Chapter II Ordinal Optimization Fundamentals

1 Two basic ideas of Ordinal Optimization (OO)

There are two basic ideas in OO:

- "Order" is much more robust against noise than "Value"
- Don't insist on getting the "Best" but be willing to settle for the "Good Enough"

Of course readers may rightly point out that these ideas are hardly new. Good engineers and designers do this all the time when confronted with difficult and complex problems of performance evaluation and optimization. Our contribution in this book is simply that we have developed a theory to **quantify** these two ideas. The practice of these two ideas is now knowledge-based instead of being experience-based. The expertise of a good designer acquired from experience will now be available to everyone who uses the tools discussed in this book. Moreover, the user will have numerical measures rather than just gut feelings. The quantification will come later in the book. For now let us simply explain the ideas in intuitive terms.

Idea No.1 Order is easier than Value. Imagine you hold two identically looking boxes with unknown content in your two hands. You are asked to judge which one is heavier than the other – an "ordinal" question. Almost all of us can correctly tell the answer, even if there is only a very slight difference between the weights of the two boxes. But if we are asked to estimate the difference in weight between the two boxes – a "cardinal" question, we will have a hard time. This is the essence of the first tenet of OO. In fact later on in this chapter we shall prove that "Order" converges exponentially fast in stochastic simulation models against the $1/(n)^{1/2}$ convergence rate of "Value" or "Confidence Interval" as discussed in chapter I.

An intuitive and graphical illustration of this idea is shown in Fig. 2.1.

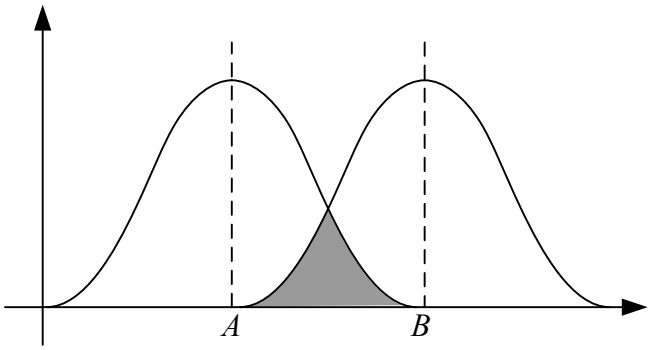

Fig. 2.1. Comparison of two values *A* and *B* corrupted by noise

We have two values *A* and *B*. But we can only observe the values through noises, which are zero mean and normally distributed. If we compare samples of these noisy values of *A* and *B*, then we will reach an incorrect answer, i.e., $A > B$, only when the samples of *A* is larger than that of *B*. This happens basically when the samples fall in the shaded triangle area. This area decreases rapidly as the values of *A* and *B* are drawn apart.

Idea No.2 Nothing but the best is very costly. The second tenet of OO rests on the idea of relaxing the goal of performance evaluation. We are asked to retreat from "nothing but the best" to a softer goal of being "good enough", e.g., settle for anything in the top-*g* choices. The small retreat can buy us quite a bit in the easing of the computational burden. Again the quantification will come later on in the chapter. However, an intuitive explanation is helpful to fix ideas at this point. When we are searching only for the best in Θ, there is only one solution (unless of course in the less common cases when there are multiple optima), and we can easily fail (i.e., the truly best design is not the observed best one). But if we are willing to settle for any one of the top-2 choices, we fail only when neither design is within the observed top-2, and if we are willing to settle for any of the top-*g* choices, there will be at least *g*! satisfactory alternatives, the possibilities increases superlinearly. More technically, for $g = 2$, the probability of which is almost the square of the previous failure probability for a large enough Θ; for top-3, we fail only when none of these three designs is within the observed top-3, the probability of which is almost the cube of the first failure probability; and for top-*g*, where *g* is small compared to the size of Θ, we fail only when none of these *g* designs is within the observed top-*g*, the probability of which is almost the first failure probability to the power of *g*.

Exercise 2.1: Prove the above statements concerning the failure probabilities.

This exponential decrease of failure probability (thus the exponential increase in successful probability) contributes to significant decrease in search, and hence computational cost. Another way of clarifying the advantage of this idea is to observe that independent noisy observation or estimation error may worsen as well as help the performance "order" of a particular choice of θ. Thus, as a group, the top-n choices can be very robust against perturbations in order so long as we don't care about the exact order within this group.

Lastly, it is not hard to convince oneself that the ease of computational burden through these two ideas is not additive but multiplicative since they are separate and independent factors. Together, as we shall demonstrate throughout this book, they produce orders of magnitude improvement in efficiency. Problems previously thought to be infeasible computationally are now within reach.

2 Definitions, terminologies, and concepts for OO

We first introduce some definitions, which will be used throughout this book

$\boldsymbol{\theta}$	The various system parameters that may be subject to design choices.
$\boldsymbol{\Theta}$	The search space for the optimization variables θ. We can simply assume it to be a very large but finite set consisting of zillions of choices.
$\boldsymbol{J}$	The performance criterion for the system as already defined in Eqs. (1.1) and (1.2).
$\hat{J}$	The estimated or observed performance for the system. This is usually done by using a crude but computationally easy model of the system performance.
$\boldsymbol{w}$	The observation noise/error, which describes the difference between the true performance and the estimated/observed performance, i.e., $\hat{J}(\theta) = J(\theta) + w$.
$\boldsymbol{\xi}$	All the random variables with known distribution used in the simulation model.

***x*(*t*; *θ*, *ξ*)** A sample trajectory of the system given the design θ and the realizetion ξ. In simulation terminology, this is often called a **replication**.

L A functional of $x(t; \theta, \xi)$ that defines the performance metric of the system on a sample path. The system's performance criterion is given by the expectation $J = E[L(x(t; \theta, \xi))]$.

N The number of designs uniformly chosen in Θ. It is understood that for each design θ, there corresponds a value of J as defined in Eqs. (1.1) and (1.2). When there is no danger of confusion we also use N to denote the set of designs chosen.

G The Good Enough set, usually the top-g designs of Θ or of N.

S The set of selected designs in N, usually the estimated top-s of N.

Selection Rule

The method that is used to determine the set S, e.g., it could be **blind pick** or **horse race**[1] or other rules using the estimated performance to order the designs (based on a crude model).

G*∩*S The truly good enough designs in S.

The relationship among Θ, G, S and the true and estimated optimal design are shown in Fig. 2.2. Note the sets G and S in OO play the analogous role as the true optimum and the estimated optimum respectively in regular optimization.

AP, Alignment Probability ≡ Prob[|*G*∩*S*|≥*k*]

The probability that there are actually k truly good enough designs in S (represented by the dotted area in Fig. 2.2 above). k is called the **alignment level**. This number quantifies how a crude model can help to assure the determination of "good enough" designs.

OPC, Ordered Performance Curve

A (conceptual) plot of the values of J as a function of the order of performance, i.e., the best, the second best, and so on. If we are minimizing then the OPC must be a non-decreasing curve[2].

---

[1] The blind choice selection rule is to arbitrarily and randomly selected s θ's from N as the set S. The horse race selection method is simply to compare the observed performance values, $\hat{J}_1, \hat{J}_2, \ldots, \hat{J}_N$ and select the designs with top-s observed values as the set S. More details can be found in Chapter III.

[2] In this book, we usually deal with finite but a large number of designs. For visualization, we show continuous curves in Fig. 2.3 and in some of the following figures, instead of a large number of discrete points. These figures can be regarded as limits of discrete points with high density.

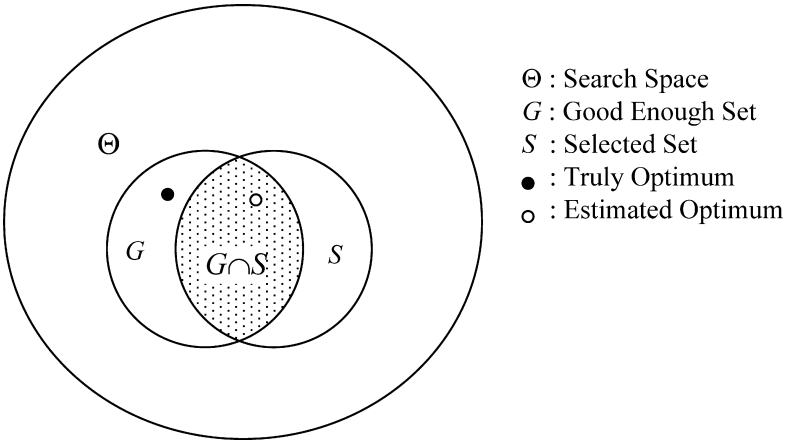

Fig. 2.2. Graphical Illustration of Θ, *G*, and *S*

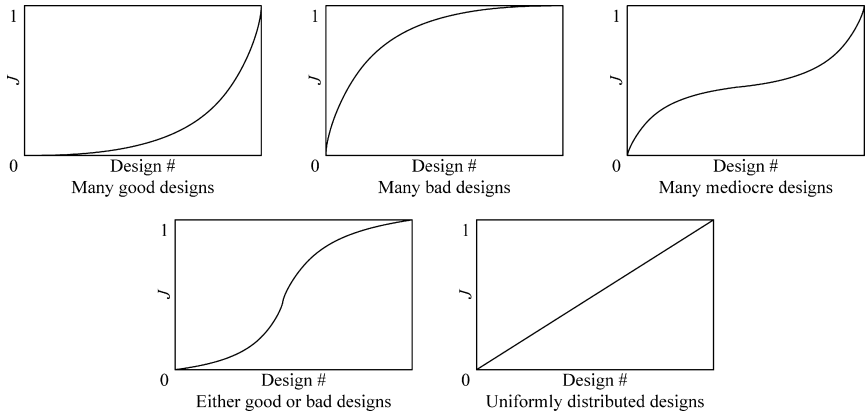

Fig. 2.3. Different types of OPC

C denotes the class or type of OPC for the problem. For non-decreasing curves, there are only five general types as shown in Fig. 2.3 below.

The fact that "order" is robust against noise (as will be shown shortly) gives us the possibility that we may use a crude model to determine the order of various designs. In other words, we visualize and set up

$$J_{\text{complex simulation model}} = J_{\text{crude model}} + \text{noise/error}^{3}. \tag{2.1}$$

In terms of Eq. (1.2) for the simulation of a complex model, the number of replications, n, is very large. However n is very small and may even be equal to one for the crude model.

Given Eq. (2.1), we define

σ^2 as the noise/error level. In simulation, this is related to the width of the confidence interval of the performance estimate. In practice we do not need to know the value of σ^2 very accurately, but whether or not the approximation of the crude model to the complex model is very bad, bad, or moderate. By definition the approximation cannot be good. For otherwise, there will not be a problem.

UAP, the Universal Alignment Probability $\equiv \text{Prob}\left[\left|G \cap S\right| \geq k \middle/ N, \sigma^2, \mathcal{C}\right]$ **≡ UAP(*N*, σ^2, $\mathcal{C}$)**

We assert here, and will establish later (in Section 5), that once N, σ^2, $\mathcal{C}$ are fixed, the alignment probability becomes independent of the specific problem under consideration. For the fixed size of G, it is possible to tabulate the required size of S in order to insure that the alignment probability, AP, is no less than a certain high value, say 0.95.

Here is an example to help picture the concepts of AP. In the world of professional tennis, players are ranked as the #1 seed, #2 seed, etc. Let us assume that as the true ranking and take the first 16 seeds as the set G. In a given tennis tournament, e.g., the U.S. Open, 16 players will reach the quarter-finals through elimination. These 16 players, thus selected, constitute the set S. And elimination tournament is the selection rule used. Without consulting a sports almanac, we can be reasonably sure that there will be a significant overlap between G and S. For example, even without any knowledge of the tennis world, we can be fairly certain that the $\text{Prob}[|G \cap S| \geq 1] \approx 1$. Furthermore, such near certainty will exist in other sports such as horse race. Thus, we claim the use of the adjective "universal" for such APs which are tabulated as a function of N, σ^2, and $\mathcal{C}$. In fact we shall show in later chapters how UAPs can be used to narrow down searches for the "good enough". There is also a simple demonstration

[3] It will be introduced later (in Section 6) that how to use OO to deal with different simulation models, i.e., both stochastic simulation models and deterministic complex simulation models. To cover both types of models, we use "noise/error" here. In Section 6 we will show that noise and error can be regarded as the same under the concept of Kolmogorov complexity.

explained immediately in Section 3 below which further illustrates the universality of the concept of alignment probability.

Finally, we summarize the spirit of ordinal optimization as:

Instead of the best for sure, we seek
the good enough with high probability.

3 A simple demonstration of OO

We present here a simple generic demonstration that everyone can do to convince themselves of the validity of the two basic tenets of OO as discussed in Section 1 and the universality of the alignment probability of OO. Let the search space Θ have 200 designs, say 1,2, …, 200. Without loss of generality, we let choice #1 be the best, #2, the second best, and so on to #200; and the performances for simplicity be 1, 2, ..., 200[4]. Mathematically, we have $J(\theta_i) = i$ in which case the best design is θ_1 (we are minimizing) and the worst design is θ_{200} and the Ordered Performance Curve (OPC) is linearly increasing. For any finite i.i.d noise w, we can implement $\hat{J}(\theta) = J(\theta) + w$ and directly observe these performances through noisy measurements as in Eq. (2.1) with noise distributed as $U[0,100]$ or $U[0,10000]$. Let the good enough performance G be 1, . . ., 12 (the top-6%), and S be the observed top-6%. We are interested in $|G \cap S|$.

All of these can be simply implemented in a spreadsheet with Θ represented by 200 rows as in Fig. 2.4.

Design # = θ	True performance $J(\theta)$	Noise $w \in U[0,W]$	Observed performance $J(\theta)+w$
1	1.00	87.98	88.98
2	2.00	1.67	3.67
.	.	.	.
.	.	.	.
.	.	.	.
199	199.00	32.92	231.92
200	200.00	24.96	224.96
			Sort on this column in ascending order

Fig. 2.4. Spread sheet implementation of generic experiment

[4] Actually any monotonically increasing numerical sequence will do.

