Chapter 4
Randomized Algorithms

There exists a very large class of problems that are computationally prohibitive when formalized in deterministic terms, but may become manageable when a probabilistic formulation can be derived and considered instead. For those problems, we are no more requesting to find *the* problem solution but *a* solution that, according to some probabilistic figure of merit, solves the problem.

Examples are the evaluation of the performance of a system when its computation is affected by perturbations (robustness analysis), verification of the satisfaction of the performance level of an embedded system or an algorithm (performance verification problem), identification of extrema of functions (function optimization problem), and design and analysis of robust controllers, just to name the few applications. The cost we have to pay to abandon determinism is that derived results will hold in probability.

Since the focus here is on embedded computation, we will see that there are some particular cases that might arise during the operational life of the embedded system violating the application constraints. However, such situations might be acceptable provided that the constraints are violated for a short time and constraints violation is a rare event. These aspects will be addressed in Chaps. 5 and 7.

Here, we request to be able to address a very large class of numerical-based problems and applications and find in the space of Lebesgue measurable functions the appropriate mathematical framework.

Definition: Lebesgue measurability

We say that a generic function $u(\psi), \psi \in \Psi \subseteq \mathbb{R}^l$ *is Lebesgue measurable with respect to* Ψ *when its generic step-function approximation* S_N *obtained by partitioning* Ψ *in N arbitrary domains grants that*

$$\lim_{N\to\infty} S_N = u(\psi)$$

holds on set $\Psi - \Omega$, $\Omega \subseteq \mathbb{R}^l$ *being a null measure set [20].*

C. Alippi, *Intelligence for Embedded Systems*, DOI: 10.1007/978-3-319-05278-6_4,

We point out that no functions generated by a finite-step, finite-time algorithm, such as *any* engineering-related mathematical computations, can be Lebesgue nonmeasurable. Indeed (see, e.g., [21]), the only way to produce nonmeasurable functions is to invoke the Axiom of Choice over an uncountable family of sets. This procedure is purely theoretical, and the objects obtained in this fashion are necessarily nonconstructible since the construction procedure would involve an uncountable number of arbitrary choices.

Under the Lebesgue measurability hypothesis and by defining a probability density function f_ψ with support Ψ, it comes out that we can transform computationally hard problems into manageable problems by sampling from Ψ and resorting to probability. Randomization comes as the main ingredient of the recipe and grants that obtained results, valid in probability, are characterized by an arbitrary accuracy and confidence levels function of the number of drawn samples. The loss in determinism is largely paid back by the possibility to solve our problem with a polynomial time algorithm.

In fact, all useful algorithms to be executed on embedded systems can be described as Lebesgue measurable functions and many interesting problems can be cast in the same formalism. However, by setting a general framework for a problem solution we can neither expect to find results in a closed form for all Lebesgue measurable applications nor pretend to solve deterministically the computationally hard problem associated with the application solution. To tackle such an issue we reformulate the deterministic problem in a probabilistic one which can be solved by Monte Carlo sampling under the control of the probabilistic theory of learning.

The chapter introduces the randomization mechanism for problem solution whose algorithmic description is known as *Randomized algorithm*.

The structure of the chapter is as follows: At first, we briefly introduce the complexity aspect associated with algorithms and problems. Since solutions will mostly be unmanageable given a generic problem described as a Lebesgue measurable function, we will resort to randomization to solve it. Monte Carlo is then presented followed by such fundamental results that are asymptotic in the number of samples n that grants convergence of some estimates to their expected values (laws of large numbers). Since asymptotic results are of scarce use in real applications (we cannot obtain a solution for a problem by taking an infinite number of sample), we need to search for bounds that grant some results to hold for a finite n. This can be achieved with randomized algorithms that integrate Monte Carlo with results coming from the theory of learning.

4.1 Computational Complexity

The computational complexity theory studies the intrinsic difficulty associated with the solution of a computable problem. Since for a computable problem an algorithm exists, i.e., the problem solution can be obtained in a finite time with a finite number of steps, it is our interest to identify “the best” algorithm solving the problem,

with optimality intended according to a given figure of merit. The complexity of an algorithm is generally evaluated as the time execution and memory resource required by an abstract machine to execute it. If time execution and memory resources are the figures of merit considered to assess the performance of the algorithm, say for solving the sorting problem, we might be interested in

- evaluating the complexity of the sorting algorithm;
- asking whether it is possible to identify a better solution for it or not.

If we focus on memory and execution time we can ask several questions whose answers are, a priori, not trivial. Which algorithm is using less memory among the ones we have? Which one is best performing on the average (i.e., the expected execution time w.r.t. random data in the sequence)? And when the sequence is ordered in the opposite way (worst case), which is the time complexity of our algorithm? Answering to these questions—and many others scholars or practitioners might raise—is fundamental if we wish to execute the algorithm on a real machine characterized by finite resources.

We comment that the questions posed above represent specific problems we wish to solve either deterministically or in probability and are of paramount relevance. In fact, even if a problem is computationally solvable in principle, it may not be addressable in practice whenever the algorithm requires an unfeasible amount of execution time or storage. The problem is general: any computer or embedded system introduces at some point hardware constraints which might make the practical execution of a given algorithm unfeasible.

4.1.1 Analysis of Algorithms

Computational complexity, in its analysis of an algorithm realm, approaches an algorithm to be investigated by observing how some extensive variables scale, e.g., the cardinality of a data set n or the dimension of the input space n.

Do we need to store the whole data set? If the answer is positive then we need n cells for storage and the Big Data paradigm is likely to become a problem. Do we need extra data structures to execute the algorithm? Then the needed storage space is the sum of all requested memory resources.

The algorithm time complexity can be decomposed in the time requested to address the basic sequences of operations (or instructions) and similarly to the memory complexity case becomes function of extensive variables.

Consider, for instance, the algorithm A given in Algorithm 1 evaluating the scalar product of two n-dimensional integer vectors $x, y \in \mathbb{N}^n$.

The complexity of algorithm A can be computed by evaluating the memory requirements $M(A)$ and the abstract computation execution time $C(A)$. For simplicity, we do not consider the complexity associated with memory assignment to the vectors and data acquisition since we wish to focus the attention on the algorithm itself. The memory requirement is simply the sum of requested variables (e.g., in memory cells, words, or bytes)

Algorithm 1: Algorithm A: a simple algorithm computing the scalar product between two vectors

```
scalar_product = 0;
i = 0;
assign memory to vectors x and y and populate the content;
while i < n do
    scalar_product = scalar_product + x[i]y[i];
    i = i+1;
end
```

$$M(A) = 2n + 2$$

while the computational complexity is

$$C(A) = (2n + 2)T_a + (n + 1)T_c + n(2T_+ + T_*)$$

where T_a is the time requested for an assignment, T_c that associated with the evaluation of a condition, T_+ and T_* represent the times requested to execute an addition and a multiplication, respectively.

We comment that all time components T assume constant values on a given processor (to ease the understanding we assume a sequential execution on a single core processor having independent instructions, e.g., for assignment, addition and multiplication); the faster the processor the shorter the execution time. It is clear that the average, the worst case or a generic case analysis coincide since complexity is not dependent here on the specific data instances but solely on the cardinality of the sequence.

The complexity of an algorithm is defined by means of the asymptotic character of the complexity figures of merit when the extensive variable n goes to infinity. The consequence is that the algorithm complexity is assessed by investigating how it scales with the problem complexity. Here, dependencies introduced by a specific machine assume constant values and can be neglected.

By referring to Algorithm 1, $M(A)$ scales as $2n$, that is to say its order is $O(n)$, while $C(A) = n(2T_+ + T_c + T_* + 2T_a)$ yields to an $O(n)$ order. It comes out that both functions C and M are linear with n. When Big Data are available we know that both the execution time and the memory complexity of the algorithm will scale linearly with n.

The "big Oh" notation characterizes the complexity of a given algorithm by hiding smaller terms contributions. The advantage in its use is that it makes the evaluation of complexity independent of the specific hardware platform or the computational model used. Other approaches evaluate the complexity of an algorithm by providing lower and upper bounds for it [209].

Let us evaluate the scalar product in a different way, within a sequential approach. The complexity of Algorithm 2 according to the two figures of merit is $M(B) = O(1)$, $C(B) = O(n)$ since the memory occupation does not scale with n and the loop is iterated $n + 1$ times.

Algorithm 2: Algorithm B: a sequential scalar product computation

```
scalar_product = 0;
i = 0;
assign memory to scalars x and y;
while i < n do
    input x and y;
    scalar_product = scalar_product + xy;
    i = i+1;
end
```

The comparison between the two algorithms is then carried out at the big Oh notation level by ordering their complexity according to the rank

$$\cdots O(k^{-n}) \prec O(n^{-k}) \prec O(n^{-1}) \prec O(1) \prec \cdots$$

$$\prec \cdots O(\log n) \prec O(n) \prec O(n^k) \prec O(k^n) \prec \cdots$$

where k is a strictly positive real value. Clearly, algorithms A and B have the same computational complexity in terms of execution time but algorithm B does not require storing the two vectors and, hence, is to be preferred to A if memory is a problem and n increases. For a small number of data the opposite might hold since the constant terms associated with arithmetic operations, the memory assignment and the input readout operations might introduce a strong influence on the final time execution. However, these situations are of no interest to computational complexity.

Complexity can be evaluated also by inspecting the behavior of the algorithm in the worst or the average case; the worst case is generally considered to compare two algorithms when their average complexity is identical.

4.1.2 P, NP-Complete, and NP-Hard Problems

We say that a problem A belongs to class P if its computational time complexity is polynomial $O(n^k)$ with constant k. In other words, the algorithm solves the problem in a polynomial execution time. Some authors, e.g., [23] claim that such a property characterizes problems that can be considered "efficiently solvable" or "tractable." This statement is only qualitatively true but sheds some light on the intrinsic complexity behind algorithms.

The problem of evaluating a scalar product belongs to class P. Many other algorithms belong to the P class: from ordering a vector of finite dimension, to verifying the presence of a given pattern within an image and carrying out a digital filtering of a signal.

Consider, as an example, the problem of sorting a numerical vector of cardinality n. *Bubblesort* shows complexity $O(n^2)$, *merge sort* $O(n \log n)$ both for the worst and

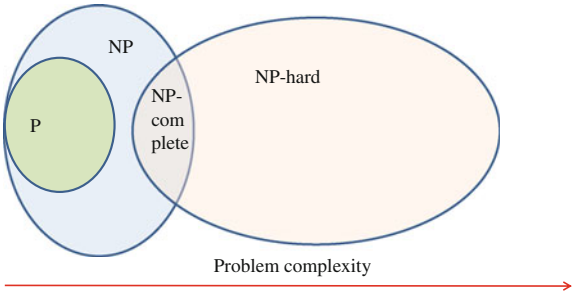

Fig. 4.1 The classes of problems *P*, *NP*, *NP-complete*, and *NP-hard* and their inclusions provided that P ≠ NP. The complexity increases when we leave the P problems and move toward the NP-hard ones

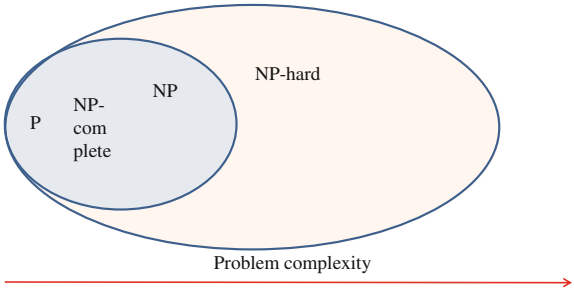

Fig. 4.2 The classes of problems *P*, *NP*, *NP-complete*, and *NP-hard* and their inclusionsprovided that P = NP. The complexity increases when we move toward NP-hard problems

the average cases, *quicksort* $O(n \log n)$ for the average case, and $O(n^2)$ for the worst case [24]; different algorithms have different complexities.

