Optimization of Slider Air Bearing Shapes using Variations of Simulated Annealing

Hong Zhu and D. B. Bogy

Computer Mechanics Laboratory
Department of Mechanical Engineering
University of California
Berkeley, CA 94720

ABSTRACT

Hard disk drives continue to increase in areal density. This requires sliders with lower and lower flying height (FH). Also the uniformity of its FHs and the flatness of its roll profile with radius become more critical as the FH gets lower. By using modern optimization techniques, it is possible to find the optimal slider air bearing surfaces (ABS) design according to our multi design goals. The simulated annealing technique is a powerful tool for addressing this strongly nonlinear problem, and it has also been widely used in many other areas. Several variations of this optimization technique have been developed and they formed the Simulated Annealing family. Among these algorithms are the Standard Boltzmann Annealing (BA), Fast Cauchy Annealing (FA) and the Adaptive Simulated Annealing(ASA). In this report, we describe the Simulated Annealing optimization technique and we also compare the performance of BA, FA and ASA by optimizing a slider ABS design.

1. INTRODUCTION

Since the first commercial hard disk drive, which was the IBM product RAMAC (Random Access Method of Accounting and Control) with an areal density of 2000 bits/in.^{2} or 100 bits/in. (BPI) and 20 tracks/in. (TPI) was delivered in 1957, the areal density increased at an average annual growth rate of about 39% from 1957 to 1991. The rate increased to 65% from 1991 to 1997 due to the use of many new technologies such as magnetoresistive (MR) read heads, smaller diameter disks and smoother disk surfaces which allow lower flying heights. Fig. 1 shows the areal density growth for the hard disk. Recently, IBM has achieved an areal density of 35.3Gb/in.^{2} in a laboratory demonstration, and it's believed that areal densities of 100Gb/in.^{2} will be demonstrated by the end of year 2001. Recently the hard disk industry has begun discussion of HDD areal density of 1Tb/in.^{2}.

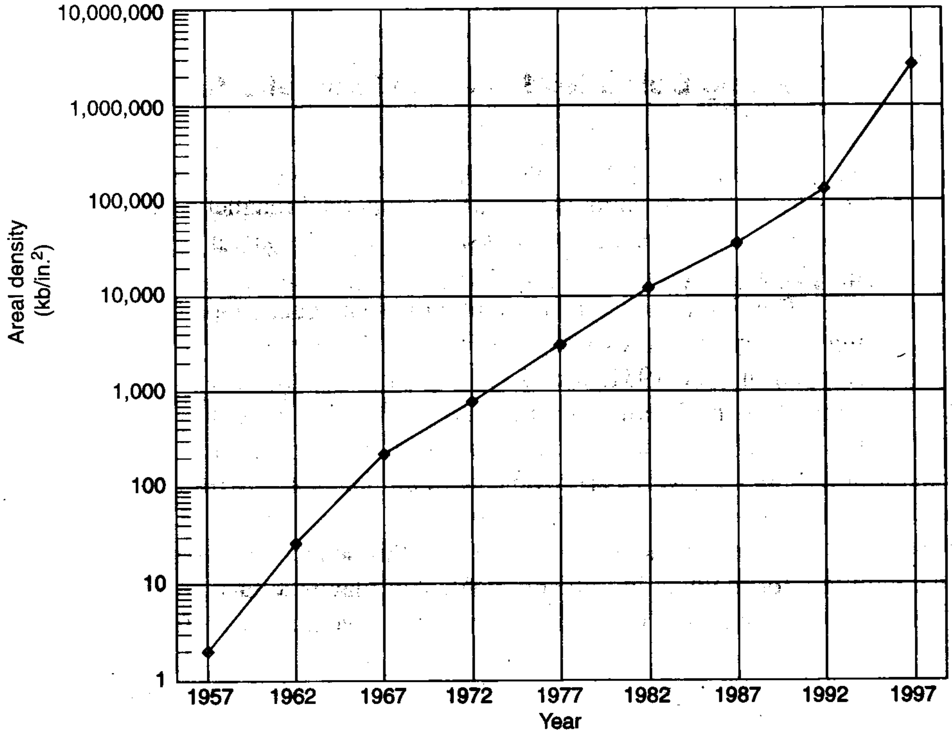


Fig.1 Hard magnetic disk areal density growth [4]

The increase of the areal density is of great economic and technical interest, and it has a huge impact on the price of the hard disk drive. Along with the increase of the areal density, the price per megabyte has been lowered from more than $100 per megabyte in 1980 to the present cost of about 2 cents per megabyte. The following figure shows the price history over the past 20 years, showing a precipitous drop in the last few years.

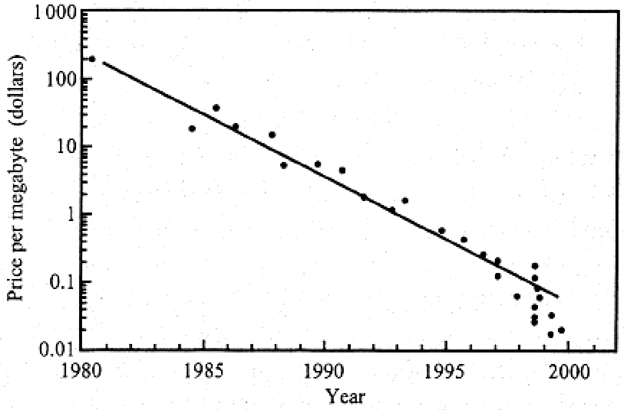


Fig.2 Price history of hard disk products [5]

To obtain higher areal density, the head-to-media spacing or flying height must be lowered. Fig.3 shows the relationship between the head-to-media spacing and the areal density.

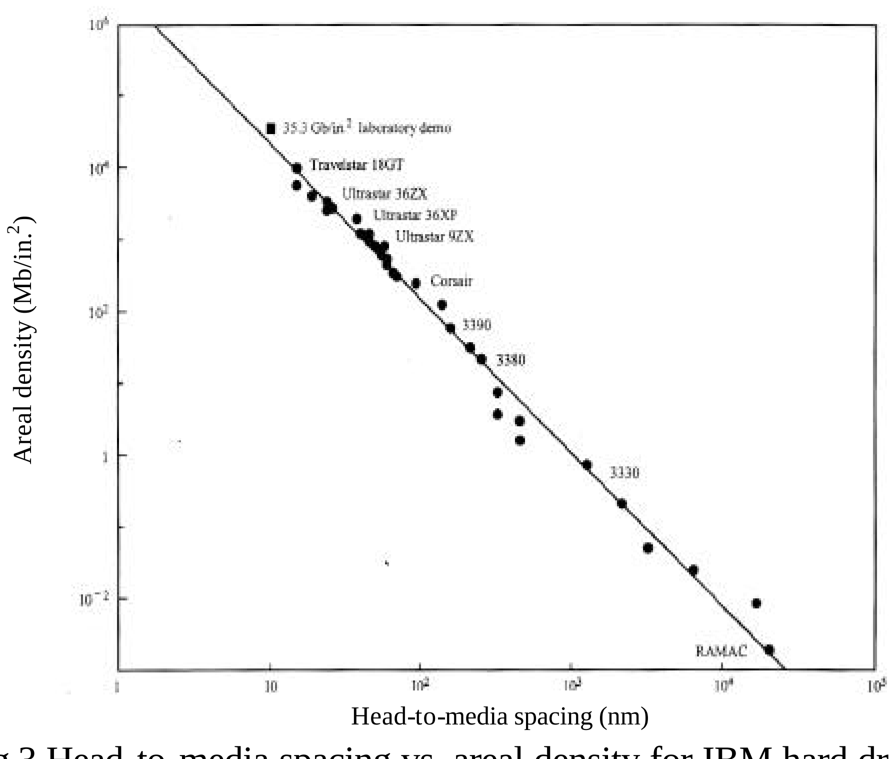


Fig.3 Head-to-media spacing vs. areal density for IBM hard drives [5]

In order to obtain the areal density of 100 Gb/in.^{2}, the flying height of the slider will need to be below 10 nanometers. At these very low flying heights, the performance of the slider becomes quite crucial and requires tighter control of flying height uniformity, roll profile flatness etc.

To meet the increasingly rigorous multi-objective slider performance criteria, modern optimization technique can be used to solve this strongly nonlinear problem. The advantages of using numerical optimization are:

- Reduction of design time.
- Optimization provides a systematized logical design procedure.
- Many design variables and constraints, which are not easy to visualize, can be handled by the optimization program.
- Generally we can always get some design improvement by using optimization.
- The optimization process is not biased by intuition or experience in engineering. Therefore it has a higher probability of obtaining improved nontraditional designs.
- Optimization requires a minimal amount of human-machine interaction.

In summary, high efficiency, simplicity and automaticity are the reasons to use optimization techniques when designing slider air bearing for HDD.

2. NUMERICAL METHOD

2.1 Introduction to the optimization method

Optimization is the process of trying to minimize a function subject to conditions on the variables. This function is generally called the objective function or cost function. The conditions set on the variables are referred to as constraints.

We can state the optimization problem as:

Minimize $\{f(x) \mid x \in S\}$, where f(x) is the objective function, S is a set of feasible solutions to the problem known as the search space and x is a single point within the set.

If there are no constraints set on the variables, the problem is referred to as an unconstrained problem. Otherwise it is called a constrained problem. The constraints reduce the size the set S by limiting the number of feasible solutions. For the unconstrained and constrained problem, the mathematical description is basically the same. The only difference is the size of the set S. The constrained problem has a smaller set S, but the constraints also make the problem much more complicated.

If the objective function f(x) and the constraints in a problem are both linear combinations of the independent variables, the problem is referred to as a linear programming problem. For this kind of problem, the solution can easily be found by the Simplex method or the interior point method. If the objective function is quadratic in nature, while its constraints take the linear form, we can decompose the problem and then find its solution by the Simplex method. If the objective function f(x) has a definite form, we can always find its solution in a predictable way. Unfortunately, many problems of interest are nonlinear problems. Air bearing design optimization problems have many objective functions with no distinct forms and the constraints can take many possible forms as well.

For nonlinear problems, the most difficult issue is multiple optima. The objective function may have many minima and the one found might not be the absolute minimum point. Instead, we only have the assurance that it is a local optimum, i.e. a feasible point x* that is an optimal solution to the problem whose feasible region is the intersection of the original region and some neighborhood of x*. It is important to avoid just finding the local optimum. That's why we adopt a global optimization technique. Global optimization is the search for the absolute minimum point of the objective function over the given search space.

