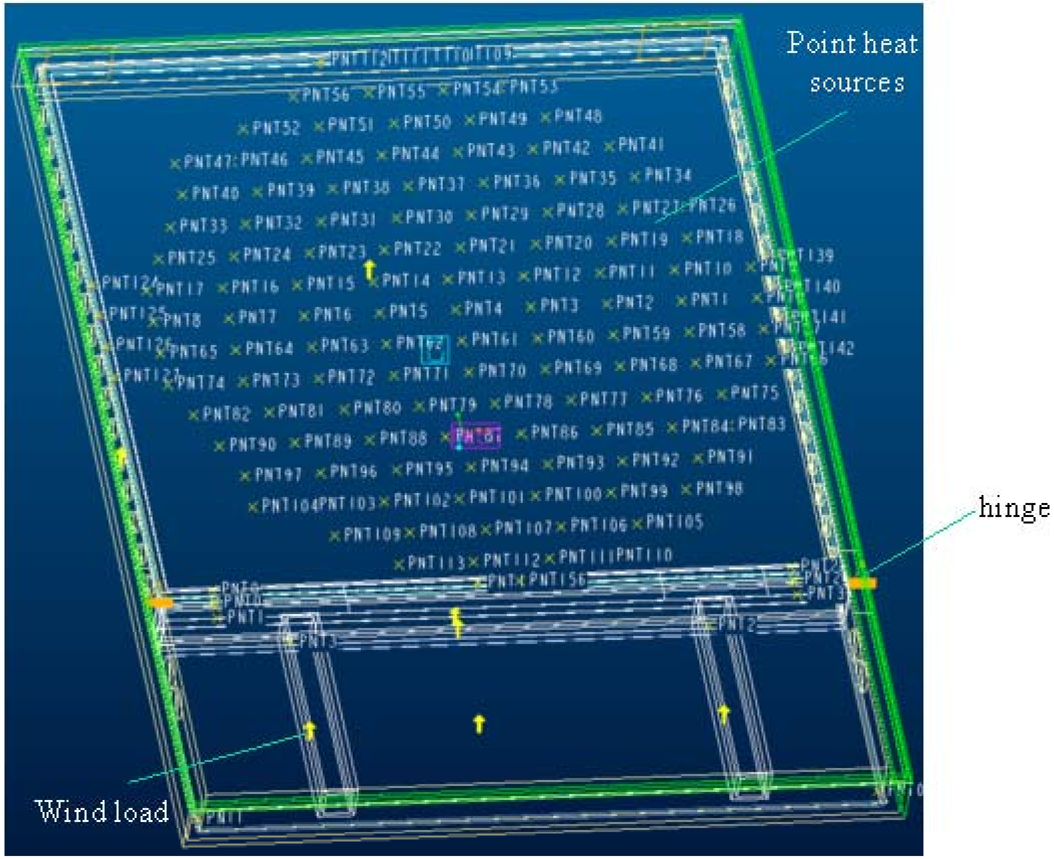



- A 4 design variables, implicit constraint ex. shows 1 restart at iteration history.
- 0.6% out of 2401combinations are evaluated by FEM.

# **Real design with implicit constraint**: Optimal decision (1)

**Purpose:** minimize weight s.t. maximum displacement constrained, determine cross-section sizes, material and shape whether rib or not.




**Opt. Model:**

min. $f(a, b, c, d) = ad[2(100-2a+b)+c]/1000000$

s.t. $disp\text{-}max(a, b, c, d) \leq 6.0$ where $d$ is the density of the material