With reference to Fig. 4.1, the class of *nondeterministic polynomial time* problems *NP* is larger than the *P* one. *NP* contains the class of decision problems for which, given a candidate solution, we can verify in polynomial time if the solution solves the problem or not (in other words, we sample a candidate solution from the solution space and verify in polynomial time whether the selected solution is effective or not). *NP* contains many important problems with the hardest called *NP*-complete. For *NP*-complete problems no polynomial-time algorithms are known to solve them. A different way to characterize a *NP*-complete problem is the following: a decision problem is said to be *NP*-complete if it is *NP* and any other *NP* problem can be reduced to it so that its complexity is bounded by a polynomial in the complexity of the original problem.

A problem *H* is said to be *NP*-hard if and only if there exists a *NP*-complete problem *L* that is reducible to *H* in polynomial time. In other words, problem *L* can be solved in polynomial time by a machine which provides an oracle for *H*. Again, a problem is *NP*-hard if each *NP* problem can be reduced to this problem. One of the still open questions is whether P = NP or not, i.e., can a *NP* problem (and hence any of the class) be solved in polynomial time? Were that be the case, then Fig. 4.1 would degenerate as depicted in Fig. 4.2.

Even if it is thought that the answer is negative, the problem is still without a formal solution. The interested reader should consider [25] for a detailed analysis about complexity issues.

An example of a *NP* hard problem of interest here is the following: Given a Lebesgue measurable function $u(\psi) \in [0, 1]$, $\psi \in \Psi \subset \mathbb{R}^l$ and a value $\gamma \in [0, 1]$, does inequality $u(\psi) \leq \gamma$ hold for any $\psi \in \Psi$? The problem, which models the situation where we wonder about the level of satisfaction of a constraint, is surely computationally intractable for a generic $u(\cdot)$ function, since we should query the oracle at each ψ_i and ask the question "is $u(\psi_i)$ below γ?" Even if the oracle responds in a single time step (polynomial time response), the number of queries needed to solve the whole problem is not polynomial for a continuous space Ψ.

We will see in subsequent sections that some hard problems can be addressed and solved by resorting to probability. Such problems are known in the literature as belonging to the class of Randomized Polynomial time (RP) problems.

4.2 Monte Carlo

Monte Carlo methods constitute a class of algorithms that use a repeated random sampling approach and a probabilistic framework to compute the requested output. Due to the possibly large sampling required to provide accurate results, their full effectiveness became available thanks to advances in the computational power exposed by current processors and supercomputers (even if the history of the method dates back to the Manhattan project and it has been formalized in a seminal paper by Metropolis and Ulam [8] already in 1949).

Monte Carlo is an effective tool for addressing problems which can hardly be solved analytically for the mathematical complexity of the involved functions (e.g., integro-differential equations coming from physics and chemistry). It should be noted that Monte Carlo is a set of methods more than a method, each of which personalized to solve a specific class of applications. For instance, we have a method with its own mathematical results to address the integration problem, another for dealing with optimization or computational mathematics. The interested reader can refer to [12, 13] for a comprehensive analysis and further advances. As mentioned above, the core idea is that of sampling from a space and observing the satisfaction of a property or generating an estimate based on the sample ensemble; results are then aggregated to provide an approximated solution to the original problem. In the following, we present at first the idea behind Monte Carlo and, then, the main results the theory provides.

4.2.1 The Idea Behind Monte Carlo

To present the Monte Carlo method with a straight and widely used example: the estimation of π.

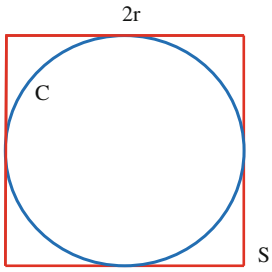

Fig. 4.3 Circle C is inscribed in *square* S representing the sampling world. Defined as $\Pr_C$ the probability of extracting a point belonging to the *circle*, then $\pi = 4 \Pr_C$

Example 1: a probabilistic estimate for π

Consider a square S of side length $2r$ and a circle C inscribed within the square (see Fig. 4.3). Assume that a uniform distribution is induced on the square so that each sample drawn from there is equiprobable. Draw then n points inside the square and observe, for each point, whether it belongs to the circle or not. In doing this a straight question would be to ask which is the probability $\Pr_C$ of extracting a point belonging to the circle.

The answer is that such a probability is simply the ratio between the area of the circle and that of the square, i.e., its value is $\frac{\pi}{4}$. Then, $4 \Pr_C$ is exactly π: we found a way to compute π with a probabilistic approach.

The issue now becomes that of computing $\Pr_C$ which, a priori, is unknown. We solve the problem with randomization by extracting n samples $s_1, \ldots, s_n$ from S according to the uniform distribution, and evaluating the number of samples n_C falling within the circle and computing the empirical probability

$$\hat{p}_n = \frac{n_C}{n}.$$

The procedure is formalized as follows: Consider the indicator function I_C

$$I_C(s_i) = \begin{cases} 1 \text{ if } s_i \in C \\ 0 \text{ if } s_i \notin C \end{cases}$$

The empirical probability $\hat{p}_n$ can be computed as

$$\hat{p}_n = \frac{1}{n} \sum_{i=1}^{n} I_C(s_i)$$

and represents an approximation of $\Pr_C$. Having an estimate for $\Pr_C$, we generate an estimate for π as

$$\hat{\pi}_n = 4\hat{p}_n = \frac{4}{n} \sum_{i=1}^{n} I_C(s_i).$$

How good is the approximation $\hat{\pi}_n$ of π? It is intuitive to believe that the larger the number of samples n the better the estimate (the smaller the error $e(n) = |\hat{\pi}_n - \pi|$).

As such, we should consider a "sufficiently large" n to obtain a good approximation according to some predefined accuracy level. Such aspect will be addressed in Sect. 4.3. Instead, the convergence issue of $\hat{\pi}_n$ to π will be studied in Sect. 4.2.2. A high level algorithm for the Monte Carlo method is given in Algorithm 3.

Example 2: a different probabilistic approach to estimate π

Let us consider a different approach to estimate π with randomization. Consider the equation of a sector of circumference $y = f(x), x, y \in [0, 1]$

$$y = \sqrt{1 - x^2}$$

and observe that π can be obtained as

$$\pi = 4 \int_0^1 \sqrt{1 - x^2} dx.$$

If we induce a uniform distribution f_x on the input domain $[0, 1]$ we have that π can also be intended as the expected value of y

$$\pi = 4E_x[y(x)] = 4 \int_0^1 \sqrt{1 - x^2} dx.$$

We comment that the variance σ_y^2 of y is bound

$$\sigma_y^2 = \int_0^1 (y(x) - E_x[y(x)])^2 \, dx \leq 1.$$

Extract then n samples x_i from $[0, 1]$ and evaluate the sample mean

$$\hat{E}_n(y(x)) = \frac{1}{n} \sum_{i=1}^{n} y(x_i).$$

Then, by invoking the Tchebychev inequality in the form

$$\Pr(|z - \mu| \geq \alpha) \leq \frac{\sigma^2}{\alpha^2}$$

where z is an i.i.d random variable of mean μ, variance σ^2, and α is a positive number [2], we have that

Algorithm 3: The Monte Carlo algorithm

1- Identify the input space D of the algorithm and a random variable s, with probability density function f_s over D;
2- Extract n samples $S_n = \{s_1, \ldots, s_n\}$ from D according to f_s;
3- Evaluate the algorithm on S_n;
4- Generate an estimate of the algorithm output.

$$\Pr\left(|\hat{E}_n(y(x)) - E_x[y(x)]| \geq \varepsilon\right) \leq \frac{\sigma_y^2}{n\varepsilon^2} \leq \frac{1}{n\varepsilon^2}.$$

where the variance of the estimator is $\text{Var}(\hat{E}_n(y(x))) = \frac{\sigma_y^2}{n}$. We can then select the confidence δ as

$$\frac{1}{n\varepsilon^2} < \delta \Longrightarrow n > \frac{1}{\delta\varepsilon^2}.$$

This says that, if we choose $n \geq \frac{1}{\delta\varepsilon^2}$, then

$$\Pr\left(|\hat{E}_n(y(x)) - E_x[y(x)]| \leq \varepsilon\right) \geq 1 - \delta$$

holds with probability $1 - \delta$ and we can estimate π as

$$\Pr\left(|4\hat{E}_n(y(x)) - 4E_x[y(x)]| \leq 4\varepsilon\right) = \Pr\left(|\hat{\pi}_n - \pi| \leq 4\varepsilon\right) \geq 1 - \delta$$

from which we derive the number of points needed to estimate π at a given tolerated level. For instance, if we select $\varepsilon = 0.025$ and $\delta = 0.01$ we need $n \geq 1600$. We extracted $n = 1600$ samples from a uniform distribution f_x and obtained the estimate $\hat{\pi}_n = 3.148$ for which $|\hat{\pi}_n - \pi| = 0.006 \leq 0.1 = 4\varepsilon$.

In the previous experiments, we have implicitly assumed that sampling is associated with a continuous random variable. However, similar results hold by sampling over a discrete space (e.g., a regular grid over $(0, 1)^k$, $k \in \mathbb{N}$).

Interestingly, the second solution proposed to estimate π provides as well the minimum number of samples satisfying a given accuracy and a confidence level. In the sequel, we will be interested in this latter approach by improving bounds so as to reduce the number of samples needed to solve a specific problem after having investigated the asymptotic behavior of the estimate.

4.2.2 Weak and Strong Laws of Large Numbers

In Sect. 4.2.1 we have seen that, by extracting n samples from S it is possible to build sequence $\hat{\pi}_1, \hat{\pi}_2, \ldots, \hat{\pi}_n$; it would be appreciable to discover that such a sequence

converges to the expected value π as the second example showed provided that $n \to \infty$. This main result is known in the literature as the *Law of large numbers*; the interested reader can refer to [14] for its proof.

4.2.2.1 Weak Law of Large Numbers

Let $x \in D$ be a continuous scalar random variable of finite expectation μ and finite variance σ_x^2 and $x_1, \ldots, x_n$ a set of n independent and identically distributed samples drawn from D (e.g., $D = \mathbb{R}$) according to the continuous probability density function f_D. Generate the empirical mean $\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} x_i$. Then, for any $\varepsilon \in D$, the weak law of large numbers guarantees that

$$\lim_{n \to +\infty} \Pr(|\hat{\mu}_n - \mu| \geq \varepsilon) = 0.$$

An identical result also holds for the discrete random variable case.

4.2.2.2 Strong Law of Large Numbers

Let $x \in D$ be a continuous random scalar variable of finite expectation μ and finite variance σ_x^2 and $x_1, \ldots, x_n$ a set of n independent and identically distributed samples drawn from D (e.g., $D = \mathbb{R}$) according to the continuous probability density function f_D. Generate the empirical mean $\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} x_i$. Then, the strong law of large numbers guarantees that relationship

$$\lim_{n \to +\infty} \hat{\mu}_n = \mu$$

holds with probability one.

Comments

The difference between the strong and the weak formulation of the laws of large numbers is in the convergence modality. In the weak case, the probability of generating an estimate $\hat{\mu}_n$ so that $|\hat{\mu}_n - \mu| \geq \varepsilon$ decreases as the number of samples increases. Differently, the strong law of large numbers implies that the sequence $\hat{\mu}_n$ converges to μ with probability one.

When we apply the laws of large numbers to the Monte Carlo method, we have that $\hat{\mu}_n$ converges to μ (and $\hat{\pi}_n$ to π).

The assumption of finite variance is not truly necessary but makes the proof easier. In fact, a large or infinite variance negatively affects the convergence rate. However, the variance must exist. When this assumption does not hold, as it happens in example 3, the laws of large numbers cannot be applied.

Example 3: breaking the law of large numbers

Let $x \in \mathbb{R}$ be a continuous random variable characterized by the Cauchy density function

$$f_x = \frac{1}{\pi(1+x^2)}.$$

Then the expectation $E[x]$ does not exist, because the integral

$$\int_{-\infty}^{+\infty} \frac{x}{\pi(1+x^2)} dx$$

diverges; likewise the variance does not exist, hence violating the assumptions requested by the laws of large numbers. If we compute the sample mean by using n samples drawn from the Cauchy density it can be proved that the average is still ruled by a Cauchy probability density function [26].