There are many global optimization algorithms, and they can be divided into two fundamentally different categories, i.e. deterministic algorithms and stochastic algorithms. For the deterministic algorithms, every new search point is chosen in a definite way and no random components are involved. For the stochastic algorithms, random elements are introduced to generate the new search points. Deterministic algorithms can handle definite objective functions very well. The stochastic algorithms can be applied to a wider range of objective function types, but usually with longer run times.

2.2 Family of simulated annealing algorithm

The simulated annealing algorithm is a global optimization technique that is based on the concept of the physical annealing process where the temperature of a system is gradually lowered in order to obtain its lowest energy state.

Simulated annealing, which is a stochastic technique, was developed to handle nonlinear problems that are extremely difficult to minimize. It is known to be a powerful and useful tool for a wide variety of minimization problems of large nonlinear systems. It has also been widely applied in many areas, such as circuit design, chemistry, economics, biology, image processing, statistics etc. Research on the simulated annealing technique has also become very active in recent years.

One of the most important elements of all simulated annealing algorithms is the Metropolis rule, which was developed by Metropolis et al. in 1953. We now briefly describe this important rule.

Recall that optimization is a process for minimizing an objective function or cost function $E(\underline{\mathbf{x}}^i)$. Here E is a function of the vector $\underline{\mathbf{x}}$ where its components x_n are derived from a certain set in a search space. The superscript i represents different states or designs generated during the process. The Metropolis rule incorporates the following three stages:

1) Given a starting design $\underline{\mathbf{x}}^i$ with cost function $E(\underline{\mathbf{x}}^i)$, a small perturbation to $\underline{\mathbf{x}}^i$ is made to obtain a new design $\underline{\mathbf{x}}^j$ according to a probability function $g_T(\underline{\mathbf{x}}^i)$.

2) Computation of the cost function difference between the two designs by $\Delta E = E(\underline{\mathbf{x}}^j) - E(\underline{\mathbf{x}}^i)$.

3) Decide whether or not to accept the new design. There are two cases:
   a. If $\Delta E \le 0$, the new design is always accepted.
   b. If $\Delta E > 0$, the new design is accepted with the probability

$$h(\Delta E) = e^{-\frac{\Delta E}{T}},$$

where T is the cost function temperature.

The procedure is repeated while the temperature T is gradually lowered. If the annealing procedure is carried out adequately, the system is expected to converge to the global minimum state. By accepting states with relatively higher cost function values according to some probability, which is called the “hill-climbing” technique shown in the following figure, the Metropolis rule can help the process avoid getting trapped in a local minimum point.

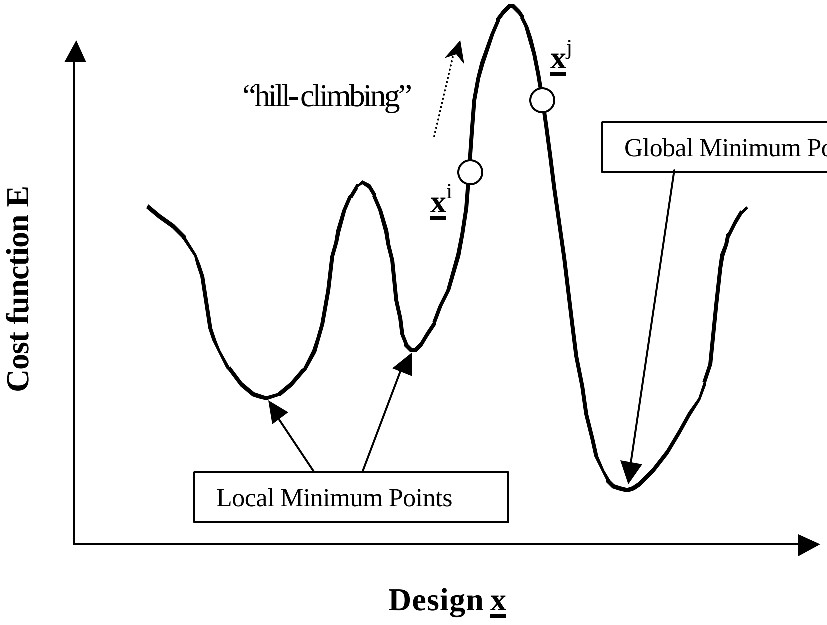


Fig.4 “Hill-climbing” technique in Metropolis rule

Several simulated annealing algorithms have developed, such as the Standard Boltzmann Annealing (BA), Fast Cauchy Annealing (FA) and Adaptive Simulated Annealing (ASA). The basic idea of all these algorithms is the same. The main differences are the selection of probability functions $g_T(\underline{\mathbf{x}})$ and $h(\Delta E)$, and the different cooling schedules resulting from those two probability functions. There are three components in all these simulated annealing algorithms:

- $g_T(\underline{\mathbf{x}})$: The probability density function in the state space of D parameters $x=\{x_n,\ n=1,D\}$, where the subscript T represents the temperature.

- $h(\Delta E)$: The probability function for acceptance of new cost functions given the most recent prior value.

- T(k): The annealing “temperature” for step k, which is also referred to as the cooling schedule. It includes the parameter temperature and the cost temperature, which will affect the perturbation of the previous state and the acceptance probability of the new state respectively. These two temperatures comply with the same cooling schedule. The cooling schedule is actually a result of the probability functions $g_T(\underline{\mathbf{x}})$ and h(ΔE).

2.2.1 Standard Boltzmann Annealing (BA)

The BA was first introduced as a stochastic method for implementing large dimension path integrals for statistical physics. **[6]** The method was developed for handling general minimization problems. The kernel of the algorithm is based on the probability density derived from Gaussian Markovian systems. The Boltzmann distribution is given by

$$g_T(x) = (2pT)^{\frac{-D}{2}} e^{\frac{-x^2}{2T}} \quad ,$$

where $x \equiv \frac{h - h_0}{b - a} \in [-1,1]$. The range $[a, b]$ is the constraint set on certain points. h_0 represents the previous point and h is the new point. So x can be viewed as the normalized deviation of h from the previous point h_0. T is the parameter temperature, which is the measure of the perturbation of the Boltzmann distribution g in the D-dimensional space h .

The 3-D and 2-D views of the Boltzmann distribution are shown in Fig.5 and Fig.6 respectively. These two figures show the change of the distribution as the temperature is lowered from 1 to 0.1.

The acceptance probability of the BA is:

$$h(\Delta E) = \frac{1}{1 + e^{\frac{\Delta E}{T}}} \quad ,$$

where ΔE signifies the “energy” difference between the present and previous values of the energies (considered here as cost functions), i.e., $\Delta E = E_{k+1} - E_k$. T is the cost temperature. The lower the cost temperature, the lower the acceptance probability.

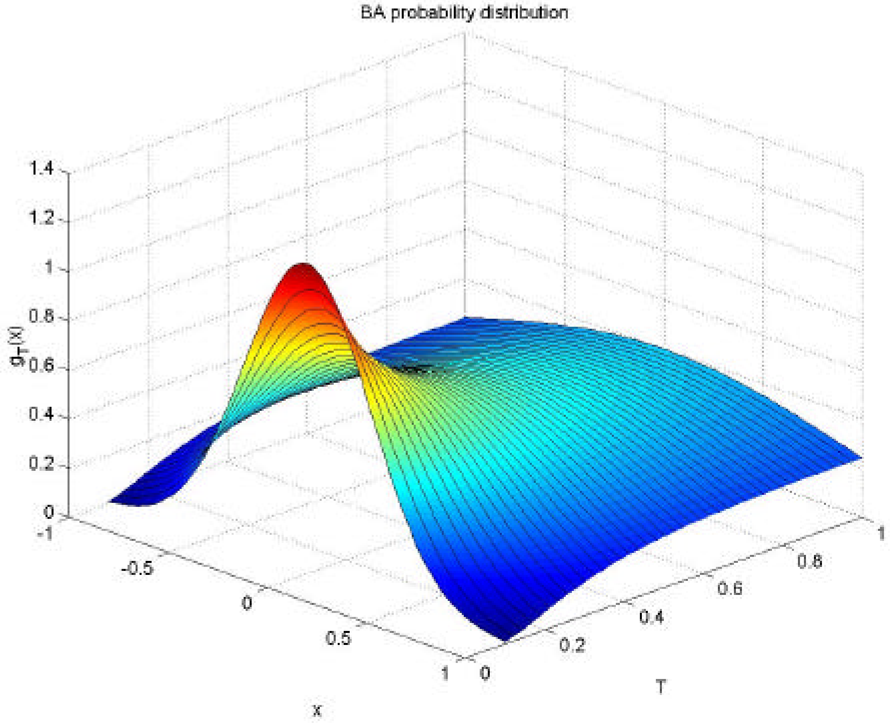


Fig.5 3-D view of the BA probability function $g_T(x)$

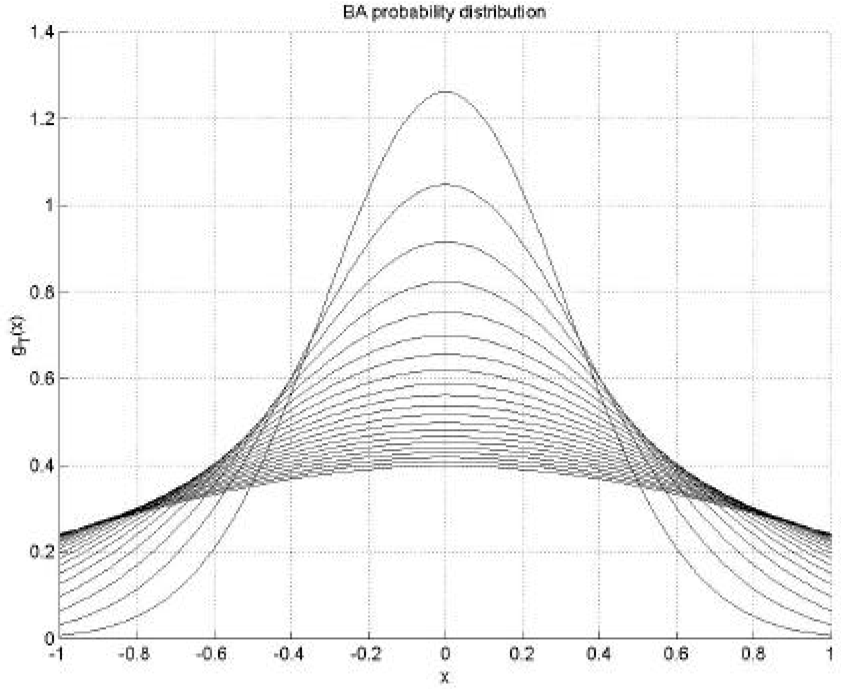


Fig.6 2-D view of the BA probability function $g_T(x)$

Given $g_T(x)$, it has been proved [7] that a global minimum of E(x) will be obtained if T is selected to be not faster than

$$T_k = \frac{T_0}{\ln(k)} \quad .$$

The strict proof will not be introduced here. Instead, we only present a heuristic demonstration here to show that T_k will suffice to give a global minimum of E(x). In order to statistically assure that any point in x-space can be sampled "Infinitely Often in annealing Time" (IOT), it suffices to prove that the products of probabilities of not generating a state x IOT for all annealing-times successive to time k_0 yield zero,

$$\prod_{k=k_0}^{\infty}(1-g_k)=0 \quad ,$$

which is equivalent to:

$$\sum_{k=k_0}^{\infty} g_k = \infty \quad .$$

If we put the expression of T_k into $g_T(x)$, then we obtain

$$\sum_{k=k_0}^{\infty} g_k \geq \sum_{k=k_0}^{\infty} e^{-\ln k} = \sum_{k=k_0}^{\infty} \frac{1}{k} = \infty \quad .$$

2.2.2 Fast Cauchy Annealing (FA)

The methodology of BA can be extended for use with any function *g* that satisfies $\prod_{k=k_0}^{\infty}(1-g_k)=0$ in conjunction with a proper cooling schedule T(k).

