Efficient Pre-Silicon Validation and Post-Silicon Tuning of Self-Healing Analog/RF Integrated Circuits

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Fa Wang

B.S., Automation, Tsinghua University

Carnegie Mellon University
Pittsburgh, PA

December, 2015

Abstract

The advent of the nanoscale integrated circuit (IC) technology makes high performance analog and RF circuit increasingly susceptible to large-scale process variations. Process variations, including inter-die variations and local mismatches, significantly impact the parametric yield of analog/RF circuit, and must be properly handled at all levels of design hierarchy. Traditional approaches based on over-design are not sufficient to maintain high parametric yield, due to the large-scale process variations and aggressive design specifications at advanced technology nodes. In this context, the self-healing circuit has emerged as promising methodology to address the variability issue. In this thesis, we propose efficient pre-silicon validation and post-silicon tuning techniques, which are essential for the practical usage of self-healing methodology.

One important problem in self-healing methodology is to efficiently and accurately predict the parametric yield in pre-silicon. The main challenge of this problem is caused by multiple circuit states related to tuning knobs. Given that these circuit states closely interact with process variations, they must be properly modeled in order to accurately estimate the parametric yield. Towards this goal, we develop an efficient performance modeling algorithm, referred to Correlated Bayesian Model Fusion (C-BMF) that explores the correlation between circuit states. Next, based on the performance model, the self-healing behavior and the parametric yield can be efficiently and accurately predicted.

Another important problem in self-healing circuit is to efficiently perform post-silicon tuning. Towards this goal, indirect performance sensing methodology has recently attracted great attention. In the indirect performance sensing paradigm, the performance of interest (PoI) is not directly measured by on-chip sensor, but is instead accurately predicted from an indirect sensor model. Such indirect sensor model takes a set of other performances as inputs, which are referred to as the performances of measurements (PoMs). The PoMs are selected such that they are highly correlated with PoI and are easy to measure. Due to the process shift associated with manufacturing lines, the indirect sensor model must be calibrated from

time to time. For the purpose of reducing the model calibration cost, we propose a Bayesian Model Fusion (BMF) algorithm that reuses the information collected in early stage of manufacturing. We further extend BMF to a Co-learning Bayesian Model Fusion (CL-BMF) algorithm that incorporates not only the early stage information, but also the current stage information that was not considered in the original modeling problem.

Acknowledgement

First of all, I would like to express my deepest appreciation to my advisor, Prof. Xin Li for his guidance and persistent help on my Ph. D. study and research. He provided me with the important academic training in analytical, writing, presentation and learning skills. He continually and convincingly conveyed a spirit of adventure and passion.

This work would not be accomplished without the invaluable academic suggestions from my Ph. D. committee members. Not only did they carefully review my proposal and thesis, but also offered constructive insights on the research presented in this thesis. I would like to express my gratefulness to Prof. Shawn Blanton for kindly providing industrial connections, which inspires me to finish the cost analysis part of the thesis. I would like to thank Prof. Tamal Mukherjee for his instructions and helpful inputs regarding the side information selection of co-learning in the thesis. I would also like to thank Dr. Jean-Olivier Plouchart for all the active collaboration and inspirational discussions.

In addition, my sincere thankfulness goes to IBM and Intel. This work cannot be done without their cooperation and support. I would like to thank Alberto Valdes-Garcia and Bodhisatwa Sadhu from IBM for providing their self-healing circuit data and continued discussion and support. I would like to thank Chenjie Gu from Intel for meaningful exchanges of ideas on Bayesian Model Fusion algorithm development. I would also like to acknowledge the financial support for this work from DARPA HEALICS program, National Science Foundation and C2S2 Focus Center.

Last but not least, I would like to thank my group-mates and friends at Carnegie Mellon University for their friendship and academic inspirations. Many thanks also belong to my parents and my beloved for their continuous encouragement and unyielding support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In 1965, Gordon E. Moore made the projection that the number of transistors in a dense integrated circuit (IC) doubles approximately every two years, which is later known as Moore's law [1]. Moore's law proved accurate for several decades, and is used in the semiconductor industry to guide long-term planning. Behind Moore's law, reduced cost and improved performances are the main driving factors for continuous IC scaling.

In past few decades, numerous technology breakthroughs in IC technology contribute to sustain Moore's law including chemically-amplified photoresist [2], deep UV excimer laser photolithography [3], chemical mechanical planarization [4]. New technology nodes enabling smaller transistor feature size continuously being developed. Larger and larger circuit functions with superior performances are manufactured on a single semiconductor substrate. However in 2012, when technology node reaches 22nm, the growth rate began to decrease [5]. The key limiting factor is the yield loss, which is defined as the proportion of the number of fabricated chips that fail to meet performance specifications [6]. The yield loss directly degrades the profitability of the IC manufacturing, and thus must be properly handled in all stages of technology node development.