Column 1 models the N (=200) alternatives and the true order 1 through N. Column 2 shows the linearly increasing OPC from 1 through N (=200). The rate of OPC increase with respect to the noise variance σ^2 essentially determines the estimation or approximation error of $\hat{J}(\theta)$. This is shown by the random noise generated in column 3 which, in this case, has a large range $U[0,100]$ or $U[0,10000]$. Column 4 displays the corrupted (or estimated or observed) performance. When we sort on column 4, we can directly observe the alignment in column 1, i.e., how many numbers 1 through g (=12) are in the top-g rows. For example, in Fig. 2.5 we show what the sorting result of the table in Fig. 2.4 may look like. Try this and you will be surprised! It takes less than two minutes to setup on Excel.

Design # = θ	True performance $J(\theta)$	Noise $w \in U[0,W]$	Observed performance $J(\theta)+w$
2	2.00	1.67	3.67
5	5.00	20.12	25.12
.	.	.	.
.	.	.	.
.	.	.	.
90	90.00	79.09	169.09
193	193.00	90.85	283.85

Fig. 2.5. The spread sheet in Fig. 2.4 after sorting on the observed performance in ascending order (Note: Column 4 is completely sorted in ascending order)

An already implemented version can also be found on http://www.hrl.harvard.edu/~ho. One can also simulate the result of blind pick by making the noise $U[0, 10000]$ and repeat the Excel simulation. It should be noted that the above demonstration is rather general. Except for the assumption of independent noise/error in Eq. (2.1), the results are applicable to any complex computational problems when approximated by a crude model. Chapters below will further discuss and exploit this generality.

At this point, except for establishing more carefully the validity of the two basic ideas of ordinal optimization (in the rest of this chapter) and other extensions (in the future chapters), we already have the procedure for the practical application of OO to complex optimization problems. This procedure is basically an elaboration of the demo above which we re-stated here for emphasis (Box 2.1):

Box 2.1. The application procedure of OO

Step 1:	Uniformly and randomly sample N designs from Θ.
Step 2:	Use a crude and computationally fast model to estimate the performance of these N designs.
Step 3:	Estimate the OPC class of the problem and the noise level of the crude model. The user specifies the size of good enough set, g, and the required alignment level, k.
Step 4:	Use Table 2.1 (in Section 5 below) to calculate $s = Z(g,k/\text{OPC class, noise level})$.
Step 5:	Select the observed top s designs of the N as estimated by the crude model as the selected set S.
Step 6:	The theory of OO ensures that S contains at least k truly good enough designs with probability no less than 0.95.

Section 7 gives two examples of application of this procedure to complex problems. Interested reader may go directly to that section. Case studies of more complex real world problems can be found in chapter VIII.

4 The exponential convergence of order and goal softening

In this section, we provide the theoretical foundation of ordinal optimization method, namely, the alignment probability converges exponentially with respect to the number of replications (Section 4.2) and with respect to the sizes of good enough set and selected set (Section 4.3). The proofs can be viewed as a more formal explanation on the two tents of OO: optimization can be made much easier by order comparison and goal softening. As a preparation for the proof of exponential convergence w.r.t. order, we introduce the large deviation theory first (Section 4.1). The large deviation theory justifies why order comparison of two values A and B corrupted by noise is easy as we pointed out in Section 2.1. Readers who are not that interested in mathematical details and are willing to accept the intuitive idea introduced in Section 1 can skip or skim this section on a first reading.

The formal verifications in this chapter are given for the most frequently used selection rule—horse race rule. In OO, recalling that for the horse race rule, we are interested in the alignment probability (AP) that the observed top-s designs (estimated good enough designs) contain at least k of the actual top-g designs (real good enough designs).

To establish the exponential convergence properties formally, we need to introduce the problem formulation first. Assume that the N designs are indexed such that

$$J(\theta_1) < J(\theta_2) < J(\theta_3) < \ldots < J(\theta_N).$$

Let $L(\theta_i,n)$ be the sampled performance for the n-th replication and assume that for each design θ_i, $L(\theta_i,1)$, $L(\theta_i,2)$,…, $L(\theta_i,n)$, … form a sequence of i.i.d. random variables with distribution such that for any $n>0$, $E[L(\theta_i,n)]= J(\theta_i)$. Let $\hat{\bar{J}}(\theta_i,n)$, $i=1,2,\ldots,N$, be the performance estimates such that

$$\hat{\bar{J}}(\theta_i,n) = \frac{1}{n}\sum_{j=1}^{n} L(\theta_i,j) = J(\theta_i) + w(\theta_i,n),$$

where $w(\theta_i,1)$, $w(\theta_i,2)$, …, $w(\theta_i,n)$, … are estimation errors and $E[w(\theta_i,n)] = 0$.

4.1 Large deviation theory

Let us consider an i.i.d. sequence $x_1,x_2,\ldots$ with distribution function F (or density function f) and finite mean μ. In our context, the numbers in this sequence are observations of performance of a given design. Let $a>\mu$ and $b<\mu$ be two constants. The law of large numbers implies that

$$\text{Prob}\left[\frac{x_1+\cdots+x_n}{n} \geq a\right] = \text{Prob}\left[x_1+\cdots+x_n \geq na\right] \to 0 \text{ as } n\to\infty$$

and

$$\text{Prob}\left[\frac{x_1+\cdots+x_n}{n} \leq b\right] = \text{Prob}\left[x_1+\cdots+x_n \leq nb\right] \to 0 \text{ as } n\to\infty.$$

A fundamental question is: how fast do these two probabilities decrease?

Although it seems that this question is about a single design, it is important to us since there is a natural way to reduce our order comparison problem to it. In Fig. 2.6, the deviation probability $\text{Prob}\left[\frac{x_1+\cdots+x_n}{n} \geq a\right]$ is

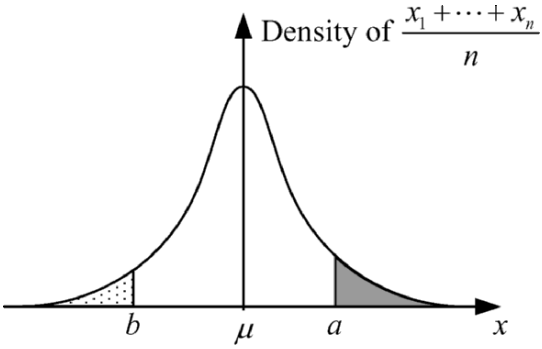

Fig. 2.6. Illustration of deviation probabilities

corresponding to the gray area and the deviation probability $\text{Prob}\left[\frac{x_1+\cdots+x_n}{n} \le b\right]$ the dotted area.

The connection between the deviation probabilities and the comparison of two fixed values A and B corrupted by noises can be interpreted as in Fig. 2.7 below. Assume $B > A$. Denote u as the position where the density functions of the two sample means meet. Then the rough estimation on misalignment probability (shaded area) in Fig. 2.1 (in Section 2.1) can be viewed as the sum of the gray area and the dotted area, where the gray area equals the deviation probability of A beyond u and the dotted area equals the deviation probability of B under u. Note u might change for a different n. A precise way of upper bounding the misalignment probability is to fix an amount of deviation less than or equal to $\delta = \frac{B-A}{2}$:

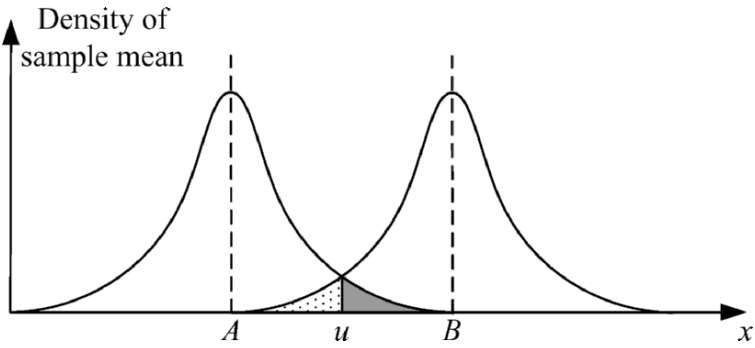

Fig. 2.7. Comparison of two values A and B corrupted by noises

Prob[Sample mean of observations of A> Sample mean of observations of B]
≤ Prob[Sample mean of observations of A>$A+\delta$]
+ Prob[Sample mean of observations of B<$B-\delta$].

The purpose of this subsection is to show that, for every constant $a > \mu$, there exists a positive β such that

$$\text{Prob}\left[x_1 + \cdots + x_n \geq na\right] \leq e^{-n\beta} \tag{2.2}$$

and, for each constant $b<\mu$

$$\text{Prob}\left[x_1 + \cdots x_n \leq nb\right] \geq e^{-n\beta}. \tag{2.3}$$

This implies that the probability for the sample mean $\frac{x_1 + \cdots + x_n}{n}$ to have finite deviation ("large deviation") from its mean decays exponentially. In the following, we shall show Eq. (2.2) and leave the similar justification of Eq. (2.3) to the readers. It is useful to define

$$M(s) \equiv E\left[e^{sx_1}\right] = \int e^{sy} dF(y) = \int e^{sy} f(y)\, dy .$$

Exercise 2.2: Let $x_1, x_2, \ldots$ be i.i.d. standard normal random variables, then please derive

$$M(s) = \frac{1}{\sqrt{2\pi}} \int e^{sy} e^{-\frac{1}{2}y^2} dy = e^{\frac{1}{2}s^2}.$$

$M(s)$ is known as the moment generating function (mgf) of the random variables x_i. $M(s)$ contains information of all order of moments, and especially we have $\mu = E\left[x_1\right] = M'(0) = \left.\frac{dM(s)}{ds}\right|_{s=0}$. It is interesting to note that $M(-s)$ is simply the Laplace transformation of the density function f. So, instead of giving the distribution function F or the density function f, the description of a random variable can also be characterized by its mgf $M(s)$. It is also natural to see that the mgf for the sum of independent random variables (r.v.s) is the product of mgfs of all r.v.s. In particular,

$\int e^{sy} dF_{x_1+\cdots+x_n}(y) = \left(\int e^{sy} dF(y)\right)^n = \left[M(s)\right]^n$, where $F_{x_1+\cdots+x_n}$ is the distribution of the r.v. $x_1+\ldots+x_n$.

For $s \geq 0$, mgf has the advantage of providing upper bounds on probability of events. In fact, we have

$$\text{Prob}\left[x_1 \geq b\right] \leq e^{-sb} M(s). \tag{2.4}$$

To see why this is true, we make the following observation.

$$\int_{y \geq b} f(y)\,dy \leq \int_{y \geq b} e^{s(y-b)} f(y)\,dy \leq \int e^{s(y-b)} f(y)\,dy. \tag{2.5}$$

The first inequality in Eq. (2.5) follows from the fact that $e^{s(y-b)} \geq 1$ when $s \geq 0$ in the range of integration $y \geq b$. The second inequality of Eq. (2.5) is due to the fact that $e^{s(y-b)} f(y)$ is always non-negative (recall f is a density function) and integration of a positive function over the entire region $(-\infty,+\infty)$ is always no less than integration over a part $[b,+\infty)$ of it . Thus Eq. (2.4) follows from Eq. (2.5) by noting

$$\text{Prob}\left[x_1 \geq b\right] = \int_{y \geq b} f(y)\,dy$$

and

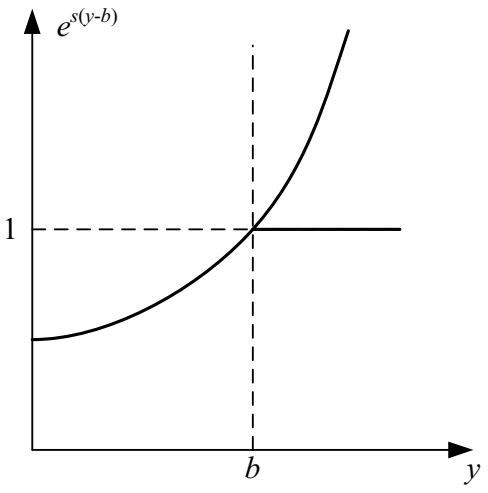

Fig. 2.8. Illustration of Eq. (2.5)

$$\int e^{s(y-b)} f(y)\,dy = e^{-sb} M(s).$$

A graphical illustration of Eq. (2.5) is given in Fig. 2.8.
Apply Eq. (2.4) to $x_1+\ldots+x_n$ and let $b = na$, we establish

$$\text{Prob}\left[x_1 + \cdots + x_n \geq na\right] \leq e^{-sna}\left[M(s)\right]^n = e^{-n\left(sa - \log M(s)\right)}, \text{ for all } s \geq 0.$$

This is known as the Chernoff bound (Chernoff 1952). Define a function $R(s)=sa-\log M(s)$. Then we have

$$\text{Prob}\left[x_1 + \cdots + x_n \geq na\right] \leq e^{-nR(s)}, \text{ for all } s \geq 0. \tag{2.6}$$

We shall use it to establish the exponential decaying rate for Prob$[x_1+\ldots+x_n\geq na]$. Note although Eq. (2.6) looks already like a bound implying exponential decaying, there is a gap between it and the desired Eq. (2.2) where we need a *positive constant* β. We close the gap by showing that there is an $s^*\geq 0$ such that $R(s^*)>0$. For simplicity, we assume that $\mu = 0$ and $a > 0$ is a constant. (The reader is required to extend the result to the general case where $\mu \neq 0$ below.) Then $\mu = M'(0) = 0$. Consider the Taylor expansion of $M(s)$ around $s = 0$,

$$R(s) = sa - \log(M(0)+M'(0)s+o(s)) = sa - o(s).$$

Thus there exists a $s^*>0$ such that

$$R(s^*) = s^* a - \log(M(s^*))>0.$$

We can then choose $\beta = R(s^*)$.

