

## PART II

### Basic Statistical Design Techniques

Part II  
Basic Statistical Design Techniques

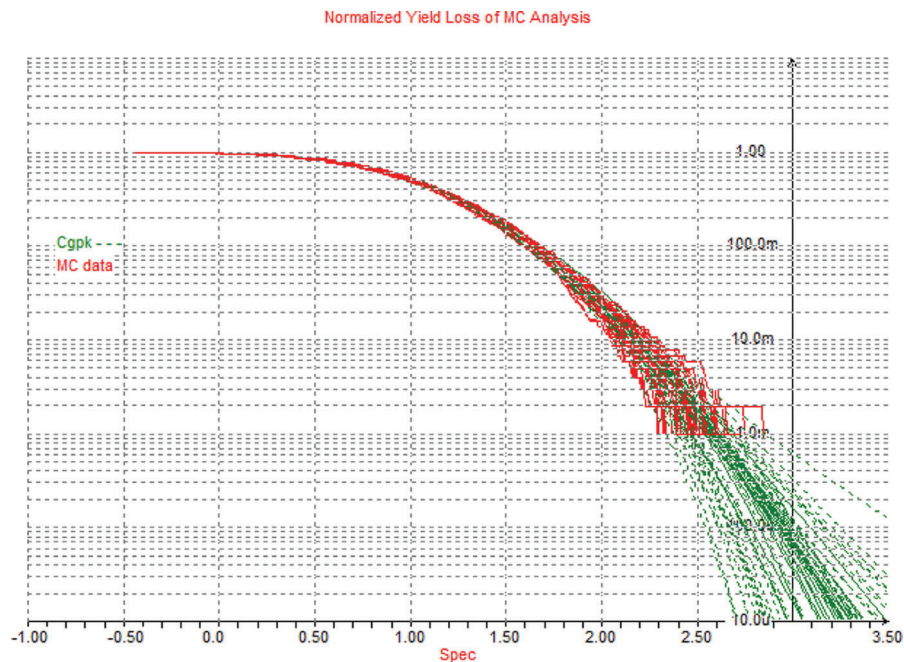
Chapter 3  
Classical Monte-Carlo and Yield Analysis

Chapter 4  
Non-Normal Data Analysis



# 3

## Classical Monte-Carlo and Data Analysis for Yield



A good statistical method can give a speed-up against running all combinations of parameters. Monte-Carlo is so such a technique and practically the most general one. For this reason, most designers use it since many years, but we also want to give clarifications on which uncertainties are usually involved with the different methods. First we focus on MC estimation methods for single real values, like the partial yield or the mean or standard deviation of a certain performance.

Statistics is often regarded as a boring topic. The statistical theory seems to come with a wild bunch of concepts and special terms. Without computers

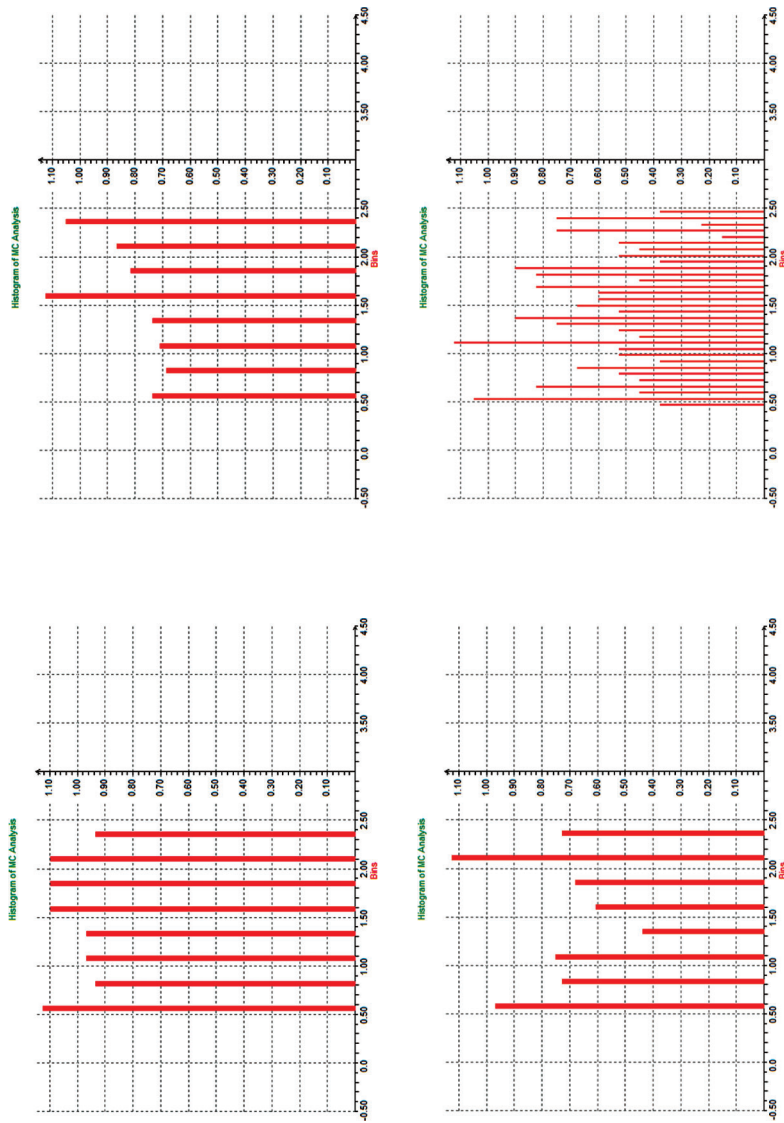
many people would probably agree on this. However, having a computer enables you to do virtual experiments, and running those many times, till you got the feeling “now the results are indeed quite stable.” This way there is no need to do any special calculation: Just setup your problem in a kind of virtual testbench, run it, and collect the data. This way you can build up a very good feeling for statistical data and the uncertainty coming with it.

Unfortunately, too often in real design work it is the other way round: You set up your Monte-Carlo analysis, wait for some hours, inspect the results briefly, and you are done—without much reflection and without detailed and critical result interpretations. Figure 3.1 shows how different even a simple uniform (rectangular) distribution can look like. It also shows nicely how random MC can be. In Chapter 6 we will also check for randomness in higher dimension, with further surprising results. Of course, also in Gaussian distributions there is such randomness *intrinsic* to the *sampling* process!

On the other hand, in spite of runtime problems and special statistical terms, there are good reasons to do problem solving in a statistical way, and surprisingly sometimes MC can be quite efficient! It can be much faster than some people claim and also some speed-up techniques can be applied further—some with very low risks, some with more, some with big speed-up, some with speed-up only in certain cases. We will tell you how to get as much as possible from MC step by step. In Chapters 3–5, we use classical MC techniques available in all circuit simulators, no need for expensive tools!

Our goal is: *You should get a feeling for statistics, as you have a feeling for circuits!*

This is important, because we have to go beyond pure descriptive statistics, and statements like “The proportion of fail samples is 2% in our current MC run”! “Speed-up” sounds good, but is sometimes risky! Sometimes speed-up methods work straight forward, but in other cases there is no one-to-one comparison possible, because two algorithms come with different prerequisites. All-in-all, we as designers have many options and for sure, in reality you have to deal with uncertainties, probability, and statistics, so in the first chapters we focus on the consequences of this for design verification, e.g., yield analysis. Later we extend this to multi-variate analysis, addressing the correlations between performances and statistical variables. This is a bit more complex but can also lead to deeper design understanding! It is also more difficult but doing it efficiently has triggered the invention of several advanced techniques beyond random Monte-Carlo. Later we also extend the



**Figure 3.1** Four histograms generated with MC program (uniform distribution,  $n = 256$ , different seeds, the last one has also a higher bin count—giving less smoothing).

**Table 3.1** When to use what regarding basic statistical techniques

Class	Analysis	Limitations	Applications
MC univariate analysis	Classical diagrams (histogram, cumulated histogram, quantile plots)	Your eye has to decide	First inspection if the design works meaningful, debugging
	Sample yield	Need a lot of points for stable statistic	Yield verification
	$C_{PK}$	Data has to be normal distributed	Yield verification. Use the generalized $C_{PK}$ for non-normal data.
MC multi-variate analysis	Classical diagrams (scatter plots)	Your eye has to decide	Check if parameters are correlated or not
	Correlations, contributions, performance model coefficients	Usually many points needed for stable results	Understand relationships between statistical variables and performances

techniques to support not only yield analysis, but also design optimization for yield improvements.

Some math should not be skipped, so we want to build up intuition about probability density functions (pdf), Monte-Carlo, yield estimation, confidence intervals, etc., but also some terms and measures—less well known in the circuit design community—are also very useful and basically simple, like percentiles, sampling methods, estimates, cumulative distribution function (cdf), worst-case distances, normal quantile plots, etc.

What do we want from numerical algorithms in general and statistical methods in specific?

- Of course speed matters, but usually the time spent is highly dominated by circuit simulation times (including netlisting and extraction of performances) and not by internal runtimes of the statistical parts. This means we need trustable results with a moderate count of simulations. Otherwise we cannot use the methods during the design tweaking phase or in an optimization loop.
- Accuracy matters as well, the results from a MC analysis depend on chance and vary statistically. Usually there is a trade-off between speed and accuracy, but algorithms have also some numerical and systematic errors. Such errors should not increase much for nonlinear problems, at high yields or non-normal distributions.
- For application to complex real-world designs, we also need scalability. Algorithms with strong increase in simulation effort for complex designs

featuring many variables (like  $>1,000$ ) are usually quite limited in application.

- We also need robustness, because circuits often work in a highly nonlinear way. Random data can contain outliers, and also simulations can fail due to nonconvergence. To be scalable, efficient, and robust we often need adaptive algorithms and models. For instance, ranking methods are usually much more robust than classical least-square methods, but this comes with the price of lower efficiency.
- The results should be easy to understand and come with an accuracy indication. For instance, trusting results based on strong assumptions (like data is normal) or extrapolation is riskier than results based on mild assumptions (like pdf has finite variance) and interpolation.
- The setup should be easy, and the results should not depend too much on user settings. Bad settings should be reported including suggestions for an improved setup.

Note [Keynes]: We will deal with many formulas and definitions. From school you may remember that probability itself has been often introduced a bit arbitrarily by the axioms of Kolmogorov, similar to the geometry axioms from Euclid! There are meaningful *other* interpretations on what probability or geometry “is,” but luckily very often for engineers such philosophical details do not matter much! As we can use in our computer near-ideal random number generators we have almost no limitations, whereas in reality often the concept of probability as a kind of limit “frequency of occurrence” is not so easily applicable, because some unknown parameters change the probabilities over time.

### For Further Reading:

Around Monte-Carlo there is a lot of literature. As a starting point, best pick the references related to circuit design, but actually it is very interesting to see also MC working in other fields of science and engineering. *Note*: If you search for “yield estimation” you will often find pure electrical engineering papers, in general or for math literature it is better to search for “percentiles.”

- R. E. Walpole, R. H. Myers, S. L. Myers, K. Ye, *Probability & Statistics for Engineers & Scientists*, 9th Edition, Prentice Hall, 2012.
- D. M. Lane, *Online Statistics Education: An Interactive Multimedia Course of Study*, (<http://onlinestatbook.com/2/estimation/confidence.html>).
- H. Schmid and A. Huber, *Measuring a Small Number of Samples, and the  $3\sigma$  Fallacy: Shedding Light on Confidence and Error Intervals*, IEEE Solid-State Circuits Magazine, vol. 6, no. 2, pp. 52–58, 2014.

- S. Kotz and N. Johnson, Process Capability Indices, Taylor & Francis, 1993.