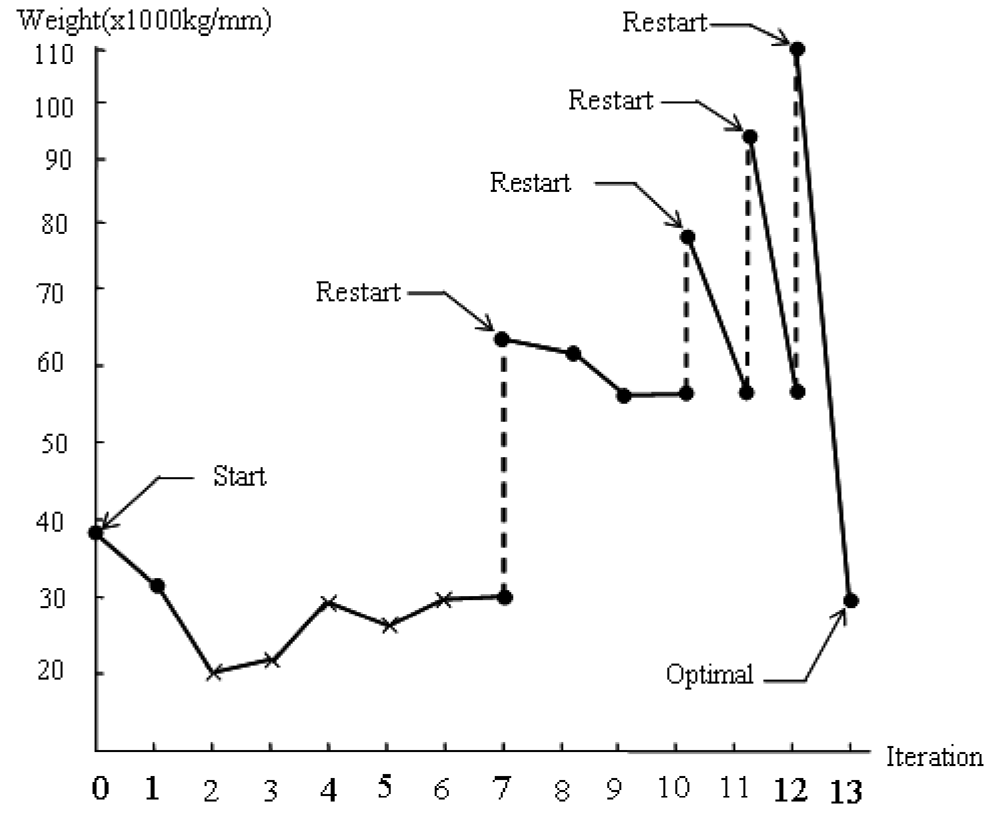
## **Real design with implicit constraint**: Optimal decision (2)

- There are only 2 discrete values for variable *b* and *d*. Therefore $L_8(4^1 \times 2^4)$ orthogonal array is used as the set of initial training data.
- Note variable c can be zero, which means that the rib is not necessary.

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| *a* | 11 | 13 | 16 | 18 | 20 | 22 |
| *b* | 250 | 262.58 | | | | |
| *c* | 0 | 25 | 50 | | | |
| *d* | 2.7 | 7.8 | | | | |

| No. | $L_8(4^1 \times 2^4)$ orthogonal array | | | | *a* | *b* | *c* | *d* | *f*(×1000kg) | Feasible |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 11 | 250 | 0 | 2.7 | 19.483 | N |
| 2 | 1 | 2 | 2 | 2 | 11 | 262.58 | 50 | 7.8 | 62.734 | Y |
| 3 | 2 | 1 | 1 | 2 | 16 | 250 | 0 | 7.8 | 79.373 | Y |
| 4 | 2 | 2 | 2 | 1 | 16 | 262.58 | 50 | 2.7 | 30.722 | N |
| 5 | 3 | 1 | 2 | 1 | 18 | 250 | 50 | 2.7 | 32.951 | N |
| 6 | 3 | 2 | 1 | 2 | 18 | 262.58 | 0 | 7.8 | 91.704 | Y |
| 7 | 4 | 1 | 2 | 2 | 22 | 250 | 50 | 7.8 | 113.599 | Y |
| 8 | 4 | 2 | 1 | 1 | 22 | 262.58 | 0 | 2.7 | **37.847** | **Y** |

# **Real design with implicit constraint**: Optimal decision (3)




•After 13 iterations, including 4 restarts, the optimal design point (18, 262.58, 0, 2.7) was found twice at the 7th iteration and the last restart.

•The objective function value at the optimal point is 16.1% lower than that of the starting point.

•Note that *c*=0, which means that the rib is not necessary, and the material selected is aluminum.