Exercise 2.3: Show that for general μ, as long as $a>\mu$, there exists a positive β such that

$$\text{Prob}[x_1+\ldots+x_n\geq na]\leq e^{-n\beta}.$$

Exercise 2.4: Show that for general μ, as long as $b<\mu$, there exists a positive β such that

$$\text{Prob}[x_1+\ldots+x_n\leq nb]\leq e^{-n\beta}.$$

4.2 Exponential convergence w.r.t. order

With the large deviation theory, given two designs with distinct true performances, it is clear from above (Section 4.1) why order comparison is easy and converges rapidly. The problem is, in general, we have a large number of designs. In this subsection, we argue that the benefit of exponential convergence on order comparison for two designs is preserved for the general situation with N designs. The idea is to upper bound the misalignment probability (the overlap area for the two design case) by the sum of probabilities that sample mean deviates from true performance by the amount δ for each design, where δ is half of the minimal gap Δ between true performances. Here $\Delta = \min_{i=1,\ldots,N-1} \left(J\left(\theta_i\right) - J\left(\theta_{i+1}\right)\right)$ can be viewed as the counterpart of *B-A* for the two-design case.

Given a size s, let S_n be the selected set of size s according to the horse race rule after we obtain the observed (estimated) performance $\bar{\hat{J}}\left(\theta, n\right)$ based on n replications for all designs θ. Given also a size g of the good enough set G and the alignment level k such that $1 \le k \le \min(g,s)$. Our purpose is to show there exists a positive β such that

$$\text{Prob}\left[|S_n \cap G| \ge k\right] = 1 - O\left(e^{-n\beta}\right) \tag{2.7}$$

as long as the moment generating function $E[e^{sL(\theta,1)}]$ exists for all $s \in (-d,d)$ for some $d > 0$. It is sufficient to show the result Eq. (2.7) for the case $k = \min(g,s)$ since $\text{Prob}[|S_n \cap G| \ge k] \le \text{Prob}[|S_n \cap G| \ge k']$ for all $k' < k$. The reason is that $\min(g,s)$ is the highest alignment level and increasing the required alignment level always makes alignment harder (lower the alignment probability). Assume that the N designs are indexed such that the true performance value J is sorted in ascending order,

$$J(\theta_1) < J(\theta_2) < J(\theta_3) < \ldots < J(\theta_N).$$

Recall that the observed values $\bar{\hat{J}}\left(\theta_i, n\right)$ used by the horse race rule are taken as sample mean of performance values of designs, that is $\bar{\hat{J}}\left(\theta_i, n\right) = \frac{1}{n}\sum_{j=1}^{n} L\left(\theta_i, j\right), i = 1, 2, \ldots, N$, with $L(\theta_i,j)$, j = 1,2,… as i.i.d. observations. Without loss of generality, we assume our optimization problem is to find the minimum. Sort the sequence $\bar{\hat{J}}\left(\theta_i, n\right),\ i = 1, 2, \ldots, N$

in ascending order and denote the design ranking no. i as $\theta_{[i]}$[5]. In other words, $\overline{\hat{J}}\left(\theta_{[1]},n\right)\le\overline{\hat{J}}\left(\theta_{[2]},n\right)\le\ldots\le\overline{\hat{J}}\left(\theta_{[N]},n\right)$. So $\theta_{[i]}$, $i=1,2,\ldots,N$ are random variables taking value from the design space $\Theta_N=\{\theta_1,\theta_2,\ldots,\theta_N\}$. The selected set by horse race rule is a random set given as $S_n=\{\theta_{[1]},\theta_{[2]},\ldots,\theta_{[s]}\}$. The alignment probability can be expressed as

$$\text{Prob}\left[\left|S_n\cap G\right|\ge k\right]=\text{Prob}\left[\left|\left\{\theta_{[1]},\theta_{[2]},\ldots,\theta_{[s]}\right\}\cap\left\{\theta_1,\theta_2,\ldots,\theta_g\right\}\right|\ge k\right].$$

Here g is the size of the good enough set G. To prove Prob[$|S_n\cap G|\ge k$]$\ge$1-$e^{-n\beta}$ where k=min(g,s), it is equivalent to show Prob[$|S_n\cap G|<\min(g,s)$]$\le Ce^{-n\beta}$ for some positive constant C. To prove this, denote the minimal gap between any two of true performance values as $\Delta=\min_{i=1,\ldots,N-1}\left(J\left(\theta_i\right)-J\left(\theta_{i+1}\right)\right)$ and introduce two events

$$\text{Event } A=\{|S_n\cap G|<\min(g,s)\}$$

$$\text{Event } B=\{\text{there exists one } \theta_i \text{ in } \Theta_N \text{ s.t. } \left|\overline{\hat{J}}\left(\theta_i,n\right)-J\left(\theta_i\right)\right|\ge\delta\}$$

where δ is half of the minimal gap Δ. Event A is the misalignment event under level k = min(g,s), i.e.,

$$\text{Prob}[|S_n\cap G|<\min(g,s)]=\text{Prob}[\text{Event } A].$$

Event B is the event that at least one design's sample mean deviates from its true value over half of the minimal gap Δ. In Fig. 2.9, Event B occurs when at least one design's sample must fall in either dotted area ($\overline{\hat{J}}\left(\theta_i,n\right)-J\left(\theta_i\right)<-\delta$) or gray area ($\overline{\hat{J}}\left(\theta_i,n\right)-J\left(\theta_i\right)>\delta$). It is clear from Fig. 2.9 that if *every* design's sample mean stays in the interval centered at the design's true performance value with width Δ (or less) or equivalently the deviation from the true performance is less than $\delta=\Delta/2$, there will be *no* swap in the order of sample means and the alignment level k = min(g,s) is achieved. This implies that, for a misalignment to occur (furthermore some swaps in sample means to occur), Event B must occur.

---

[5] Please note that $\theta_{[i]}$ depends on n. We assign indices randomly to designs when they tie.

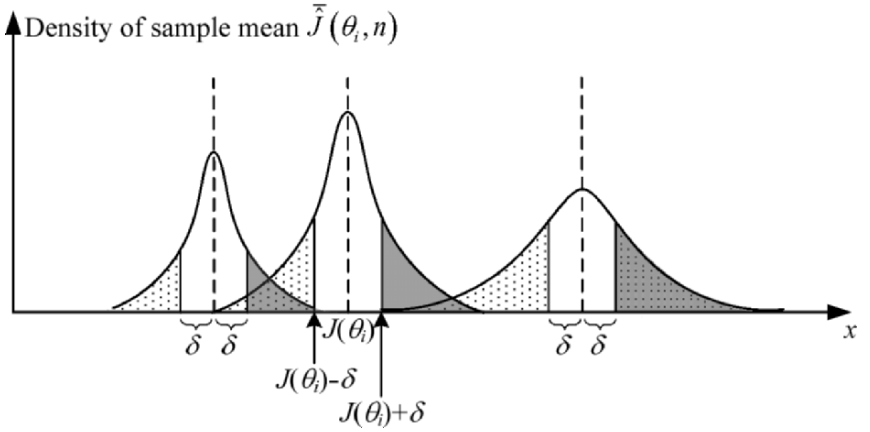

Fig. 2.9. If sample mean stays within δ distance from true performance for each design, no misalignment will happen.

Thus we know that Event A is a subset of Event B and

$$\text{Prob}\left[\left|S_n \cap G\right| < \min\left(g, s\right)\right] = \text{Prob}\left[\text{Event } A\right] \le \text{Prob}\left[\text{Event } B\right]. \quad (2.8)$$

Simple estimation gives

$$\begin{aligned} \text{Prob}\left[\text{Event } B\right] &= \text{Prob}\left[\bigcup_{i=1,\ldots,N}\left\{\left|\bar{\hat{J}}\left(\theta_i, n\right) - J\left(\theta_i\right)\right| \ge \delta\right\}\right] \\ &\le \sum_{i=1}^{N} \text{Prob}\left[\left|\bar{\hat{J}}\left(\theta_i, n\right) - J\left(\theta_i\right)\right| \ge \delta\right] \\ &= \sum_{i=1}^{N} \text{Prob}\left[\bar{\hat{J}}\left(\theta_i, n\right) \ge J\left(\theta_i\right) + \delta\right] \\ &\quad + \sum_{i=1}^{N} \text{Prob}\left[\bar{\hat{J}}\left(\theta_i, n\right) \le J\left(\theta_i\right) - \delta\right]. \end{aligned} \quad (2.9)$$

This is a direct extension of our estimation on misalignment probability for the two-design case (using the sum of gray area and dotted area in Fig. 2.7) to the general case where we have N designs as shown in Fig. 2.9. Now an upper bound for the misalignment probability is given by the sum of N gray areas and N dotted areas associated to the N designs. Since for every design, the true performance J is the mean value of its observed value L

and $\delta>0$, it follows the Large Deviation Theory, there exist positive numbers β_i and β_i' such that

$$\text{Prob}\left[\overline{\hat{J}}\left(\theta_i,n\right)\geq J\left(\theta_i\right)+\delta\right]\leq e^{-n\beta_i}$$
$$\text{Prob}\left[\overline{\hat{J}}\left(\theta_i,n\right)\leq J\left(\theta_i\right)-\delta\right]\leq e^{-n\beta_i'}.$$

Let $\beta=\min(\beta_1,\ldots,\beta_N,\beta_1',\ldots\beta_N')$, then we have from Eqs. (2.8) and (2.9)

$$\text{Prob}\left[|S_n\cap G|<\min\left(g,s\right)\right]\leq 2Ne^{-n\beta}.$$

The exponential convergence of the alignment probability is hence established by noting the size N of design space is fixed.

So far, we have shown the exponential convergence of OO w.r.t. order using the large deviation theory and estimation on misalignment probability for designs with i.i.d. observations. The exponential convergence of the alignment probability can be generalized to the situation of regenerative simulation[6], where performances are estimated by taking time average over a single sample path based on the ergodic properties of discrete event systems. When we carry out a simulation of length t and obtain some observed value $\overline{\hat{J}}(\theta,t)$ for all designs θ, by applying horse race rule, we can decide a selected set S_t. Since S_t depends on t, which is a measure of computation budget, the question now becomes where Prob[$|S_t\cap G|\geq k$] converge exponentially as $t\to\infty$? The exponential convergence for this case means that there exists $\beta>0$ such that

$$\text{Prob}\left[\left|S_t\cap G\right|\geq k\right]=1-O\left(e^{-t\beta}\right). \tag{2.10}$$

It was proved in (Xie 1997) that when the Heidelberger and Meketon's esmtiators (Heidelberger and Meketon 1980) defined in Eq. (2.11) or time average estimators defined in Eq. (2.12) below are used as observed value $\overline{\hat{J}}\left(\theta,n\right)$, under mild condition, there must exist a $\beta>0$ such that Eq. (2.10) holds.

[6] The basic idea of the approach of regenerative simulation is that a stochastic process may be characterized by random points in time when it "regenerative" itself and become independent of its past history. (See also Appendix A.)

Intuitively, since a regenerative simulation is equivalent to many periods of statistically independent replications of the system sample path, the validity of Eq. (2.10) is totally reasonable. The proofs can be found in (Xie 1997) and will be omitted here.

Note, this type of results were first obtained in (Dai 1996) showing that the best observed design is indeed a "good" design. (Xie 1997) extended these results to the general setting we describe here. Extensions of (Dai 1996) to the situation using common random variables in simulation have been made in (Dai and Chen 1997).

To define the estimators and the exponential convergence mathematically, we will introduce some notations. Let $T_i(\theta)$ be the i-th regeneration epoch, $i = 0,1,2,\ldots$, where $T_0(\theta)$ is the initial delay. Let $\tau_i(\theta)$ be the length of i-th regeneration cycle, $i = 0,1,2,\ldots$. Then $T_i(\theta) = \sum_{j=0}^{i} \tau_j(\theta)$. Suppose the interested performance value on sample path at time t is $L_t(\theta)$ with $|L_t(\theta)| \le C$ for some constant C. Let $L(\theta, i) = \int_{s=T_{i-1}(\theta)}^{T_i(\theta)} L_s(\theta) ds$ be the total sample performance in the i-th regeneration cycle.

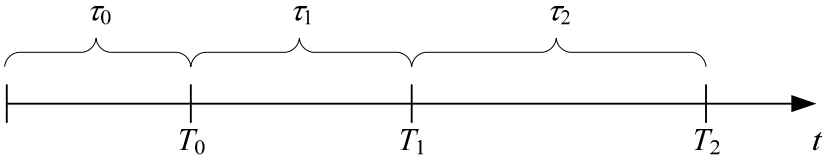

Fig. 2.10. An illustration for the regeneration cycles

Let $K(\theta,t)$ be the number of regeneration cycles completed by time t. Then the Heidelberger and Meketon's estimator is defined as

$$\bar{\hat{J}}(\theta,t) = \frac{\sum_{i=1}^{K(\theta,t)+1} L(\theta,i)}{\sum_{i=1}^{K(\theta,t)+1} \tau_i(\theta)}, \tag{2.11}$$

and the time average estimator is defined as

$$\bar{\hat{J}}(\theta,t)=\frac{1}{t}\int_{s=0}^{t} L_s(\theta)\,ds. \tag{2.12}$$

We assume that the regeneration process has i.i.d cycles, i.e., $\{(\tau_i(\theta), L(\theta,i)), i=1,2,\ldots\}$ is a sequence of i.i.d. random variables. Denote $m(s,\theta)=E[e^{s\tau_0(\theta)}]$ as the mgf (moment generating function) of the initial delay $\tau_0(\theta)$ and $M(s,\theta)=E[e^{s\tau_1(\theta)}]$ as the mgf of the length $\tau_1(\theta)$ of the first regeneration cycle $\tau_1(\theta)$. A sufficient condition for Eq. (2.10) to hold for the estimators in Eq. (2.11) or Eq. (2.12) is that $m(s,\theta)$ and $M(s,\theta)$ exist for all $s\in(-\delta,\delta)$ for some $\delta>0$. Note that this existence of a finite mgf was later shown by Fu and Jin to be both a necessary and sufficient condition in (Fu and Jin 2001). They have also shown how one can recover the exponential convergence rate in cases where the mfg is not finite. (Well-known distributions that do not possess a finite mgf include the lognormal distribution and certain gamma distributions.) In particular, by working with appropriately truncated versions of the original random variables, the exponential convergence can be recovered.