It may be desirable for the function *g* to enable a faster convergence rate. The use of a Cauchy distribution is a good example of how this may be achieved. The Cauchy distribution is defined by the following equation:

$$g_T(x) = \frac{T}{\left(x^2 + T^2\right)^{\frac{D+1}{2}}} \quad .$$

The 3-D and 2-D views of the Cauchy distribution are shown in Fig.6 and Fig.7 respectively. These two figures show the change of the distribution as the temperature is lowered from 1 to 0.1.

The simulated annealing procedure using the Cauchy distribution converges to a global minimum with temperature declining not faster than

$$T(k) = \frac{T_0}{k} \quad .$$

Then

$$\sum_{k=k_0}^{\infty} g_k \approx \frac{T_0}{x^{D+1}} \sum_{k=k_0}^{\infty} \frac{1}{k} = \infty \quad .$$

Thus the FA method statistically has an annealing schedule exponentially faster than the method of BA.

2.2.3 Adaptive Simulated Annealing (ASA)

Many physical problems have a D-dimensional parameter space. Different parameters have different finite ranges, given by physical considerations, and different annealing-time-dependent sensitivities, measured by the curvature of the cost function at local minima. BA and FA have g distributions that sample infinite ranges, and there is no provision for considering differences in each parameter-dimension, e.g., different sensitivities might require different cooling rates. Adaptive Simulated Annealing (ASA) was developed to meet the following goals:

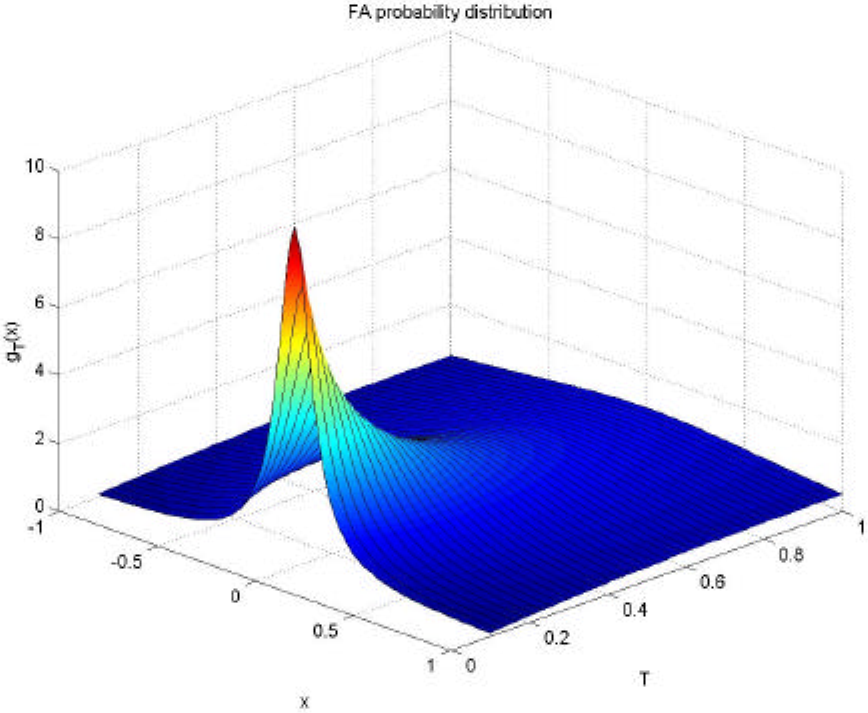


Fig.6 3-D view of the FA probability function $g_T(x)$

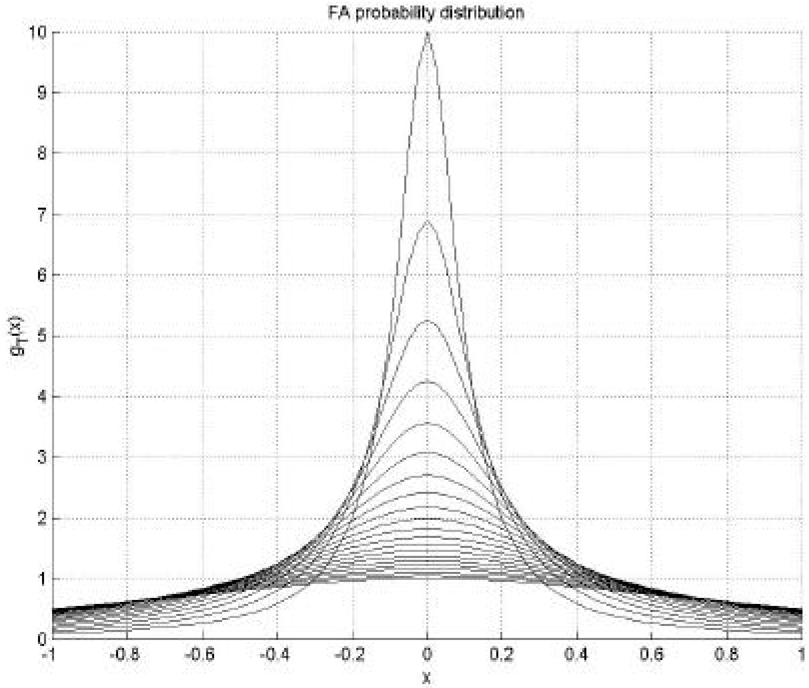


Fig.7 2-D view of the FA probability function $g_T(x)$

- Obtain a solution from a bounded parameter space instead of an unbounded space.
- Use a faster cooling schedule so as to get faster convergence.
- Consider the sensitivities of for each parameter-dimension, i.e., different constraint points should have different cooling rates.

Each of the parameters α^{i} at annealing time k is bounded within the range

$$a_k^i \in [A_i, B_i] \ .$$

The parameters are generated at each new step by a random variable x^i $\in$ [-1,1] as follows:

$$a_{k+1}^i = a_k^i + x^i (B_i - A_i) \ .$$

Define the generating function as

$$g_T(x) = \prod_{i=1}^{D} g_T^i(x^i) = \prod_{i=1}^{D} \frac{1}{2(|x^i| + T_i)\ln(1 + \frac{1}{T_i})} \ .$$

The 3-D and 2-D views of this distribution are shown in Fig.8 and Fig.9 respectively. These two figures show the change of the distribution as the temperature is lowered from 1 to 0.1.

Then the cumulative probability distribution is

$$G_T(x) = \int_{-1}^{x^1}\int_{-1}^{x^2} \cdots \int_{-1}^{x^D} \prod_{i=1}^{D} g_T^i(x^i) dx^1 dx^2 \cdots dx^D \equiv \prod_{i=1}^{D} G_T^i(x^i) \ ,$$

where

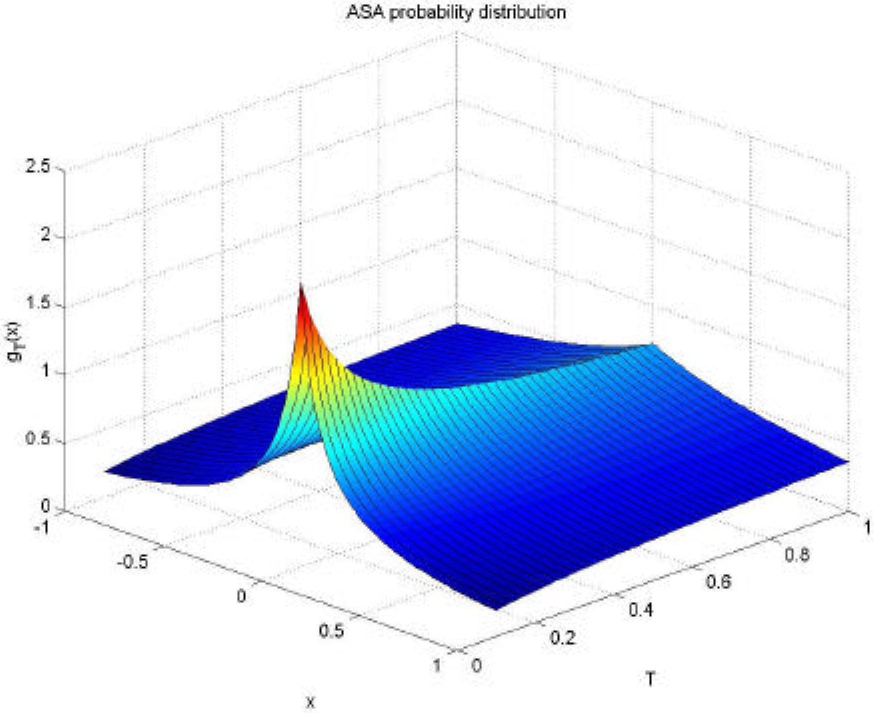


Fig.8 3-D view of the ASA probability function $g_T(x)$

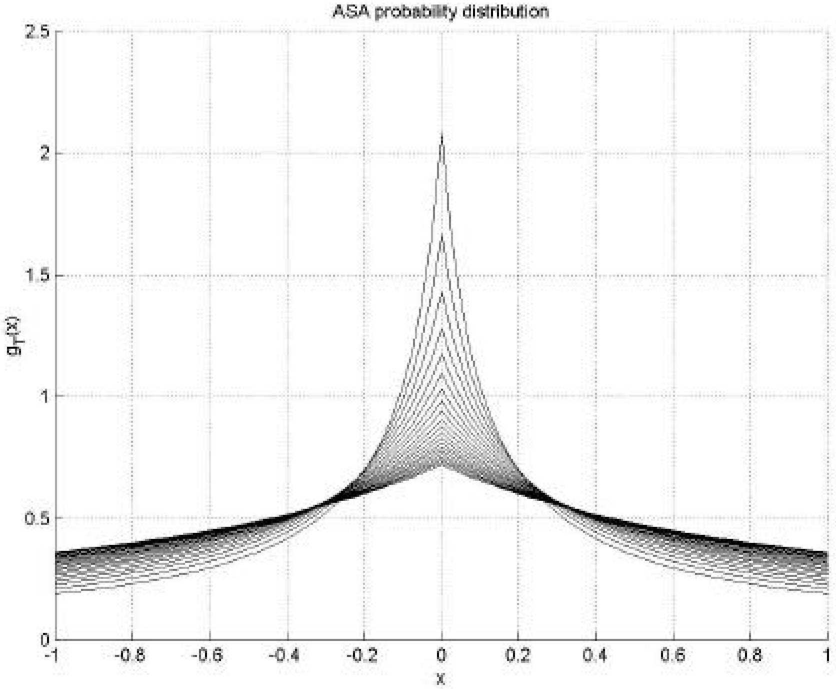


Fig.9 2-D view of the ASA probability function $g_T(x)$

$$G_T^i(x^i) = \frac{1}{2} + \frac{\operatorname{sgn}(x^i)}{2} \frac{\ln(1 + \frac{|x^i|}{T_i})}{\ln(1 + \frac{1}{T_i})} \quad .$$