A main consequence is that if the noise affecting measurements is ruled by a Cauchy density and we average over a number of measurements (think of the estimation module we introduced in Sect. 2.1.1) to mitigate the presence of uncertainty, then the average cannot be expected to be more accurate than any individual measurement!

4.2.3 Some Convergence Results

The laws of large numbers are rather general and can be applied to several interesting cases among which those related to probability and expected value estimation.

Define the real function $u(\psi)$, $\psi \in \Psi \subseteq \mathbb{R}^l$ to be measurable according to Lebesgue in Ψ and denote by f_ψ the probability density function of a random variable ψ with support on the input space Ψ. Assume that ψ has finite mean and variance.

4.2.3.1 Probability Function Estimation

The problem can be formalized as follows: Given a generic value $\gamma \in \mathbb{R}$, evaluate the probability $p(\gamma)$ for which $u(\psi)$ is below γ when ψ spans Ψ, i.e., compute

$$p(\gamma) = \Pr(u(\psi) \leq \gamma).$$

In other terms, we are asking if the embedded system is satisfying a given constraint γ given performance function $u(\psi)$. Formulation of probability $p(\gamma)$ in a closed form can be achieved only in particular cases, e.g., for very specific choices of $u(\psi)$ and f_ψ. However, the problem can be addressed and solved by resorting

Algorithm 4: Estimating the probability that a requested performance value is attained

1- Extract n independent and identically distributed samples $Z_n = \{\psi_1, \ldots, \psi_n\}$ from Ψ according to f_ψ;
2- Evaluate, for the i-th sample ψ_i, the indicator function

$$I(\psi_i) = \begin{cases} 1 \text{ if } u(\psi_i) \leq \bar{\gamma} \\ 0 \text{ if } u(\psi_i) > \bar{\gamma}. \end{cases}$$

3- Construct the estimate $\hat{p}(\bar{\gamma})$ of $p(\bar{\gamma})$ as

$$\hat{p}_n(\bar{\gamma}) = \frac{1}{n} \sum_{i=1}^{n} I(\psi_i)$$

to randomization. In the following, we aim at solving the problem with the laws of large numbers and, to this end, we assume at first that γ is given and assumes value $\bar{\gamma}$. However, obtained results are valid for any $\bar{\gamma}$.

Extract n independent and identically distributed samples $Z_n = \{\psi_1, \ldots, \psi_n\}$ from $\psi \in \Psi$ according to f_ψ and evaluate the indicator function

$$I(\psi_i) = \begin{cases} 1 \text{ if } u(\psi_i) \leq \bar{\gamma} \\ 0 \text{ if } u(\psi_i) > \bar{\gamma} \end{cases}$$

The estimate $\hat{p}_n(\bar{\gamma})$ of $p(\bar{\gamma})$ is

$$\hat{p}_n(\bar{\gamma}) = \frac{1}{n} \sum_{i=1}^{n} I(\psi_i)$$

Algorithm 4 summarizes the needed steps to provide an estimate $\hat{p}_n(\bar{\gamma})$ of $p(\bar{\gamma})$.

The laws of large numbers hold under the respective hypotheses and, for any $\varepsilon \in (0, 1)$ we have that

weak law of large numbers

$$\lim_{n \to +\infty} \Pr(|\hat{p}_n(\bar{\gamma}) - p(\bar{\gamma})| \geq \varepsilon) = 0$$

strong law of large numbers

$$\lim_{n \to +\infty} \hat{p}_n(\bar{\gamma}) = p(\bar{\gamma})$$

with probability one.

In other terms $\hat{p}_n(\bar{\gamma})$ converges to $p(\bar{\gamma})$. The obtained results, evaluated for a given $\bar{\gamma}$ value, can now be extended to deal with any given γ value (different

γs will experience different convergence rates). We can then write, for an arbitrary given γ that

weak law of large numbers

$$\lim_{n\to+\infty} \Pr(|\hat{p}_n(\gamma) - p(\gamma)| \geq \varepsilon) = 0, \quad \forall \gamma \in \mathbb{R}$$

strong law of large numbers

$$\lim_{n\to+\infty} \hat{p}_n(\gamma) = p(\gamma), \quad \forall \gamma \in \mathbb{R}$$

with probability one.

4.2.3.2 Expected Value Estimation

Another interesting case, which can be immediately derived from the theory, refers to the problem of evaluating the expected value

$$E_\Psi[u(\psi)] = \int_\Psi u(\psi) f_\psi d\psi$$

through the empirical mean

$$\hat{E}_n(u(\psi)) = \frac{1}{n} \sum_{i=1}^{n} u(\psi_i).$$

where ψ_is have been extracted according to f_ψ.

In this case, we wish to evaluate some expected performance the embedded system should have based on measured instances telling us how the system performs for a given input.

Convergence of $\hat{E}_n(u(\psi))$ to $E_\Psi[u(\psi)]$ is granted under the assumptions of the laws of large numbers.

weak law

$$\lim_{n\to+\infty} \Pr(|\hat{E}_n(u(\psi)) - E_\Psi[u(\psi)]| \geq \varepsilon) = 0$$

strong law

$$\lim_{n\to+\infty} \hat{E}_n(u(\psi)) = E_\Psi[u(\psi)]$$

with probability one.

The goodness of the estimate can be evaluated by taking expectation with respect to the sequence of n samples in Z_n. In particular, it can be proved, e.g., by referring to [22], that the variance of the estimate is

$$\text{Var}\left(\hat{E}_n(u(\psi))\right) = E_{Z_n}\left[\left(E_\Psi[u(\psi)] - \hat{E}_n(u(\psi))\right)^2\right] = \frac{\text{Var}(u(\psi))}{n}.$$

The result has a main conceptual impact and states that the variance of the estimate is the variance of function $u(\psi)$ scaled by n^{-1}. The above expression states that if $\text{Var}(u(\psi))$ and $\text{Var}(\hat{E}_n(u(\psi)))$ are bound, we can estimate a priori the number of samples needed to obtain a required accuracy in the estimate. In fact, if we know the variance $\text{Var}(u(\psi))$ (or it is possible to provide a bound for it) and we set $\text{Var}(\hat{E}_n(u(\psi)))$ at a tolerated level c, then the number of samples to be drawn is

$$n \geq \frac{\text{Var}\,(u(\psi))}{c}.$$

4.2.4 The Curse of Dimensionality and Monte Carlo

The *Curse of dimensionality* refers to the bad scaling of the number of points n needed to explore a space as its dimension d increases. Consider the segment $\Psi = [0, 1)$ and subdivide it into $N = 10$ points so that each segment has resolution of 0.1. It comes out that, if we wish to keep the same grid resolution for a d dimensional space, the number of points we need to consider to "explore" the space is $n = N^d$. Such an exploration of the space grows exponentially with d and, soon, becomes computationally prohibitive.

The "curse of dimensionality" represents a major problem every time we need to sample a space and take future actions, e.g., if our task is to estimate the function $E_\Psi[u(\psi)]$ through $\hat{E}_n(u(\psi))$.

However, as nicely pointed out in [2] the mean square error of the Monte Carlo estimate of the expected value does not depend on the dimension d of the space which, somehow, breaks the "curse of dimensionality." As it will be clear in Sect. 4.3 this is a consequence of the fact that we associated a probability density function to Ψ: instead of exploring Ψ with a uniformly-spaced grid we do that by extracting the due number of points according to f_ψ. In other words, the curse of dimensionality can be avoided if we move our analysis from a strictly deterministic to a probabilistic framework.

4.3 Bounds on the Number of Samples

With Monte Carlo, we have seen that it is difficult to estimate the number of samples n we should consider to solve a given problem. Results, e.g., see [15–17], exploit some trial tests or a priori information about the specific problem to decide when

stopping the sampling procedure. In other cases, e.g., as it happens in Example 2, we were able to identify the minimum number of points required to satisfy the accuracy and confidence requirements.

However, this cannot be granted for a generic application, characterized by a generic Lebesgue measurable function. Moreover, since we are looking for generality so as to cover a large set of applications, a pdf-free approach must be considered. The price we have to pay in a pdf-free framework is the a priori larger number of samples needed to solve our problem compared with that we would need by knowing the probability density function.

Several improved bounds on the number of samples n have been presented in the literature to solve large classes of problems through randomization. We will review such bounds starting from Bernoulli's one.

The theoretical framework is that of a Bernoulli process where the random variable x assumes value 1 with probability p and value 0 with probability $1-p$. The expected value is $E[x] = p$ and the variance $\text{Var}(x) = p(1-p)$. Denote by $x_1, \ldots, x_n$ the sequence of n independent samples drawn from x and compute the empirical mean

$$\hat{E}_n = \frac{1}{n}\sum_{i=1}^{n} x_i$$

which represents the estimate of the probability that $x = 1$ in the n trials. $\hat{E}_n$ is a binomially distributed variable with expected value $E[\hat{E}_n] = p$ and variance $\text{Var}(\hat{E}_n) = \frac{p(1-p)}{n}$.

4.3.1 The Bernoulli Bound

Inequality

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - \delta$$

holds for any accuracy level $\varepsilon \in (0, 1)$ *and confidence* $1 - \delta, \delta \in (0, 1)$ *provided that at least* $n \geq \frac{1}{4\delta\varepsilon^2}$ *independent and identically distributed samples are drawn.*

The proof follows by recalling the Tchebychev theorem in the form

$$\Pr\left(|x - \mu| \geq \alpha\right) \leq \frac{\sigma^2}{\alpha^2}$$

where x is the random variable of mean μ, variance σ^2, and α is a positive number. By substituting x with $\hat{E}_n$ and α with the accuracy variable ε, we obtain

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| \geq \varepsilon\right) \leq \frac{p(1-p)}{n\varepsilon^2}. \tag{4.1}$$

Since $p(1-p)$ is maximized by $\frac{1}{4}$, we can be finally bound (4.1) as

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| \geq \varepsilon\right) \leq \frac{1}{4n\varepsilon^2}. \tag{4.2}$$

By introducing a confidence value $\delta \in (0, 1)$, we can rewrite (4.2) as

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) \geq 1 - \delta. \tag{4.3}$$

By setting

$$\frac{1}{4n\varepsilon^2} \leq \delta$$

we derive the number of samples granting (4.3) to hold.

$$n \geq \frac{1}{4\delta\varepsilon^2} \tag{4.4}$$

Comments

The Bernoulli bound shows that the number of required samples grows quadratically (inversely proportional) with the requested accuracy for the estimate ε and linearly (inversely proportional) with the requested confidence δ. We can obtain a good estimate of $\hat{E}_n$ with a polynomial sampling of the space. For instance, with the choice $\varepsilon = 0.05, \delta = 0.01$ we need to extract at least $n = 10000$ samples; with the choice $\varepsilon = 0.02, \delta = 0.01$ we need to extract at least $n = 62500$ samples. Figure 4.4 shows how the Bernoulli bound scales with δ and ε. We recall we shall consider small values for δ and ε to have enough confidence and accuracy.

The cost of sampling is the drawback we have to pay for generality (i.e., any p or application). Fortunately, the Bernoulli bound can be tightened with the Chernoff's one.

4.3.2 The Chernoff Bound

The Chernoff bound [1] largely improves over the Bernoulli's bound by reducing the number of samples to be drawn. We study at first the case where variable x is a Bernoulli random variable.