4.3 Proof of goal softening

In Section 2.1, we have argued intuitively **nothing but the best is very costly.** If we retreat from "nothing but the best" to a softer goal of "good enough", e.g., settle for anything in the top-*g* choices, then the small retreat can buy us quite a bit in the ease of the computational burden. In this subsection, we make a rigorous justification for this point and will show that the alignment probability for both blind pick selection rule and horse race selection rule converges exponentially to 1, as the size *g* of the good enough set and the size *s* of the selected set increase.

4.3.1 Blind pick

First, let us show the exponential convergence result for blind pick. It will be used as a base for proving the exponential convergence of the horse race rule. In fact, we will prove that the alignment probability of blind pick rule is always a lower bound for the alignment probability of horse race rule. This is reasonable, since no knowledge is used in BP, and in HR some, though imperfect, knowledge is used to select the set *S*. Let *N* be the size of the design space, *g* and *s* be the size of good enough set *G* and the size of selected set *S* respectively. For blind pick, the misalignment probability

Prob[$|S\cap G|=0$] is given by (see full derivation in Eq. (2.37) in Section 5.1 below)

$$\frac{\binom{N-g}{s}}{\binom{N}{s}},$$

where $\binom{N}{s}$ represents the number of different choices of s designs out of N distinguished ones, i.e.,

$$\binom{N}{s}=\frac{N!}{s!(N-s)!}.$$

Thus, the alignment probability Prob[$|S\cap G|\geq 1$] is given by

$$\text{Prob}\left[\left|S\cap G\right|\geq 1\right]=1-\text{Prob}\left[\left|S\cap G\right|=0\right]=1-\frac{\binom{N-g}{s}}{\binom{N}{s}} \tag{2.13}$$

$$=1-\frac{\frac{(N-g)!}{s!(N-g-s)!}}{\frac{N!}{s!(N-s)!}}=1-\frac{(N-g)(N-g-1)\cdots(N-g-s+1)}{(N)(N-1)\cdots(N-s+1)}. \tag{2.14}$$

Since $\frac{N-g-i}{N-i}\leq\frac{N-g}{N}=1-\frac{g}{N}$ for all i = 0,1…,s-1, we have

$$\text{Prob}\left[\left|S\cap G\right|\geq 1\right]\geq 1-\left(1-\frac{g}{N}\right)^{s}. \tag{2.15}$$

Furthermore, since 1-$x\leq e^{-x}$ holds for all x, we can bound Prob[$|S\cap G|\geq 1$] from below as

$$\text{Prob}\left[\left|S\cap G\right|\geq 1\right]\geq 1-e^{-\frac{gs}{N}}, \tag{2.16}$$

which is the desired result, i.e., alignment probability for blind pick converges exponentially w.r.t. the size of the set *G* and *S*.

Exercise 2.5: Draw a figure to verify the above statement that $1-x \le e^{-x}$.

4.3.2 Horse race

Now we are going to present the convergence result for the horse race selection rule alignment probability. We assume that the i.i.d. noise $w(\theta_i, n) = W_i(n)$, i=1,2,…,N, has the common cumulative continuous distribution function $F_n(x)$ and density function $f_n(x)$ and has zero mean.

With the help of the relation

$$\text{Prob}\left[|\, S \cap G \,| \geq 1\right] = 1 - \text{Prob}\left[|\, S \cap G \,| = 0\right],$$

we will show that for horse race selection rule, the alignment probability $\text{Prob}\left[|\, S \cap G \,| \geq 1\right]$ is bounded from below by the function $1 - e^{-\frac{gs}{N}}$, that is

$$\text{Prob}\left[|\, S \cap G \,| \geq 1\right] \geq 1 - e^{-\frac{gs}{N}}. \tag{2.17}$$

This is quite reasonable, since BP utilizes no knowledge of the problem, while the *S* selected by the horse race rule can only improve upon the AP. We can expect the same exponential convergence. While it is intuitively reasonable to suppose that any crude model for picking the set *S* must result in better performance than blind pick, it is nevertheless important to rule out crude models that may appear sensible on the surface but actually favor bad designs unknowingly. Consequently, we must prove that the *S* obtained based on a horse race model will indeed perform better and results in better AP than on a blind pick model. This is the purpose of this section.

To establish exponential convergence for horse race by leveraging the results in Section 4.3.1 on AP for BP, we should follow two steps:

Step 1. Identify Least Favorable Configuration (LFC) for horse race misalignment probability.

Step 2. Evaluate misalignment probability under LFC and prove its equivalence to that of blind pick.

Least Favorable Configuration (LFC) is well known in Ranking and Selection literature (Barr and Rizvi 1966). The general idea is to find and

take advantage of some monotone properties in a set of distributions with a parameter (such as mean value) to specific setting of the parameter under which certain ranking or selection probability of interest achieves maximum or minimum. In our case, we should use true performance as the parameter and aim at finding LFC for misalignment probability under the horse race selection rule. Let us first take a close look at the misalignment event under horse race.

For a direct derivation of the result, the basic idea is this: whenever the observed performance of every design in G is no better than that of at least s designs not in G, none of the designs in G will be selected in which case [$|S \cap G|=0$]. Fig. 2.11 shows a case of $N = 6$ designs with $g = 2$ and $s = 3$. In the figure, we show the procedure of generation of observation performance $\hat{\bar{J}}(\theta_i, n) = J(\theta_i) + W_i(n)$ from the true performance $J(\theta_i)$ by adding noise $W_i(n)$, $i = 1, 2, \ldots, 6$. The best observed performance of the two good enough designs (black balls in the lower part of the figure) is indicated by value A. Since it is greater than the observed performance of the three designs not in G (white balls), the select set by horse race contains only white balls which means a misalignment occurs. In the figure, we order the observed performance of all four designs not in G and indicate the s-th (third) value as B. We observe that $B<A$ is true, when misalignment happens.

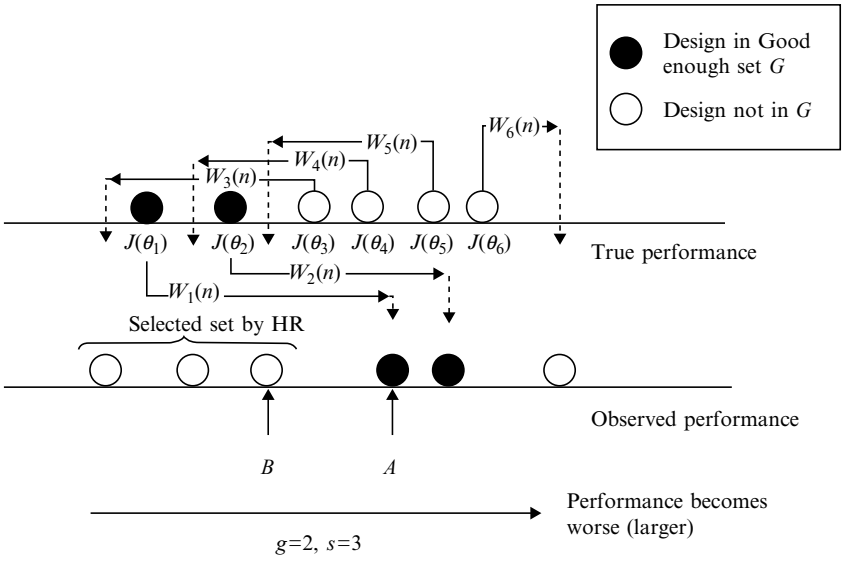

Fig. 2.11. An illustration for the misalignment event under horse race rule

To characterize $\text{Prob}\left[|S \cap G| = 0\right]$ in detail, we divide a given set of observation data $\hat{\overline{J}}(\theta_i, n) = J(\theta_i) + W_i(n)$ into two groups, the observed data for good designs $G=\{\theta_1,\ldots\theta_g\}$ and the data for bad designs $(\theta_{g+1},\ldots\theta_N)$. We order the N-g observation data for bad designs such that

$$J\left(\theta_{[g+1]}\right) + W_{[g+1]}(n) < \ldots < J\left(\theta_{[N]}\right) + W_{[N]}(n). \tag{2.18}$$

For a misalignment to happen when using horse race, we observe that, there must be at least s "bad" designs outperform good designs $\theta_1,\ldots,\theta_g$. Or put it in another way, all observed performances for good designs must be larger than $B=J(\theta_{[g+s]}) + W_{[g+s]}(n)$. Denote the best observed performance of good designs as $A=\min_{j\in\{1,\ldots,g\}}(J(\theta_j) + W_j(n))$. Then a misalignment simply means $B<A$ holds.

$$\text{Prob}\left[|S \cap G| = 0\right] = \text{Prob}\left[B < A\right]. \tag{2.19}$$

For example, in Fig. 2.11, the value of A is $\min(J(\theta_1) + W_1(n), J(\theta_2) + W_2(n)) = J(\theta_1) + W_1(n)$, the ordered N-g = 4 observation data for bad designs are

$$J(\theta_3) + W_3(n) < J(\theta_4) + W_4(n) < J(\theta_5) + W_5(n) < J(\theta_6) + W_6(n).$$

So, $\theta_{[5]} = \theta_5$ and the value of B is $J(\theta_{[5]}) + W_{[5]}(n) = J(\theta_5) + W_5(n)$.

Now we work on finding the LFC for $\text{Prob}\left[B < A\right]$=$\text{Prob}\left[|S \cap G| = 0\right]$. Our idea is to shift mean value $J(\theta_i)$ of all distributions of $\hat{\overline{J}}(\theta_i, n)$ to a common value $J(\theta_g)$. More specifically, we move the mean value of the distribution of every good design up to $J(\theta_g)$ and to move the mean value of the distribution of every bad design *down* to $J(\theta_g)$. Then we have a new set of N observed data $J(\theta_g) + W_i(n)$ sharing the same noise sample $W_i(n)$ with $\hat{\overline{J}}(\theta_i, n) = J(\theta_i) + W_i(n)$, the original observation with noise. We order the N-g data $J(\theta_g) + W_i(n)$ associated with bad designs such that

$$J\left(\theta_g\right)+W_{[g+1]}(n)<\ldots<J\left(\theta_g\right)+W_{[N]}(n). \tag{2.20}$$

Note since the mean values are reduced for the data of bad designs, the value $J(\theta_g)+W_{[g+s]}(n)$ appearing in the ordered sequence in Eq. (2.20) must be no greater than its counterpart $B=J(\theta_{[g+s]})+W_{[g+s]}(n)$ appeared in Eq. (2.18). At the same time, as the counterpart of A, the random variable $\min_{j\in\{1,\ldots,g\}}(J(\theta_g)+W_j(n))$ must be no less than A. Let us denote $A'=\min_{j\in\{1,\ldots,g\}}(J(\theta_g)+W_j(n))$ and $B'=J(\theta_g)+W_{[g+s]}(n)$. Fig. 2.12 shows this procedure of finding LFC for the designs in Fig. 2.11. Recall, we have g = 2 and s = 3. We move the mean value of the observed performance's distribution to $J(\theta_2)$ but keep the sample of noise $W_i(n)$ the same as in Fig. 2.11. The new observation data become $J(\theta_2)+W_i(n)$, $i=1,2,\ldots,6$, and their values are as shown in the line "observed performance (LFC)". For reader's convenience, we also show original observation data in the bottom of the figure. The best value for good designs under new observation i.e., A', equals $\min(J(\theta_2)+W_1(n),J(\theta_2)+W_2(n))=J(\theta_2)+W_2(n)$. We order the new observations of the four bad designs as $J(\theta_2)+W_3(n)<J(\theta_2)+W_4(n)<J(\theta_2)+W_5(n)<J(\theta_2)+W_6(n)$. So, the third ($s$-th) value in this sequence is $J(\theta_2)+W_5(n)$, $\theta_{(5)}=\theta_5$ and the value of B' is $J(\theta_2)+W_5(n)$. Two important observations from this example are $B'<B$ and $A<A'$, as a result of our way of generating new observations.

With the new distributions defined above, we are able to establish an upper bound for $\text{Prob}[B<A]$ in Eq. (2.19), that is

$$\text{Prob}[B<A]\le\text{Prob}[B'<A']. \tag{2.21}$$

In fact, we have noted that $B'<B$ and $A<A'$. As a result, $B<A$ always implies $B'<A'$. This shows indeed the shift we made leads to the LFC for the misalignment probability.

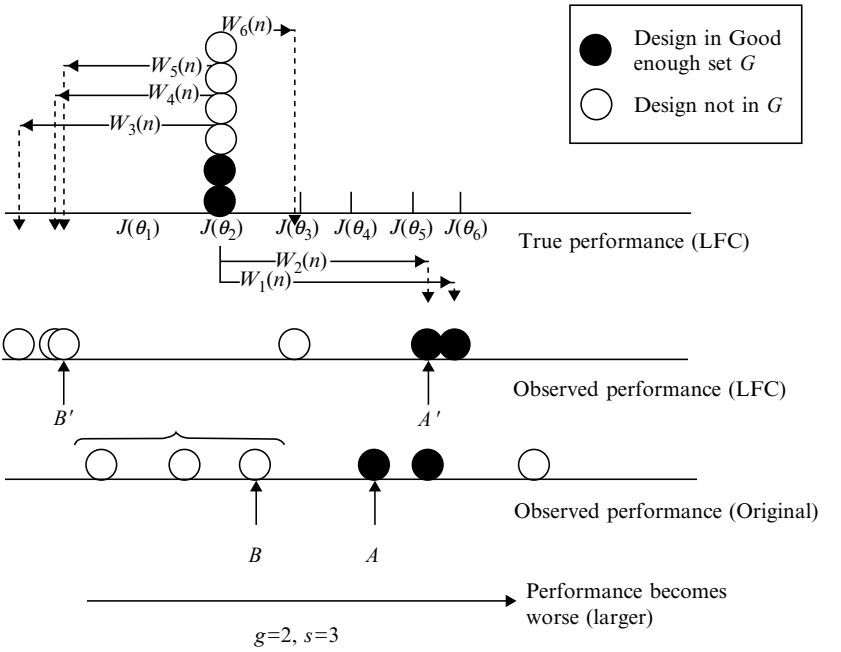

Fig. 2.12. An illustration of the Least Favorable Configuration for designs in Fig. 2.11

Exercise 2.6: Show Eq. (2.21) is true.