### 3.1 Corners vs. Monte-Carlo

In a corner analysis the design is stressed at well-defined critical parameter combinations, and this is quite a straightforward scheme. However, what is really Monte-Carlo? How general is it? Besides that Monte-Carlo is a city in the south of Europe, we found no single best definition.

This is the nicest statement about Monte-Carlo I ever heard:

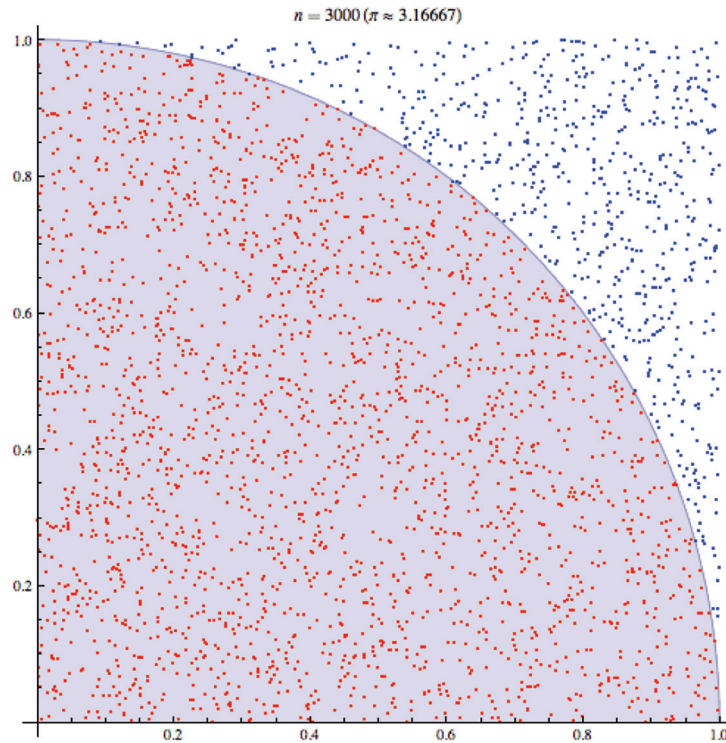
*One single Monte-Carlo point can tell you more about the circuit, than hundred nominal simulations.*

In MC we mimic our real-world system in a computer and use statistical models to include production variations. Even if models are not accurate, MC is useful because we can check our design on robustness. So even one MC sample result is much closer to real world production samples than a nominal simulation, which is actually (partly) an artificial best case (e.g., regarding mismatch)—a kind of Potemkin village. So do not fool yourself and skip doing a MC analysis! A nominal simulation is actually simply not so much “nominal” as you may think! It is more a concept for testing ideas, and to put the design in an almost ideal state. If designers create a real prototype, e.g., on a PCB, they will not create something close to a nominal simulation, they will create one Monte-Carlo sample!

One important measure for robustness is the production yield, but also others (like mean and standard deviation sigma of our output performances) can be of designer’s and quality engineer’s interest. Therefore, MC has found a huge number of applications, like in weather forecasting, chart analysis, biology, etc.

However, mathematicians have another view on MC; here you find things like “Monte-Carlo integration has this and that characteristics,” so basically it works “amazingly well,” e.g., completely independent on shape and dimensions and correlations! So MC is integration? The good thing with math is you can really prove something under certain well-defined prerequisites. And indeed the yield can be related to an integral, and we can prove accurate convergence of the sample yield to the true yield. In a wider sense MC is any technique making use of random numbers for solving problems! The problem itself might have no real relation to random numbers, e.g., integration is a perfect example (see Figure 3.2), like also possible with many other methods (like Simson’s rule, etc.).





**Figure 3.2** Monte-Carlo integration on a circle for calculating  $\pi$ .

Note: The mathematicians seem to “love” integrals and the yield, because there we have proven convergence! You cannot prove that the sample mean, standard deviation, median, etc. will converge in general! In addition: The simplest way of doing MC would be just to run it and look to the performances plots graphically, just to get a feeling how large the performance spreads are, e.g., by placing markers. However, to get histograms you also need an automated performance evaluation (e.g., to get the 3dB-bandwidth BW). For yield analysis you also need specifications (like  $BW > 20$  MHz). These additional entries are almost a prerequisite for all automated methods, so we will not always mention them (Figure 3.2).

One can really prove under very mild assumptions (namely that the samples are identically and independently distributed—i.i.d.) that the MC convergence speed is  $1/\sqrt{n}$  for the yield integral! This sounds that MC speed is quite moderate compared to algorithms like Newton-Raphson having quadratic convergence. For example integration by Riemann’s sum (giving  $1/n$  speed)

or even by Simpson rule is significantly faster, but amazingly not if you do that in higher dimensions (each random variable gives one new dimension) as we have in real circuit problems!

Note that the good behavior of Monte-Carlo also in cases of high complexity is a huge advantage over practically all other algorithms! A corner analysis becomes more difficult if you need to treat many variables, but MC yield estimation not! This problem of “dimensionality” comes back to us and to anybody if we want to improve Monte-Carlo or just replace it by something faster!

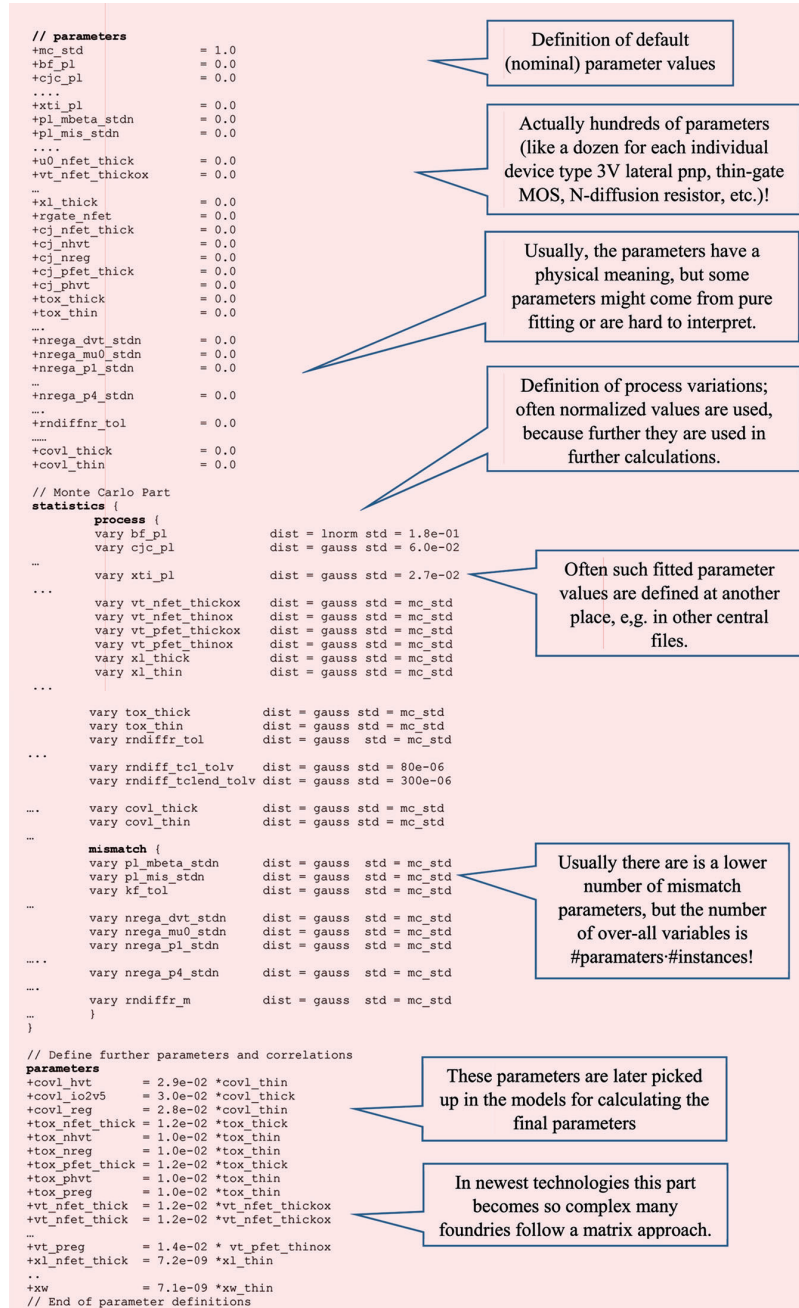
For its general applicability, we should see in MC not only integration, more something like an art of dealing with statistics in a numerical way. In a computer we have many more options and access to variables than we would have in statistical data coming from a vote or from a fab! And we can also take ideas from analytical and combinatorial approaches to tailor the algorithm to our problem structure.

It looks like MC is inaccurate due to slow convergence, but it can be even worse. For more difficult estimates than the sample yield, we may need many more quite fuzzy prerequisites and maybe we cannot prove that the distribution is normal but only asymptotically normal or we simply cannot easily tell the  $1/\sqrt{n}$  convergence starts with a low  $n$  like 20 or with a large  $n$  like 200,000. In some cases, already simple estimates like the sample mean will not even converge at all (inconsistent estimates)! On the other hand, advanced MC schemes may give a much higher convergence rate, but only under restrictions and it may happen that also the convergence stops at some point, like beyond 20,000!

Luckily MC can often be done this way that some self-checking for accuracy is possible (beside just to make a “golden” run with  $1000\times$  more points). A simple way is splitting the data in two equal parts, evaluate them separately, and then compare the results. This kind of cross-correlation analysis is the simplest, not the most efficient one, and many other ways exist. A very crafty method is “bootstrap,” which we will discuss in Chapter 5.

As mentioned, behind the scenery MC uses statistical models (see Figure 3.3), so each statistical parameter is described by its probability density function (pdf); in many of the cases as normal Gaussian distribution given by a certain mean  $\mu$  and standard deviation  $\sigma$ .

However, the designer’s real interest is usually in the performance variations, and in between both there is a long often highly nonlinear circuit simulation. So there is usually a kind of curtain between the user getting just the MC results and the true population defined by the statistical models and the often very complex circuit behavior! In MC (but not in a production)



**Figure 3.3** Statistical section of a simulator model card for a typical ultra-deep sub-um process.

we can inspect the statistical models and can obtain the exact value of mean and sigma for all statistical parameters, but we cannot do that for the circuit performances. It is not even clear what kind of distribution the circuit performances follow! As circuits often act nonlinear, the originally Gaussian distributions usually appear “distorted,” so becoming non-Gaussian at the output!

To give a first summary of corner vs. MC and statistics:

- Verification using foundry-provided corner combined with environmental corners is only leading to a trustable verification if your design is pure CMOS logic and if mismatch has almost no impact!
- Foundry-corner-based verification is inaccurate for typical analog circuits and performances, so it can lead to under-design. It might also lead to over-design, e.g., if the foundry corners are related to  $6\sigma$ , but you design a high-performance circuit and you are already happy with  $2\sigma$  yield.
- For yield analysis you need statistical techniques, but by only inspecting the sample yield you need many MC samples, especially for high-yield verification ☹.
- The corner method may become time-consuming too if you have to cover many variables and performances.

**Is “Worst-Case” a precise term?** Indeed if something is bad, you can probably make it still even worse, but of course if our design spec is for a certain temperature range like  $-40$ – $125^{\circ}\text{C}$ , it makes not much sense add too much further margins. Only some margin can be usually justified due to model inaccuracies. When we talk about WC it is something like a “realistic” WC, i.e., we search for the WC parameter combination within the specified environmental ranges and with a certain (minimum) yield. Pure “digital” WC corners sets like FF, SS, FS, and SF will only represent the speed WC for CMOS logic (for a certain yield like  $5\sigma$ ). To extend the corner idea for analog many PDK model set-ups come with further corners, like slowR, fastR, slowC, and fastC. So in principle including also these and all combinations in a corner verification gives you a further insurance. However, the effort increases a lot and you can still not treat well mismatch and correlations. Also on “sigma” you will typically over-design, when combining the  $5\sigma$  slowR with  $5\sigma$ -FF and  $5\sigma$ -slowC corner. Quantifying the overall sigma of such combination is not easy and would rely on further assumptions. Better directly use statistical methods and use corners more to test design improvements and ideas, from time to time, and as small double-check.