Process variations in analog/RF circuit is an important factor that causes yield loss [6]-[7]. Process variations, including inter-die variations and local mismatches, manifest themselves as the uncertainties associated with the geometrical and electrical parameters of semiconductor devices. Designing analog/RF circuit becomes increasingly challenging in advanced technology nodes. With the deep submicron IC technology scaling, the device-level process variations have become larger and larger, which are not easily mitigated by foundries. This lays a fundamental challenge on robust design techniques for analog/RF circuits.

However, traditional robust design techniques [8]-[32] are not sufficient to maintain high parametric yield with process variations. In this context, on-chip self-healing methodology has been proposed [41]-[53]. In self-healing circuit, tuning knobs (e.g. tunable current source) are employed which allow performance (e.g. voltage gain, power, etc.) flexibility. The key idea of on-chip self-healing is to actively measure and adaptively tune circuit performances in post-silicon on-chip. As such, for each manufactured chip, the knob configurations can be tuned to optimize the circuit performance for its particular process corner. The parametric yield of circuit can therefore be significantly improved.

The design flow of self-healing analog/RF IC will be discussed in detail in Section 1.2.2. Pre-silicon validation and post-silicon tuning are two important tasks in the design flow. In pre-silicon, the self-healing circuit design must be verified by parametric yield analysis before moving to the manufacturing process. Once the circuit is manufactured, post-silicon tuning is required to exploit the benefit of configurability and improve the parametric yield. The main challenge for both pre-silicon validation and post-silicon tuning is how to maintain low overhead in computation, hardware and testing cost so that self-healing methodology can be practically implemented.

In this thesis, we develop efficient pre-silicon validation and post-silicon tuning techniques for self-healing analog/RF IC. The core task of pre-silicon validation is the parametric yield estimation. We propose an efficient parametric yield estimation algorithm based on correlated performance modeling which considers the correlation among tuning knob configurations. The post-silicon tuning of self-healing circuit consists of two core tasks: (i) on-chip performance sensing, and (ii) on-chip performance tuning. In on-chip performance sensing, circuit performances are actively measured by on-chip sensors. Then in on-chip performance tuning, a set of on-chip tuning knobs (e.g. bias current) are adaptively adjusted in order to satisfy all performance specifications. While the on-chip performance tuning is also an important topic, we restrict our discussion to on-chip performance sensing in this thesis. We develop a novel indirect performance sensing technique to facilitate inexpensive-yet-accurate on-chip performance sensing. In the rest of this chapter, we will briefly review the background on traditional robust design techniques, on-chip self-healing methodology, pre-silicon validation of self-healing circuit, and post-silicon tuning of self-healing circuit, and then outline the overall structure of the thesis.

1.1 Traditional Robust Design Techniques

Traditionally, various design techniques have been proposed to handle the performance uncertainty of analog/RF IC. The traditional design techniques can be classified into two broad categories: (i) corner-based design optimization [8]-[24], and (ii) statistical design optimization [25]-[32].

In corner-based design optimization [8]-[24], the circuit is optimized at all process corners by combining the extreme values of all process parameters. However, the corner-based approach has two major issues. First, the performance corner in an analog circuit does not necessarily to be the process parameter corner. In other words, the extreme performance value may not occur when all process parameters reach extreme values. In fact, the actual performance corner is topology dependent and performance dependent [7]. Second, the total number of the combinations of the process corners exponentially grow with the number of process parameters. In an advanced technology node, the total number of process parameters can easily reach 1000. In such case, the total number of corner combinations will become astronomical, and the corner-based approach would be impractical.

In statistical design optimization [25]-[36], the statistical distribution of the circuit performance is first analyzed. Then the circuit design is optimized such that an only sufficient and necessary margin is chosen. As such, the design is robust with respect to process variations, and meanwhile the over-design is minimized. Statistical design optimization has three categories: (i) direct yield estimation [25]-[28], (ii) design centering [29]-[32], and (iii) worst-case optimization [33]-[36]. The direct yield estimation approach [25]-[28] essentially traverses the design space and estimate parametric yield associated with each design point. The yield is usually estimated by transistor-level Monte Carlo simulations. Eventually the design point with the maximum yield is selected. However, since the simulation based yield estimation can be very expensive, the direct yield estimation based design optimization is not very efficient. The design centering approach [29]-[32] is based on the geometrical insight in the parameter space, which aims to find the optimal design point that has the maximum distance from all boundaries posed by performance constraints. In design centering approach, the robustness of the circuit is measured by the volume of the ellipsoid that maximally inscribed or minimally circumscribed the boundaries. However, since design-centering does not directly optimize parametric yield, the optimized design point might not provide the maximum yield.