Eq. (2.21) is the LFC result in our context. Its advantage is, now instead of dealing with the observation data $\hat{\overline{J}}(\theta_i, n) = J(\theta_i) + W_i(n)$ which are all following different distributions with different means, we need only to deal with the case of i.i.d. observations $W_j(n)$ plus a constant.

In summary, Fig. 2.12 and our arguments above show, for horse race selection rule, misalignment occurs when the best value (A) of good designs is greater (for minimization problem) than the values of s bad designs (we denote B as their maximum) in the observation data. Although evaluating the probability of $B<A$ is generally difficult due to the heterogeneous nature of distributions generating observations of designs, this characterization of misalignment enables us to find the new setting that provides a tractable upper bound for misalignment probability, which turns out to be the same as blind pick. The new setting is tractable since observations of all designs obey i.i.d. distributions. The new setting provides an upper bound on misalignment probability (thus it is a LFC)

because the gap between the maximum of s bad designs B' and its best observed value for good designs A' is always greater than that of B and A. The connection between the new setting and blind pick is natural because independent draw of samples from the same distribution gives no preference on any specific design and all designs have equal chance to be selected which is the same as blind pick.

Now we proceed to evaluate misalignment probability under the LFC, or equivalently to calculate $\text{Prob}\left[B' < A'\right]$. This is the second step in order to establish the exponential convergence for horse race. It will be shown below that $\text{Prob}\left[B' < A'\right] = \binom{N-g}{s}\binom{N}{s}^{-1}$ which is exactly the misalignment probability already given in Section 4.3.1 for blind pick. For readers not interested in the mathematical details, you can go directly to the end of this section (the texts below Eq. (2.36)).

Without loss of generality, we can simplify the analysis of misalignment probability of our LFC by ignoring the common constant $J(\theta_g)$ in all observations $J(\theta_g)+W_i(n)$ and directly define $A^* = \min_{j\in\{1,\ldots,g\}} W_j(n)$ and $B^* = W_{[g+s]}(n)$ based on the zero mean sample $W_i(n)$. Denote their densities and distributions as $\phi_{A^*}(x), \phi_{B^*}(y)$ and $\Phi_{A^*}(x), \Phi_{B^*}(y)$ respectively. Then we can write

$$\begin{aligned}\text{Prob}\left[B' < A'\right] &= \text{Prob}\left[B^* < A^*\right] = \int_{-\infty}^{+\infty} \text{Prob}\left[B^* < x\right]\phi_{A^*}(x)\,dx \\ &= \int_{-\infty}^{+\infty}\int_{--\infty}^{x} \phi_{B^*}(y)\,dy\,\phi_{A^*}(x)\,dx\end{aligned} \tag{2.22}$$

based on the fact that $A^* = \min_{j\in\{1,\ldots,g\}} W_j(n)$ and $B^* = W_{[g+s]}(n)$ are independent.

Exercise 2.7: Verify $\text{Prob}\left[B' < A'\right] = \text{Prob}\left[B^* < A^*\right]$.

Hint: Compare the samples $J(\theta_g)+W_i(n)$ and $W_i(n)$ $i = 1, 2, \ldots, N$.

Exercise 2.8: Show

$$\phi_{A^*}(x) = g\left[1-F_n(x)\right]^{g-1} f_n(x). \tag{2.23}$$

Hint: use Eq. (2.24) below.

In order to evaluate $\phi_{A^*}(x)$, we first decide the distribution function $\Phi_{A^*}(x)$. It is straightforward to see that

$$\begin{aligned}\Phi_{A^*}(x) &= \text{Prob}\left[A^* < x\right] = 1 - \text{Prob}\left[\min_{j\in\{1,\ldots,g\}} W_j(n) > x\right] \\ &= 1 - \prod_{j=1}^{g} \text{Prob}\left[W_j(n) > x\right] = 1 - \left[1 - F_n(x)\right]^g .\end{aligned} \tag{2.24}$$

To evaluate $\phi_{B^*}(y)$, we first find its distribution $\Phi_{B^*}(y)$. For a given value *y*, when $B^* = W_{[g+s]} < y$ is true, the bad design data $W_{g+1}(n), \ldots, W_N(n)$ are divided further into two groups, one group Λ contains *m* (≥s) values which are less than A^*, the other group Λ^- contains all the remaining *N*-*g*-*m* bad designs which are greater than *y*. According to the value of *m* and the way two groups formed, we can express the distribution $\Phi_{B^*}(y)$ as the following exclusive unions:

$$\Phi_{B^*}(y) = \text{Prob}\left[W_{[g+s]} < y\right]$$

$$= \text{Prob}\left[\bigcup_{\substack{\Lambda\subset\{g+1,\ldots,N\} \\ \Lambda \text{ has at least } s \text{ elements}}} \left\{\max_{i\in\Lambda} W_i(n) < y \text{ and } \min_{j\in\Lambda^-} W_j(n) > y\right\}\right] \tag{2.25}$$

$$= \sum_{m=s}^{N-g} \sum_{\substack{\Lambda\subset\{g+1,\ldots,N\} \\ \Lambda \text{ has } m \text{ elements}}} \text{Prob}\left[\max_{i\in\Lambda} W_i(n) < y \text{ and } \min_{j\in\Lambda^-} W_j(n) > y\right]. \tag{2.26}$$

We have

$$\text{Prob}\left[\max_{i\in\Lambda} W_i(n) < y\right] = \left[F_n(y)\right]^m \tag{2.27}$$

and

$$\text{Prob}\left[\min_{j\in\Lambda^-} W_j(n) > y\right] = \left[1 - F_n(y)\right]^{N-g-m} . \tag{2.28}$$

Notice that $\max_{i\in\Lambda} W_i(n)$ and $\min_{j\in\Lambda^-} W_j(n)$ are independent, we have from Eqs. (2.26)–(2.28) that

$$\begin{aligned}\Phi_{B^*}(y) &= \sum_{m=s}^{N-g} \sum_{\substack{\Lambda\subset\{g+1,\ldots,N\}\\ \Lambda \text{ has } m \text{ elements}}} \left[F_n(y)\right]^m \left[1-F_n(y)\right]^{N-g-m} \\ &= \sum_{m=s}^{N-g} \binom{N-g}{m} \left[F_n(y)\right]^m \left[1-F_n(y)\right]^{N-g-m}.\end{aligned} \tag{2.29}$$

Taking derivative on Eq. (2.29) yields

$$\phi_{B^*}(y) = s\binom{N-g}{s} \left[F_n(y)\right]^{s-1} \left[1-F_n(y)\right]^{N-g-s} f_n(y). \tag{2.30}$$

Exercise 2.9: Show Eq. (2.30).

Now with expressions Eqs. (2.23) and (2.30) of $\phi_{A^*}(x)$ and $\phi_{B^*}(y)$ plugged in, we are ready to proceed on the integration in Eq. (2.22). We have

$$\begin{aligned}&\text{Prob}\left[B' < A'\right] \\ &= \int_{-\infty}^{+\infty} \int_{-\infty}^{x} s\binom{N-g}{s} \left[F_n(y)\right]^{s-1} \left[1-F_n(y)\right]^{N-g-s} f_n(y)dy\, g\left[1-F_n(x)\right]^{g-1} f_n(x)dx.\end{aligned} \tag{2.31}$$

Using substitution, letting $u = F_n(x)$ and $v = F_n(y)$, we can further simplify Eq. (2.31)

$$\text{Prob}\left[B' < A'\right] = gs\binom{N-g}{s} \int_0^1 \int_0^u v^{s-1} \left[1-v\right]^{N-g-s} dv \left[1-u\right]^{g-1} du. \tag{2.32}$$

Using induction method, one can show that

$$gs \int_0^1 \int_0^u v^{s-1} \left[1-v\right]^{N-g-s} dv \left[1-u\right]^{g-1} du = \binom{N}{s}^{-1}. \tag{2.33}$$

As a consequence, we have

$$\text{Prob}\left[B' < A'\right] = \frac{\binom{N-g}{s}}{\binom{N}{s}}, \tag{2.34}$$

which implies

$$\text{Prob}\left[\left|S \cap G\right| = 0\right] = \frac{\binom{N-g}{s}}{\binom{N}{s}} \tag{2.35}$$

and furthermore

$$\text{Prob}\left[\left|S \cap G\right| \geq 1\right] = 1 - \text{Prob}\left[\left|S \cap G\right| = 0\right] = 1 - \frac{\binom{N-g}{s}}{\binom{N}{s}}. \tag{2.36}$$

This shows that for the worst case where all observed performance values $\hat{\bar{J}}\left(\theta_i, n\right)$ are i.i.d., the alignment probability of horse race rule is the same as that of blind pick. As a result, for general cases where the true performance values *are* different, the alignment probability of horse race rule is bounded from below by the blind pick alignment probability

$$1 - \frac{\binom{N-g}{s}}{\binom{N}{s}}.$$

The desired result Eq. (2.17) then follows from the exponential convergence w.r.t. the size of *G* and *S* for blind pick.

Note exponential convergence of OO w.r.t. the size of *G* and *S* was originally given in (Lee et al. 1999). The noises were assumed to obey normal distributions. This assumption allows one to find the same LFC for noises with non-identical distributions, but as we have shown above, the

normal distribution assumption is not necessary for our case where noises obey identical distribution.

Exercise 2.10: If in the observation data $\overline{\hat{J}}(\theta_i, n) = J(\theta_i) + W_i(n)$, the noises $W_i(n)$ obey normal distributions $N(0, \sigma_i^2 / n)$. Show the misalignment probability $\text{Prob}[|S \cap G| = 0]$ is no greater than the scenario where the noises $W_i(n)$ obey normal distributions $N(0, \sigma^2 / n)$ with $\sigma^2 = \max_{i=1,\ldots,N} \sigma_i^2$.

5 Universal alignment probabilities

The discussion in Section 2 and the demonstration in Section 3 suggest that the concept of alignment probability Prob[$|G \cap S| \geq k$] is rather general and can be problem independent. Thus it is possible to establish some universal scheme for all optimization problems to help narrow down the search for "good enough" designs as a function of the number of crude samples taken, N, the approximate size of the estimation error, σ^2, the type of problem class, $\mathcal{C}$, and finally the selection procedure used. This can be very useful during the initial phase in many problems that involve (i) a structureless and immense search space and (ii) performance evaluation that is corrupted by large noise/error and/or is computationally intensive. We explore this possibility below (see also (Lau and Ho 1997)). Alert reader may point out here that our aim here bears resemblance to the extensive literature in statistics on rank and selection (R&S) (Gupta and Panchapakesan 1979; Santer and Tamhane 1984; Goldman and Nelson 1994). There are, however, two major differences. First, the R&S schemes deal with a search space of usually less than a hundred[7], often in tens (such as in comparison study of the efficacy of different drugs) while we consider subset selection from Θ that has size in billions and zillions. Second, the cardinal notions of "distance of the best from the rest" and the probability of "coincidence of the observed best with the true best" used in R&S have very little significance in our problem domain. Instead, we focus on softened criterion and different selection procedures.

[7] Although recent development of R&S allows to deal with a design space as large as 500 (Nelson et al. 2001; Wilson 2001), this is still comparatively small than the size that OO can handle.

5.1 Blind pick selection rule

We obtain the simplest result on alignment probability by using the blind pick selection rule, i.e., we blindly pick out the members from the selected set, S without any evaluation of the performances from N samples. Equivalently, we can say that the performances in Eq. (2.1) is sampled with the noise variance being infinite (in the demonstration of Section 3, we used the noise distribution of $U[0,10000]$ to approximate the blind pick selection rule when the range of the true performance is [0,200]). For given size of S and G being s and g respectively, the alignment probability for blind pick (BP) is

$$AP(s,g,k,N/BP) = \sum_{i=k}^{\min(g,s)} \frac{\binom{g}{i}\binom{N-g}{s-i}}{\binom{N}{s}} = \text{Prob}\left[\left|G \cap S\right| \geq k\right]. \quad (2.37)$$

Exercise 2.11: Try to derive Eq. (2.37) before reading the explanation below.

The validity of Eq. (2.37) can be seen as follows: There are total $\binom{N}{s}$[8] ways of picking s out of N designs. Suppose i of these s designs actually belong to G, then there are $\binom{g}{i}$ ways for which this is possible. The remaining s-i designs can be distributed in $\binom{N-g}{s-i}$ ways. The product of these two factors constitutes the total number of ways that we find exact i members of G by picking out s designs out of N. Dividing this product by $\binom{N}{s}$ yields the probability for i. Summing over all $i \geq k$ gives Eq. (2.37).

---

[8] $\binom{N}{s}$ represents the number of different choices of s designs out of N distinguished ones, i.e., $\binom{N}{s} = \frac{N!}{s!(N-s)!}$.