x^i is generated from the uniform distribution U[0,1] through a parameter $u^i \in$ U[0,1]:

$$x^i = \operatorname{sgn}(u^i - \tfrac{1}{2}) T_i [(1 + \frac{1}{T_i})^{|2u^i - 1|} - 1] \quad .$$

For the cooling schedule

$$T_i(k) = T_{0i} e^{-c_i k^{\frac{1}{D}}} \quad ,$$

a global minimum can be obtained, i.e.,

$$\sum_{k=k_0}^{\infty} g_k \approx \sum_{k=k_0}^{\infty} \left(\prod_{i=1}^{D} \frac{1}{2|x^i| c_i} \right) \frac{1}{k} = \infty \quad .$$

The parameter c_i is controlled by the following relations:

$$\begin{aligned} k_{fi} &= e^{n_i} \\ T_{fi} &= T_{0i} e^{-m_i} \\ c_i &= m_i e^{\frac{-n_i}{D}} \end{aligned} \quad ,$$

where k_{fi} and T_{fi} are the final time step and the final parameter temperature.

The acceptance probability function is defined as:

$$h(\Delta E) = e^{\frac{-\Delta E}{T}} \quad .$$

For a multi-dimensional search the cost function value generally has different sensitivities with respect to different parameters. So at any annealing time, it is sensible to attempt to "stretch out" the ranges over which the relatively insensitive parameters are being searched, as compared to the ranges of the more sensitive parameters. In the algorithm, that is equivalent to resetting the annealing time k for the different parameters.

This mechanism is referred as "Re-annealing" or "Adaptation" and it is the reason why the algorithm is called Adaptive Simulated Annealing. The adaptation is accomplished by calculating the energy sensitivities with respect to the different parameters at the most current minimum value of the cost function:

$$s_i = (B_i - A_i)\frac{\partial E}{\partial a^i} \quad .$$

The annealing time k_i is rescaled for each parameter a^i by making use of the maximum sensitivity $s_{\max}$=max(s_1, s_2, …s_D) :

$$k_i^{'} = \left(\frac{\ln\left(\frac{T_{i0}}{T_{ik}} \frac{s_i}{s_{\max}} \right)}{c_i} \right)^D \quad .$$

T_{i0} is set to unity at the beginning of the search, which is ample to span each parameter dimension.

2.2.4 Summary

The following table gives a comparison among these three simulated annealing algorithms:

	BA	**FA**	**ASA**
Generating Probability $g_T(x)$	$(2pT)^{\frac{-D}{2}} e^{\frac{-x^2}{2T}}$	$\frac{T}{\left(x^2+T^2\right)^{\frac{D+1}{2}}}$	$\prod_{i=1}^{D} \frac{1}{2(\lvert x^i \rvert + T_i)\ln(1+\frac{1}{T_i})}$
Acceptance Probability h(ΔE)	$\frac{1}{1+e^{\frac{\Delta E}{T}}}$	$\frac{1}{1+e^{\frac{\Delta E}{T}}}$	$e^{\frac{-\Delta E}{T}}$
Cooling Schedule T(k)	$\frac{T_0}{\ln(k)}$	$\frac{T_0}{k}$	$T_{0i}e^{-c_i k^{\frac{1}{D}}}$
Adaptive Re-annealing	No	No	Yes

Among these three algorithms, ASA has the fastest cooling rate since the temperature is decreased exponentially. That means it has the fastest convergence rate. Also, ASA features an adaptive re-annealing mechanism. This unique feature enables the ASA to set different cooling rates for parameters with different sensitivities. These properties make ASA the most efficient and robust algorithm among the three.

3. IMPLEMENTATION OF THE ALGORITHMS

3.1 Structure of the optimization program

To implement the optimization, two closely integrated parts are needed. One is the optimization algorithm, and the other is the solver.

The optimization algorithm is used to generate different sample designs, which are then sent to the solver for calculation of the parameters. From the results the algorithm evaluates the quality of the current design and generates a new design based on the result.

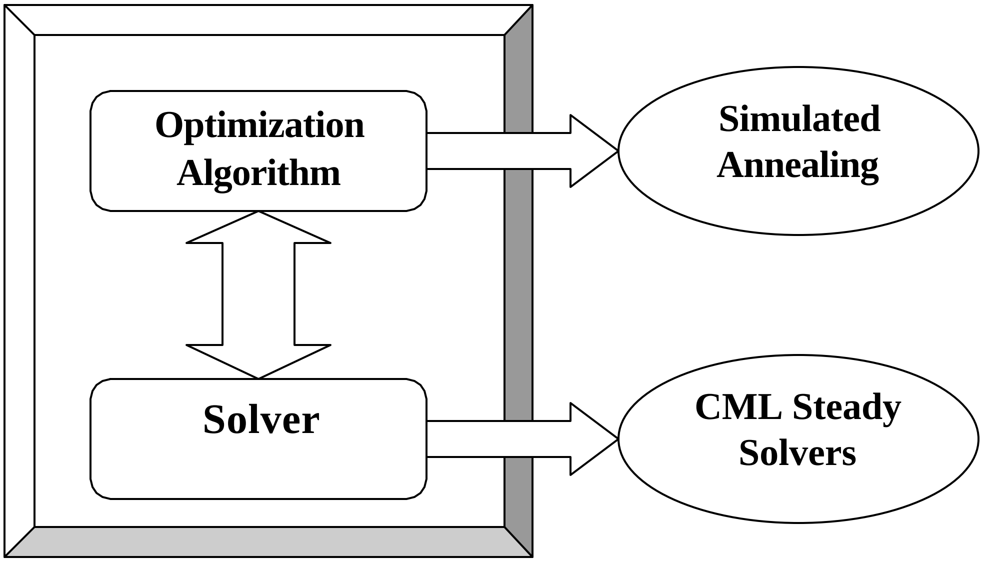


Fig.10 Structure of the optimization program

Here we use the three simulated annealing algorithms, including the Standard Boltzmann Annealing (BA), Fast Cauchy Annealing (FA) and the Adaptive Simulated Annealing (ASA). The solver is the CML slider ABS design programs, which were developed by the Computer Mechanics Laboratory of University of California at Berkeley, including the CML rectangular mesh solver Quick419 and the CML triangular mesh solver Quick5.

3.2 Flow chart of the optimization program

The flow chart of the optimization program is shown below, where **N** represents the number of the designs, $\mathbf{N_{max}}$ the maximum number of designs prescribed, **T** the annealing temperature and $\mathbf{T_{min}}$ the prescribed minimum annealing temperature.

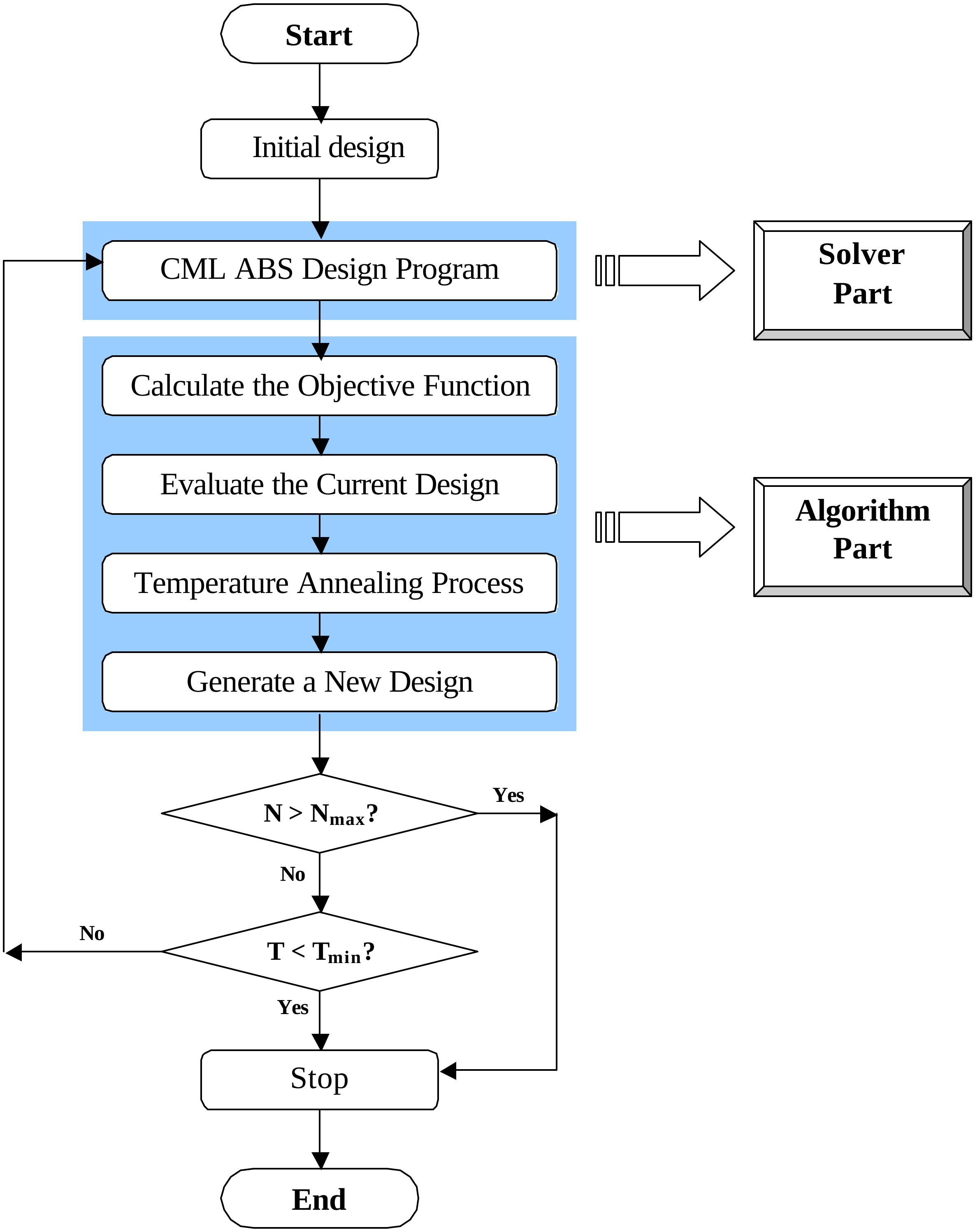


Fig.11 Flow chart of the CML optimization program

4. AIR BEARING DESIGN OPTIMIZATION PROBLEM

The optimization problem defined here is: given a prototype slider ABS design, optimize it to get uniform flying heights near the target flying height and at flat roll profile. Also increase its air bearing stiffness if possible.