4.3.2.1 The Bernoulli Case

In the Bernoulli case, the main result states that

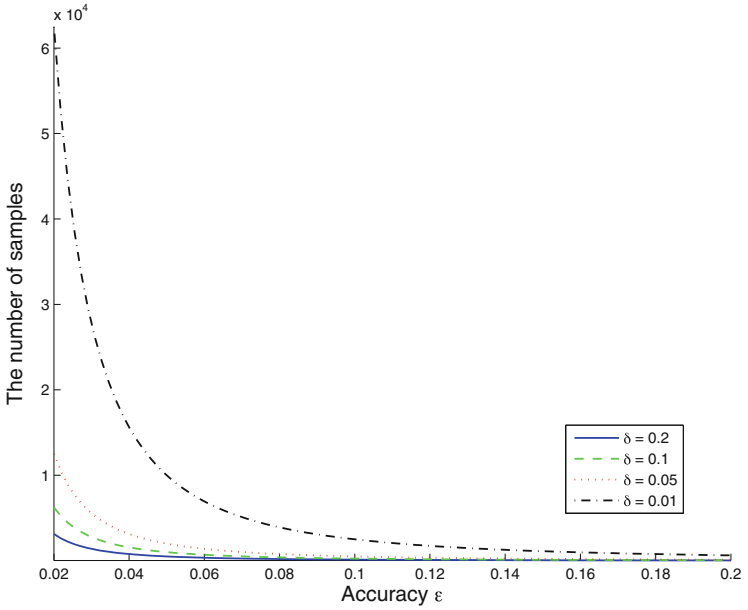

Fig. 4.4 The number of samples requested by the Bernoulli bound

Inequality

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - \delta$$

holds for any accuracy level $\varepsilon \in (0, 1)$ *and confidence* $1 - \delta, \delta \in (0, 1)$ *provided that at least*

$$n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}$$

independent and identically distributed samples x are drawn.

To prove the bound we recall that $E[\hat{E}_n] = p$ and

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) = \Pr\left(|\hat{E}_n - p| < \varepsilon\right) \leq$$

$$\Pr\left(\hat{E}_n < p + \varepsilon\right) + \Pr\left(\hat{E}_n > p - \varepsilon\right).$$

By relying on the Binomial distribution, we can derive analytically those probabilities

$$\Pr\left(\hat{E}_n > p + \varepsilon\right) = \Pr\left(n\hat{E}_n > n(p+\varepsilon)\right) = \sum_{k>n(p+\varepsilon)}^{n} \binom{n}{k} p^k (1-p)^{n-k}$$

and

$$\Pr\left(\hat{E}_n < p - \varepsilon\right) = \Pr\left(n\hat{E}_n < n(p-\varepsilon)\right) = \sum_{k=0}^{k\leq n(p-\varepsilon)} \binom{n}{k} p^k (1-p)^{n-k}.$$

From those expression it is possible to derive the smallest n such that the sum of the two probabilities is greater than $1-\delta$, but no close form solution is known for the problem. Chernoff provided a bound for each of the above terms. In its additive form, we have that

$$\Pr\left(\hat{E}_n \geq p + \varepsilon\right) \leq e^{-2n\varepsilon^2}$$

and

$$\Pr\left(\hat{E}_n \leq p - \varepsilon\right) \leq e^{-2n\varepsilon^2}.$$

thus

$$\Pr\left(|\hat{E}_n - p| \geq \varepsilon\right) \leq 2e^{-2n\varepsilon^2}$$

i.e.,

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - 2e^{-2n\varepsilon^2}.$$

It comes out that

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - \delta$$

holds if we extract at least n samples so that $\delta \leq 2e^{-2n\varepsilon^2}$. This happens if we select

$$n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}.$$

Results, obtained in the case of x distributed as a Bernoulli variable, can be extended to cover the continuous case where the distribution is generic.

4.3.2.2 The General Case: The Hoeffding Inequality

The Chernoff bound for a generic probability density function and continuous variable ψ can be derived from the Hoeffding inequality [18]

Hoeffding inequality

Let $x_1, \ldots x_n$ be a sequence of independent random variables so that each x_i is almost surely bounded by the interval $[a_i, b_i]$, i.e., $\Pr(x_i \in [a_i, b_i]) = 1$. *Then, defined the empirical mean $\hat{E}_n = \frac{1}{n}\sum_{i=1}^{n} x_i$, we have that for any ε value inequality*

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| \geq \varepsilon\right) \leq 2e^{\frac{-2\varepsilon^2 n^2}{\sum_{i=1}^{n}(b_i - a_i)^2}} \tag{4.5}$$

holds.

Under the above assumptions, we can rewrite (4.5) as

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - 2e^{\frac{-2\varepsilon^2}{\sum_{i=1}^{n}(b_i - a_i)^2}}. \tag{4.6}$$

In the interesting case where $\hat{E}_n$ represents the estimate $\hat{p}_n(\gamma)$ of a probability, e.g., $p(\gamma) = \Pr(u(\psi) \leq \gamma)$ for a given positive scalar γ (but any other event applies), we have that for a generic random variable ψ_i the indicator function

$$I\left(u(\psi_i) \leq \gamma\right) = \begin{cases} 1 \text{ if } u(\psi_i) \leq \gamma \\ 0 \text{ if } u(\psi_i) > \gamma \end{cases}$$

I assumes values in $\{0, 1\}$. As a consequence, $a_i = 0$, $b_i = 1$ and (4.6) becomes

$$\Pr\left(|\hat{E}_n - E[\hat{E}_n]| < \varepsilon\right) > 1 - 2e^{-2n\varepsilon^2}.$$

Since, $\hat{p}_n(\gamma) = \hat{E}_n$ and $E(\hat{p}_n(\gamma)) = p(\gamma)$ the expression becomes

$$\Pr\left(|\hat{p}_n(\gamma) - p(\gamma)| < \varepsilon\right) > 1 - 2e^{-2n\varepsilon^2}. \tag{4.7}$$

from which we derive the Chernoff bound by requesting $\delta \leq 2e^{-2n\varepsilon^2}$

$$n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}. \tag{4.8}$$

The Hoeffding inequality plays a major role since it allows us to

- derive the Chernoff bound of (4.8) that will be used to determine the number of samples needed to estimate the probability of performance satisfaction;
- derive the Chernoff bound formally identical to that of (4.8) granting the empirical mean to converge to its expected value with given accuracy and confidence levels;
- derive a set of bounds for estimating the maximum/minimum value of a function within a probabilistic framework.

Figure 4.5 presents the number of samples as function of δ and ε requested by the Chernoff bound.

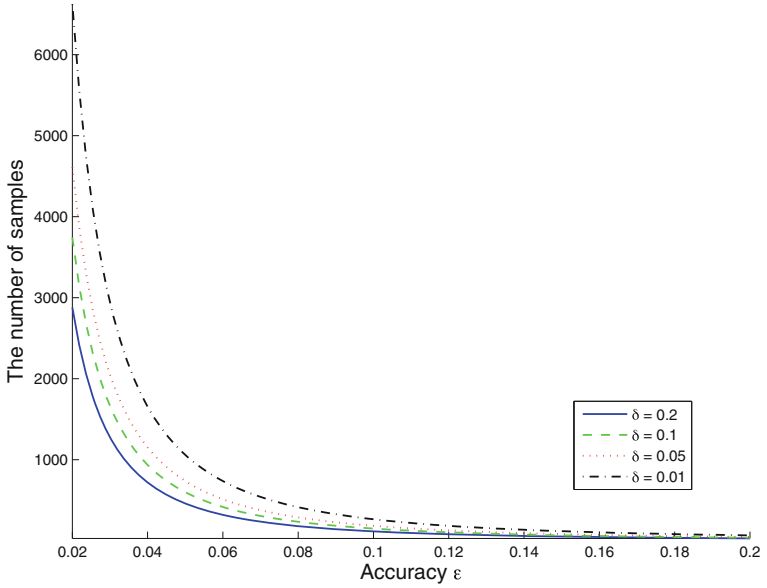

Fig. 4.5 The number of samples requested by the Chernoff bound as function of confidence δ and accuracy ε

Table 4.1 The number of samples $n = n(\varepsilon, \delta)$

Bound	$\varepsilon = 0.05, \delta = 0.02$	$\varepsilon = 0.05, \delta = 0.01$	$\varepsilon = 0.02, \delta = 0.01$	$\varepsilon = 0.01, \delta = 0.01$
Bernoulli	5000	10000	62500	250000
Chernoff	922	1060	6623	26492

Comments

The Chernoff bound shows that the number of required samples grows quadratically (inversely proportional) with the requested accuracy of the estimate ε but logarithmically with the confidence δ. Even if it might appear as a limited gain in reality it is not and represents a true achievement. In fact, if we refer to Table 4.1 we appreciate the significant improvement of the Chernoff bound over the Bernoulli one.

Interestingly, it appears that accuracy is more sampling demanding than confidence since the former is ruled by a quadratic term whereas the latter is bound by a linear one. Figure 4.6 compares the Bernoulli and the Chernoff bound. When δ and ε assume small values, as generally requested by applications since we wish to get high confidence and accuracy, the Chernoff bound significantly improves over the Bernoulli one with a gain $n_c = 2\delta \ln \frac{2}{\delta} n_b$ where n_c and n_b represent the number of samples requested by Chernoff and Bernoulli, respectively.

Other interesting bounds can be obtained by assuming some a priori information about p. For instance, the Chernoff-Okamoto bound [4] is tighter than the Chernoff

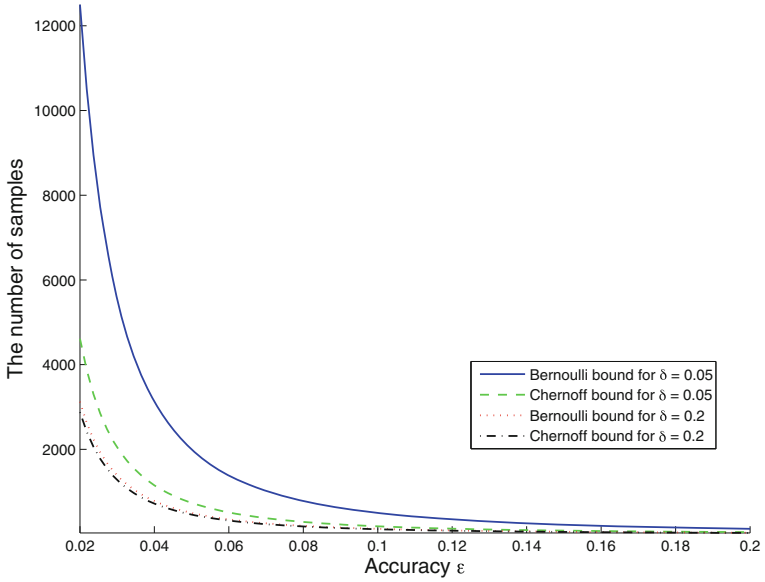

Fig. 4.6 The number of samples requested by the Bernoulli and the Chernoff bounds as function of confidence δ and accuracy ε. Chernoff largely improves over Bernoulli provided that δ and ε assume small values, as requested by applications

one but assumes that $p \leq 0.5$. Other bounds use only one side of the Chernoff bound and can be used to deal with special cases. The interested reader can refer to [2, 4].

As it will be clear in Sect. 4.3.3 the Chernoff bound is one of those main results which make the use of randomized algorithms viable.

4.3.3 A Bound on Samples to Estimate the Maximum Value of a Function

Sections 4.3.1 and 4.3.2 have shown how it is possible to derive bounds on the number of samples needed to guarantee convergence of the empirical mean to its expectation. We show here that many problems such as the verification of a constraint satisfaction problem can be modeled as a realization of a Bernoulli process; at the same time many problems can be reduced to the evaluation of the empirical mean of a quantity.

In this section, we aim at using a sampling technique (randomization) to estimate the maximum value of a function (and, of course, its minimum by changing the sign of the function). Say that we wish to maximize function $u(\psi) \in \mathbb{U} \subset \mathbb{R}$, $\psi \in \Psi \subseteq \mathbb{R}^l$ by identifying the maximum value $u_{\max}$

$$u_{\max} = \max_{\psi \in \Psi} u(\psi).$$

There exists a very large literature addressing the function optimization problem. Different techniques exploit a priori information about the function to be optimized, e.g., as it happens with gradient descent techniques where differentiability is requested. Some techniques explore the search space by looking for regularity and building blocks such as in the case of genetic algorithms; others, explore the search space with a probabilistic approach as in simulated annealing or introduce a blind search strategy as it happens with Monte Carlo. It can be proven that under mild hypotheses on the function to be optimized, all the above techniques converge in probability to the maximum value, also in the case of a blind random search exploration of the parameter space [19]. Different methods either differ in performance accuracy or convergence rate.