# **Zooming Strategy Example**: Achieve Mirror Surface (1)

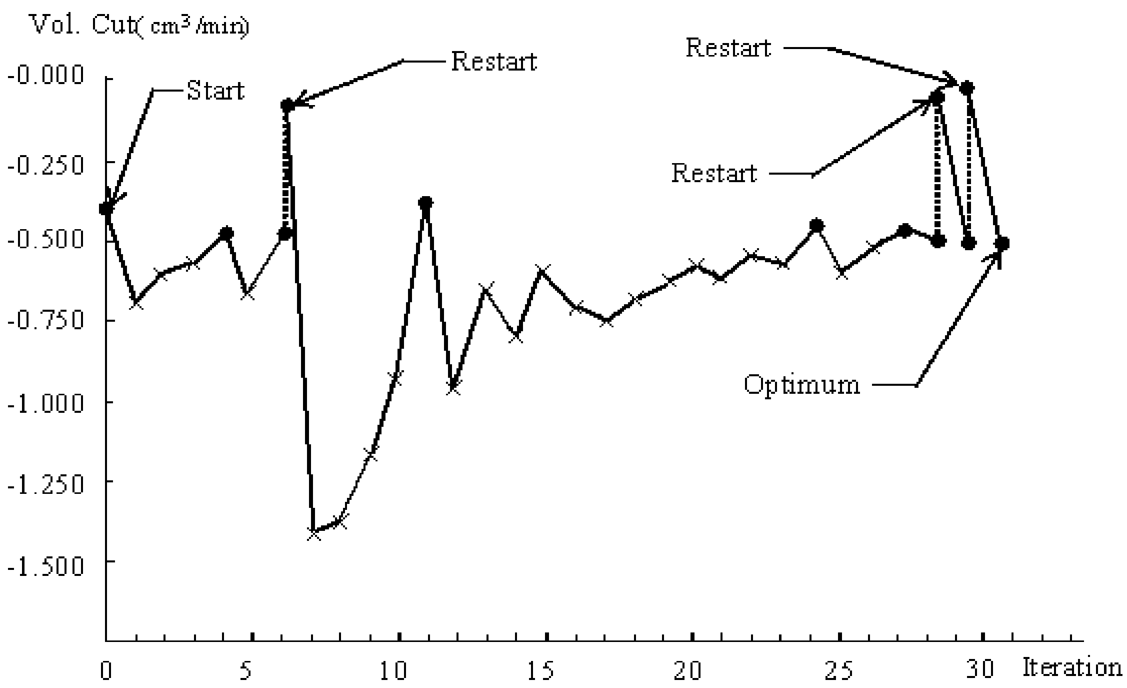
- Al. plate on one-pass milling to get max. volume cut
- Opt. model: max. $Q = 0.055 d_c \times v_s \times r_f \times z$

  s.t. implicit constraint on average roughness $R_a = \frac{\sum_{i=1}^{n} y_i}{n} \le 0.05\ \mu m$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $v_s$ ($10^3$rpm) | 2 | 2.5 | 3 | 3.5 | 4 | 4.5 | 5 | 5.5 | 6 | 6.5 | 7 | | |
| $r_f$ (mm/t) | 0.5 | 1.0 | 1.5 | 2.0 | 2.5 | 3.0 | 3.5 | 4.0 | 4.5 | | | | |
| $d_c$ (mm) | .025 | .03 | .035 | .04 | .045 | .05 | .055 | .06 | .065 | .07 | 0.75 | .08 | .085 |
| $z$ (tooth) | 2 | 4 | 6 | 8 | 10 | | | | | | | | |

| no. | $L_9$ orthogonal array | | | | $v_s$ (rpm) | $r_f$ (mm/t) | $d_c$ (mm) | z(tooth) | $Q$($cm^3$/min) | $R_a$(μm) | Feasible |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 2000 | 0.5 | 0.025 | 2 | 0.00275 | 0.08627 | N |
| 2 | 1 | 2 | 2 | 2 | 2000 | 2.5 | 0.055 | 6 | 0.09075 | 0.04587 | Y |
| 3 | 1 | 3 | 3 | 3 | 2000 | 4.5 | 0.085 | 10 | 0.42075 | 0.04786 | Y |
| 4 | 2 | 1 | 2 | 3 | 4500 | 0.5 | 0.055 | 10 | 0.06806 | 0.02997 | Y |
| 5 | 2 | 2 | 3 | 1 | 4500 | 2.5 | 0.085 | 2 | 0.10519 | 0.05580 | N |
| 6 | 2 | 3 | 1 | 2 | 4500 | 4.5 | 0.025 | 6 | 0.16706 | 0.05088 | N |
| 7 | 3 | 1 | 3 | 2 | 7000 | 0.5 | 0.085 | 6 | 0.09818 | 0.04744 | Y |
| 8 | 3 | 2 | 1 | 3 | 7000 | 2.5 | 0.025 | 10 | 0.24063 | 0.06227 | N |
| 9 | 3 | 3 | 2 | 1 | 7000 | 4.5 | 0.055 | 2 | 0.19058 | 0.10546 | N |

# **Zooming Strategy Example**: Achieve Mirror Surface (2)




- 37 points evaluated.
- Start & last 2 restarts find the optimum.