Here the NSIC 7nm flying height slider is used as the prototype slider. The rail shape and the 3-dimensional rail geometry are shown in Fig. 12 and Fig. 13, respectively.

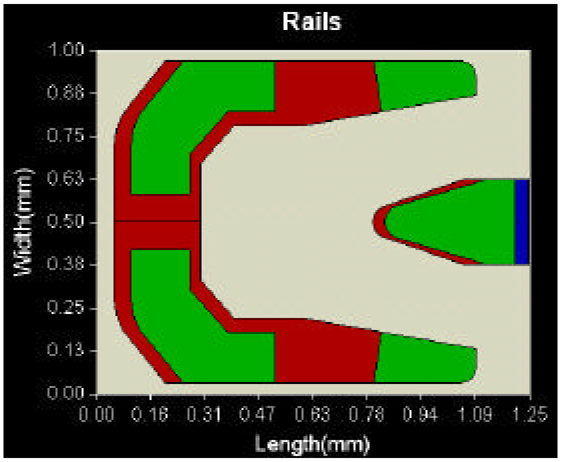


Fig.12 Rail shape of the initial ABS design

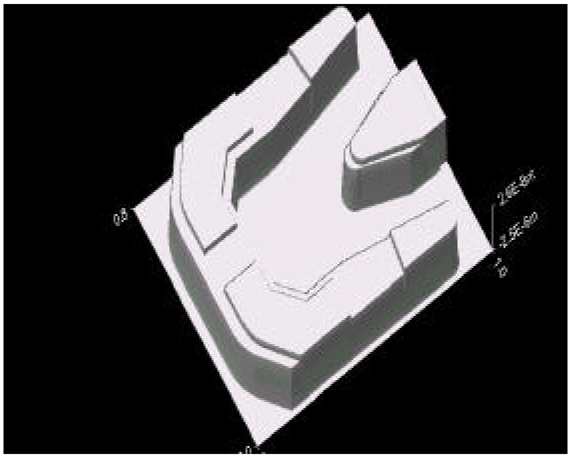


Fig.13 3-D rail shape of the initial ABS design

The slider is a Pico slider (1.25×1.0mm), which flies over a disk rotating at 7200 RPM. Its flying heights are all around 7nm from OD to ID. Now we want to lower its flying heights to the target flying height, i.e. 5nm and at the same time maintain a flat roll profile at the three different radial positions OD, MD and ID. The objective function or cost function is defined as:

1*´ (FH Max Difference term) +* ***9*** *´ (FH term) +* ***1*** *´ (Roll term) +* ***1*** *´ (Roll Cutoff term) +* ***1*** *´ (Pitch Cutoff term) +* ***1*** *´ (Vertical Sensitivity term) +* ***1*** *´ (Pitch Sensitivity term) +* ***1*** *´ (Roll Sensitivity term) +* ***1*** *´ (Negative Force term)*

So the goal of the optimization is to minimize this multi-objective function under the given constraints. Note that since we are primarily concerned with the flying heights, we put a heavier weight (9) on that term. All the objective terms are normalized and their definitions can be found in the "CML optimization program version 2.0 user's manual". The constraints we defined here are shown in the following figure. The definition of these constraints can also be found in the user's manual.

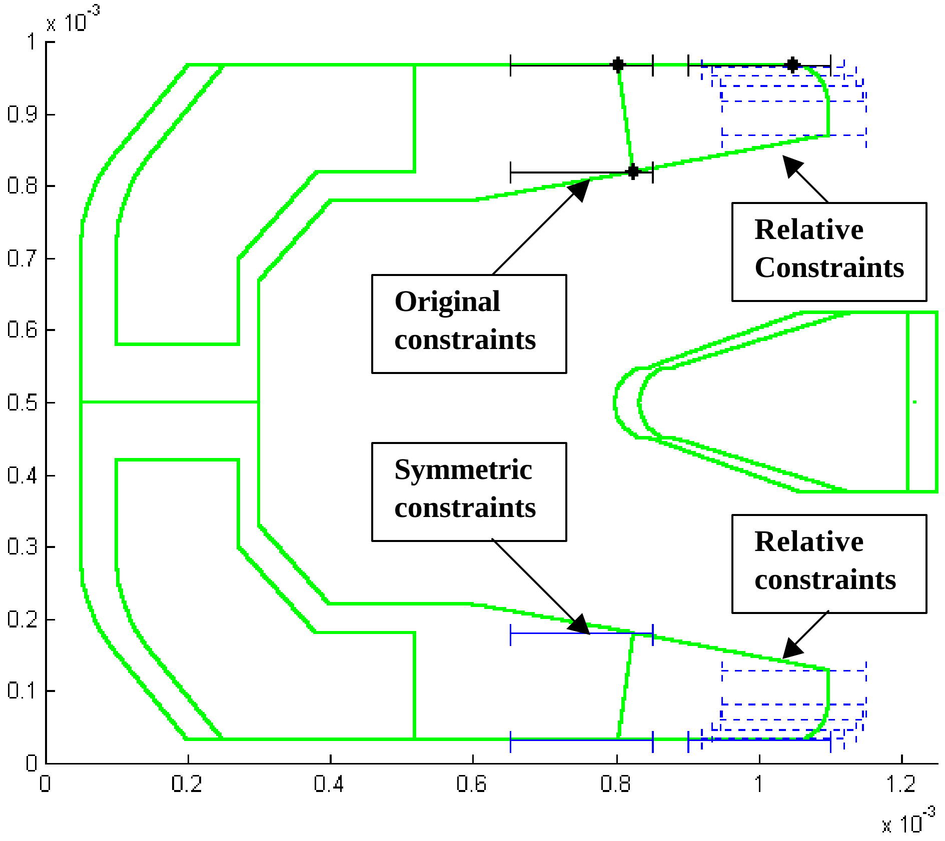


Fig.14 Constraints defined on the initial design

5. SIMULATION RESULTS

With the same initial design, constraints and objective function, we carried out the optimization using the BA, FA and ASA respectively.

The following three figures show the variation of the objective function values during the optimization process for BA, FA and ASA respectively.

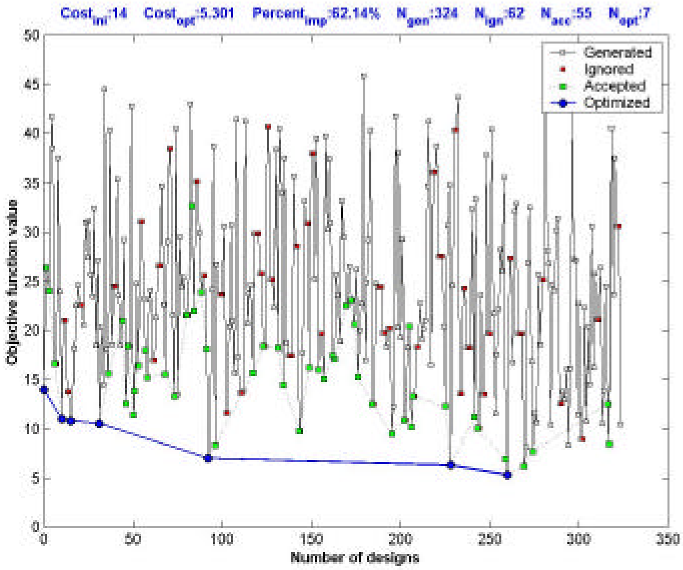


Fig. 15 Variation of the objective function value for BA

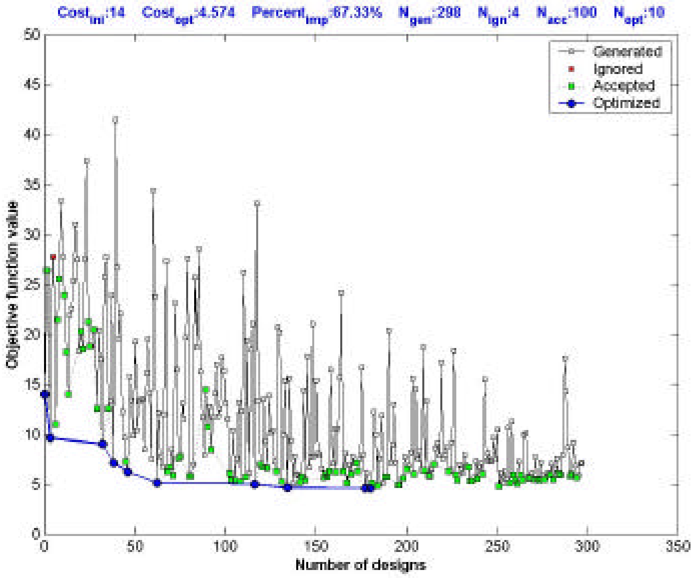


Fig. 16 Variation of the objective function value for FA

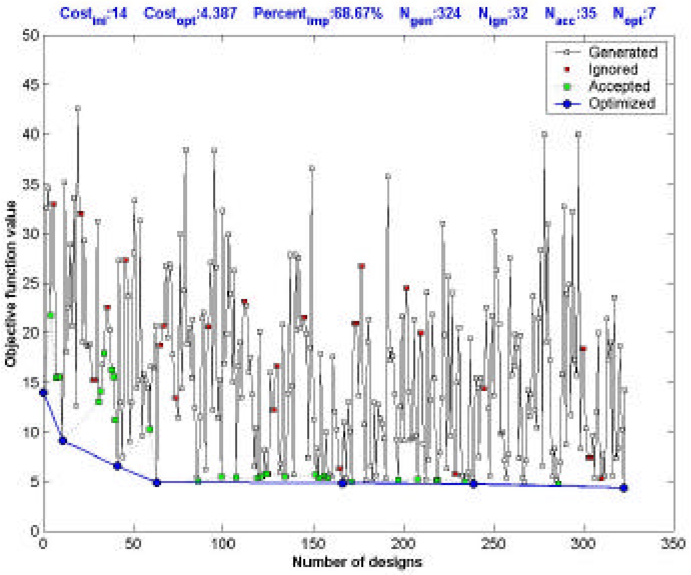


Fig. 17 Variation of the objective function value for ASA

In all of the above figures, $Cost_{ini}$ means the initial objective function value, and $Cost_{opt}$ means the objective function value for the final optimized design. The $Percent_{imp}$ signifies the percentage of improvement for the cost function value which is defined as:

$$Percent_{imp} = \frac{Cost_{ini} - Cost_{opt}}{Cost_{ini}} \times 100\%$$

N_{gen}, N_{ign}, N_{acc}, N_{opt} in these figures represent the number of the designs generated, ignored, accepted and optimized, respectively.