Consider the case where random variable ψ, with probability density function f_ψ, is defined over Ψ and generate the estimate

$$\hat{u}_{\max} = \max_{i=1,\ldots,n} u(\psi_i)$$

after having drawn n random samples $\{\psi_1, \ldots, \psi_n\}$. To move back to embedded systems consider $u(\psi)$ as a performance function and ask which is the maximum (minimum) value the function assumes given the fact we can only provide n measurements $u(\psi_i)$. That said, how good is the estimate $\hat{u}_{\max}$? The answer is given by the laws of large numbers.

4.3.3.1 Weak and Strong Laws of Large Numbers for Empirical Maximum

Assume that $u(\psi)$ is continuous in $\psi_{\max} = \text{argmax}_{\psi \in \Psi} u(\psi)$ and that f_ψ assigns a non-null probability to every neighborhood of $\psi_{\max}$.

Then, for any $\varepsilon > 0$ we have that

weak law of large numbers

$$\lim_{n \to +\infty} \Pr(u_{\max} - \hat{u}_{\max} \geq \varepsilon) = 0$$

strong law of large numbers

$$\lim_{n \to +\infty} \hat{u}_{\max} = u_{\max}$$

with probability one.

Since asymptotic results are of scarce utility in real applications we determine a bound on the number of samples granting $\hat{u}_{\max}$ and $u_{\max}$ to be close in probabilistic terms [2].

4.3.3.2 A Bound for a Probabilistic Estimate of the Maximum of a Function

The problem can be simply solved by noting that the determination of the maximum of a function is related to the probability estimation problem addressed in Sect. 4.3.2 and, in particular, Eq. (4.7):

$$\Pr\left(|\hat{p}_n(\gamma) - p(\gamma)| < \varepsilon\right) > 1 - 2e^{-2n\varepsilon^2}. \tag{4.9}$$

In fact, if we set $\gamma = \hat{u}_{\max}$ we have that

$$p(\gamma) = \Pr\left(u(\psi) \leq \hat{u}_{\max}\right) = 1 - \Pr\left(u(\psi) > \hat{u}_{\max}\right)$$

and

$$\hat{p}_n(\gamma) = 1$$

since all taken samples satisfy inequality $u(\psi) \leq \hat{u}_{\max}$ by construction. Therefore, from (4.9)

$$\Pr\left(|\hat{p}_n(\gamma) - p(\gamma)| < \varepsilon\right) = \Pr\left(\Pr\left(u(\psi) > \hat{u}_{\max}\right) < \varepsilon\right) > 1 - 2e^{-2n\varepsilon^2}$$

which holds by selecting n according to the Chernoff bound. However, the bound can be improved as shown in [2] and leads to the final result:

Inequality

$$\Pr\left(\Pr\left(u(\psi) > \hat{u}_{\max}\right) \leq \varepsilon\right) \geq 1 - \delta$$

holds for any accuracy level $\varepsilon \in (0, 1)$ *and confidence* $1 - \delta, \delta \in (0, 1)$ *provided that at least*

$$n \geq \frac{\ln \delta}{\ln(1 - \varepsilon)} \tag{4.10}$$

independent and identically distributed samples are drawn.

Other results about convergence exist, but are outside the goal of this book. The interested reader can refer to [14] where a more complete analysis is carried out. Derived results will be used in Sect. 4.4.2.

4.4 Randomized Algorithms

Consider a problem influenced by some variables grouped in vector ψ with a pdf f_ψ over the space Ψ. Randomized algorithms are algorithms that, by sampling from space Ψ according to f_ψ, provide results valid in probability. The method is general

Algorithm 5: The algorithm behind randomized algorithms

1- Transform the deterministic problem into a probabilistic problem;
2- Identify the input space Ψ of the algorithm and define a random variable ψ, with probability density function f_ψ over Ψ;
3- Identify the accuracy and the confidence levels and, then, the number of samples n required by the randomization process;
4- Draw n samples $S_n = \{s_1, \ldots, s_n\}$ from Ψ according to f_ψ;
5- Evaluate the algorithm on samples in S_n;
6- Provide the probabilistic outcome of the algorithm.

and can be applied to a very large class of functions, namely those Lebesgue measurable: a filter bank, a Fast Fourier Transform (FFT), a discrete cosine transform, wavelets transform, and a generic circuit response function are some very simple examples of Lebesgue measurable functions.

At a very high abstraction level, the procedure behind a randomized algorithm is given in Algorithm 5.

In the following, we will apply randomized algorithm to an interesting class of problems. In Chaps. 5 and 7 results will be applied to the robustness problem and to characterize the level of approximate computation, respectively. Randomized algorithms will also be used to assess the performance of embedded applications as well as evaluate the level of constraints satisfaction within a noise-affected environment.

4.4.1 The Algorithm Verification Problem

The algorithm verification problem aims at evaluating the satisfaction level of an inequality. Even though solving this problem might appear strange, we will see that it constitutes the core of many problems.

Consider function $u(\psi) \in \mathbb{U} \subset \mathbb{R}, \psi \in \Psi \subseteq \mathbb{R}^l$ Lebesgue measurable over Ψ onto which a random variable ψ is defined, with pdf f_ψ over Ψ, and a given, but generic, $\gamma \in \mathbb{R}$ scalar. As we already pointed out the problem models the case where we wish to determine the level of satisfaction of performance function $u(\psi)$ given a constant value γ, generally acting as a tolerated performance. Without loss of generality we study here and in next sections a scalar performance function. However, the simultaneous attainment of several scalar performance functions may be easily handled with the introduced techniques. The problem can be finalized as:

Verify the level of satisfaction of inequality

$$u(\psi) \leq \gamma, \forall \psi \in \Psi.$$

In other words, we wish to determine the "percentage" of points of Ψ satisfying the inequality. Such a value is simply the ratio

$$n_{u(\psi)\leq\gamma} = \frac{\int_{u(\psi)\leq\gamma,\psi\in\Psi} d\psi}{\int_{\Psi} d\psi}.$$

Determination of $n_{u(\psi)\leq\gamma}$ is surely a computationally hard problem for a generic $u(\psi)$ function and cannot be computed in a closed form unless $u(\cdot)$ presents a form that makes the mathematics amenable. Differently, the problem can be solved with a randomized algorithm by transforming the deterministic problem into a probabilistic one. By relying on the previously mentioned probability density function f_{ψ} defined over Ψ, we are able to evaluate the probability

$$p(\gamma) = \frac{\int_{u(\psi)\leq\gamma,\psi\in\Psi} f_{\psi}(\psi)d\psi}{\int_{\Psi} f_{\psi}(\psi)d\psi} = \Pr\left(u(\psi)\leq\gamma\right), \forall\psi\in\Psi.$$

We have seen in Sect. 4.2.3 that $p(\gamma)$ can be evaluated through randomization and that, given a γ value, the event

$$u(\psi)\leq\gamma$$

is associated with the Bernoulli variable

$$\psi\in\Psi : I\left(u(\psi)\leq\gamma\right) = \begin{cases} 1 \text{ if } u(\psi)\leq\gamma \\ 0 \text{ if } u(\psi)>\gamma \end{cases}$$

and by sampling n i.i.d. realizations $\{\psi_1,\ldots,\psi_n\}$ from ψ

$$\hat{p}_n(\gamma) = \frac{1}{n}\sum_{i=1}^{n} I\left(u(\psi_i)\leq\gamma\right).$$

We invoke the Chernoff inequality with $\hat{E}_n = \hat{p}_n(\gamma)$, and $E[\hat{E}_n] = p(\gamma)$ and provide the main result

Performance verification problem

Let $u(\psi)\in\mathbb{U}\subset\mathbb{R}$ be a performance function measurable according to Lebesgue on its input domain $\Psi\subseteq\mathbb{R}^l$ and ψ be a random variable, with probability density function f_{ψ} over Ψ. *Define*

$$p(\gamma) = \Pr\left(u(\psi)\leq\gamma\right)$$

and evaluate the estimate $\hat{p}_n$ from the n i.i.d. samples $\psi_1,\ldots,\psi_n$. Then,

$$\Pr\left(|\hat{p}_n(\gamma) - p(\gamma)|\leq\varepsilon\right)\geq 1-\delta$$

holds for any accuracy level $\varepsilon\in(0,1)$, confidence $\delta\in(0,1)$ and $\forall\gamma\in\mathbb{R}$ provided that

$$n\geq\frac{1}{2\varepsilon^2}\ln\frac{2}{\delta}.$$

Algorithm 6: Randomized algorithms for the algorithm performance verification problem: the given performance loss case $\overline{\gamma}$

1- The probabilistic problem requires evaluation of $p(\gamma) = \Pr(u(\psi) \leq \overline{\gamma})$ for a given $\overline{\gamma}$;
2- Identify the input space Ψ and a random variable ψ, with density function f_ψ over Ψ;
3- Select accuracy ε and confidence δ;
4- Draw $n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}$ samples $\psi_1, \ldots, \psi_n$ from ψ;
5- Estimate

$$\hat{p}_n(\overline{\gamma}) = \frac{1}{n} \sum_{i=1}^{n} I(u(\psi_i) \leq \overline{\gamma}), \qquad I(u(\psi_i) \leq \overline{\gamma}) = \begin{cases} 1 \text{ if } u(\psi_i) \leq \overline{\gamma} \\ 0 \text{ if } u(\psi_i) > \overline{\gamma} \end{cases}$$

6- use $\hat{p}_n(\overline{\gamma})$;

Value $\hat{p}_n(\gamma)$ is the probabilistic outcome of the algorithm.

By using the algorithm given in Algorithm 6 we estimate $p(\gamma)$ for a given $\bar{\gamma}$ so as to solve the problem of determine the level of satisfaction for the inequality, i.e.,

$$\Pr(u(\psi) \leq \bar{\gamma}), \forall \psi \in \Psi.$$

In other applications, we could be interested in constructing function $p(\gamma)$ for an arbitrary large but given and finite set of γs. The natural solution to this problem is to provide a decomposition of the feasible interval of γ, $[a_\gamma, b_\gamma]$ (e.g., with an equally spaced grid) and obtain for each $\gamma \in \Gamma = \{\gamma_1, \ldots, \gamma_k\}$ an estimate $\hat{p}_n(\gamma_i)$ by invoking Algorithm 6 for $i \in \{1, \ldots K\}$. In such a case the algorithm can be extended as in Algorithm 7.

Comments

Randomization has allowed us to solve the algorithm verification problem by transforming the deterministic problem in a probabilistic one. At the same time the Chernoff bound has provided the number of samples satisfying it a given accuracy ε and confidence δ.

Having provided a first complete algorithm based on randomization it is worth to shed light on some operational aspects somehow hidden within the theory.

Here, ε represents the accuracy of estimating $p(\gamma)$, given γ, with $\hat{p}_n(\gamma)$, that is to say it represents an upper bound for the error $|\hat{p}_n(\gamma) - p(\gamma)|$. If ε is small then we can confuse $\hat{p}_n(\gamma)$ with $p(\gamma)$ in our subsequent use of $p(\gamma)$. At the same time we shall note that $|\hat{p}_n(\gamma) - p(\gamma)|$ is a random variable depending on the particular realization of the sampling set. A different sampling set would have provided a different estimate $\hat{p}_n(\gamma)$.