**Table 3.2** Overview on corner analysis vs. Monte-Carlo

	Corners Process	MC
Simulation effort for pure CMOS	Low	Medium
Simulation effort for large # of device types	High	Medium
Check timing for full-custom digital designs	Yes	Not efficient
Correct device correlation	No	Yes
Check operating conditions of analog designs	Yes	Yes, but harder to analyze
Check analog performance variability	Inaccurate	Yes
Estimate production yield	No	Yes
Estimate for worst-case performance	Inaccurate	Yield-related
Obtaining process parameter sensitivities	Too inaccurate	Yes <sup>1)</sup>
Obtaining parameter & performance correlations	No	Yes <sup>1)</sup>

<sup>1)</sup> See Chapter 5.

With MC methods or gathering and analyzing measured data, you can almost never prove anything, at best only disprove. For instance, your analysis based on assuming a normal Gaussian distribution might be completely meaningful, but this does not mean that the data is really coming from a normal Gaussian pdf, it might be probably also from a Gaussian mix or from a Gaussian distribution with cut at  $\pm 9\sigma$ , or from a Student-100, etc. Only if you would fully disprove all such alternatives, you might be able to convince other people that in this case assuming a normal Gaussian is really the only correct method. Typically at some point you have to take the risk of relying on statistical methods, as you also take some risk in relying on models, etc.

Note: We named this subsection “Corners vs. Monte-Carlo,” but later (in Chapters 7 and 9) we will see that both methods can be combined, which means we can make the—native and quite fast—corner verification methodology more accurate by fully adapting it to our analog problems and to include mismatch. This way we get better understanding and can also solve difficult problems efficiently like the verification of high yield targets (like  $6\sigma$  or 1 ppb).

### 3.2 Questions and Answers: Test Yourself

There are some limitations to MC and also other questions come up, especially as many designers have at least some basic knowledge about statistics which they may remember:

1. What would happen if all our element parameter distributions would be uniform instead of Gaussian? How would that change the histogram of PSRR or  $I_{DD}$ ?

*It would usually change not much, so the histograms may still look quite Gaussian! This is due to “central limit theorem” CLT. The uniform pdf has a clear cut (roughly at  $1.5\sigma$ , whereas the normal pdf has no limit), these cuts would almost disappear. For instance, a differential pair could give  $3\sigma$  maximum offset roughly, because one transistor could be at  $1.5\sigma$  in the worst-case and the other too, giving  $3\sigma$  in total. And the more variables are involved the less the cuts have an impact! Already summing e.g., ten uniform variables will give a distribution which is very similar to a true (uncut) normal Gaussian distribution!*

2. The “central limit theorem CLT” tells us that if we add the samples from many different statistical variables we will anyway approach the normal distribution to a high degree—even independently on the distribution shape of the original variates! So also a MC analysis on a circuit design should give normal histograms?

*No, because the CLT comes with further restrictions like need for finite sigmas of the individual distributions. Also in circuit design we often do not add up enough statistical variables to obtain really a good approximation; and circuits do not always simply add variables, also multiplication and division appear!*

3. Assume the requirements for the CLT are fulfilled, how fast will we approach the normal distribution?

*Also this depends on several factors: If we add samples from a uniform distribution, then a good approximation is often already reached if adding just 10 samples. However, this approximation is usually only good in the distribution center, like  $\mu \pm 2\sigma$ , but not at  $5\sigma$ !*



4. MC is an almost universal method if you are interested in the yield—that is mathematically proven. And another assumption is usually that if we extend the number of MC points, we can always improve accuracy on estimates like the mean.

*Even this assumption is not correct, it is not true for the mean on a Pareto distribution or for the standard deviation of a Student-2! In both cases we would observe that the sample variations grow instead of becoming smaller.*



5. Is MC working correctly if we have an infinite number of statistical variables in our design? Can we allow a random number of random variables?

*Good news, random MC will still converge, e.g., the yield estimation would be not impacted at all, but unfortunately many MC extensions run into problems (e.g., LHS and LDS, see Chapter 6).*

6. If the distribution of a certain output performance is not Gaussian, can we still make estimations (e.g., on yield) from this MC data?



*Also here MC is flexible and reliable! For instance you may assume another specific distribution (like lognormal or Weibull) or use more general theorems like Chebyshev theorem (which makes no pdf shape assumptions, only the variance has to exist)!*

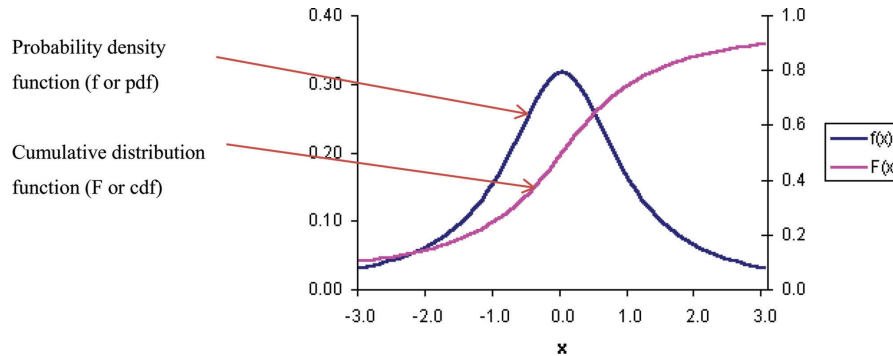
*Chebyshev's theorem states that the proportion of observations falling within  $k$  standard deviations  $\sigma$  of the mean  $\mu$  is at least  $1 - (1/k^2)$  (for  $k \geq 1$ ), so the  $\pm 3\sigma$  area covers at least only 90% (much less than 99.7% for a normal distribution!).*

### 3.3 Important Definitions and Concepts

To prepare a MC analysis, the design environment or the simulator needs access to statistical models (see Figure 3.3). And typically the technology parameters follow a *continuous* probability density function pdf ( $x$ ), e.g., they show a normal or lognormal behavior but there are also many other well-known distributions and all have their meaning and application.  $\text{pdf}(x)dx$  gives us the probability  $P$  that the random variable is within the interval  $(x, x + dx)$ , so the pdf is related to the *frequency* of occurrence.

When taking random samples ( $X_1, X_2, X_3, \dots, X_n$ ) from the pdf we can put the data into a histogram and get a staircase approximation to the probability density function pdf. The cumulated histogram, showing the yield, is giving an approximation to the cdf—the cumulated distribution function (sometimes also called integrated distribution function). This approximation is called the empirical cdf and is a staircase-shape monotonous function, like the cdf starting from  $y$  (first sample) = 0 to  $y$  (last sample) = 1.

Figure 3.4 shows the Cauchy distribution, not the normal distribution! This gives an example that it is quite easy to choose the wrong distribution. Actually *many* distributions have a center and tail regions (featuring the rare events causing pain); and look bell-shaped.

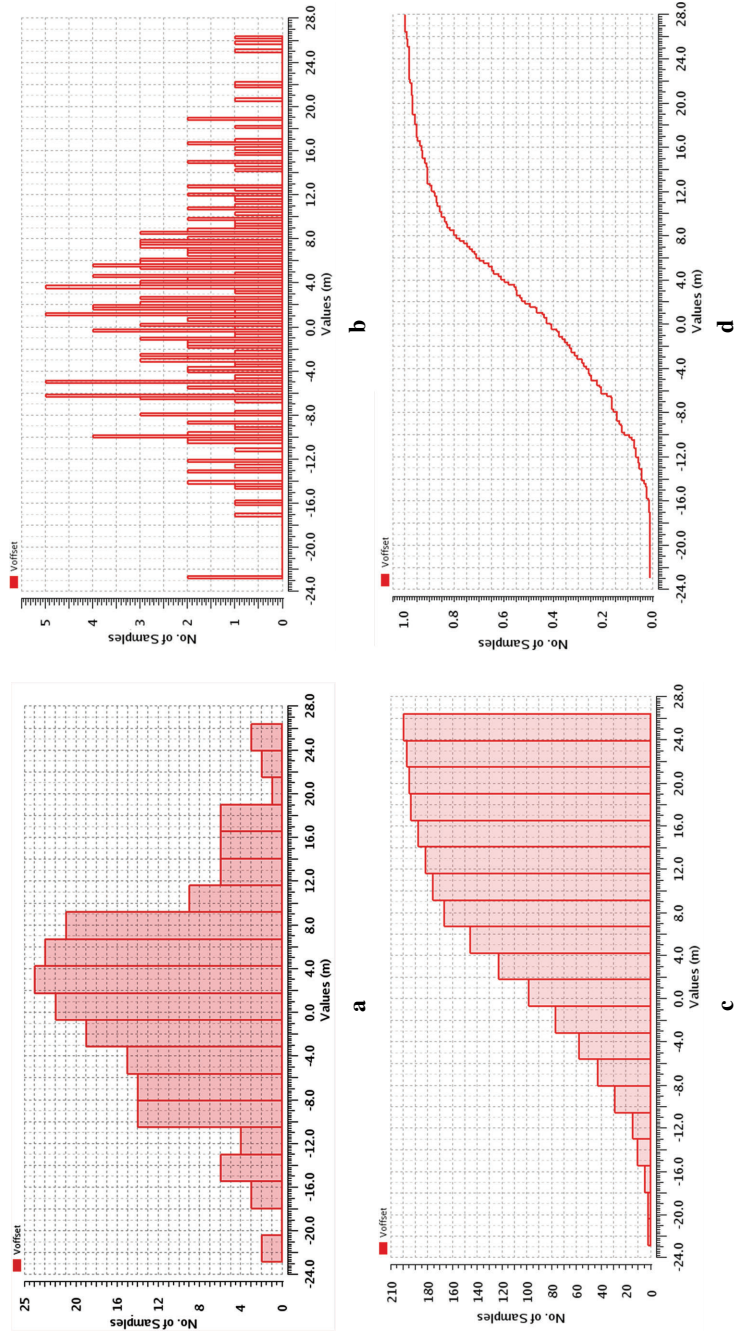


**Figure 3.4** Pdf and cdf of a typical continuous distribution.

Also discrete pdfs exist, e.g., the sample yield or dice can only take discrete values. Mathematically the differences often do not matter, e.g., both kinds can be displayed in a histogram, and we use the same formulas for yield estimation, mean, standard deviation, etc. Only the concept of density is more difficult in the discrete case, because mathematically we require Dirac pulse functions; the cdf is a staircase function for discrete distributions (like the empirical cdf, Figure 3.5d.).

**Manipulating Histograms?** To visualize MC data the histogram is a good starting point, but one big question is how to choose the number of bins. If you want high resolution you need to select a large bin count (like 30 for 200 MC points). This gives quite a noisy histogram with many small peaks, so if you want to demonstrate that your MC data depends highly on chance this is a good method! If you want to demonstrate that you can trust the MC data a lot better use a very small bin count. Actually the optimum number of bins depends on MC count and on distribution type. For normal data and not too small counts, Sturges formula might be used  $\text{bin} = \log_2(n) + 1$ , but it smoothes quite much, so you will probably not see if your distribution has two or more modes! Many programs use  $\text{bin} = \sqrt{n}$  or  $2n^{0.33}$  (Rice's rule). Note, that the cumulated histogram has several advantages: You can directly readout the yield and the binning does not matter so much, as even with  $\text{bin count} = n$  you would still get a monotonic plot, i.e., actually there is no need for binning! A general problem with histograms is that the tail region that dominates the yield is hard to check in the cumulated plot just the deviation from 1.0 counts, and 0.1% is almost impossible to read out.