The dark circles represent the optimized designs generated during the process. The optimized designs are the ones with the best-so-far objective function values. We know that the lower the objective function value, the better the design.

The objective function values for the three final optimized designs by using BA, FA and ASA, respectively, are 5.301, 4.574 and 4.387. That means ASA obtained the best optimized design in this case. Also, ASA had the fastest convergence rate. This can be verified by the following figure, which shows the cost temperature variation during the optimization process. ASA had the fastest cooling rate while BA had the slowest one.

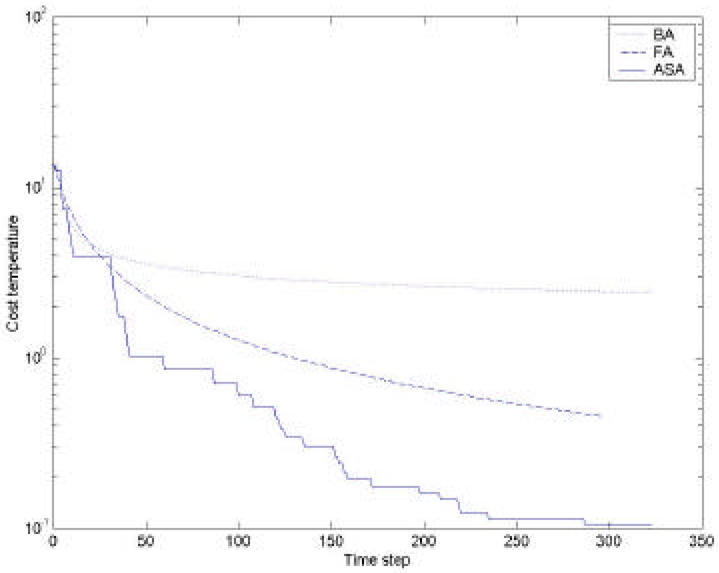


Fig.18 Cost temperature variations for BA, FA and ASA

The so-called "freezing" phenomenon for FA can be observed in Fig. 16. After certain stages, in this case, after generating about 200 designs, the fluctuation of the cost function value becomes much smaller. This indicates the parameter temperature is so low that the newly generated designs are very small perturbations from the previously accepted design. Thus the new designs have been "frozen" there. This is due to the fast cooling rate of FA. (This phenomenon is not observed for BA at this stage, since its cooling rate is quite slow) The question arises: f ASA has an even faster cooling rate than FA, why was there no freezing phenomenon for ASA? The answer is that ASA has an adaptive re-annealing process. Different parameters will have different cooling rates according to their different sensitivities. For those parameters with lower sensitivities, their temperatures get raised to let them vary more freely in the following stages. So this mechanism actually lets ASA avoid the "freezing" phenomenon while maintaining its fast cooling rate. Therefore ASA is more efficient and more robust than BA or FA.

The comparison between the initial and optimized designs produced by BA, FA and ASA are shown in the following three figures, in which the green lines (light-colored) show the rail shape of the initial design and the blue lines (deep-colored) show the rail shape of the optimized design. Similar results are obtained by all these algorithms.

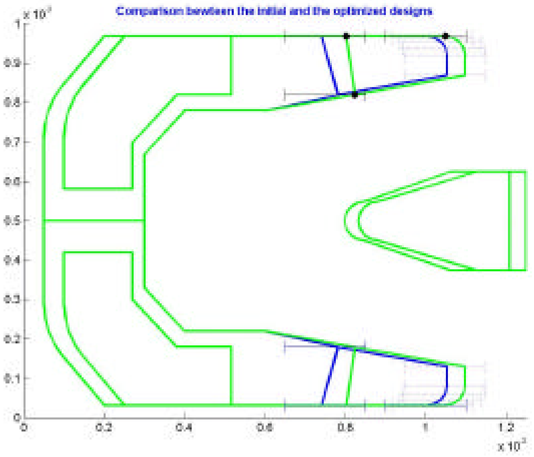


Fig.19 Optimization results by using BA

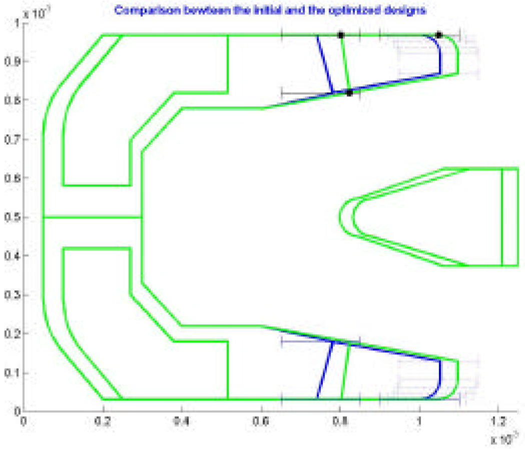


Fig.20 Optimization results by using FA

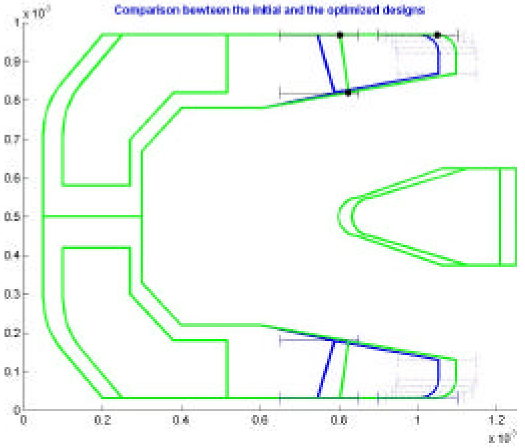


Fig.21 Optimization results by using ASA

We also show the variation of the objective function terms for the three optimized designs by using BA, FA and ASA in the following figures.