Then one should ask how credible the statement $|\hat{p}_n(\gamma) - p(\gamma)| \leq \varepsilon$ is $\forall \psi \in \Psi$; the answer is that the statement holds with probability $1 - \delta$. This means that we

Algorithm 7: Randomized algorithms for solving the algorithm verification problem

1- The probabilistic problem requires evaluation of $p(\gamma) = \Pr(u(\psi) \leq \gamma)$ for any γ belonging to a finite set of arbitrary γ values;
2- Identify the input space Ψ and a random variable ψ, with density function f_ψ over Ψ;
3- Select accuracy ε and confidence δ;
4- Identify the interested performance level set $\Gamma = \{\gamma_1, \ldots, \gamma_k\}$;
5- $\hat{p}_{n,\Gamma}(\gamma)$ = verification-problem $(\Psi, f_\psi, u(\psi), \Gamma, \varepsilon, \delta)$;
6- use $\hat{p}_{n,\Gamma}(\gamma)$;

function verification-problem $(\Psi, f_\psi, u(\psi), \Gamma, \varepsilon, \delta)$
Draw $n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}$ samples $\psi_1, \ldots, \psi_n$ from ψ;
For each $\gamma \in \Gamma$ estimate

$$\hat{p}_n(\gamma) = \frac{1}{n} \sum_{i=1}^{n} I(u(\psi_i) \leq \gamma), \qquad I(u(\psi_i) \leq \gamma) = \begin{cases} 1 \text{ if } u(\psi_i) \leq \gamma \\ 0 \text{ if } u(\psi_i) > \gamma \end{cases}$$

Group all $\hat{p}_n(\gamma)$s in vector $\hat{p}_{n,\Gamma}$;
Return $\hat{p}_{n,\Gamma}$

could extract a sequence of points for which the inequality $|\hat{p}_n(\gamma) - p(\gamma)| \leq \varepsilon$ is not verified but this happens with probability δ, which needs to be kept small.

As a last note we observe that the sampling space is $\mathbb{R}^l$: the Chernoff bound is independent from the dimension l of the input sampling space. A *small dimension or a large dimension requires the same number of samples*: again we find that randomization has somehow broken the "curse of dimensionality."

4.4.2 The Maximum Value Estimation Problem

The maximum value estimation problem, also known in the literature as worst-case analysis, aims at estimating the maximum value a function can assume.

Consider a $u(\psi) \in \mathbb{U} \subset \mathbb{R}$ function which is Lebesgue measurable over $\Psi \subseteq \mathbb{R}^l$. The problem can be cast in the canonical form requesting the evaluation of

$$u_{\max} = \max_{\psi \in \Psi} u(\psi). \tag{4.11}$$

Analytical determination of $u_{\max}$ is impossible for a large class of functions as the Lebesgue measurable one is and its evaluation might be a computational hard problem.

As we did for the verification case, we generate a probabilistic version of the problem. Observe that the (4.11) can be reformulated as searching for that value $u_{\max}$ of $u(\psi)$ for which

$$u(\psi) \leq u_{\max}, \quad \forall \psi \in \Psi. \tag{4.12}$$

Now we resort to probability by relaxing the deterministic approach intrinsic with (4.12). In particular, we are looking for an estimate $\hat{u}_{\max}$ of $u_{\max}$ and say that the estimate is good if the probability of receiving a ψ for which $u(\psi) > \hat{u}_{\max}$ is small, say assumes value τ.

In other words we are requesting that

$$\Pr\left(u(\psi) > \hat{u}_{\max}\right) \leq \tau. \tag{4.13}$$

Assume that a random variable ψ, with probability density function f_ψ, is defined over Ψ and draw n i.i.d. samples $\psi_1, \ldots, \psi_n$ from ψ. Construct estimate $\hat{u}_{\max}$ as

$$\hat{u}_{\max} = \max_{i=1,\ldots,n} u(\psi_i).$$

As we have seen in Sect. 4.3.3 the weak and strong laws of large numbers grant convergence of $\hat{u}_{\max}$ to $u_{\max}$ in probability.

Unfortunately, solution of (4.13) requires a number of points which is exponential in the dimension of the input space and, as such, the problem solution is computationally hard [6]. To solve this issue we note that (4.13) is again a random variable since different realizations of the sampling set would provide different estimates of $\hat{u}_{\max}$. To address this last aspect, we introduce a confidence value δ and use a second level of probability. Since we have reformulated our problem in a canonical form, we immediately use the bound given in (4.10).

Maximum value estimation problem

Let $u(\psi) \in \mathbb{U} \subset \mathbb{R}$ be a performance function measurable according to Lebesgue on its input domain $\Psi \subseteq \mathbb{R}^l$ onto which is defined a random variable ψ with probability density function f_ψ. Define value $u_{\max}$ to be the maximum value function $u(\psi)$ assumes, i.e.,

$$u(\psi) \leq u_{\max}, \quad \forall \psi \in \Psi.$$

Draw n i.i.d. samples $\psi_1, \ldots, \psi_n$ according to f_ψ and generate the estimate $\hat{u}_{\max}$

$$\hat{u}_{\max} = \max_{i=1,\ldots,n} u(\psi_i)$$

then,

$$\Pr\left(\Pr\left(u(\psi) \geq \hat{u}_{\max}\right) \leq \varepsilon\right) \geq 1 - \delta$$

holds for any accuracy level $\varepsilon \in (0, 1)$, confidence $\delta \in (0, 1)$ and $\forall \psi \in \Psi$ provided that

$$n \geq \frac{\ln \delta}{\ln(1 - \varepsilon)}$$

Algorithm 8: Randomized algorithm to estimate the maximum value of a function

1- The probabilistic problem requires evaluation of $\hat{u}_{\max}$;
2- Identify the input space Ψ and a random variable ψ with pdf f_ψ over Ψ;
3- Select the accuracy ε and the confidence δ levels;
4- $\hat{u}_{\max} =$ Max-estimate $(\Psi, f_\psi, u(\psi), \varepsilon, \delta)$;
5- use $\hat{u}_{\max}$;

Max-estimate $(\Psi, f_\psi, u(\psi), \varepsilon, \delta)$
Draw $n \geq \frac{\ln \delta}{\ln(1-\varepsilon)}$ samples $\psi_1, \ldots, \psi_n$ from ψ according to f_ψ ;
Compute $\hat{u}_{\max} = \max_{i=1,\ldots,n} u(\psi_i)$;
Return $\hat{u}_{\max}$

Table 4.2 The number of samples $n = n(\varepsilon, \delta)$

	$\varepsilon = 0.05, \delta = 0.02$	$\varepsilon = 0.05, \delta = 0.01$	$\varepsilon = 0.02, \delta = 0.01$	$\varepsilon = 0.01, \delta = 0.01$
n	77	90	228	459

Value $\hat{u}_{\max}$ *is the probabilistic outcome of the algorithm.*

The algorithm solving the maximum value estimation problem, i.e., the probabilistic version of the worst case analysis, is given in Algorithm 8.

Comments

As it can be seen from Table 4.2, the required number of samples $n \geq \frac{\ln \delta}{\ln(1-\varepsilon)}$ is well below the one requested by Chernoff to solve the performance verification problem. In fact, for a sufficiently small ε, $\ln(1-\varepsilon) \simeq -\varepsilon$: the number of samples scales as $\frac{1}{\varepsilon^2}$ with Chernoff and $\frac{1}{\varepsilon}$ for the above.

Figure 4.7 compares the number of samples requested by Chernoff with those requested to solve the maximum value estimation problem. We appreciate the fact that the latter bound significantly improves over the former with a gain set by $\frac{1}{\varepsilon}$.

However, since there does not exist a free lunch, the price we have to pay is that our estimate requires two levels of probability

$$\Pr\left(\Pr\left(u(\psi) \geq \hat{u}_{\max}\right) \leq \varepsilon\right) \geq 1 - \delta.$$

The inner inequality $\Pr\left(u(\psi) \geq \hat{u}_{\max}\right) \leq \varepsilon$ states that we are requesting an estimate which is good not in terms of classic accuracy but according to Lebesgue. In other terms the inequality requires that, at least with probability $1-\delta$, the probability of encountering points whose $u(\psi)$ is larger than $\hat{u}_{\max}$ is below ε.

Figure 4.8 shows the situation. Function $u(\psi)$ is given and $\hat{u}_{\max}$ determined as discussed above. Those points $u(\psi) \geq \hat{u}_{\max}$ belong to two intervals Ψ_1, Ψ_2 so that

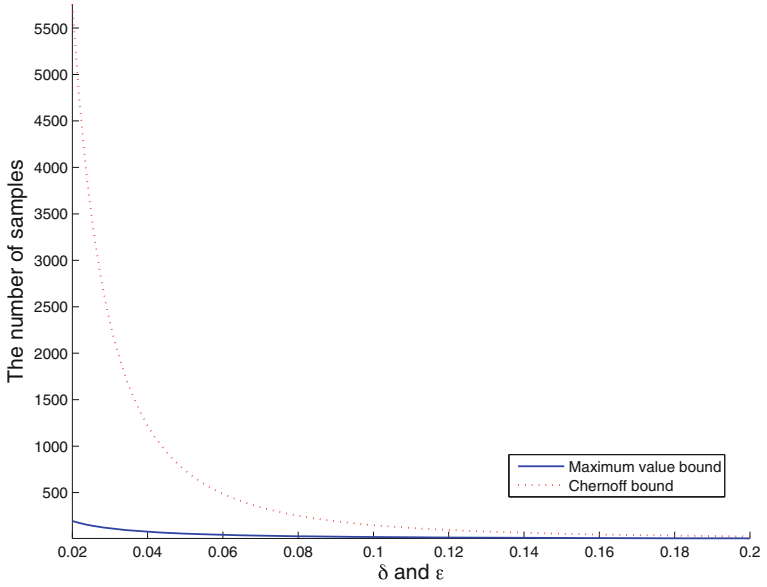

Fig. 4.7 The number of samples requested by the Chernoff bound and that requested to solve the maximum value estimation problem. ε and δ assume the same values to ease the comparison

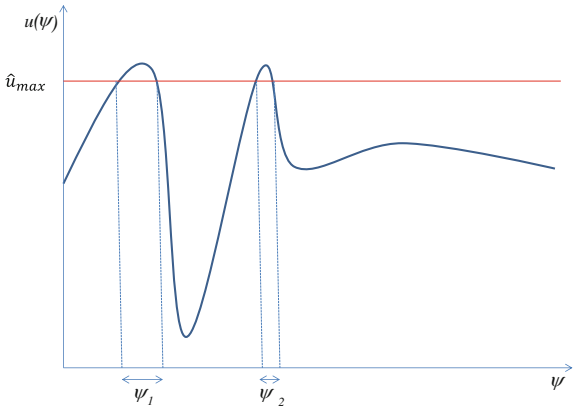

Fig. 4.8 The maximum estimated value for function $u(\psi)$ is $\hat{u}_{\max}$. The probability of having points $u(\psi) \geq \hat{u}_{\max}$ is associated with two supports Ψ_1, Ψ_2, for which $\Pr\left(u(\psi)|_{\psi\in\Psi_1} \geq \hat{u}_{\max}\right) \leq \varepsilon_1$, $\Pr\left(u(\psi)|_{\psi\in\Psi_2} \geq \hat{u}_{\max}\right) \leq \varepsilon_2$ and sum $\varepsilon_1 + \varepsilon_2 \leq \varepsilon$

$\Pr\left(u(\psi)|_{\psi\in\Psi_1} \geq \hat{u}_{\max}\right) \leq \varepsilon_1$ and $\Pr\left(u(\psi)|_{\psi\in\Psi_2} \geq \hat{u}_{\max}\right) \leq \varepsilon_2$, respectively. However, the sum $\varepsilon_1 + \varepsilon_2 \leq \varepsilon$ at least with confidence $1 - \delta$.

There might even be an infinity of points ψ for which $u(\psi)$ is larger than the estimated $\hat{u}_{\max}$ but the probability of encountering such points is no more than ε. This note should be carefully recalled when using the obtained estimates.

It can be proved that the bound is tight under regularization and smoothness hypotheses on the probability function of the random variable $u(\psi)$, for instance continuity (e.g., refer to [7]).