**Figure 3.5** Histograms with different bin counts, cumulated histogram, and empirical cdf plot (op-amp offset voltage, 200 points MC run)—tail region marked yellow.

The cdf behaves like the yield, so it starts at  $y = 0$ , ends at  $y = 1$ . Often the  $x$ -range is from  $-\infty$  to  $\infty$ , but sometimes it is limited (as for the uniform), to positive values or a certain range.

$$\text{cdf}(s) = \int_{-\infty}^s \text{pdf} \, dx \quad (3.1)$$

The cdf and pdf are programmed into the random generators of the simulator, and the parameters (like mean and standard deviation for a normal distribution) are read from model files. Often the reverse task is required, like you want to hit a certain yield  $Y = 90\%$  so the cdf is 0.9, and now you want to search which spec-setting is required to hit this point. This requires the inverse function to the cdf; the  $\text{cdf}^{-1}$  is usually just called the percentile function. For the uniform distribution, the pdf is constant and the cdf (or  $\text{cdf}^{-1}$ ) is a linear ramp. For a normal Gaussian distribution the cdf or  $\text{cdf}^{-1}$  is nonlinear, but if we have uniform random variables between 0 and 1 we can generate any other kind of distribution by using the  $\text{cdf}^{-1}$  as transformation. This is not necessarily the easiest way to practically generate random numbers for a certain wanted distribution (like normal, Cauchy, exponential, lognormal, etc.), but for theoretical analysis this can be helpful, so later in the chapter about advanced Monte-Carlo sampling methods we focus often on uniform distributions.

Prometheus – Johann Wolfgang von Goethe:

<i>Bedecke deinen Himmel, Zeus,</i>	<i>Cover thy spacious heavens,</i>
<i>Mit Wolkendunst.</i>	<i>Zeus, With clouds of mist.</i>

It looks that Zeus followed Prometheus “advise”, and covered not only the sky but many other things too. Statistics can be interpreted in different ways, like talking about “frequencies of occurrence” or use it in where we have a “lack of information”.

Note, the parameters defining a certain distribution are fix numbers—sometimes known (if you inspect the model setup files), sometimes unknown (if the sampling involves a nonlinear circuit simulation)! We need “tricky” inference techniques to obtain such true fix parameters, if we only have the random samples available, and the accuracy of such *inference* can be quite limited.

Many things rely on modeling—not only MC models—so designers often need to add some extra-margin for this, like make wires wider than needed acc. to electro-migration or IR drop requirements or let circuit work to 20% higher  $f_{\text{clk}}$  than needed or add 0.25 dB because of some test equipment limitations.

Also, MC analysis requires margins due to confidence interval widths. Fortunately, even if the models are not perfect, using them is the best you can do and they can help to make a design robust against changes (like in  $T$ , in  $V_{DD}$  or from mismatch). Truly at some points designers have to trust or make a decision how much they trust (like defining a confidence level) or use another algorithm, run more MC points, etc.

Many variables follow a normal Gaussian distribution, and the pdf of the standard normal distribution is given by:

$$N(x, 0, 1) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \quad (3.2)$$

“Standard” means its mean is zero and the standard deviation is unity. The cdf of a normal distribution is related to the error function, which is unfortunately not easy to express in terms of simpler functions; it is just a new function like Bessel functions, etc.

If we have a sample  $Y$  from  $N(0, 1)$ , we can get normal distribution with mean  $\mu$  and standard deviation  $\sigma$   $N(\mu, \sigma)$  by this linear transformation:

$$Y' = \mu + \sigma Y \quad (3.3)$$

$\mu$  is a measure of location and  $\sigma$  a measure of scale.

Moreover, note that this transformation can be applied also for many non-Gaussian distributions, so the concept of location and scale can be used quite generally. Not only the mean is a measure of location, and the standard deviation is not the only measure of scale. For each class of distribution fitting to the concept of location-scale there is an optimum estimator, for instance for the Cauchy distribution the median is stable, but the sample mean would not even converge!

Linear circuits like amplifiers perform the same linear transformation (often unfortunately we often do not know the parameters). So if we look to linear measures in an amplifier (like voltage and current, but not power or level in dB), also the output measures will be normally distributed, just scaled, and shifted. A diode characteristic is often an almost exponential function that would lead to lognormal data. Leakage currents often follow this kind of distribution.

**Normal or Gaussian?** Should we call the discussed famous distribution the “normal” distribution or “Gaussian” distribution? Normal fits a bit better because also several generalizations of the “normal Gaussian”

distribution are called Gaussian as well! In opposite to the normal Gaussian distributions, these feature more parameters, so they are more “flexible,” e.g., can also be tweaked to fit for asymmetric or long tail data. The simplest and most important “generalized” Gaussian is the Student’s t distribution (so it’s not called according to Carl Friedrich Gauss, but another famous statistician who has published a work on it under the pseudonym “Student”).

### 3.4 Expected Values

One major outcome from a MC analysis is getting sample values—usually displayed in a histogram—for the different circuit performances. These samples, either a single MC result or the whole collection, depend on chance, but what the user wants to know are the real distribution parameters behind them. Besides the distribution parameters itself (like mean  $\mu$  and sigma  $\sigma$  for a normal Gaussian distribution), also other characteristics are very essential and can be accurately calculated if we know the pdf analytically. This is often not the case unfortunately, so we aim for a statistical estimate, which is an estimate of a property of a distribution, calculated from given samples from the distribution. It is quite important to realize that the parameter  $\mu$  is not something dependent on chance, but a sample estimate like the sample mean  $m$  depends on chance, as the whole data set depends on chance.

Let us start with an example: If we roll dices, we are often interested in the expected value  $E$  (or mean value or average value), when betting on dices. It can be easily calculated; the pdf is discrete and we assume a fair dice with  $\text{pdf}(i) = 1/6$ .

$$E(X) = 1 \cdot 1/6 + 2 \cdot 1/6 + 3 \cdot 1/6 + 4 \cdot 1/6 + 5 \cdot 1/6 + 6 \cdot 1/6 = 3.5 \quad (3.4)$$

We can easily generalize this example to make it applicable for other cases:

$$\begin{aligned} \text{Expected value } E(X) &= x_1 P_1 + x_2 P_2 + \dots \text{ or } \int x \cdot p \cdot dx \\ \text{Remember: } \int p \cdot dx &= 1 \end{aligned} \quad (3.5)$$

$$\text{Mean } \mu = \int_{-\infty}^{\infty} x p \, dx = E(X) \quad (3.6)$$

Lookup: The mean can be calculated for most random distributions in general. The same name  $\mu$  is often also used for the location parameter for a normal distribution, but there is a function parameter. Also for lognormal distributions

we often use a parameter named  $\mu$ , but in this case it is not identical to the expected value!

Another measure of location is the median or the mid-point, and a measure for the width of a distribution is e.g., the spread or the standard variation  $\sigma$ .

In the general case, the expected value is an integral, and we can express other important definitions by using integrals or expected values:

$$\text{Variance } V = E([X - \mu]^2) =: \int_{-\infty}^{\infty} (X - \mu)^2 \cdot p \cdot dx \quad (3.7)$$

$$\text{Standard deviation } \sigma = \sqrt{V} \quad (3.8)$$

$$\text{Yield } Y = E(\delta(x)) = \int \delta(x) p dx \quad \text{with } \delta(x) = 1 \text{ if circuit pass else } 0 \quad (3.9)$$

As mentioned, mathematically the overall yield is given as the full parameter space volume integral over the product of the indicator function  $\delta$  and the joint pdf (note: in almost all real cases we have more than one statistical variable, so the pdf is a function of multiple variables  $\mathbf{x}$ ). For independent random variables the joint pdf is given as the product of the individual pdfs. However, taking correlations into account is actually also easy, because you can usually *decompose* the overall distribution into independent “principal” variables (using so-called principal component analysis PCA), and in fact this is often done in the model files anyway.

The indicator function gives a 1 in the pass (or acceptability) region (the region where all performances are in-spec) and 0 in the fail regions. As the indicator function is 0 in the fail regions, we can alternatively also calculate the yield as volume integral over the joint pdf only over the pass region.

All these measures like mean, standard variation, yield, etc. rely on the true distribution pdf (and their integrals), but as circuit designers we usually do not know the pdf of our outputs and usually we cannot accurately integrate (only finite sums)! Actually, all the formulas look similar, and we can indeed use the same methods for integration, but not all methods converge equally well and a method may work fast and accurate on the mean and variance, but not on the yield. The reason is simply that the pdf is often a smooth and easy to integrate function; and this is often also true for many circuit performances like offset voltage. However, the indicator function needed for yield analysis is nonsmooth, so we can expect more difficulties. So especially for yield and high-yield analysis many special techniques have been developed, whereas for getting the mean or variance just Monte-Carlo integration is often good enough (although not perfect, regarding speed).

### 3.5 Estimates, Bias Error, and Confidence Intervals

Remember: Usually the real measures of the circuit performances (pdf, mean, variance, etc.) are not available analytically, we can only estimate them from our actual MC result. This means that any estimate (e.g., the mean of the MC data) depends on chance! The circuit is actually doing a mapping from the (element) parameter space (often containing thousands of variables) to the performance space (often a dozen).

Some important estimators for the measures we discussed in the previous chapter are:

$$\text{Sample yield} = n_{\text{pass}}/n \quad (3.10)$$

$$\text{Sample mean } m = 1/n \sum x_i \quad (3.11)$$

$$\text{Sample variance } V = 1/(n-1) \sum (x_i - m)^2 \quad (3.12)$$

$$\text{Sample standard deviation } \sigma = \sqrt{V} \quad (3.13)$$

$$\text{Sample median (50\% point): } \text{cdf}_{\text{emp}}(p_{50}) = 0.5 \quad (3.14)$$

Estimates are not the same as the true distribution values or expected values, actually even different names should be used, like mean  $\mu$  vs. sample mean  $m$ , but often this is not done due to laziness, unfortunately. Often the laziness comes with small risks only, because  $s$  and  $\sigma$  might differ by only 5%, but in other cases, like yield analysis, the differences can be much larger.

The big question is: If mean and sample mean are not the same, how much can we trust such so-called statistical estimates? Actually, we can even use different estimators for the inference on the mean  $\mu$ , e.g., for a Gaussian distribution the mean and the median are identical, so should we use the sample mean or the sample median for inference on parameter  $\mu$ ?

As an engineer you know it is often not good enough to have only a point estimate, you also need an error estimation. Usually there are two kinds of errors:

1. Uncertainty due to statistical variance

- Reduce it as much as you want by increasing the number of MC samples

2. Systematic errors (bias)

- $1/n \sum (x_i - \mu)^2$  is also converging to variance but has finite-sample bias
- Outliers impact the mean much more than the median

Systematical errors often cannot be reduced so easily by just increasing the MC count. However, if you can indeed run a huge MC analysis you can estimate the error in yield estimation quite well (because the sample yield has no bias error). We can also do many MC analyses and look to the variations in the estimates from one MC analysis to the other. By running many thousands of MC analyses, we can “easily” find out in which interval 90% of the estimates are in, but unfortunately this is very time-consuming. Indeed such *statistical* variations can be treated by so-called confidence intervals; in many cases you can calculate confidence intervals giving a lower and upper confidence bound  $CI = [LCB, UCB]$  also without doing such huge repeated MC analysis. However, the user must be aware of the fact that also confidence intervals are derived from the available MC data depending on chance, which means that also confidence intervals depend on chance and are nothing else than estimates [Hoekstra]! In addition, you almost never get 100% confidence that the true measure (yield, mean, standard deviation, etc.) is in a certain range. Such statistical uncertainties lead to the need of a kind of statistical design margin, e.g., even if your sample yield and your yield target is 99%, you still should not fully trust it! What you can trust (more) is the lower confidence bound LCB, which might be only 97%. So actually you need some amount of *over-design*, because only if your design is a bit better giving 99.7% then the lower confidence bound (LCB) might be indeed equal or above your 99% target. So in this case we actually work with 0.7% over-design; how large this statistical over-design margin is depends on the used estimator (like sample yield,  $C_{PK}$ , etc.) and the number of MC points. Later, in Chapter 5 when discussing worst-case distances WCD, we will also learn about statistical methods *without* such sampling error—so in theory without need for statistical design margins and so potentially less or even zero over-design.