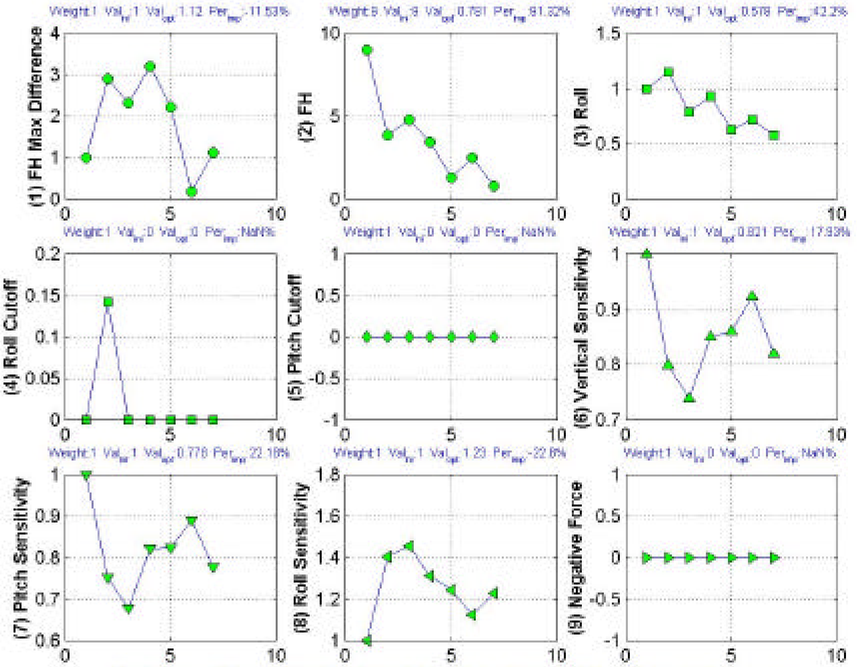


Fig.22 Variation of the objective function terms by using BA

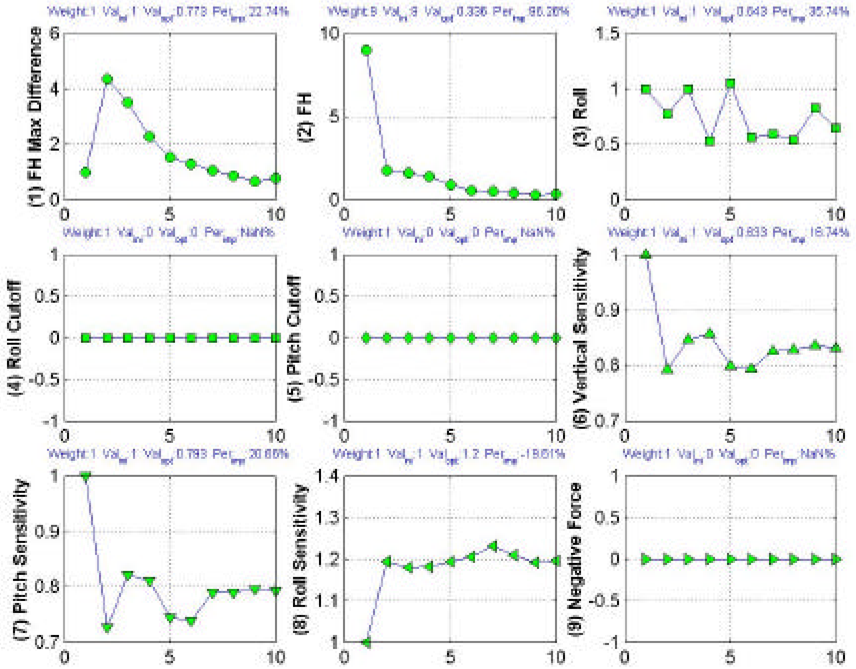


Fig.23 Variation of the objective function terms by using FA

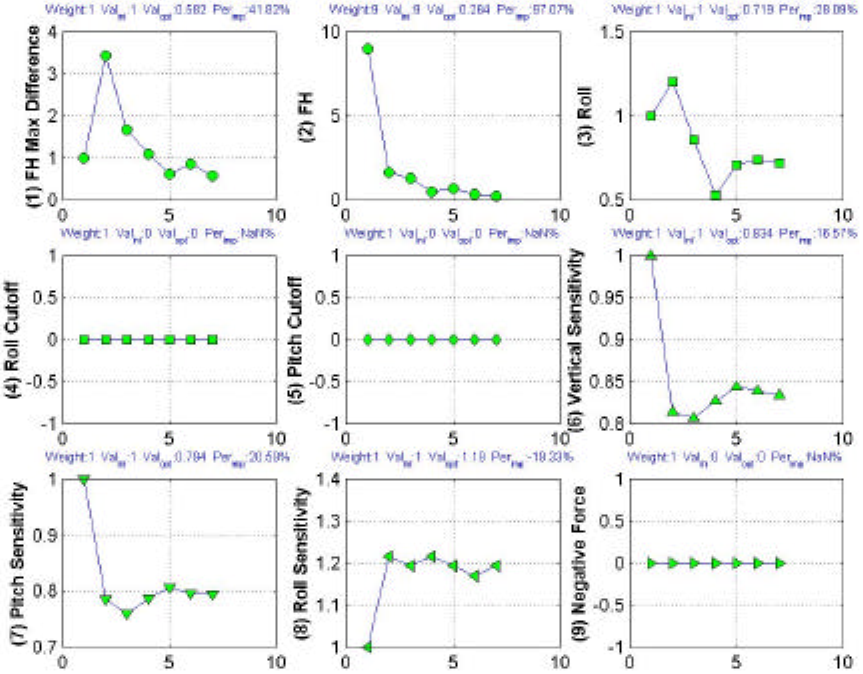
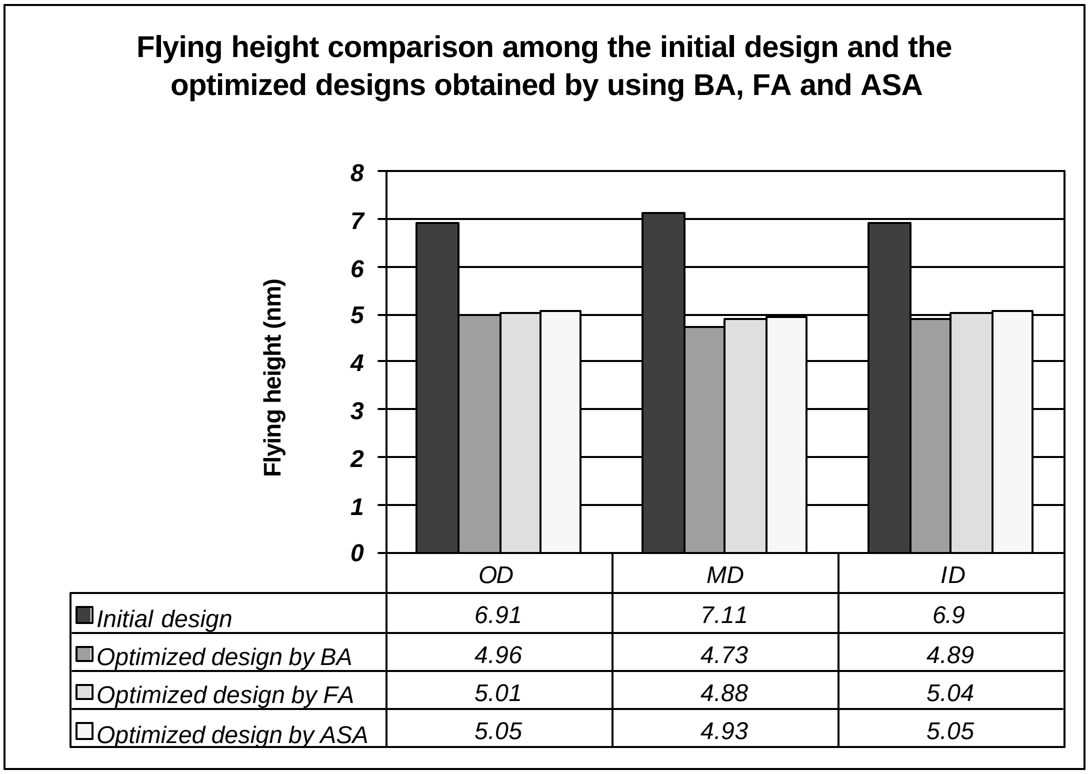


Fig.24 Variation of the objective function terms by using ASA

All three algorithms provide impressive minimization in the Flying Height term, i.e. the 2nd objective function term, which was weighted more heavily. There was also improvement for the roll term as well as some improvement on the Vertical Sensitivity term and the Pitch Sensitivity term. But the Roll Sensitivity was not improved. Some objective terms such as the Pitch cutoff term and Negative Force term remained zero for all of the optimized designs. The combinatorial effects are the minimization of the total value of the objective function. By minimizing the multi-objective cost function we obtained the final optimized designs.

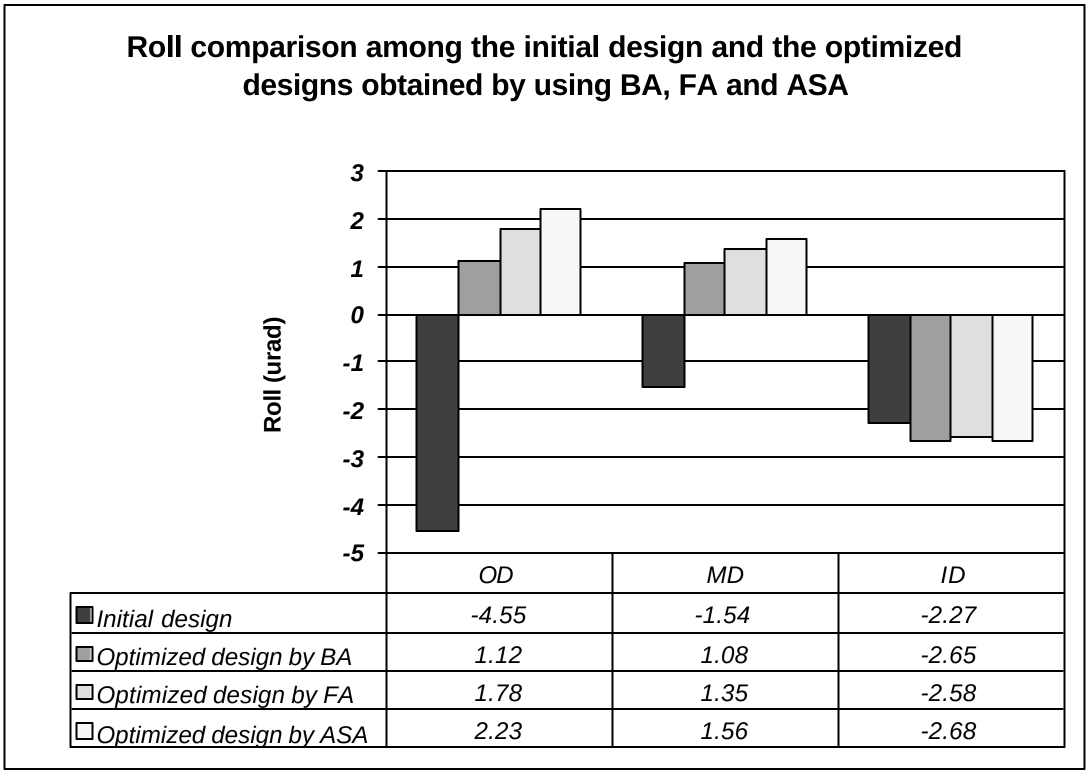
The comparison of the performance parameters of most concern, i.e. the flying heights and the rolls, are given in the following two figures.

It is clear that all of the optimized ABS designs have quite constant flying heights around the target flying height, which is 5nm. Also they all maintain a reasonably flat roll profile. The optimized design obtained by using ASA has the most uniform flying height profile and the best overall performance.



	OD	MD	ID
■ Initial design	6.91	7.11	6.9
▩ Optimized design by BA	4.96	4.73	4.89
▨ Optimized design by FA	5.01	4.88	5.04
□ Optimized design by ASA	5.05	4.93	5.05

Fig.25 Flying height distribution from OD to ID for different designs



	OD	MD	ID
■ Initial design	-4.55	-1.54	-2.27
▩ Optimized design by BA	1.12	1.08	-2.65
▨ Optimized design by FA	1.78	1.35	-2.58
□ Optimized design by ASA	2.23	1.56	-2.68

Fig.26 Roll distribution from OD to ID for different designs

6. CONCLUSION

Slider ABS designs that satisfy very strict multi-objective goals are of great importance for the performance of magnetic hard disk drives. This is a strongly non-linear problem.

Use the simulated annealing optimization technique, which is a global stochastic optimization method, provide the optimized designs automatically for a given initial design and constraints.

By putting different weights on different objective function terms, the objective function steers the designs to its goals.

Three main members of the simulated annealing family, namely the Standard Boltzmann Annealing (BA), the Fast Cauchy Annealing (FA) and the Adaptive Simulated Annealing (ASA), were shown to produce similar optimized ABS designs with greatly improved performance, i.e. uniform flying heights around the target flying height, flat rolls and improved stiffness. This illuminates that the simulated annealing algorithm is quite suitable for the optimization of the ABS designs.

The ASA was found to be the most efficient and robust scheme due to its fastest cooling schedule and its unique adaptive re-annealing mechanism. These features gave it the fastest convergence rate and let it effectively avoid the "freezing" phenomenon, which is generally a side-effect of the fast cooling rate. FA also had a faster convergence rate than BA.

Among the optimized designs obtained by using BA, FA and ASA, the one obtained by ASA had the most uniform flying height profiles and the smallest objective function value, which means its design had the best overall performance.

ACKNOWLEDGEMENTS

This study is supported by the Computer Mechanics Laboratory (CML) at the University of California at Berkeley and partially supported by the Extremely High Density Recordings (EHDR) project of the National Storage Industry Consortium (NSIC).

REFERENCES

1. O'Hara, M., 1997,"*Optimization of Hard Disk Drive Components*", Ph.D. Thesis, University of California at Berkeley.

2. O'Hara, M. and Bogy, D., 1997, "*The CML Air Bearing Optimization Program Version 1.5*", Technical Report, Computer Mechanics Laboratory, University of California at Berkeley.

3. Zhu, H. and Bogy, D., 2000, "*The CML Air Bearing Optimization Program Version 2.0*", Technical Report, Computer Mechanics Laboratory, University of California at Berkeley.