# **Zooming Strategy Example**: Achieve Mirror Surface (3)

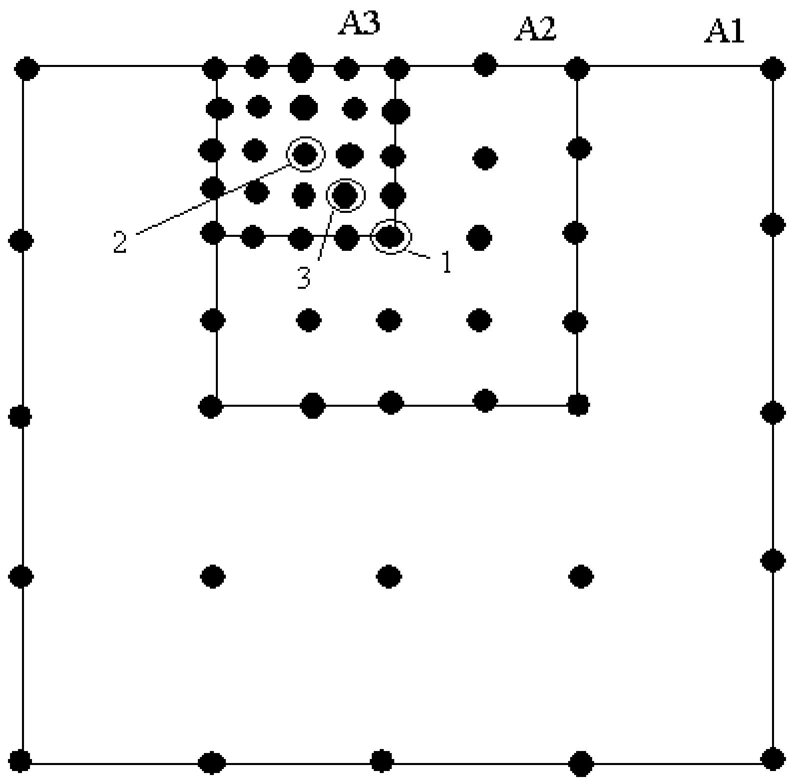
❖Purpose of zooming strategy

–Obtain a finer resolution for the optimal solution.

❖ 2 variables simple example

–Each variable with 5 discrete values

Training data include local points' data of optimum for better simulation.



# Zooming Strategy Example: Achieve Mirror Surface (4)

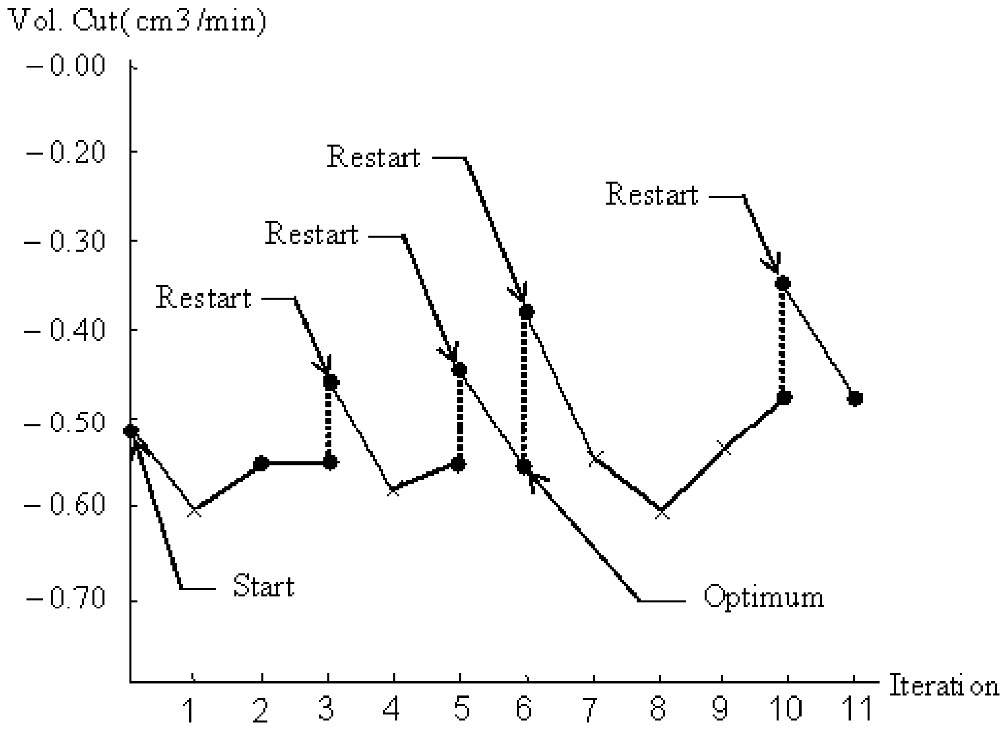
- Zoom in neighborhood of the optimal point (2500, 4.5, 0.080) to increase the no. of discrete grids for controlling the optimum resolution.