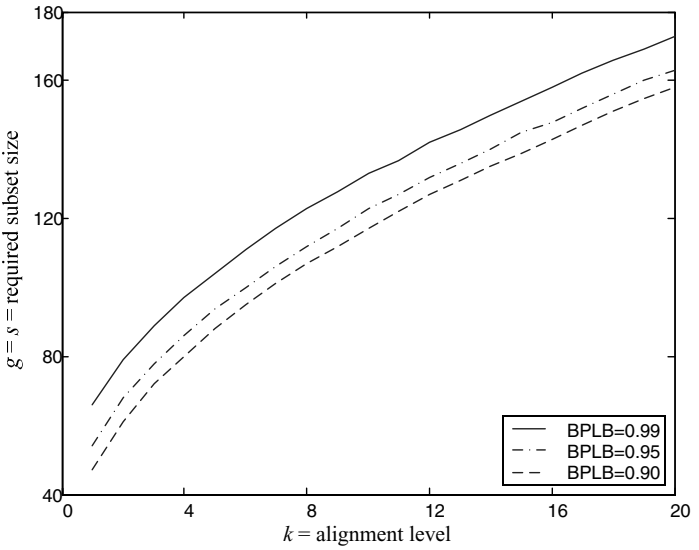

Fig. 2.13. Required subset size vs. alignment level for different alignment probabilities

Fig. 2.13 shows a plot for required size of s, with $g = s$ as a function of alignment level k for AP = 0.99, 0.95, and 0.90. These curves can be used as a lower bound (LB) for the UAP for any problems. It is instructive to see that to insure with probability (w.p. for short) 0.99 that there are at least two top 8% choices out of 1000 choices, we only need to blindly pick 80 samples – a more than ten fold reduction in search effort. Note this number s is an upper bound for selection since it is done **without any knowledge**. Imagine how much better we can do with some approximate knowledge about the problem. *This is the essence of ordinal optimization!* The next subsection will discuss the first of such less random selection rules.

5.2 Horse race selection rule

In Section 3 we demonstrated the horse race selection rule for S. The procedure of this rule is:

- We take N samples uniformly from Θ
- Using a crude model, we estimate the performances of these N samples as $\hat{J}(\theta_1),\ldots,\hat{J}(\theta_N)$
- We sort these samples according to their estimated performances as $\hat{J}(\theta_{[1]}),\ \ldots,\hat{J}(\theta_{[N]})$
- Select the observed top-s members of the N samples as the selected set S.

Then the alignment probability AP≡Prob[$|G\cap S|\geq k$] is defined the same way as the blind pick probability in the above subsection. However, in this case we no longer have a closed form solution as in Eq. (2.37). Furthermore, it is intuitively clear that the AP will also depend on the nature of the problem, i.e., the class of Ordered Performance Curve (OPC) of the problem as illustrated in Fig. 2.3 in Section 2. Hence we write AP= $\mathcal{F}(g,s,k,N,C/$Horse Race). If we normalize the OPC by defining

$$y_i = (J_{[i]} - J_{[1]})/(J_{[N]} - J_{[1]}) \tag{2.38}$$

$$x(\theta_{[i]}) \equiv x_i = (i-1)/(N-1) \tag{2.39}$$

we can attempt to fit the five different types of OPC (see Section 3) by the Incomplete Beta Function with parameters α and β given by

$$F(x/\alpha,\beta) \equiv \int_0^x \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} z^{\alpha-1}(1-z)^{\beta-1}\,dz \tag{2.40}$$

with the normalized OPC as

$$\Lambda(x/\alpha,\beta) \equiv F\left(x\Big/\frac{1}{\alpha},\frac{1}{\beta}\right). \tag{2.41}$$

For different values of α, β we can describe the different shapes of five different types of OPC and their significances in Fig. 2.14 and Fig. 2.15 below, where in Fig. 2.14 the Normalized Performance Densities are the derivatives of the inverse functions of the Normalized OPCs, respectively.

For a given pair of α and β, we can determine the AP by a simple simulation model in the same way as the Excel demo example outlined in Section 3. Extensive simulation has been done on these normalized OPCs (Lau and Ho 1997).

The principal utility of AP in practice is to determine the required size of the set S. For a given problem, the designer/optimizer picks the crude but computationally easy model to estimate the performance. S/he specifies what is meant by "good enough", namely the size of G, g. S/he also have some rough idea of the parameters, σ^2 and C [9] (hence the values α and β in Eq. (2.41) above). For practicality we set AP≥0.95. Then we can

[9] A rough idea of C can be gleamed always from the N samples $\hat{J}(\theta_1),\ldots,\hat{J}(\theta_N)$.

experimentally determine a function $Z(g,k/N,C,\sigma^2,\text{AP})$ which tells how large must the set S be in order to insure that it contains at least k member of the set G with high probability. The significance of this information is obvious. We have engineered a reduction of the search and computational effort from Θ to N to $|S| = s$.

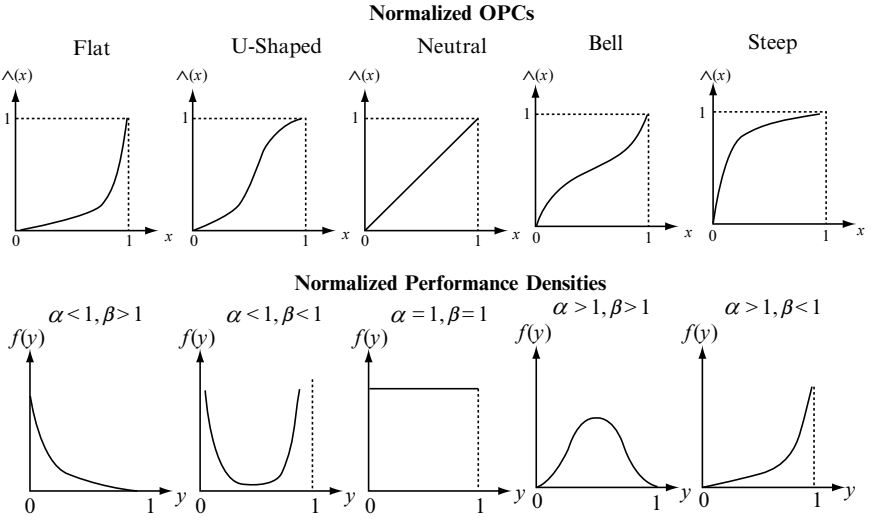

Fig. 2.14. Examples of beta density and corresponding standardized OPCs

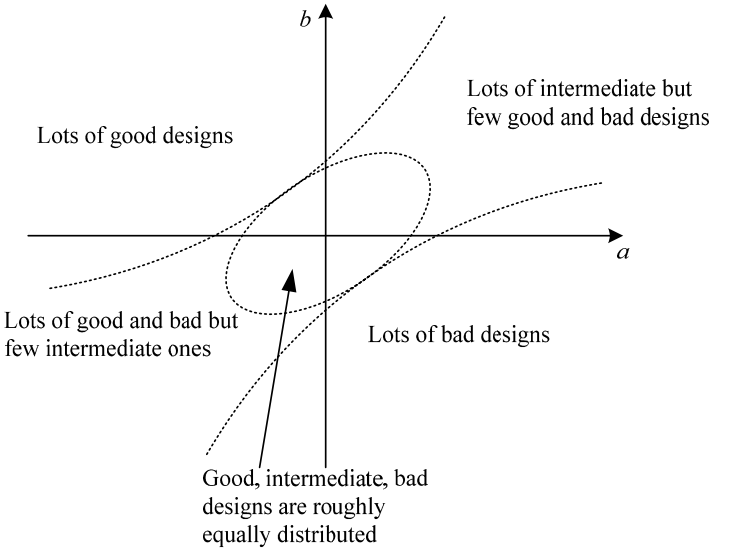

Fig. 2.15. Partitions of the *ab*-plane for five OPC categories, where $a = \log\alpha$, $b = \log\beta$

Extensive simulation experiments have been carried out with a total of 88 normalized OPCs (classified into 5 types of OPCs) using different α and β's

$$\alpha, \beta \in \{\, 0.15, 0.25, 0.4, 0.65, 1.0, 1{,}5, 2.0, 3.0, 4.5, 8.0\}$$

which covers the ab plane ($a = \log\alpha$, $b = \log\beta$) in Fig. 2.15 above with 10 U-shape class OPCs, 19 neutral class OPCs, 15 bell shaped class OPCs, and 22 each of the flat and steep class as defined in Fig. 2.14. The required sizes of S, the function $Z(g,k/N,C,\sigma^2,\mathrm{AP})$, are then tabulated as well as fitted via regression by

$$Z(k,g) = e^{Z_1} k^{Z_2} g^{Z_3} + Z_4\,, \tag{2.42}$$

Table 2.1. Regressed values of Z_1, Z_2, Z_3, Z_4 in $Z(k,g)$

Noise	∞	U[–0.5,0.5]				
OPC class	B-Pick	Flat	U-shape	Neutral	Bell	Steep
Z_1	7.8189	8.1378	8.1200	7.9000	8.1998	7.7998
Z_2	0.6877	0.8974	1.0044	1.0144	1.9164	1.5099
Z_3	–0.9550	–1.2058	–1.3695	–1.3995	–2.0250	–2.0719
Z_4	0.00	6.00	9.00	7.00	10.00	10.00
Noise	∞	U[–1.0,1.0]				
OPC class	B-Pick	Flat	U-shape	Neutral	Bell	Steep
Z_1	7.8189	8.4299	7.9399	8.0200	8.5988	7.5966
Z_2	0.6877	0.7844	0.8989	0.9554	1.4089	1.9801
Z_3	–0.9550	–1.1795	–1.2358	–1.3167	–1.6789	–1.8884
Z_4	0.00	2.00	7.00	10.00	9.00	10.00
Noise	∞	U[–2.5,2.5]				
OPC class	B-Pick	Flat	U-shape	Neutral	Bell	Steep
Z_1	7.8189	8.5200	8.2232	8.4832	8.8697	8.2995
Z_2	0.6877	0.8944	0.9426	1.0207	1.1489	1.3777
Z_3	–0.9550	–1.2286	–1.2677	–1.3761	–1.4734	–1.4986
Z_4	0.00	5.00	6.00	6.00	7.00	8.00

where Z_1, Z_2, Z_3, Z_4 are constants of regression depending on OPC types, the noise level, g, and k values[10]. These results are tabulated in Table 2.1, Fig. 2.16 and Fig. 2.17(a)–(e) (where we assume the noise contains uniform distribution with half-width W, i.e., $U[-W, W]$, W = 0.5, 1.0, 2.5.) against normalized OPC in [0,1].

[10] We do not believe a linear regression function would fit the data well. Thus, a product form is the next simplest nonlinear function we can try.

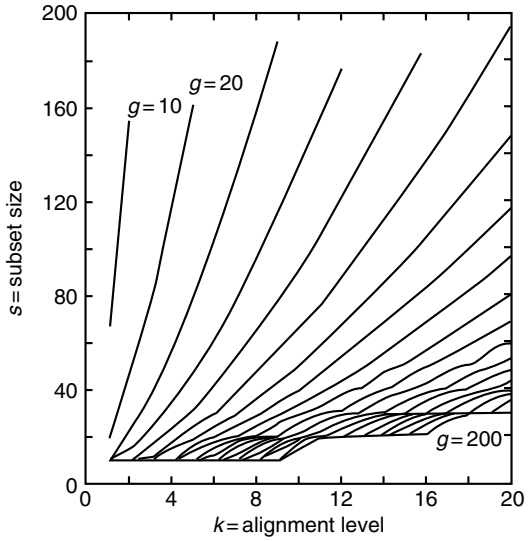

Fig. 2.16. Subset size interpolated from simulated data for the neutral class OPC and $W = 1.0$

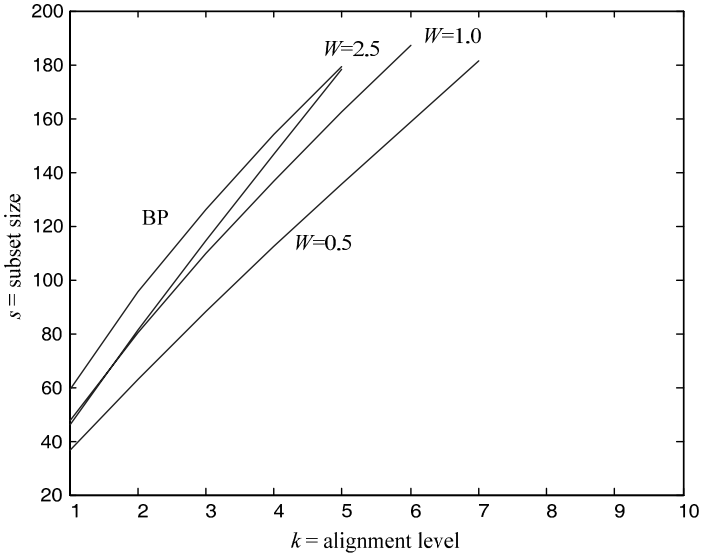

Fig. 2.17(a). Subset size for the flat OPC class at different noise levels with $g = 50$

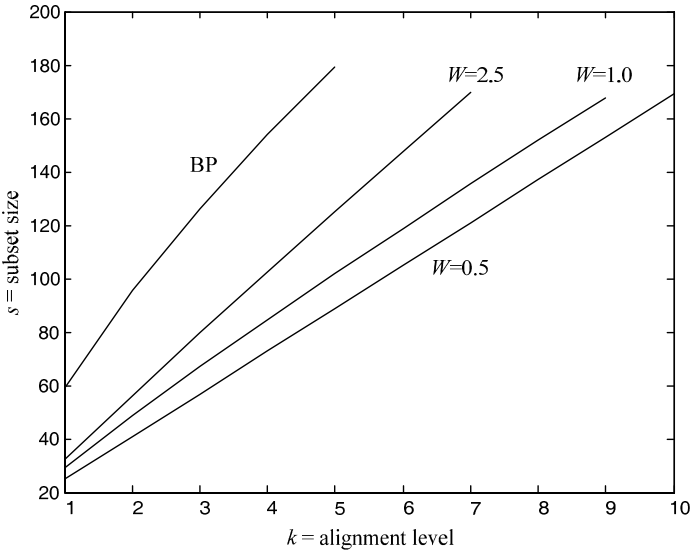

Fig. 2.17(b). Subset size for U-shaped OPC class at different noise levels with $g = 50$