4.4.3 The Expectation Estimation Problem

In many applications it is crucial to be able to estimate the expected value of a given function $u(\psi)$, operation generally carried out by estimating the empirical mean. Again, the problem is to identify the minimum number of samples granting an arbitrary level of accuracy and confidence.

Consider a $u(\psi) \in [0, 1]$ function which is Lebesgue measurable over $\Psi \subseteq \mathbb{R}^l$ and let f_ψ be the probability density function of a random variable ψ defined over Ψ. Expectation estimation requires evaluation of

$$E[u(\psi)] = \int_\Psi u(\psi) f_\psi(\psi) d\psi. \tag{4.14}$$

As in other problems, evaluation of (4.14) is computationally hard for a generic u function and the empirical mean

$$\hat{E}_n(u(\psi)) = \frac{1}{n} \sum_{i=1}^{n} u(\psi_i) \tag{4.15}$$

is constructed instead based on the n i.i.d. samples $\psi_1, \ldots, \psi_i, \ldots, \psi_n$ drawn from ψ according to f_ψ. Of course, $\hat{E}_n(u(\psi))$ is a random variable depending on the particular realization of the n samples. By invoking the Hoeffding inequality (4.5) where $a_i = 0, b_i = 1, i \in \{1, \ldots, n\}$

$$\Pr\left(|\hat{E}_n(u(\psi)) - E[u(\psi)]| \geq \varepsilon\right) \leq 2e^{-2\varepsilon^2 n} \tag{4.16}$$

we derive the Chernoff bound

$$n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}. \tag{4.17}$$

Expectation estimation problem

Let $u(\psi) \in [0, 1]$ be a performance function measurable according to Lebesgue on its input domain $\Psi \subseteq \mathbb{R}^l$ onto which is defined a random variable ψ with probability density function f_ψ. Define $E[u(\psi)]$ to be the expectation of function $u(\psi)$.

Draw n i.i.d. samples $\psi_1, \ldots, \psi_n$ according to f_ψ and generate the estimate

$$\hat{E}_n(u(\psi)) = \frac{1}{n} \sum_{i=1}^{n} u(\psi_i)$$

then,

$$\Pr\left(|\hat{E}_n(u(\psi)) - E[u(\psi)]| \leq \varepsilon\right) \geq 1 - \delta$$

holds for any accuracy level $\varepsilon \in (0, 1)$, confidence $\delta \in (0, 1)$

Algorithm 9: Randomized algorithm to estimate the expected value of a function

1- The probabilistic problem requires evaluation of $E[u(\psi)]$;
2- Identify the input space Ψ and a random variable ψ with pdf f_ψ over Ψ;
3- Select the accuracy ε and the confidence δ levels;
4- Draw $n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}$ samples $\psi_1, \ldots, \psi_n$ from ψ according to f_ψ;
5- Compute $\hat{E}_n(u(\psi)) = \frac{1}{n} \sum_{i=1}^{n} u(\psi_i)$;
6- use $\hat{E}_n(u(\psi))$;

$$n \geq \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}.$$

Value $\hat{E}_n(u(\psi))$ is the probabilistic outcome of the randomized algorithm.

The randomized algorithm to estimate the expected value of a function is given in Algorithm 9.

Comments

Interestingly, the determination of the expected value problem can be addressed with the same number of samples (Chernoff bound) used to address the probability estimation problem. The structural difference is in the use of the empirical sum in one case and the indicator function in the other. Even if their derivations came from a different perspective, both cases are a special case of the Hoeffding inequality (which leads to the Chernoff bound). As a consequence, the request that $u(\psi) \in [0, 1]$ is only made to ease the derivation of the bound through the Hoeffding's inequality. In general, it is enough to require $u(\psi_i)$ bound, e.g., to the same $a_i = a, b_i = b, i = 1, \ldots, n$. As a consequence, the bound on the number of samples would become

$$n \geq \frac{(b-a)^2}{2\varepsilon^2} \ln \frac{2}{\delta} \tag{4.18}$$

Another aspect which should be addressed is the relationship between the number of needed samples as per the Chernoff bound and that which could be derived by applying the central limit theorem. In fact, if $f_{u(\psi)} = f_{u(\psi)}(\mu, \sigma^2)$ the central limit theorem states that as n increases the distribution of $\hat{E}_n(u(\psi))$ approaches the normal distribution with mean value $E[u(\psi)] = \mu$ and variance $\frac{\sigma^2}{n}$ irrespective of $f_{u(\psi)}$. Said that, we can write that

$$\Pr\left(|\hat{E}_n(u(\psi)) - \mu| \leq \lambda \frac{\sigma}{\sqrt{n}}\right) = \operatorname{erf}\left(\frac{\lambda}{\sqrt{2}}\right) \tag{4.19}$$

If we select $\varepsilon > 0$ so that $\varepsilon = \lambda \frac{\sigma}{\sqrt{n}}$, then the implicit relationship between ε, δ, and n is

$$\delta = 1 - \text{erf}\left(\frac{\varepsilon\sqrt{n}}{\sigma\sqrt{2}}\right)$$

since for $x > 0$ we can provide the Chernoff-Rabin bound

$$\frac{1}{2}\left(1 - \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right) \leq \frac{1}{2}e^{\frac{-x^2}{2}}$$

then, being $x = \frac{\varepsilon\sqrt{n}}{\sigma}$ we can write that

$$\delta \leq e^{\frac{-\varepsilon^2 n}{2\sigma^2}}$$

from which

$$n \geq \frac{2\sigma^2}{\varepsilon^2} \ln \frac{1}{\delta}. \tag{4.20}$$

We recall that the Chernoff bound requires $u(\psi) \in [0, 1]$ as a working hypothesis but we commented that results can be extended provided that the variable is bounded. The variance σ^2 might be small. In such a case the bound (4.20) provided by the central limit theorem could slightly improve over the Chernoff bound (the opposite holds). That said, the Chernoff bound should always be preferred independently of the value assumed by σ^2. In fact, (4.20) relies on the assumption that the distribution of the empirical mean is Gaussian which is only true asymptotically with the increasing n and its convergence ratio depends on σ. Differently, the (4.17) is general and does not require any particular assumption on the distribution.

As an example, let us assume that $u(\psi)$ is uniformly distributed in interval $[a, b] = [0, 1]$. Then, the central limit theorem (using Eq. 4.20) would lead to

$$n \geq \frac{2(b-a)^2}{12\varepsilon^2} \ln \frac{1}{\delta} = \frac{1}{6\varepsilon^2} \ln \frac{1}{\delta}$$

against the bound derived from the Hoeffding inequality (Eq. 4.18)

$$n \geq \frac{(b-a)^2}{2\varepsilon^2} \ln \frac{2}{\delta} = \frac{1}{2\varepsilon^2} \ln \frac{2}{\delta}$$

Figure 4.9 presents the bound set by the central limit theorem against the Chernoff one for the choice $\delta = \varepsilon$. As we see the CLT, by taking advantage of the fact the distribution of the empirical mean is Gaussian (when it is only asymptotically), improves over Chernoff that is not assuming any particular distribution.

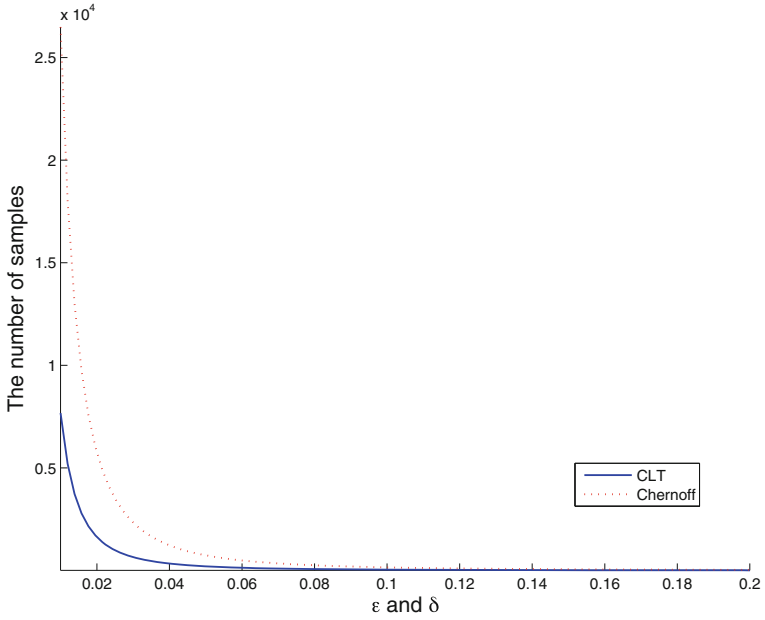

Fig. 4.9 The number of samples requested by the Chernoff bound and the Central limit theorem as function of confidence and accuracy $\delta = \varepsilon$

4.4.4 The Minimum (Maximum) Expectation Problem

The minimum (maximum) expectation problem aims at estimating the minimum (maximum) value of the expectation of a function. Without any loss in generality, we consider here the minimization problem by keeping the same structure given in [2].

Consider the Lebesgue measurable function $u(\psi, \Delta) \in [0, 1]$, $\psi \in \Psi \subseteq \mathbb{R}^l$ and $\Delta \in D \subseteq \mathbb{R}^k$. Define f_Ψ and f_Δ to be the probability density functions associated to random variables ψ and Δ defined over Ψ and D, respectively. The problem requires minimization either of function

$$u_{\min} = \min_{\psi \in \Psi} E_\Delta[u(\psi, \Delta)] \tag{4.21}$$

or

$$u_{\min} = \min_{\Delta \in D} E_\Psi[u(\psi, \Delta)].$$

The two problems are structurally equivalent; as such we consider the first one and the other follows immediately. The problem can then be described by the system

$$\begin{cases} \phi(\psi) = E_\Delta[u(\psi, \Delta)] \\ u_{\min} = \min_{\psi \in \Psi} \phi(\psi). \end{cases}$$

In Sect. 4.4.3, we have seen how the empirical mean converges to its expectation if we draw a number of samples satisfying the Chernoff bound. Let us then consider a given value $\bar{\psi}$ and estimate the expected value $E_{\Delta}[u(\bar{\psi}, \Delta)]$ with its empirical mean

$$\hat{E}_n(u(\bar{\psi})) = \frac{1}{n}\sum_{j=1}^{n} u(\bar{\psi}, \Delta_j) \tag{4.22}$$

based on n i.i.d. samples $\Delta_1, \ldots, \Delta_n$. The Hoeffding inequality can then be applied and leads to

$$\Pr\left(|\hat{E}_n(u(\bar{\psi})] - E_{\Delta}[u(\bar{\psi}, \Delta)]| \geq \varepsilon\right) \leq 2e^{-2n\varepsilon^2} \tag{4.23}$$

from which we derived the Chernoff bound (4.23) holds for $\bar{\psi}$ but it also independently holds *for any* finite sequence of $\bar{\psi} \in \{\psi_1, \ldots, \psi_m\}$ drawn from ψ.

Moreover, we can interpret $u(\bar{\psi}, \Delta)$, as a set of functions parameterized in $\bar{\psi}$ composing the function family A.

We would appreciate the actual mean evaluated on the generic i-th sample $\hat{E}_n(u(\psi_i))$ to be close to the expected value $E_{\Delta}[u(\psi_i, \Delta)]$ for any $\psi_i, i = 1, \ldots, m$.

In other words, we would like the empirical mean to converge to its expectation *uniformly* as n goes to infinity and for each element of the family $A = \{u(\psi_i, \Delta), i = 1, \ldots, m\}$. When this holds we say that function family A satisfies the *Uniform Convergence of Empirical Mean* (UCEM) property. If the family A is finite (say composed of m functions) then, by repeated application of the Hoeffding inequality, we have that

$$\Pr\left(\sup_{u \in A} |\hat{E}_n(u(\psi)) - E_{\Delta}[u(\psi, \Delta)]| > \varepsilon\right) \leq 2me^{-2n\varepsilon^2} \tag{4.24}$$

and, when $n \to \infty$, (4.24) goes to zero. The UCEM property then holds for any finite function family. However, the property might hold also for an infinite function family, e.g. $A = \{u(\psi, \Delta), \psi \in \Psi\}$. It can be proved that the UCEM property holds for all those families for which the Pollard dimension d_P of A is finite [4].