There are also many other aspects in our inference, like: Can we guarantee that for an almost infinite number of MC points the error will really approach zero, or will there be a remaining bias error? How much will the calculation be impacted by outliers? In many cases the mean is more efficient than the median, but the median is far more robust against outliers!

Truly these aspects are important for EDA software implementations and need careful tweaking. There is simply no best estimator regarding efficiency, bias, robustness, and calculation effort. Only for a certain class of distributions some algorithms may outperform others, but usually you can always provide counter example cases, so only quite complex algorithms are flexible enough to deal with difficult real-world problems. In effect the progress in EDA tools is often in such details, not directly observable by the user.

A MC key problem is that the variance in the estimates is often quite large. More advanced techniques, beyond MC, are typically much better in this aspect, but they may come with more assumptions and if these assumptions are not valid such advanced methods often introduce a significant bias error! For this reason designers using such advanced methods should be clearly aware of which underlying assumptions have been taken and if that is compatible to the design under investigation and the wishes on accuracy.

**How to measure “speed-up” and “design efficiency”?** EDA vendors are often asked how large efficiency improvements are in a new software version or by using a new feature. In math this is interestingly by far not so easy to tell compared to the use of a faster compute server. “Speed-up” sounds always good, but is sometimes risky! And the risk is sometimes hard or impossible to quantify. In the statistical sense speed-up often means “variance reduction”. Lowering the standard deviation by  $2\times$  usually translates to the option to use  $4\times$  less simulation points, but only if the variance reduction can be achieved *without* adding a systematic (bias) error. In addition, one assumption often used is that we have the “normal”  $1/\sqrt{n}$  relationship, which is also not always the case.

Sometimes variance reduction methods work straight forward: If you measure something in lab, you hope that your single measurement value  $x_1$  is close to the real value. Of course doing the measurement again can lead to another value, and e.g., always taking the last value is not a too bad approach, but if noise is present we can expect quite significant variations still. To get a more stable estimate we could take the *average* value; and another approach would be to ignore all extreme values, so taking the *median* value. However, not all cases are simple like this, and not always it is so easy to see (or even calculate) the gain in accuracy, to check for prerequisites, and to clarify the advantages and disadvantages.

In circuit design you can do even more than a clever data *analysis*, e.g., you can inspect the statistical model parameters, you can create clever testbenches and do hand calculations on offsets; or we can run not only random sets for the setting of statistical variables, but set them in a systematic way. Some methods are based on sampling and with confidence intervals, we can tell about accuracy, but we cannot easily quantify if an assumption like “data is Gaussian” is valid! Also we sometimes need to compare statistical and non-statistical methods, and hard error limits like  $\epsilon < \epsilon_{max}$  should be treated differently than statements about variance, and the choice of test cases has a big impact on statements about speed-up too.



If one estimate has  $1/\sqrt{n}$  convergence and second one has  $[\log(n)]^s/n$ , then the speed-up of the second one might be impressive, but actually it depends on which settings of  $n$  (e.g., representing number of simulations) and  $s$  (e.g., number of statistical parameters) is regarded as meaningful. Of course, we hope that we pick realistic values, like designers can effort running  $n = 500$  points, but seldom 500M ones. Also the type of circuits can vary a lot (having  $s$  ranging e.g., from 1 to 30 or more). In addition, during the design tweaking phase we may take more risks than for sign-off.

Last not least, sometimes the speed-up is a bit theoretical, e.g., maybe you simply do not really need to know the sigma of an offset voltage with 0.5% accuracy which may require indeed 10,000 simulations using an old standard method; or a new method needs 10 times less simulation points, but it cannot run all these in parallel like the other old-fashioned “slow” method.

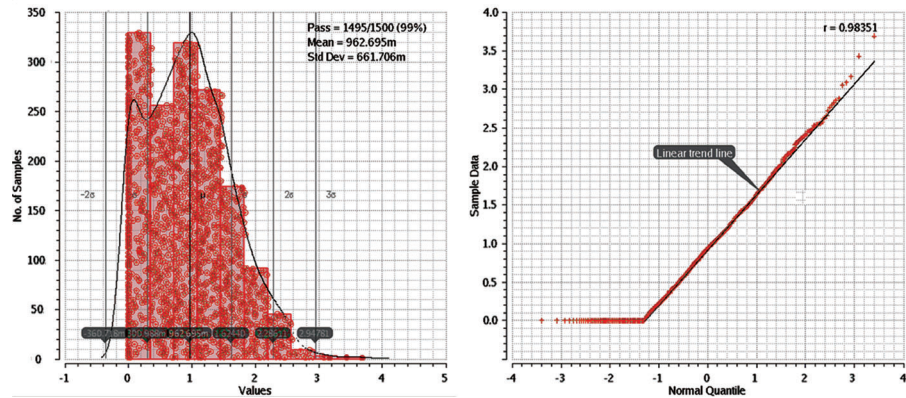
### 3.6 Basic Data Analysis for Normal Gaussian Data

If the data is normally distributed you can make quite easily a detailed data analysis, using many school book techniques, but how to check for normality? The easiest way is an eye inspection of the histogram, which provides a picture approximating the probability density pdf of the performance data. Here the problem arises on how many bins you should set: More bins leads to more noise, but too few bins can make an inspection also difficult. Most difficult is to decide whether the data follows a normal distribution also regarding tail shape, because here you have often only few data points and also it is not easy to decide if a certain curvature is normal Gaussian (i.e., according to  $e^{-x^2}$ ) or not. So looking to many histogram examples is a good training, but we can do even better.

Indeed, the so-called normal quantile plot solves the problem of curvature inspection, because it shows a kind of transformed cdf plot (actually the x-axis is the inverse normal cdf, also named z-score, and the y-axis is the sorted data).

For normal distributions the data should fit to a straight line in the (normal) quantile plot. Also an interpretation for nonstraight lines is quite easy. For instance if there is a clear lower limit the quantile plot will start horizontally (see Figure 3.6).

If data is long-tailed, the quantile plot gets shaped like arctan, for short tails it would look like hyperbolic tangent (like  $I_C(V_{in})$  of a differential pair),



**Figure 3.6** (a) Histogram and (b) normal quantile plot from an op-amp – tail region marked in yellow.

for asymmetric (skewed) data we get a kind of J or reversed J shape. For more details, look at Figure 3.7; it also shows the relation to the kurtosis  $k$  (normalized central 4th order moment of the distribution, more in Chapter 4 when we discuss non-normal data).

The “trick” is usually how to know if a deviation in the quantile plot is just a random effect or really a systematic non-normality. For this reason

	All but a few points fall on a line	Outliers in the data
	Left end of the pattern is below the line while the right end of the pattern is above the line	Symmetric, long tails at both ends Typically $k > 3$
	Left end of the pattern is above the line while the right end of the pattern is below the line	Symmetric, short tails at both ends Typically $k < 3$
	Curved pattern with slope increasing from left to right	Skewed to right Typically $\text{skew} > 0$
	Curved pattern with slope decreasing from left to right	Skewed to left Typically $\text{skew} < 0$



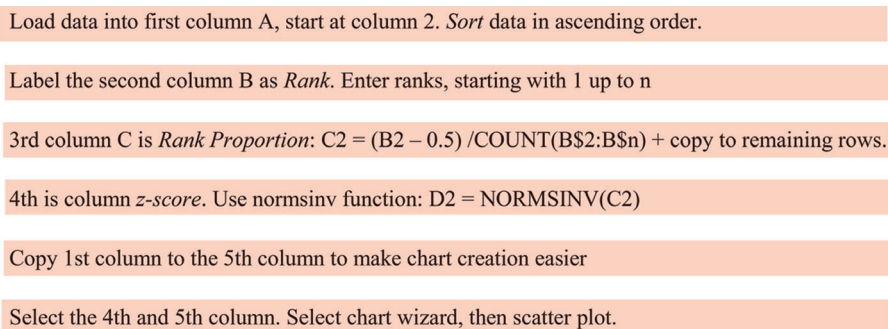
**Figure 3.7** How to interpret normal quantile plots.

some tools (like RealTime MC) add (sample yield) confidence intervals to the quantile plot, but if the non-normality is only mild, then a decision is usually still difficult. In particular, in the tail regions, a decision is often tough to make because there we have usually only a few samples, so *any* statistic will not be very stable. In Chapter 4 we will discuss non-normal distributions in more detail, and we also demonstrate *numerical* tests for normality (Figure 3.8).

If we are sure that the data follows a normal distribution, then we can calculate also confidence intervals (CI) quite easily, because the normal cdf and pdf is analytically tractable. Due to this, like we can calculate the sample mean directly from the data, we can also calculate CI from the data. Note that we calculate the CI for the sample estimate like sample mean  $m$ , not for the fix (but often unknown) distribution parameter  $\mu$ ! Like the sample estimate also the CI depend on chance! We can only expect that in average it will correct, according to its confidence level. If our assumption on normality is violated, then also the CI calculation will get wrong. In such cases confidence intervals from a normal approximation can be at best approximated CI and usually better methods exist (like bootstrap, Chapter 6).

Another method to get a confidence interval would be just doing our MC analysis again and again. For instance to calculate the CI on sample mean  $m$ , we can do a MC analysis very often, with different seeds but with the same count  $n$ . And we will observe that also the sample mean  $m$  will be approximately normally distributed.

In this case the standard error (SE) as the standard deviation of the sampling distribution of a statistic (most commonly of the mean) is given by  $SE = s/\sqrt{n}$  (with standard deviation  $s$ ). And the confidence interval for a confidence level



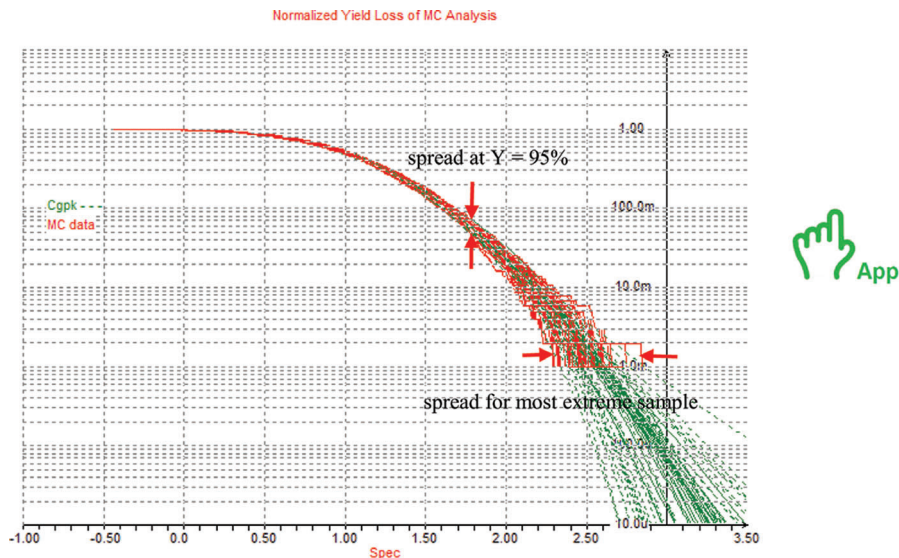
**Figure 3.8** Excel steps to show normal quantile plot.

of 95% will be around the mean with  $\pm 1.96SE = \pm 1.96s/\sqrt{n}$ ; so  $\pm 2s/\sqrt{n}$  is a good rule of thumb for the 95% confidence interval (for the sample mean of a normal distribution)—compare this to Figure 1.11.