| | 1 | 2 | 3 |
|---|---|---|---|
| $v_s$ ($10^3$rpm) | 2 | 2.5 | 3 |
| $r_f$ (mm/t) | 0.5 | 1.0 | 1.5 |
| $d_c$ (mm) | .025 | .03 | .035 |

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $v_s$ (rpm) | 2000 | 2250 | 2500 | 2750 | 3000 |
| $r_f$ (mm/t) | 4.0 | 4.25 | 4.5 | 4.75 | 5.0 |
| $d_c$ (mm) | 0.0750 | 0.0775 | 0.0800 | 0.0825 | 0.0850 |

| no. | $L_9$ orthogonal array | | | $v_s(rpm)$ | $r_f(\mu m/t)$ | $d_c(mm)$ | $R_a(10^{-3}mm)$ | $Q(cm^3/min)$ | Feasible |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2000 | 4.0 | 0.075 | 0.04243 | -0.33 | Y |
| 2 | 1 | 2 | 2 | 2000 | 4.5 | 0.080 | 0.04479 | -0.396 | Y |
| 3 | 1 | 3 | 3 | 2000 | 5.0 | 0.085 | 0.04867 | -0.4675 | Y |
| 4 | 2 | 1 | 2 | 2500 | 4.0 | 0.080 | 0.04735 | -0.44 | Y |
| 5 | 2 | 2 | 3 | 2500 | 4.5 | 0.085 | 0.05230 | -0.5259375 | N |
| 6 | 2 | 3 | 1 | 2500 | 5.0 | 0.075 | 0.04667 | -0.515625 | Y |
| 7 | 3 | 1 | 3 | 3000 | 4.0 | 0.085 | 0.05483 | -0.561 | N |
| 8 | 3 | 2 | 1 | 3000 | 4.5 | 0.075 | 0.05044 | -0.556875 | N |
| 9 | 3 | 3 | 2 | 3000 | 5.0 | 0.080 | 0.05643 | -0.66 | N |

# **Zooming Strategy Example**: Achieve Mirror Surface (5)




- It improves 14.6% of total volume cut at first zooming.
- Opt. point is found after 3 iterations.

# Conclusions and Discussion

# Conclusions: Summary (1)

- In the SNA method no precise function value or sensitivity calculation is required.
  Thus the SNA method does not require continuity or differentiability of the problem functions.

- Only one evaluation of the implicit constraints is needed in each iteration to see whether the current design point is feasible.

# Conclusions: Summary (2)

- The feasible domain is simulated by a NN.
  A clear 0-1 binary pattern is used in the input-output layers of the NN.
  Thus the computational cost of the training of the NN is relatively small.

- In our numerical experience, the error goal of 1e-6 is usually met within 100 epochs, even for cases with many variables and training points.

## Conclusions: Summary (3)

- In all examples, only a small fraction of the total possible combinatorial combinations of design variables were evaluated.

- The precious function evaluations of the implicit constraints are accumulated to progressively form a better approximation to the feasible domain using the NN.

## Conclusions: Summary (4)

- When the no. of discrete values that a design variable takes is large, the "resolution" of the discrete grids of the search space can be controlled using "zooming strategy".

- Instead of taking all discrete values all at once, a few discrete values can be chosen first, so that the size of the search space is acceptable.

- When search terminates at a discrete point, the search space can be reduced to neighborhood of this current point, but increase resolution of the discrete grids, then start iterations again.

# Discussion: SNA Method Limitation (1)

- When number of design variables/discrete values that a design variable can take is large, the search space will be large. It might affect efficiency of the search algorithm.
- No. of points searched in each iteration increases in order of $3^n$-1, where *n* is no. of variables. It is very expensive with large no. of variables even though function evaluation is cheap, which poses limitation to the SNA method .
- As demonstrated by the examples presented in this paper, practicality of SNA method can handle up to 10 design variables.

# Discussion: SNA Method Limitation (2)

- Another limitation of SNA is that the objective function is assumed to be explicit.

- If the objective function is implicit, use NN to simulate the function values. This is a potential direction of future development of the SNA method.