4.4.4.1 The Pollard Dimension

Let Ψ be a measurable space and $F \subseteq [0, 1]^k$ a family of measurable functions. A set of points $\psi_1, \ldots, \psi_n$ is said to be *P-shattered* by F if there exists a real vector $c \in [0, 1]^n$ such that, for every binary vector $b \in \{0, 1\}^n$, there exists a function $f_b \in F$ such that

$$\begin{cases} f_b(\psi_i) < c_i & \text{if } b_i = 0 \\ f_b(\psi_i) \geq c_i & \text{if } b_i = 1 \end{cases}$$

The Pollard dimension d_P of F is the largest integer n for which there exists a set of cardinality n P-shattered by F [4].

To better understand the concept of P-shattered consider a real vector $c \in [0, 1]^n$ and the generic point ψ_i. For each function $f \in F$ we have that $f(\psi_i)$ can be larger (or equal) or smaller than value c_i. Then there are 2^n possible behaviors as f varies in F. Set $\psi_1, \ldots, \psi_n$ is said to be *P-shattered* by F if *each* of the possible 2^n behaviors is realized by some $f \in F$.

The d_P is a generalization of the Vapnik–Chervonenkis (VC) dimension defined on binary valued functions F. Moreover, for binary valued functions $d_P = d_{VC}$ where d_{VC} is the VC-dimension.

When the Pollard's dimension is known, we can state the important Corollary [2]:

The minimum expectation problem. Corollary:

Let $u(\psi, \Delta) \in [0, 1]$ be a performance function measurable according to Lebesgue on its domains $\Psi \subseteq \mathbb{R}^l$ and $D \subseteq \mathbb{R}^k$, onto which are defined the random variables ψ and Δ, respectively, with probability density functions f_Ψ and f_Δ. Let d_P of function $u(\cdot)$ be finite.
Draw m i.i.d. samples $\psi_1, \ldots, \psi_i, \ldots, \psi_m$ from ψ and n i.i.d. samples $\Delta_1, \ldots, \Delta_j, \ldots \Delta_n$ from Δ and compute

$$\hat{E}_n(u(\psi)) = \frac{1}{n} \sum_{j=1}^{n} u(\psi, \Delta_j)$$

$$\hat{u}_{\min} = \min_{i=1,\ldots,m} E_n[u(\psi_i)]$$

then,

$$\Pr\left(\Pr\left(E_\Delta[u(\psi, \Delta)] \leq \hat{u}_{\min} - \varepsilon_1\right) \leq \varepsilon_2\right) \geq 1 - \delta$$

holds for any accuracy level $\varepsilon_1, \varepsilon_2 \in (0, 1)$, confidence $\delta \in (0, 1)$ provided that

$$m \geq \frac{\ln \frac{2}{\delta}}{\ln(\frac{1}{1-\varepsilon_2})}$$

and

$$n \geq \frac{32}{\varepsilon_1^2}\left[\ln \frac{16}{\delta} + d_P\left(\ln \frac{16e}{\varepsilon_1} + \ln \frac{16e}{\varepsilon_1}\right)\right]$$

Value $\hat{u}_{\min}$ is the probabilistic outcome of the algorithm.

Instead, when the Pollard dimension is not know we can use the main result given in the following theorem [2]

Table 4.3 The number of samples $n, m = g(\varepsilon, \delta)$

$\varepsilon_1 = \varepsilon_2 = \varepsilon$	$\varepsilon = 0.05, \delta = 0.02$	$\varepsilon = 0.05, \delta = 0.01$	$\varepsilon = 0.02, \delta = 0.01$	$\varepsilon = 0.01, \delta = 0.01$
(m, n)	(89, 1960)	(104, 2126)	(263, 14451)	(528, 61296)

The minimum expectation problem. Theorem:

Let $u(\psi, \Delta) \in [0, 1]$ *be a performance function measurable according to Lebesgue on its input domains* $\Psi \subseteq \mathbb{R}^l$ *and* $D \subseteq \mathbb{R}^k$, *onto which are defined the random variables* ψ *and* Δ, *respectively, with probability density functions* f_Ψ *and* f_Δ.
Draw m i.i.d. samples $\psi_1, \ldots, \psi_i, \ldots, \psi_m$ *from* ψ *and n i.i.d. samples* $\Delta_1, \ldots, \Delta_j, \ldots \Delta_n$ *from* Δ, *compute*

$$\hat{E}_n(u(\psi)) = \frac{1}{n} \sum_{j=1}^{n} u(\psi, \Delta_j)$$

$$\hat{u}_{\min} = \min_{i=1,\ldots,m} E[u(\psi_i)]$$

then,

$$\Pr\left(\Pr\left(E_\Delta[u(\psi, \Delta)] \leq \hat{u}_{\min} - \varepsilon_1\right) \leq \varepsilon_2\right) \geq 1 - \delta$$

holds for any accuracy level $\varepsilon_1, \varepsilon_2 \in (0, 1)$, *confidence* $\delta \in (0, 1)$ *provided that*

$$m \geq \frac{\ln \frac{2}{\delta}}{\ln(\frac{1}{1-\varepsilon_2})}$$

and

$$n \geq \frac{1}{2\varepsilon_1^2} \ln \frac{4m}{\delta}$$

Value $\hat{u}_{\min}$ is the probabilistic outcome of the algorithm.

Comments

We see from Table 4.3 that the required number of samples can be very high depending on the selected accuracy and confidence levels since the number of samples n is function of the number of samples m, yet through alogarithm.

However, the number of samples required by the corollary is significantly higher than those requested by the theorem. For instance, if we choose $\varepsilon_1 = \varepsilon_2 = \varepsilon = 0.02$

Algorithm 10: Randomized algorithm for the minimum expectation problem

1- The probabilistic problem requires to estimate $\min E[u(\psi, \Delta)]$;
2- Identify the input spaces Ψ, D and random variables ψ and Δ with probability density function f_ψ over Ψ and f_Δ over D, respectively;
3- Select the accuracy ε and the confidence δ levels;
4- Draw $m \geq \frac{\ln \frac{2}{\delta}}{\ln(\frac{1}{1-\varepsilon})}$ i.i.d. samples $\psi_1, \ldots, \psi_i, \ldots, \psi_m$ from ψ;
5- Draw $n \geq \frac{1}{2\varepsilon^2} \ln \frac{4m}{\delta}$ i.i.d. samples $\Delta_1, \ldots, \Delta_j, \ldots \Delta_n$ from Δ according to f_Δ;
6- Compute $\hat{u}_{\min}(\psi_i) = \frac{1}{n} \sum_{j=1}^{n} u(\psi_i, \Delta_j)$ for each i;
7- use $\hat{u}_{\min} = \min_{i=1,\ldots,m} \hat{u}_{\min}(\psi_i)$ and $\hat{\psi} = arg \min_{i=1,\ldots,m} \hat{u}_{\min}(\psi_i)$;

and $\delta = 0.01$ then $m = 263, n = 14,451$ from the Theorem and $m = 263, n = 1,367,851$ from the Corollary with the easiest (yet unlikely) dimension $d_P = 1$. For this reason, we surely use the Theorem's results in the Randomized algorithm framework, mostly with the choice $\varepsilon_1 = \varepsilon_2 = \varepsilon$.

The randomized algorithm for solving the minimum expectation problem is finally summarized in Algorithm 10.

4.5 Controlling the Statistical Volume of the Sampling Space

Randomization requests to sample from a given space Ψ and a random variable with probability density function f_ψ defined over Ψ. By acting on some controlling parameter of f_ψ, we can tune the statistical volume Ψ defined as

$$\text{Vol}(\Psi) = \int_\Psi f_\psi d\Psi$$

which is a very useful operation in many applications. For instance, if we wish to control the space of uncertainty affecting a computation we find useful to introduce a control parameter that allows the shrinkage/enlargement of the space. A norm applied to the vector is a first element that can control it. Another possibility—which can be related to the norm—is the introduction of a mechanism controlling the scattering of points in the space. For their nature, the variance for a scalar and the covariance matrix for a vector can control effectively the statistical volume of a space: the larger the scattering index the larger the embraced volume.

If $\Psi \subset \mathbb{R}^l$, it is common to describe it either in terms of a controllable hypercube or a controllable ball onto which ϕ is defined with pdf f_ψ (both situations can be managed by introducing the concept of norm). In the former case, a common description is such that each component $\psi(i)$ of ψ belongs to a bounded interval, i.e., $\psi(i) \in [a_i, b_i]$. Here, the control of the volume is on a_i and b_i. If we set identical and symmetrical values for a_i and b_i so that $a_i = -\rho, b_i = \rho$, then we have that each edge of the hypercube has length 2ρ and Ψ can be controlled in expansion and contraction with the single parameter ρ and $\Psi = \Psi(\rho)$.

Algorithm 11: The algorithm for extracting vectors according to a uniform distribution from a l_p norm-ball

1- Generate l independent random real scalars ξ_i distributed according to the generalized gamma density function

$$G(x) = \frac{p}{\Gamma(\frac{1}{p})} e^{\xi^p}, \xi \geq 0,$$

where Γ is the gamma function and p the norm value.

2- Construct random vector $x \in \mathbb{R}^l$ of components $x_i = s_i \xi_i$ where s_i is a random sign. Random vector $y = \frac{x}{\|x\|_p}$ is uniformly distributed on the boundary of $\mathbb{B}_\rho$.

3- Return $\psi = \rho y w^{\frac{1}{l}}$, where w is a random variable uniformly distributed in [0, 1]

We recall that if we have a uniform distribution defined in the $[-\rho, \rho]$ interval, the variance is $\frac{\rho^2}{3}$, and the control of ρ implies a control in variance. This situation is formalized by the $\|\psi\|_\infty$ norm

$$\|\psi\|_\infty = \max\{|\psi(1)|, |\psi(2)|, \cdots, |\psi(l)|\}$$

being $\psi(i)$ the i-th component of vector ψ. Following the definition, $\|\psi\|_\infty = \rho$ induces a hypercube of edge 2ρ. In the latter case, e.g., the norm-ball controlled case, ψ is restricted within $\Psi(\rho)$ described in terms of norm-bounded balls of radius ρ

$$\Psi(\rho) = \{\|\psi\|_p \leq \rho\}$$

where

$$\|\psi\|_p = \left(\sum_{i=1}^{l} |\psi(i)|^p \right)^{\frac{1}{p}}.$$

In general, the L^2-norm is used but other norms can be considered to bound Ψ and having it controlled as $\Psi(\rho)$. Interestingly, the maximum norm $\|\psi\|_\infty$ is the limit of the $\|\psi\|_p$ norm when $p \to \infty$.

Though a uniform distribution sample extraction algorithm is immediate for a $\|\psi\|_\infty$ norm where we simply need to uniformly sample from each axis, the problem is more complex if we wish to generate a uniform sampling from a norm-bounded ball. Clearly, verifying the appurtenance of a sample to the ball as we did when estimating π with the square-circle mechanism of Sect. 4.2.1, instead of a hypercube is not an effective solution. Fortunately, [3] provides a simple algorithm that returns a sample ψ belonging to a ball $\mathbb{B}_\rho$

$$\mathbb{B}_\rho = \Psi(\rho) = \{\psi \in \Psi : \|\psi\|_p \leq \rho\}.$$

The algorithm is given in Algorithm 11. Interestingly, if we arrest the algorithm to the second step, we obtain a sample that is uniformly distributed on the boundary $\|\psi\|_p = \rho$.

If $\Psi = \mathbb{R}^l$ and a multivariate probability density function f_ψ is defined for ψ, say Gaussian, then we can control the statistical volume by acting on the covariance matrix C_ψ. The interested reader can refer to [2] for a deeper investigation.