Note ☺: Often we are only interested in single-sided specs or single-sided CIs, so only the upper “outliers” degrade the yield, so the risk for being out of desired range like  $\geq 2\sigma$  is actually only app. 2.5% not 5%. If you are already happy with lower confidence then the CI becomes narrower, but you take more risk, so usual confidence levels range from 75% to 99%. The higher the confidence level, the lower the risk making a wrong decision, but actually already small deviation in the model can lead to severe errors also in the confidence intervals!

A detailed analysis has been done by William Sealy Gosset on normal distributions. He derived that the confidence interval on the sample mean  $m$  is related to the Student’s  $t$  distribution, and the factor 2 in our formula is slightly a function of the sample count  $n$ . In older days engineers used lookup tables, now almost all statistical tools and EDA software directly provide the user the confidence intervals based on Student’s  $t$  distribution; “Student” was the pseudonym which Mr. Gosset used for his article.

Also the CI for the standard deviation can be calculated analytically, and again we could double-check it by repeated MC (like in Figure 3.9).



**Figure 3.9** 50 MC results on the sample yield ( $\log(1 - Y)$  in red) for a Gaussian distribution.

The CI for  $s$  is related to the  $\chi^2$  distribution. As a rule of thumb, for  $n = 200$  the 95% CI on  $s$  is app.  $\pm 10\%$ . Actually the distribution is slightly asymmetric, so the real value is  $+11/-9\%$ . This is no surprise, because upper “outliers” are more likely than lower ones (there will be no negative values).

Note: You can find an incredible number of papers on confidence intervals CI, but most of them are related to the mean. However, circuit designers are usually most interested in the standard deviation, like for offset voltage, or in the yield. CIs on these are a bit more difficult, but usually also available in design environments. In particular, the tail region of a distribution is of high interest and here different CI methods can really give different results. In tail regions like beyond 99% any statistic will rely on quite few samples, so CI become wide, maybe too wide to make design decisions! This is a major motivation for the advanced methods in Chapters 6 and 7.

**Functions and Distributions.** Most people think of the probability *density* function pdf when talking about distributions like the normal or uniform ones. This is just because the pdf looks like the most important graph, the histogram, giving the *frequency* of occurrence. The histogram is also good for identifying distributions by eye inspection. However, we have seen that for check on normality, for yield estimation or confidence intervals also other functions matter. Maybe go through this chapter again, and look up what function is for what! Often indeed the *cumulated* distribution function cdf (the integral of the pdf) is even more helpful, because it is directly related to yield and to the probability that a variable  $X$  falls into a certain interval  $[a, b]$ . The reason why the cdf is not so famous is just that it often *looks* not so characteristic, because the integration e.g., smooth out edges, and the point of highest density is much harder to see. Also the *inverse* cdf (the percentile function) matters. For instance, the factor “2” used in our approximation for the 95% confidence interval is coming from the Student-t distribution, but not its pdf but from the inverse cdf. Actually knowing this (and not much more) is already extremely helpful when doing statistics.

### 3.6.1 The Yield Estimation Problem

We discussed basic estimates and confidence intervals. And one outcome was that for simple estimates like sample mean or standard deviation we usually do *not* need many MC points—often 200 points are enough to decide if offset

voltage is small enough, but how many points are needed to verify the *yield* with a certain significance?

Using the sample yield the accuracy (e.g.,  $Y$  confidence interval) depends on  $Y$  itself and on the sample count  $n$ . For  $\pm 1\%$  absolute accuracy  $\Delta Y$  and a yield of 50% app. 11000 points are needed (confidence level at 95%—quite a typical value), and at 98% yield we need roughly 800 points. Unfortunately also 800 simulations is not that low, and 1% absolute error in yield is not that accurate, because it matters much if your loss is 1% or 3%, or even 0.1% vs. 2.1%! Actually looking to the yield loss or to the error in terms of sigma yield estimation is generally more stable in the distribution center than for tail regions.

Focusing on the *relative* yield loss error we would need to look at  $\log(1 - Y)$ , and Figure 3.9 is showing this for a repeated MC run. Looking to the spread in y-direction gives quite a native feeling on how “instable” MC results can be (a constant  $\Delta y$  in this plot is related to a certain fix relative error, due to the log y-axis).

Note: In this plot the spec is set to 3.0 giving a true yield of  $4\sigma$ . Therefore, we almost never get a fail within  $n = 1024$  MC points, so at some point we reach  $Y = 1.0$  and  $\log(1 - Y)$  becomes infinite (the red curve plot stops there). The green curve is an extrapolation, and in the extrapolation region the variations are even larger.

If your circuit is well designed, then often a short MC run shows no fails, so the sample yield becomes 100%, but of course this is typically too optimistic. To be on the safe side you may ask again for a confidence interval. The yield confidence interval problem has been solved by Clopper-Pearson, resulting in a quite complex formula using the beta function, but if you have no fails, then already the “Rule of Three” is a very good approximation for the 95% CI:

$$CI(Y) = [1 - 3/n, 1.0] \quad (3.15)$$

This way we can also easily calculate how many points the MC run should include till we can decide with 95% confidence if the design fulfills a certain desired yield:

$$n \geq 3/(1 - Y_{\text{target}}) \quad (3.16)$$

Already this simple formula demonstrates very well our verification problems, because it will lead to the fact that high yield verification needs a lot of MC points, usually much more than for obtaining stable values for sample mean and standard deviation of the performance values! Look at Table 3.3 for more

**Table 3.3** Verification with sample yield according to the rule of three

Yield in Sigma	True $C_{PK}$	Single-sided Yield Loss	Number of MC Points (if No Fails)	Comment
0 sigma	0	50%	4	is of low interest
1 sigma	0.33	15.9%	17	is of low interest
2 sigma	0.67	2.3%	130	the minimum realistic yield target
3 sigma	1	0.14%	2200	often used as target
4 sigma	1.33	0.003%	95K	often the limit for pure MC sample yield
5 sigma	1.67	290 ppb	10M	typical for blocks in high-volume chips
6 sigma	2	1 ppb	3G	typical for memory

details, it includes also the process capability index  $C_{PK}$  which has a strong connection to the “yield in sigma” (see next Section 3.6.2). Also note that there is no simple reciprocal relation between yield loss (failure rate) and sigma; it is a special nonlinear relation you have to be aware of. For instance from  $2\sigma$  to  $3\sigma$  we have roughly to divide by 20 to treat the loss, but from  $5\sigma$  to  $6\sigma$  it is already 290 (look also at Table 1.3).

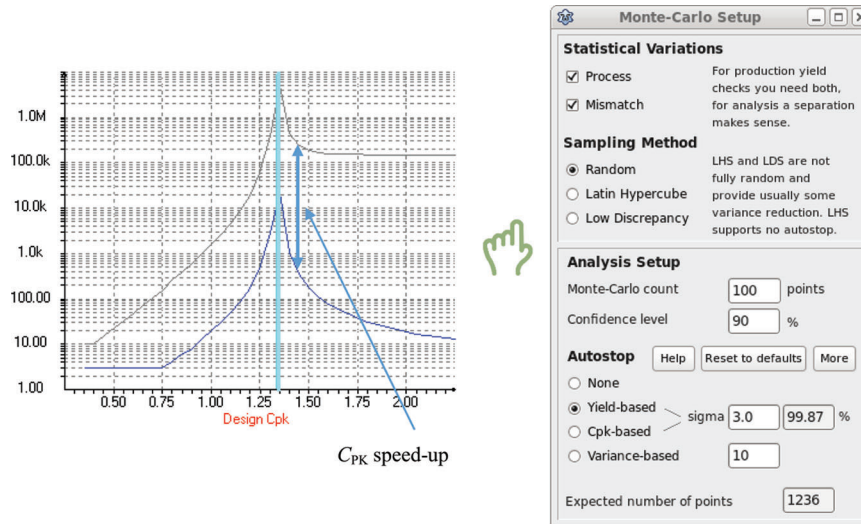
A distribution with a constant failure rate is the exponential distribution, playing a key role in radioactivity. One distribution with such tail behavior but a “Gaussian center” is the so-called logistic distribution; if you want a power law tail instead, you would end up in the Student’s  $t$  distribution. Actually many distributions exist, and all have their meaning and application.

Already these numbers will typically lead to long simulation runs, but if you have indeed failed samples the lower yield confidence limit will be even lower and we need even more MC samples (see Figure 3.10, some more details and further screenshots can be found at [Iastate]), and the convergence rate will be  $1/\sqrt{n}$  – not  $1/n$  as the simple rule of 3 may suggest!

Note⊗: There are many confidence interval approximations in the mathematical literature. Look up that the interval from a normal approximation can be very bad [Schmid], because the sample yield distribution can be highly non-normal! Do not use it, better use Clopper-Pearson or Agresti-Coull (both are slightly on the pessimistic side compared to the “Rule of Three”).

If the design is perfect (like  $>6\sigma$ , giving almost no fails) and we want to ensure just  $3\sigma$  only we will need typically app. 2000 points, but if the design is truly only  $3.15\sigma$  we need app. 16,000 points to make an accurate enough decision based on counting failed samples. If we use the  $C_{PK}$  (next chapter) we would need only approximately 1100 points. However, if we allow no





**Figure 3.10** Required random MC count to verify a  $4\sigma$  design based on sample yield vs.  $C_{PK}$  [Iastate].

design margin both sample yield and  $C_{PK}$  would require an infinite number of points. So we have an over-design vs. speed trade-off.

**The Binomial Distribution.** Looking to yield means dealing with pass and fail only, i.e., doing investigations on a binomial distribution. Its “accurate” confidence interval has been first calculated by C.J. Clopper and E.S. Pearson, in 1934. Although many laws exist which give the normal Gaussian distribution a strong preference, especially for a large number of samples, it is by far not the only important distribution. If we use the normal approximation to the binomial distribution for the confidence interval, we get the simpler so-called Wald interval. It is easier to calculate and often used, but unfortunately it can be far too optimistic. For instance, having a  $C_{PK}$  of 1.5 and a huge set of 100,000 MC points the CI from the normal approximation would be still 3× too optimistic on yield loss, for IC design better forget the Wald interval (even the “Rule of 3” is better in this case).

The reason for the large number of points at high yields is that such yield analysis based on outer samples, the tail samples—and these are rare; any statistics based on them will have quite large variations and will be quite



unstable. When looking just to  $3\sigma$  verification, one may still accept to run 2,000 to 16,000 samples at least for fast-running testbenches, but during the design phase you typically need many such runs, and remember already for a failure rate of 1000 ppb (0.0001% or  $4.75\sigma$ ) the numbers get huge: we need 1,000,000 samples as a real minimum giving us hope just only to observe a fail (no confidence interval included!); having no fails (so the design should be even much better than  $4.75\sigma$ ) the 95% confidence limit would dictate us 3,500,000 samples, and for a design margin like  $0.33\sigma$  (roughly  $5\times$  less loss) we would typically need 6,000,000 MC points, and for less over-design even more!