4. Daniel, E., Mee, C. and Clark, M., 1999, "*Magnetic Recording, The First 100 Years*", IEEE, New York.

5. Tompson, D., 2000, "*The future of magnetic data storage technology*", IBM J. Res. Develop., Vol. 44, No. 3, pp311-321.

6. Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E., 1953, "*Equation of state calculations by fast computing machines*", J. Chem. Phys., 21, pp1087-1092.

7. Geman, S. and Geman, D., 1984, "*Stochastic relaxation, Gibbs distribution and the Bayesian restoration in images*", IEEE Trans. Patt. Anal. Mac. Int., 6, pp721-741.

8. Szu, H. and Hartley, R., 1987, "*Fast simulated annealing*", Phys. Lett. A 122, 3-4, pp157-162.

9. Ingber, L., 1989, "*Very Fast Simulated Re-Annealing*", J Math. Comput. Modelling, V12, pp967-973.

10. Ingber, L., 1993, "*Simulated annealing: Practice versus theory*", Math. Comput. Modelling, 18, 11, pp 29-57.

11. Ingber, L., 1996, "*Adaptive Simulated Annealing (ASA): Lessons learned*", Journal of Control and Cybernetics, Vol.25, pp33-54.

12. Rosen, B., 1992, *"Function optimization based on advanced simulated annealing*", IEEE Workshop on Physics and Computation PhysComp '92, pp 289-293.

13. Cohen, B., 1994, "*Training synaptic delays in a recurrent neural network*", M.S. Thesis, Tel-Aviv, Israel, Tel-Aviv University.

14. Sen, M. and Stoffa, P., 1995, "*Global Optimization Methods in Geophysical Inversion*", ELSEVIER, Amsterdam – Lausanne – New York – Oxford – Shannon – Tokyo.

15. Vanderplaats, G., 1984, "*Numerical Optimization Techniques for Engineering Design with Applications*", McGraw – Hill Book Company.

16. Bird, G., 1976, "*Molecular Gas Dynamics*", Oxford: Clarendon Press.

Appendix A

Sampling from a prescribed distribution [16]

During the process of optimization, the generation of new design requires the generation of representative values of variables that are distributed in a prescribed manner. This is done through random numbers and is a key step in the implementation of simulated annealing algorithms.

Here we assume u is a random number which is uniformly distributed between 0 and 1, i.e. u∈ U[0,1].

The distribution of the variable x can be described by a normalized probability density function g(x) such that the probability of a value of x lying between x and x+dx is given by g(x)dx.

If x∈ [a, b], then the total probability is

$$\int_a^b g(x)dx = 1 \quad .$$

Now let's define the cumulative distribution function G(x) as

$$G(x) = \int_a^x g(x)dx \quad .$$

Then we may invert the cumulative distribution function G(x) and get

$$x = x(G) \quad .$$

Next we can generate a random number u and set it equal to G. Then the representative value of x is given by

$$x = x(u) \quad .$$

This method is therefore referred as the **inverse-cumulative method**. The operation of this method is shown graphically in Fig. A1.

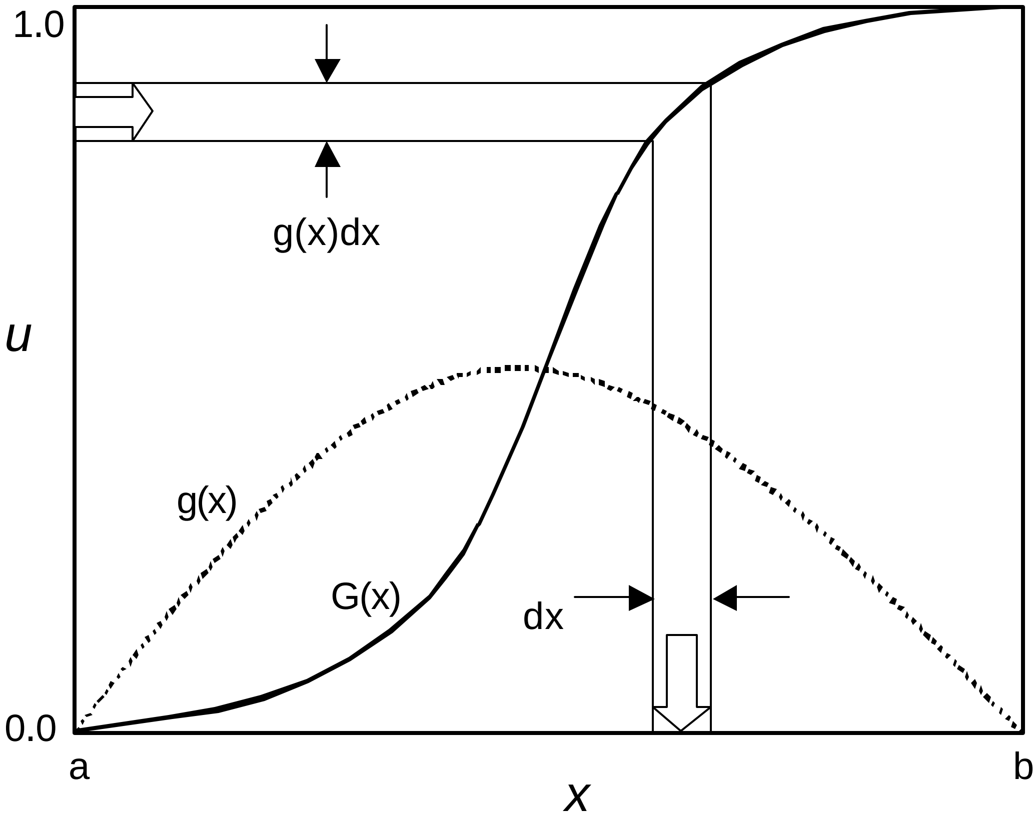


Fig. A1 Relationship between the typical normalized probability density function g(x) and the cumulative distribution function G(x)

First let's consider a trivial example in which the x is uniformly distributed between a and b. For this case g(x) is a constant and we have

$$g(x) = 1/(b-a) \ .$$

So, from the above we get

$$G(x) = \int_a^x g(x)dx = \frac{x-a}{b-a} \ .$$

Inverting G(x), we have

$$x = a + G \times (b - a)$$.

Then let G be equal to a random number u∈ U[0,1] to get

$$x = a + u \times (b - a)$$.

For the case of the Adaptive Simulated Annealing algorithm (ASA), the probability density function is defined as

$$g_T(x) = \prod_{i=1}^{D} g_T^i(x^i) = \prod_{i=1}^{D} \frac{1}{2(\left|x^i\right| + T_i)\ln(1 + \frac{1}{T_i})}$$,

where T stands for the annealing temperature and D the dimension of parameter space. We can easily verify that

$$\int_{-1}^{1}\int_{-1}^{1}\cdots\int_{-1}^{1}\prod_{i=1}^{D} g_T^i(x^i)dx^1dx^2\cdots dx^D = 1$$.

Its cumulative probability function is

$$G_T(x) = \int_{-1}^{x^1}\int_{-1}^{x^2}\cdots\int_{-1}^{x^D}\prod_{i=1}^{D} g_T^i(x^i)dx^1dx^2\cdots dx^D \equiv \prod_{i=1}^{D} G_T^i(x^i)$$,

where

$$G_T^i(x^i) = \frac{1}{2} + \frac{\operatorname{sgn}(x^i)}{2}\frac{\ln(1 + \frac{\left|x^i\right|}{T_i})}{\ln(1 + \frac{1}{T_i})}$$.

Then as before, we can generate a set of random numbers u^i from the uniform distribution u^i∈ U[0,1] . After inverting the cumulative distribution function and letting

$$G_T^i(x^i) = u^i\ ,$$

we get

$$x^i = \mathrm{sgn}(u^i - \tfrac{1}{2})T_i[(1+\frac{1}{T_i})^{\left|2u^i-1\right|} - 1]\ .$$

So far we have discussed the inverse-cumulative method and its applications. However, this method can be used only when it is possible to invert the cumulative distribution function G(x) to obtain an explicit function for x. But sometimes it's impossible to obtain the inverse cumulative distribution function, if for example, the probability density function is

$$g(x) = \frac{1}{\sqrt{2p}} e^{\frac{-x^2}{2}}\ ,$$

then

$$G(x) = \int_{-\infty}^{x} g(x)dx = \frac{1}{2} + \frac{1}{2} erf\left(\frac{x}{\sqrt{2}}\right)\ .$$

This expression can't be inverted to give x in terms of G and thus the inverse-cumulative method fails.

The general alternative is to apply the **acceptance-rejection method**. In order to make direct use of the random number u, the probability function is normalized by dividing it by its maximum value g$_{max}$ to give

$$\hat{g}(x) = \frac{g(x)}{g_{\max}} \quad .$$

A value of x is then chosen at random on the basis of x being uniformly distributed between its limits, i.e.

$$x = a + u \times (b - a) \quad .$$

The function $\hat{g}(x)$ is then calculated for this value of x and a second random number is generated. The value of x is then either accepted or rejected according to whether $\hat{g}(x)$ is greater or less than this second random number. This procedure is repeated until a value of x is accepted. Since the random number u is uniformly distributed between 0 and 1, the probability of a particular value of x being accepted is clearly proportional to $\hat{g}(x)$ and the accepted values conform to this distribution.

For the Boltzmann Annealing (BA) and Fast Annealing (FA) of the simulated annealing algorithm family, when the inverse-cumulative method fails because we can't invert their cumulative distribution functions, we can use the acceptance-rejection method.

For the Boltzmann Annealing, the probability density function is

$$g_T(x) = (2pT)^{\frac{-D}{2}} e^{\frac{-x^2}{2T}} \quad .$$

So

$$\hat{g}_T(x) = \frac{g_T(x)}{g_{T\max}} = e^{\frac{-x^2}{2T}} \quad .$$

Then the first random number $u_1 \in U[0,1]$ can be generated. If we denote η_0 as the previous point and η as the new point, and define the range of this point as [a, b], then

$$h = a + u_1(b - a) \quad .$$

From this value of η we can obtain x since x is defined as

$$x \equiv \frac{h - h_0}{b - a} \in [-1,1] \quad .$$

Then the value of $\hat{g}_T(x)$ follows from x. Next the second random number $u_2 \in$ U[0,1] is generated and compared with the value of $\hat{g}_T(x)$. If $\hat{g}_T(x)$ is greater than u_2 , the x value gets accepted. Otherwise the x value is rejected and the above procedure is repeated until a value of x is accepted.

For Fast Annealing, the probability density function is

$$g_T(x) = \frac{T}{\left(x^2 + T^2\right)^{\frac{D+1}{2}}} \quad .$$

By following the above procedure we again get the value of x according to this prescribed probability density distribution.