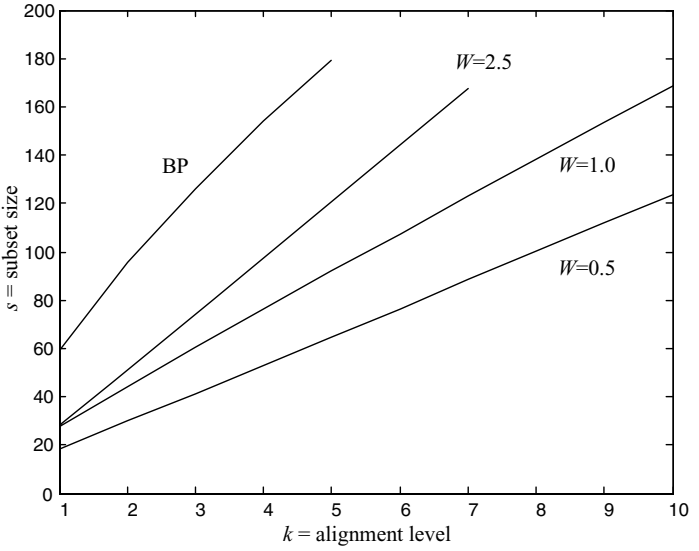

Fig. 2.17(c). Subset size for the neutral OPC class at different noise level with $g = 50$

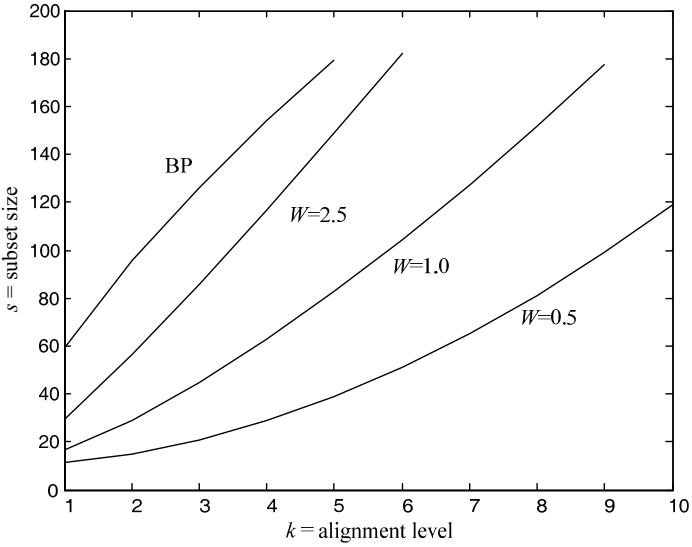

Fig. 2.17(d). Subset size for the bell-shaped OPC class at different noise levels with $g = 50$

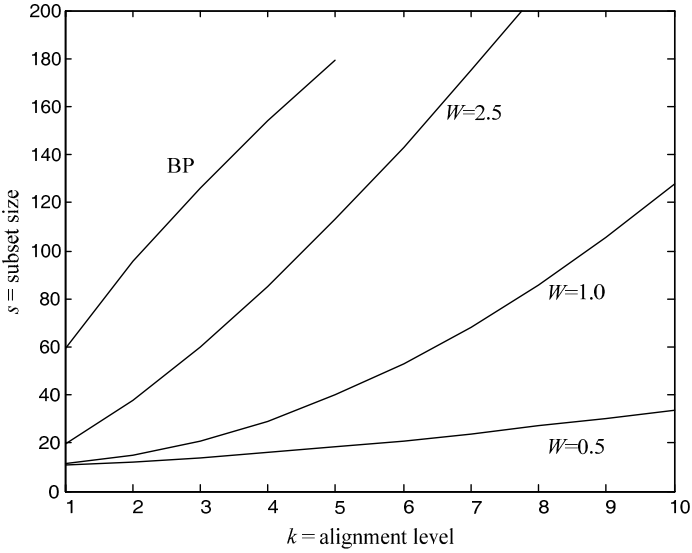

Fig. 2.17(e). Subset size for the steep OPC class at different noise levels with $g = 50$

These results have been extensively tested and found to be very reliable in large number of studies ((Lau and Ho 1997), and also see reference list in this book and at CFINS website:
http://www.cfins.au.tsinghua.edu.cn/en/resource/index.php).
Consequently, we designated these AP as ***"universal"***.

As an example, consider the following function defined on the range Θ=[0,1]

$$J(\theta) = a_1 \sin(2\pi\rho\theta) + a_2\theta + a_3, \tag{2.43}$$

where $a_1 = 3$, $a_2 = 5$, $a_3 = 2$. For $\rho = 500$, i.e., there are five hundred cycles in the range [0,1]. To estimate the exact functional form of Eq. (2.43) without prior knowledge, it may require extensive evaluation of the entire domain [0,1] at many points. However, here we shall consider using a crude model to approximate Eq. (2.43). In particular, based on the observation that there is a general rising trend in [0,1], we use a linear function

$$\hat{J}(\theta) = 5\theta. \tag{2.44}$$

Notice that only the linear part of Eq. (2.43) is contained in the crude model. In other words,

$$\hat{J}(\theta) = J(\theta) + \text{error}.$$

By generating N=1000 uniform samples from [0,1] using the crude model, we have

$$\hat{\Theta}_N = \{\theta_1, \theta_2, ..., \theta_{1000}\}.$$[11]

[11] Astute readers may notice that in the following we apply ordinal optimization to find good enough designs in N, and might wonder how to find good enough designs in Θ instead. The quick answer is that N is representative of Θ. When both N and Θ are large enough (which this example satisfies) the selected set that is selected from N also has a high probability to contain good enough designs in Θ, and the difference between the two alignment probabilities can be ignored for engineering purpose. But this notion will be quantified and made precise in Chapter VII Section 1.

The noise/error range can be estimated by $W = \max_{\theta_i \in \Theta} \left(\left| J(\theta_i) - \hat{J}(\theta_i) \right| \right)$ after adjusting for the mean values. We then select the neutral OPC class for this example. Once the good enough criterion g and the alignment level k are specified, the required selected subset size s from the crude model Eq. (2.44) is given by the function $Z(g,k/\text{neutral},W)$ in Table 2.1. Notice that these selected elements correspond to the first s members of N, because of the monotone property of the crude model. Then, we compare the selected subset with the true model to determine which indeed matches the good enough designs. 1000 experiments, each with a different N, are generated, so as to validate the actual observed alignment probability against AP = 0.95. We determined the alignment of each subset of size $s = Z(g,k/\text{neutral},W)$, where $g = 20, 30, \ldots, 200$ and $1 \le k \le 10$. Some of the alignment probabilities are plotted in Fig. 2.18. Each line in Fig. 2.18 represents the fraction of the 1000 experiments in which there are at least k of g good enough designs matched in the selected set. Note that:

1. The alignment probabilities are in general greater than 0.95, and this can be attributed to the conservative estimate of the function $Z(\bullet/\bullet)$.

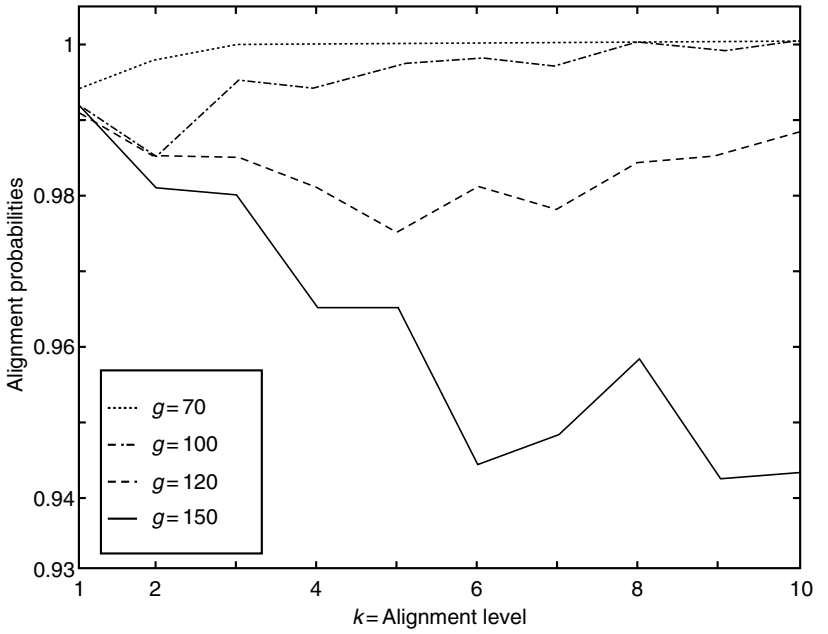

Fig. 2.18. Alignment probability validation for the example

2. Some fluctuation of the alignment probabilities are observed, which is due to the residues of the regression function *Z*.

The concept of universal alignment probability and the function $Z(\bullet/\bullet)$ have been validated many times in all papers on OO (Lau and Ho 1997; Shen and Bai 2005) and more examples will be shown later on in this book. Finally, we note that the blind pick AP of Eq. (2.37) of Section 5.1 is always a quick-and-dirty lower bound that is useful.

Exercise 2.12: Recall that in Section 2 we introduced 5 types of OPCs. Suppose we have a problem with many good designs (the flat type), and a problem with many bad designs (the steep type). Suppose the noise level is small. If we define the top-5% designs in both problems as the good enough designs, please use the UAP table just introduced to calculate the value of *s* such that Prob[$|G \cap S| \geq 1$]$\geq$0.95. Which problem requires a larger selected set? Is this result counter intuitive? Shall we set the same value of *g* for both problems?

6 Deterministic complex optimization problem and Kolmogorov equivalence

In previous sections, OO was developed to deal with stochastic complex simulation-based optimization problems (SCP), in which the crude model is a stochastic model of fewer replications, i.e.,

$$J_{\text{est}}(\theta) = J_{\text{true}}(\theta) + \text{random noise}(\theta). \tag{2.45}$$

There is another type of simulation-based optimization problems, where the true performance can only be obtained by deterministic and complex calculation (e.g., a large-scale finite element calculation). The crude model is usually a deterministic but computationally fast model, i.e.,

$$J_{\text{est}}(\theta) = J_{\text{true}}(\theta) + \text{deterministic complex error}(\theta). \tag{2.46}$$

This is called the deterministic complex problems (DCP). In fact, the example shown in Section 5 above is just one such DCP. There are also many successful applications in both types, especially for the DCP, (Yang 1998; Guan et al. 2001; Lin et al. 2004) just to name a few. One question immediately arises:

In what sense are OO in DCP and OO in SCP equivalent s.t. the UAP table in Section 5 can be used in both cases?

We address this question in this section.

First, let us compare the two problem formulations in Eqs. (2.45) and (2.46). Digital computers have pervasive applications in simulation-based optimization. We cannot generate pure random numbers in a digital computer. Instead, we use pseudo random number generator (PRNG). When both the PRNG and the seed are fixed, all the numbers thus generated compose a deterministic and complex sequence. As long as either the PRNG or the seed is not known to the user, which is the case in any engineering practice, the number thus generated is unpredictable. Then tremendous amount of simulation literature (Fishman 1996; Yakowitz 1977; Gentle 1998; Landau and Binder 2000) have established that we can regard the number generated by a PRNG as a random number since they pass rigorous statistical tests. The concept of Kolmogorov complexity (Li and Vitányi 1997) also justified that we can regard the unpredictable deterministic number as a random number, which means that there is no fundamental difference between the two problem formulations in Eqs. (2.45) and (2.46), from an engineering viewpoint.[12]

Second, let us look at the application procedures for OO in SCP and OO in DCP (Box 2.2), which are *almost* identical.

There are three differences between the above two columns: step 2, 3, and 4. In Step 2 and 3, the differences are mainly about the names. The two Step 2's are equivalent in the sense that the performance evaluation is a complex and time-consuming calculation. The two step 3's are equivalent in the sense that a complex deterministic error and a random noise is equivalent w.r.t. Kolmogorov complexity, as aforementioned. We now focus on Step 4 and answer why the UAP table in Section 5 for SCP can be also used for DCP. Suppose we want to regress another UAP table for DCP. Then we need to repeat the experiments, exactly as we did in Section 5, when Θ is extremely large that almost no design can be selected more than once in the initial random sampling of N designs. Thus all the experimental data are statistically equivalent to those obtained when regressing the UAP table for SCP. So the table thus regressed should be

[12] In principle, for any DCP for which we wish to apply OO, we should go through the same rigorous statistical analysis as we have done in the simulation literature to establish that the errors can indeed be equated to random noises in Eq. (2.46). For engineering applications, we often take as an article of faith based on the Kolmogorov equivalence that the complex incompressible and unpredictable error sequence in Eq. (2.46) are indeed random. So far this assumption has worked in all references cited.

the same as the UAP table in Section 5, subject to statistic error. This is why we can use the same UAP table in both SCP and DCP.

Box 2.2. Comparison of the procedures for OO in SCP and for OO in DCP

OO in SCP	OO in DCP
Step 1: randomly sample N designs from Θ	Step 1: randomly sample N designs from Θ
Step 2: *stochastic crude model*-based performance evaluation	Step 2: *deterministic crude model*-based performance evaluation
Step 3: estimate OPC and the *noise level*. User specifies g and k.	Step 3: estimate OPC and the *error level*. User specifies g and k.
Step 4: calculate s using the UAP table	Step 4: calculate s using the UAP table
Step 5: select the observed top-s designs as S	Step 5: select the observed top-s designs as S
Step 6: The theory of OO ensures there are at least k good enough designs in S with high probability.	Step 6: The theory of OO ensures there are at least k good enough designs in S with high probability.