There are many known attempts to solve this general yield verification problem. One is to go back to the original yield definition and solving the yield integral by other means than MC. For instance, a numerical integration by Riemann's sum would have  $1/n$  convergence rate if we would only have one statistical variable, and Simpsons's rule would be even faster! However, both methods will slow down in higher dimensions – and may become even slower than MC. On the other hand, we will also demonstrate methods which require theoretically even no such “design margin,” so coming in theory with almost no need for over-design.

### 3.6.2 Sample Yield vs. $C_{PK}$

If we can assume a normal distribution, we can solve the problem of verifying the partial yield also in another way (Figure 3.11). We can calculate a much less quantized estimation by creating a Gaussian fit to the data and we can calculate a yield estimate from this fit by using the cdf of the normal distribution, which is related to the error function. The process capability index method is exactly doing this, and the big advantage of the  $C_{PK}$  is that we can even get a realistic yield estimate (so below 100%) if there are no fail samples! This way we can obtain a yield estimate with tighter confidence interval, but the user should be aware of potential systematic errors.

Note: To get the total yield from the different  $C_{PK}$ s for each spec we need multi-variate techniques and correlations. This will be discussed in Chapter 5.

The  $C_{PK}$  is given as:

$$\begin{aligned} C_{PK} &= |USL - \mu|/3\sigma \quad (\text{for single-sided upper spec limit}) \\ C_{PK} &= |\mu - LSL|/3\sigma \quad (\text{for single-sided lower spec limit}) \\ C_{PK} &= \min(|\mu - LSL|/3\sigma, |USL - \mu|/3\sigma) \quad (\text{for double-sided specs}) \end{aligned} \quad (3.17)$$

Check if data is close enough to a normal distribution

Approximate data with a normal Gaussian distribution  
For this estimate the parameters via sample mean and sample standard deviation

Calculate the  $C_{PK}$  value according to spec type and value

Calculate  $Y(C_{PK}) = \frac{1}{2} \cdot (1 + \text{erf}(3C_{PK} / \sqrt{2}))$

**Figure 3.11**  $C_{PK}$  flow (prerequisite is of course a stable process).

Note: From the mathematical view point the use of the min function is a horrible approach, because there would be no sensitivity to the nondominating spec-side till we reach the balance. A much better approach—also in conjunction with optimization—is to calculate two  $C_{PK}$ s for both spec-sides and then calculating back to yield for both. To combine them into one we just have to add the yield loss and can again transform back from yield to  $C_{PK}$  via inverse error function! Later we will have a similar problem for the worst-case distance (WCD) approach, and we can solve it similarly.

The  $C_{PK}$  formula is actually a kind of normalized spec margin or performance margin approach. The normalization is done in terms of sigma of the output distribution, this sigma and the “yield in sigma” are only the same if we really have a normal Gaussian output (performance) distribution, which often cannot be assured so easily. The whole approach is a continuous one, whereas the sample yield is more a yes/no or 1-bit ADC approach. From circuit design you know 1-bit ADCs have higher quantization noise but no nonlinearity, whereas multi-bit ADC have much better SNR, so the  $C_{PK}$  method is more similar to analog or multi-bit ADC style! Also for optimization and debugging avoid binary specs; it is simple waste of information! Sometimes it might be indeed “native” to use a spec like “power-up circuit OK”, but in such case better look to start-up time directly and create an “analog” spec!

The  $C_{PK}$  measures the relative performance variation (e.g., due to mismatch and process variations); actually by putting the distance of spec limit vs. mean in relation to the standard deviation, for double-sided specs it also takes the

spec-centering into account. One practical advantage in using the  $C_{PK}$  instead of yield values is that the  $C_{PK}$  values are more manageable, e.g.,  $C_{PK} = 1$  means 0.135% loss and is equivalent to a spec distance of  $3\sigma$ . A  $C_{PK}$  of 2 means 1ppb loss, which is already a very small value. Usually you require at least a  $C_{PK}$  beyond unity.

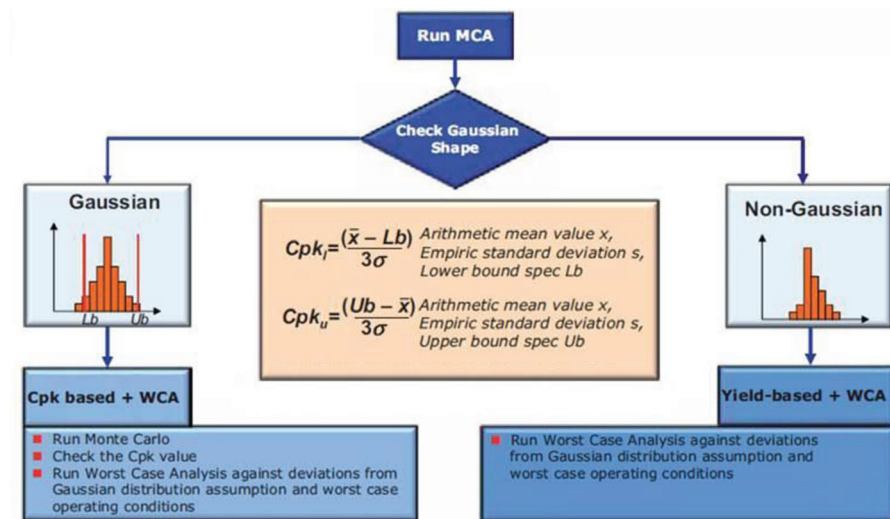
As the  $C_{PK}$  depends on  $\mu$  and  $\sigma$ , we can also easily calculate a  $C_{PK}$  variance  $V = \sigma^2$  and the  $C_{PK}$  confidence interval ( $\pm 2\sigma$  for 95% confidence).

$$\sigma C_{PK}^2 1/9n + C_{PK}^2/2n \quad (3.18)$$

e.g.,  $\sigma C_{PK} \cong 5\%$  at  $n = 250$  &  $C_{PK} = 1$

Even for  $C_{PK} = 2$  the variance of the  $C_{PK}$  ( $\sigma C_{PK}$  in Equation (3.18)) is quite small for already moderate MC counts. This means that using the  $C_{PK}$  we only need MC counts in the order of hundreds, even for high-yield verification! Whereas yield estimation by sample yield  $Y = n_{\text{pass}}/n$  would require often billions of points (Figure 3.12).

One may wonder whether the  $C_{PK}$  method—equivalent to a Gaussian fit—is the “best” method. In fact, it really is, in some way, but only if it is really sure that the data comes from a normal Gaussian distribution! In this case also the method for determining the two parameters, mean and sigma,



**Figure 3.12** Classical split flow using  $C_{PK}$  for normal data and sample yield as backup (in addition we can use most advanced methods described in following chapters) (courtesy: MunEDA).

by the well-known estimates sample mean and sample standard deviation is optimum, e.g., more efficient than the use of the median instead of the mean. The underlying theory is fundamentally given by the concept of maximum likelihood (ML). In ML estimation (MLE) we determine the parameters so that the probability to get the given data is maximized. At this point we recommend looking to dedicated mathematical literature, and we also want to mention that although the concept of maximum likelihood sounds so general, there are still some cases in which it should be complemented with further techniques. For instance, ML parameter estimation is often quite sensitive to outliers and often also not the easiest method (e.g., it is hard to apply for a tri-angular or a Cauchy distribution).

**The Moment Method.** To categorize distributions we can e.g., look to the pdf or to the quantile plot, but we can also do it based on the so-called distribution moments. Look at the equations for mean and variance; these are the first and second moment of the distribution. We can simply extend the idea by using higher exponents. The 3rd moment is called skew  $s$  and measures the asymmetry, and the 4th order moment is the kurtosis  $k$ , measuring the tail behavior (to some degree). Usually the moment are taken around the mean (central moments), also the higher moments are usually normalized to the standard deviation. This avoids getting too extreme numbers, and makes the “shape” measurement independent from the distribution scale. For instance a Gaussian distribution has a kurtosis of 3.0 and zero skew. In the past, moment fitting was the most commonly chosen method for data fitting, but MLE is even more general (e.g., it can be applied even if the higher moments are infinite, like for the Cauchy distribution).

Table 3.4 lists also other methods for yield estimation, as a non-normal distribution can be non-normal in many different ways there is almost no single best method. Also the result interpretation is usually more difficult than for the normal case: For the  $C_{PK}$  we have to deal with the two distribution parameters for location and scale plus the spec limit; for non-normal cases things become more complicated.

### 3.6.3 Confidence Interval-Based Autostop for MC

The user is interested in MC results having a certain minimum accuracy, which is related to the width of the confidence intervals. As confidence interval

**Table 3.4** MC yield estimation techniques

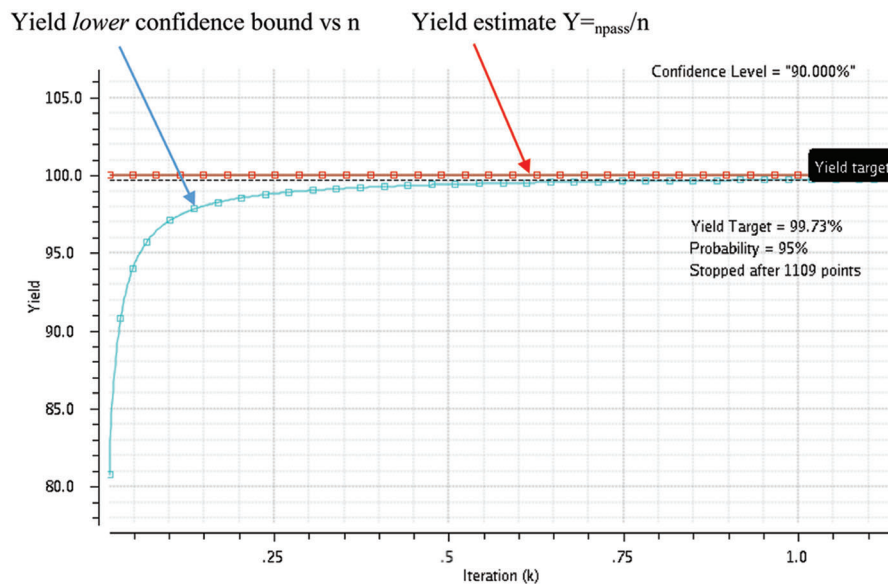
Name	Method	Limitations	Comment
Sample yield	Nonparametric yield estimation	Large variance, so wide CI	Standard method in design environments, no bias error
$C_{PK}$	Gaussian fit	Only accurate for normal distributions	Standard in QA
Kernel density estimation (KDE)	Kernel density fit (almost nonparametric)	Limited extrapolation capabilities, so hardly suited for high-yield estimation	Difficult setting for smoothing bandwidth, available in math packages
Multi-parameter fit	e.g., [Lange] using generalized lambda distributions	Bad fit e.g., for multimodal distributions or for long tails with cuts	Available in advanced design environments
Nonparametric fit plus tail modeling	e.g., KDE and Pareto fit to tail	Limited accuracy for Gaussian distributions or for long tails with cuts	No easy interpretation of parameters,
Generalized $C_{PK}$	See Chapter 4 and [Weber]	Limited accuracy e.g., for long tails with cuts	Available in RealTime MC

calculation for the sample yield is quite simple, this has been exploited in many design environments to implement a kind of MC autostop feature. This could reduce the setup effort for the designer, e.g., he/she only has to set a certain minimum and maximum number of points, a certain yield target to be verified, and the simulator “decides” when to stop.

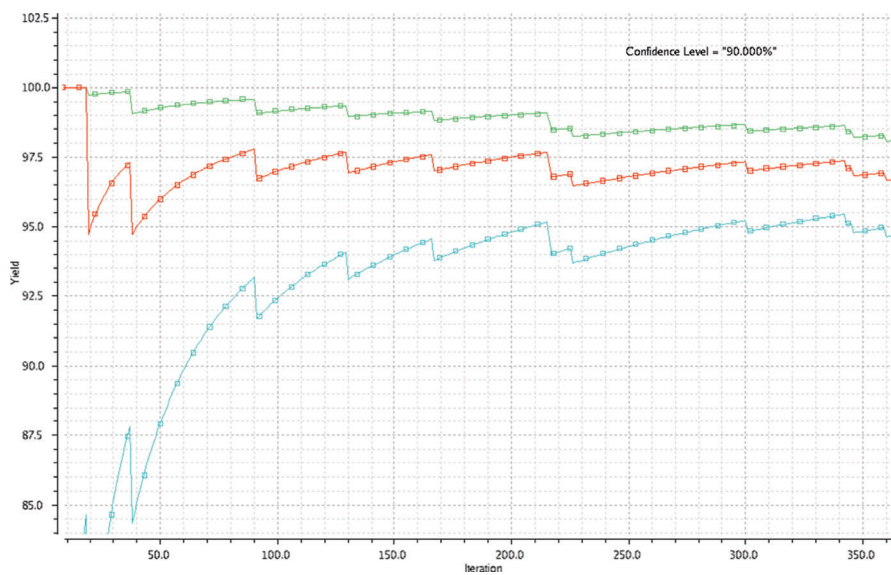
Figure 3.13 show the MC run of a very good design with no spec fails, so the yield and the lower CI limit always increases. This way we will cross the desired yield level at some count  $n$ .

A spec-fail would push down  $Y$  and CI limits (see Figure 3.14).

The position for spec fails of course depends on the random MC walk, so on MC seed value. In this case the design is too bad to achieve desired yield target; so even the upper yield confidence bound UCB is worse than the yield target! If your design is really bad, like giving a fail already in the first five MC points, the autostop could come very early. Here the 95% CI is approximately [0.08,0.90], so if your target yield is 90% or higher, the MC autostop would be triggered. Statistically this is correct and you would save a lot of time. On the other hand, it might be inconvenient, maybe the yield is low due to quite an uninteresting spec that is not fully confirmed or you are also interested



**Figure 3.13** Sample yield confidence limits for a MC run with no fails.

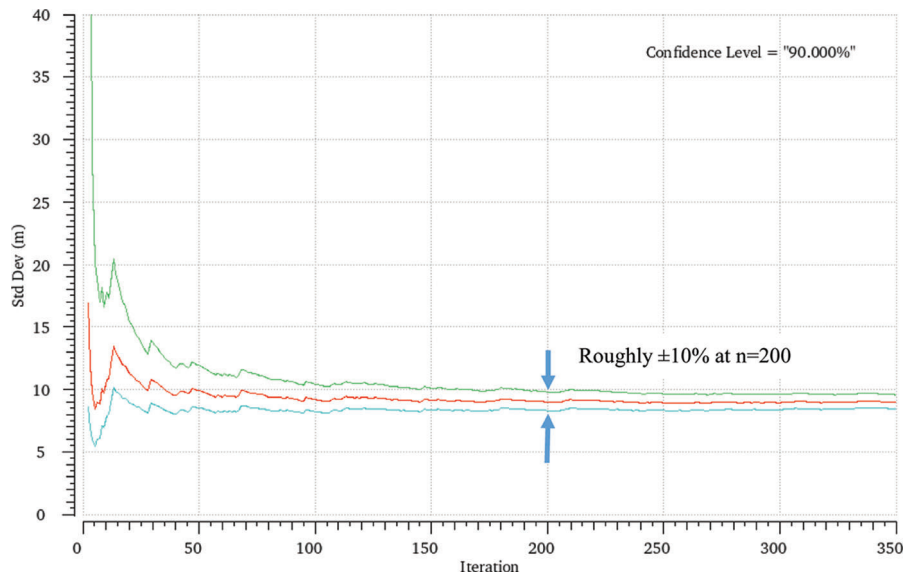


**Figure 3.14** Sample yield confidence limits for a MC run containing fails.

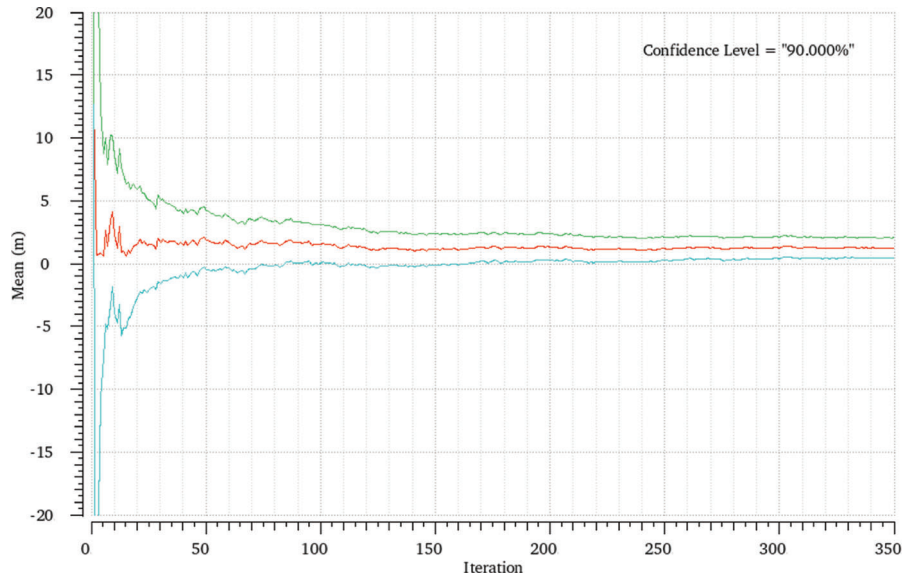
in looking to histograms, for which you should have many more points. It could also happen that the autostop comes very late, just because your target yield and your actual design yield are close together. In such cases, autostop does not help on speed but of course on accuracy. For these reasons, even when using autostop, the specification of a minimum and maximum count (or runtime) makes sense.

An autostop feature might be also implemented based on other confidence intervals, like on sample standard deviation  $s$  (Figure 3.15) or when reaching a certain accuracy level for a contribution analysis (Chapter 5). Usually also plots for checking how stable the mean estimation is are available, but often the MC mean  $m$  is of lower interest, because it is for Gaussian distributions close to the nominal simulation (Figure 3.16). As you know how stable  $m$  depends on  $\sigma$ .

CI width follows the  $\chi^2$  distribution and approximately a  $1/\sqrt{n}$  law. Also you can see that it is quite symmetric if it is tight enough (like for large  $n$ ). With larger confidence level (like 99%) the CI would be larger. A meaningful autostop criteria could be e.g., obtained from the relative error on the standard deviation, like  $|\text{UCB}(s) - \text{LCB}(s)|/s < 0.05$ . Note: Often CI calculations for mean and sigma are based on normal approximations, so you may add some safety margin to be prepared for non-normal distributions (see next chapter).



**Figure 3.15** Sample standard deviation confidence limits for typical MC run.



**Figure 3.16** Sample mean confidence limits for typical MC run (same circuit as in Figure 3.15).

Essentially all most advanced statistical analyses typically feature such autostop (although they may execute much more steps than a pure MC analysis). This way the user does not need to know in advance how many simulations are needed; instead he/she sets a certain accuracy level for a certain estimate, like sample yield or standard deviation, or maybe more advanced estimates like  $C_{PK}$  or generalized  $C_{PK}$  (see Chapter 4) or correlations (see Chapter 5).

**Testing in Quality Assurance.** In MC with autostop, we do basically one test and decide whether we should continue our analysis or not. In quality assurance this is a standard technique too, but often reality creates more problems. Imagine that e.g., the testing of chip is costly or even destructive. How should a company check if a delivery e.g., of 1,000,000 parts, is fine or not? For cost reasons, it makes sense to test only a small subset, and if e.g., all chosen samples are fine, we could extend the testing (more samples, more costly tests). This is called sequential testing, and in principle designers or EDA tools can do the same, e.g., going beyond simple MC autostop. For instance, it may make sense to exploit that some



tests in the MC setup run quickly (e.g., simple DC or AC simulations), but others take much more simulation time (transient noise analysis, load-pull analysis, etc.).

### 3.7 Questions and Answers

1. In PDK's normal distributions often have cuts for the process parameters, e.g., at  $5\sigma$  (so you will never get samples for these variables beyond  $\pm 5\sigma$ ). Can you still use the  $C_{PK}$  and sample yield?

*Yes, the sample yield works for all kinds of distributions anyway! The  $C_{PK}$  is slightly too pessimistic if your distribution is a normal distribution with cuts.*

2. Can MC handle correlations correctly?

*Yes, there is no problem at all on this. For some advanced non-MC statistical analyses correlations might cause problems, but not in random MC.*

3. How many points do you need to get a stable sample standard deviation and  $C_{PK}$ , like  $\pm 10\%$ ?

*There is no hard limit; it depends on confidence level. For near-normal data, you typically need only about 200 samples to make the CI  $\pm 10\%$  with 95% confidence. For very long-tail data the CI might be much wider!*



4. How many points do you need roughly to get in your MC result a sample that is as extreme as  $+3\sigma$  or  $-3\sigma$ ?

*There is again no hard limit for a uniform distribution you would never get a sample  $3\sigma$  off from the mean! For a normal distribution you may need 100 to 300 samples typically, it depends (pretty much) on chance. Unfortunately you need many more points for  $6\sigma$ !*



5. Imagine you get 1,000,000 result samples like the height of a good, so you can calculate mean and sigma, maybe to 1.70m and 0.1m, respectively. So we can also calculate the approximated 95% confidence interval to  $1.70\text{m} \pm 2 \cdot 0.1/\sqrt{1,000,000} = 1.70\text{m} \pm 0.0002\text{m}$ ! This means the CI is very tight! Imagine you ask 1,000,000 people about the height of the emperor of China, and you would get these numbers, do you believe you can get the height this way to an accuracy of  $0.0002\text{m} = 0.2\text{mm}$ ?

*Think a bit before you check out Google!*

6. How can I calculate the spec setting for a certain yield?

*Via  $C_{PK}$  you only have to solve the  $C_{PK}$  formula for the spec limit, so it is easy, but only correct if you really have a normal distribution. Via sample yield you can do so too, but the result is much more quantized and only acceptable for low yields, like 90% for a MC run with 100 points.*



7. Imagine you are in a certain city and you know all taxis are numbered from 1 to  $n$  with no gaps. You take a trip and from time to time you see a taxi. How can you estimate the total number of taxis from your observations?

*We can expect that all numbers appear with the same probability, so we do estimations on a discrete uniform distribution. One method is to calculate the mean value and multiply it by 2, but this is not the best way to do it. Do you have ideas for faster convergence? What about taking the maximum?*



8. Figure 3.10 shows an almost flat curve for the sample yield at high yield levels. Please look at it and explain why this makes sense!

*For high design yields we will get fails only with a very low probability, and it matters not much anymore if the design is  $5\sigma$  or  $6\sigma$  if you want to verify only  $3\sigma$ . For the  $C_{PK}$  this is not the case, because it really exploits the spec margin.*



9. We use the term spec or performance margin; and we use differences; why not ratios? For which type of distributions it would be better using ratios? *With ratios you run into problems with performances which can have both signs. For lognormal distributions using ratios would be optimums!*

10. Imagine you have an outlier in your almost Gaussian data and you are using the  $C_{PK}$  method for yield estimation? What happens if you have an upper spec only, but an outlier at the negative (non-spec) side?

*Indeed the mean and even more the sigma would be impacted, so a large sigma could decrease the  $C_{PK}$ , although the outlier would be a pass sample! Also look at *cpk.xls* from the River webpage for experiments.*