Readers may also consider the case when there are correlations among the deterministic errors in Eq. (2.46) for different designs. This can be regarded as correlated noise or independent non-identical noise, which will be addressed in Chapter VII, Section 3. Here we just summarize that it has already been shown by numerical experiments and theoretical explanations that the correlation among the noises seldom can hurt and actually helps most of the time (Deng et al. 1992). For the case of independent non-identical noise, there are ways to divide the designs into several groups, within each of which the noise are i.i.d. Then we can easily extend the method in this chapter to deal with the problem (Yang 1998).

In short, as long as J_{true} in Eq. (2.46) can be assumed to be Kolmogorov complex, we can apply OO to deal with the optimization problem

$$\min_{\theta\in\Theta} J_{\text{true}}(\theta), \tag{2.47}$$

given the crude model

$$J_{\text{est}}(\theta) = J_{\text{true}}(\theta) + \text{noise/error}(\theta). \tag{2.48}$$

We estimate the noise/error level, and the ordered performance curve. As long as the design space Θ is extremely large, we can use the UAP table (Table 2.1 in Section 5) to decide the appropriate selection size.

7 Example applications

7.1 Stochastic simulation models

Let us consider the cyclic server problem discussed in (Ho et al. 1992). The system has 10 buffers (of unlimited capacity) for 10 arrival streams modeled by Poisson processes with rates $\lambda_1, \ldots, \lambda_{10}$ respectively. There is a single cyclic server serving the 10 buffers in a round-robin fashion: at buffer i, m_i jobs are served (if there are less than m_i jobs in the buffer, then serve all the jobs until the buffer becomes empty); then, the server moves from buffer i to buffer $i + 1$ with a changeover time of length δ_i (Fig. 2.19). A holding cost of C_i units at buffer i is incurred. The objective is to find a service policy $(m_1, m_2, \ldots, m_{10})$ such that it minimizes the average holding cost per job per unit time in the system. We assume that $0 < m_i < 10$ for all i; in other words, no more than 10 jobs may be served at each buffer for any policy. The design space Θ is therefore the lattice

$$\Theta = \left\{ m = \left(m_1, m_2, ..., m_{10} \right) / 0 \le m_i \le 10, \forall i \right\}.$$

The cost coefficients and arrival rates are respectively

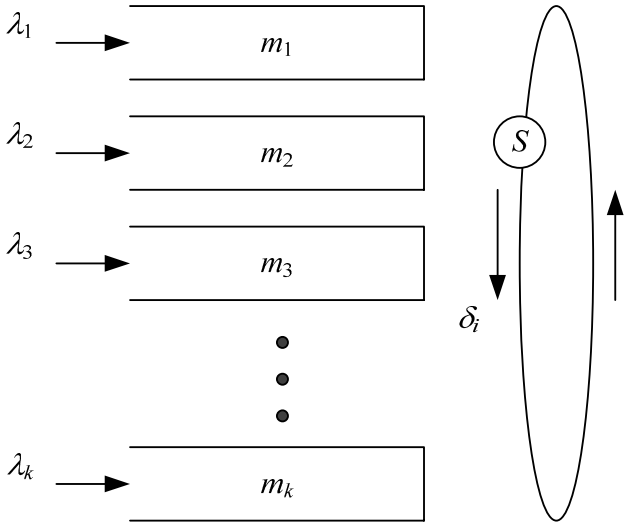

Fig. 2.19. Cyclic server serving K stream of arrivals

$$(C_1,...,C_{10}) = (1,1,1,10,1,50,1,1,1,1),$$
$$(\lambda_1,...,\lambda_{10}) = (1,1,1,1,1,1,1,1,1,1),$$

with a service rate of the server $\mu = 20$, and the mean changeover time of δ_i is

$$E(\delta_i) = 1/30, \text{ for all } i.$$

All random quantities are exponentially distributed. Notice that buffer 4 and buffer 6 have much higher cost coefficients.

We have generated 1000 policies (designs) from Θ and run long simulations for each policy to obtain their true ordering.[13] After 16753 jobs have arrived the system, the best 20 ordered designs are

$$\{\theta_{[1]},\theta_{[2]},\ldots,\theta_{[20]}\}=\{761,166,843,785,417,456,205,925,234,70,586,91,93,493,818,565,928,250,716,840\},^{[14]}$$

which will be taken as the true ordering of the top 20 designs. Assume that we are interested in obtaining any of these top 20 designs; i.e., they form the good enough subset from the 1000 design samples; then, we could have stopped the simulation at much earlier time instants. Suppose that we had terminated the simulation at the time when 161 and 330 jobs had arrived in the system.[15] Let us call these two time instants T_1 and T_2, respectively, and we have taken the corresponding noise levels to be large and medium. Without any prior knowledge, we conjectured a neutral OPC for the 1,000 designs. Then, the required subset selection sizes at these two instants are given as

$$s_{T_1} = Z(20,1/\text{neutral OPC, large noise}) = 65,$$
$$s_{T_2} = Z(20,1/\text{neutral OPC, medium noise}) = 47.$$

---

[13] Each policy is generated as follows: a buffer size between 0 and 10 inclusive is generated for each m_i, $i = 1,..., 10$. Thus, each design is a point sampled from the lattice Θ.

[14] The numbers are the indexes of designs.

[15] The number of jobs 161, 330 and 16753 correspond respectively to 500, 1000, and 50000 standard clock ticks. Simulation up to 50000 clock ticks is needed for the confidence intervals of the performance values of all designs to separate from each other. A standard clock tick is equivalent to an event happening to all 1000 systems operating under all policies. See Chapter VII Section b for further details about the standard clock.

Let us first examine the 65 designs at T_1,

S_{T_1} = {201, ***166***, ***565***, ***818***, 702, 335, 487, 471, 73, 331, ***843***, 172, 139, 595, 945, 905, 156, 658, 649, 431, 969, 233, 130, 204,307, 105, ***840***, 29, 179, 189, 58, 305, 40, 38, 9, 525, 31, 286, 17, 366, 982, 914, 529, 655, 567, 828, 640, 621, 53, 301, 527, 924, 165, 459, 126, 597, 285, 643, ***761***, 958, 681, 242, 379, 83, 927};

we see that six designs (in boldface and larger italics) are included in S_{T_1}. At T_2, the 47 selected designs are

S_{T_2} = {***761***, 595, ***565***, 873, ***843***, 139, 525, 105, ***166***, ***818***, 477, 643, 567, 447, 417, 980, 969, ***234***, ***928***, 366, 686, 201, 702, 738, 704, 111, 255, 314, 982, 361, ***785***, 640, 773, 910, 901, 235, 455, ***70***, 914, 172, 925, 335, 897, 31, ***456***, 217, 176};

we see that ten designs (in boldface and larger italics) from the good enough subset have been captured. The true top-20 designs in order by definition are

{***761,166,843,785,417,456,205,925,234,70,586,91,93,493,818,565, 928, 250,716,840***}

It is also interesting to point out that, from our experiments, we have observed a very fast convergence of design orders. (See Section 4 for more details on the exponential convergence of ordinal comparison.)

7.2 Deterministic complex models

After the discussion in Section 6, we now can look at the example discussed in the end of Section 5 from another aspect. The optimization problem is to minimize

$$J(\theta) = a_1 \sin(2\pi\rho\theta) + a_2\theta + a_3, \tag{2.49}$$

where a_1 = 3, a_2 = 5, a_3 = 2, ρ = 500, $\theta \in [0, 1]$. The deterministic crude model used to describe the increasing trend of $J(\theta)$ is

$$\hat{J}(\theta) = 5\theta. \tag{2.50}$$

Since designs are taken randomly from the interval [0,1], the steps given in Section 5 are the steps to apply UAP table to decide selection set size s to

solve this deterministic complex optimization problems. As seen from this example, by adopting a softened criterion, one can indeed achieve good alignment results by employing a very crude model in lieu of a complex model. This shows the importance of capturing the trend or general behavior of a system prior to the study of essential details. Perhaps, this also explains why a designer's intuition is often more valuable in the initial phase of a design process. Once a number of good enough designs are singled out, detailed studies of these designs can be done in the subsequent stages of the design process.

8 Preview of remaining chapters

So far, what we have presented in the first two chapters are introductory OO methodology and its foundations. Following the steps of OO and examples given, the readers can apply OO to solve real-world problems. In fact the majority of the 200 some references on OO employed no more than the theory and tools presented so far.

The remaining part of the book can be read more or less independently as shown in the logical dependency graph in Fig. 2.20 of the below. It is divided as three parts: Chapter III, IV, V and VI are major extensions of the OO method; Chapter VII deals with additional extensions; Chapter VIII presents case study for real-world examples.

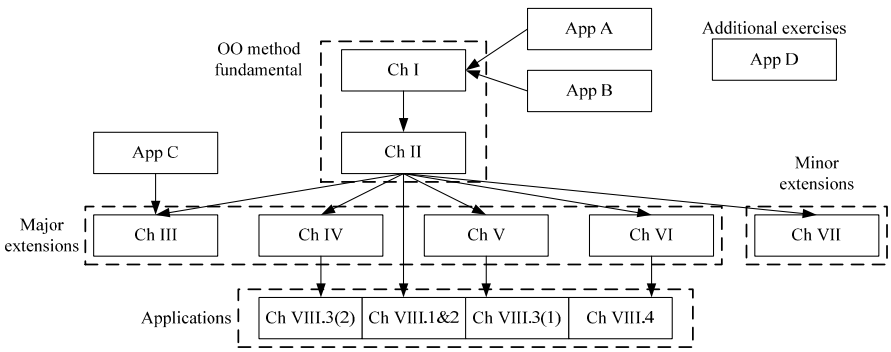

Fig. 2.20. Organization of the contents of the book

For the major extensions, we focus on Selection Rules in Chapter III. So far, we have studied two basic selection rules, namely, blind pick and horse race. We established analytical expression for blind pick and UAP table for the horse race. Although it is sufficient to use these rules to solve most application problems, it is still interesting to ask the natural question:

how about other selection rules? The purpose of Chapter III is to introduce more selection rules, compare the efficiencies of different selection rules, and give guideline in choosing selection rules based on the availability of computing budget.

As a second major extension to Ordinal Optimization method, we focus on optimization problems with multiple objective functions in Chapter IV. When there are multiple criteria (refers to as the vector case), ordinal comparison has to be done in a more complicated way than the scalar case of single objective function. As a result, the operative concept in multi-criterion optimization becomes the concept of Pareto optimum which was first formulated way back by Vilfredo Pareto. A design is said to be Pareto-optimal if it is not dominated by any other designs (i.e., there exists no other design that is better for at least one objective function value, and equal or superior with respect to the other objective functions). By introducing a natural order "layers" in design space, we generalize ordinal optimization from the scalar case to the vector case. We quantify how many observed layers are enough to contain the required number of designs in the Pareto frontier with high probability.

As a third major extension to Ordinal Optimization method, we focus in Chapter V on optimization problems with constraints. Similar to the objective function, we assume that the evaluation of constraints is also time consuming. So, the simple method of re-defining the design space as the feasible set then applying the tools of unconstrained OO does not work. To get around the time consuming evaluation barrier in constraints, we follow the idea of "crude model" in OO. Our key idea is to use a rough estimate of feasibility and allow the selected set to include some infeasible designs. Naturally to achieve the same level of alignment, more designs should be selected (thus a larger selected set is needed) for constrained OO. We quantify this additional correction.

A fourth extension to Ordinal Optimization method is given in Chapter VI. We deal with the memory limitation problem when we are trying to store a design on a computer. This problem comes naturally when we consider strategy optimization problems such as searching for good enough feedback control strategy for a complex system. Since for anything other than toy problems, the search space for admissible strategies can be enormously large, and the representation of a multi-dimensional function can be taxing on any size of computer memory, we need a way to search systematically in the strategy space that takes the limitation of memory storage into account. OO is incorporated into such a framework to search in the strategy space that can be implemented on a computer.

Further extensions of OO methodology requiring relatively little changes in solving real world problems will be discussed in Chapter VII.

Firstly, in previous study, no matter how large the design space is, we randomly sample $N = 1000$ designs and then apply OO to find some of the truly good enough designs (of these 1000 designs) with high probability. We show that the difference between the truly good enough designs (top-g%) of these 1000 designs, and the truly good enough designs (top-g%) of the entire design space is negligible for practical purpose. Thus further justify the practical use of OO for purists. Secondly, we show how we can take advantage of parallel computing ideas when applying OO to speed up the computation. The technique described is general. But the explanation is done by way of a specific example for clarity. Thirdly, in the previous consideration, only the i.i.d. additive noise is considered. However, the additive noise in practice might not be i.i.d. For example, the well adopted common random number generator technique is usually used in practice to reduce the variance of the observation noise. The observation noise then may be correlated. In some other times, the additive noise may be related to the performance of the solution, i.e., the noise is independent, but non-identical. We show that knowledge of these dependencies can help to improve the efficiency of OO method. Finally, as we mentioned earlier, Ordinal Optimization is not intended to replace the other optimization techniques. Instead, there are natural ways to combine ordinal optimization (including the key element of ordinal comparison) with other techniques, such as genetic algorithm to further improve the performance thus found.

In Chapter VIII, we present four real world application examples of applying OO method. The first example is a clothing manufacturing example. The problem is difficult and it is prohibitive to search for the best solution considering the tremendous computing budget involved. Using ordinal optimization ideas introduced in Chapter I and Chapter II, we obtained very encouraging results – not only have we achieved a high proportion of "good enough" designs but also tight profit margins compared with a pre-calculated upper bound. There is also a saving of at least 1/2000 of the computation time if brute-force simulations were otherwise used. The second real-world example is the Turbine blade design problem. We demonstrate how OO in Chapter I and II can be applied to solve such a deterministic complex problem. The third real-world example is the resource planning of a complex remanufacturing system involving two conflicting performance indices can only be evaluated by simulation. We demonstrate the application of Constrained Ordinal Optimization method developed in Chapter V and Vector Ordinal Optimization method developed in Chapter IV to the problem. At last, we demonstrate and apply extension of OO under limited memory developed in Chapter VI to the long standing strategy optimization problem known as the Witsenhausen Problem.