The worst-case optimization approach [33]-[36] achieves robust circuit design by optimizing the worst case circuit performances considering process variations. The worse-cast performance values can be defined as a particular percentile point (e.g. 99%) of the performance cumulative distribution function. However, worst-case optimization has the limitation that it could only guarantee good optimized design with a single performance. In the case of multiple performance specifications, it will ignore the performance correlation and thus cannot provide an accurate yield estimation, rendering sub-optimal design point.

In this context, the robust analog/digital optimization algorithm (ROAD) has been proposed [35]-[36]. ROAD approach is consisted of one initialization step and three core steps. In the initialization step, a coarse initial design is found by performance centering algorithm [37]. Then, a number of transistor-level simulations are run and the performance model in terms of both design variable and process parameters is fitted. After this, worst-case performance is analyzed and the worst-case performance model is fitted in terms of design variables. Such worst-case performance model is then used to optimize circuit performance via nonlinear programming.

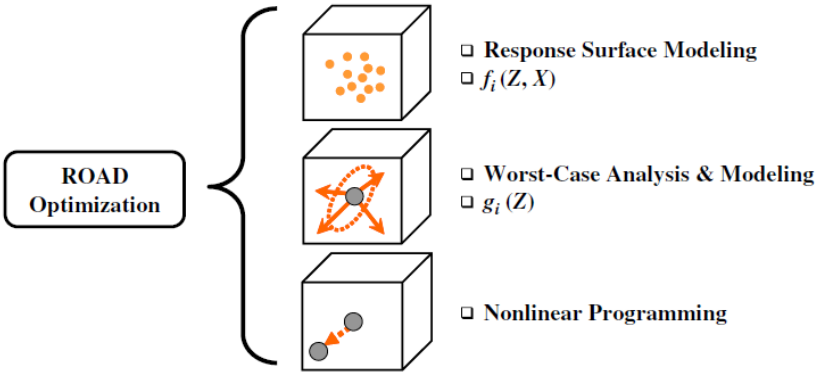


Figure 1-1. Robust analog/digital optimization algorithm flow [7].

With the scaling of IC technology, robust circuit design has become more and more challenging, if not impossible. The reasons are two-folds here. On one hand, process variations have become larger and larger with the transistors continuously scaling to deep submicron. On the other hand, more and more aggressive specifications on speed, power and area are set to accommodate costumer demands. For example, with the world quickly moving into the Internet of Things (IoT) era, ultra-high speed

communication circuits with low power consumption are required [38]-[40]. In this context, robust circuit design becomes extremely difficult. Figure 1-2 illustrates the challenge. The advance in IC technology makes the performance distribution becoming increasingly scattered. As such, the performance specification keeps pushing to squeeze the feasible design region. Even if the optimal robust design point can be found, the trend of the parametric yield decrease is inevitable for the traditional design methodology. To address this fundamental challenge, on-chip self-healing methodology has been proposed, which will be explained in Section 1.2 in detail.

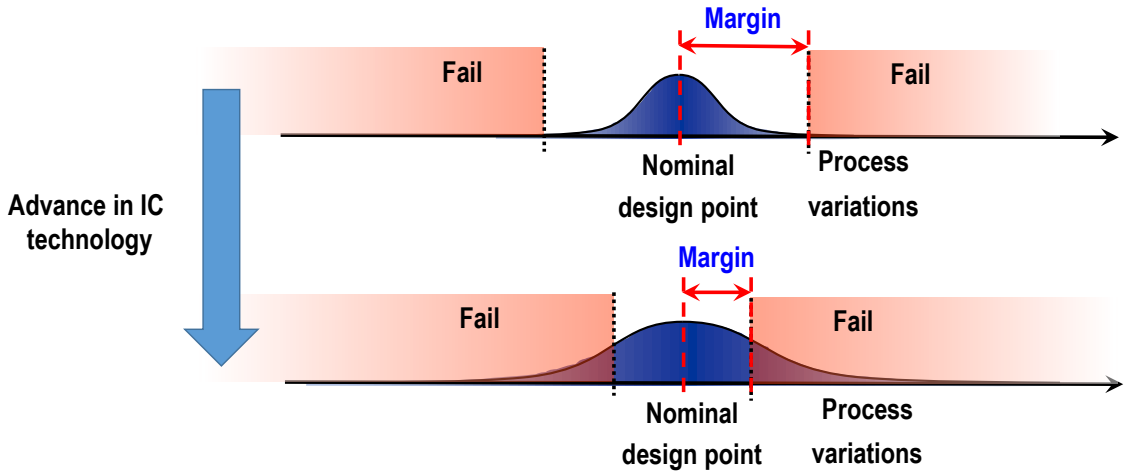


Figure 1-2. Challenges of robust circuit design is shown. With the scaling of IC technology, on one hand the process variations become larger, while on the other hand the circuit specifications become more aggressive. As a result, the design margin dramatically shrinks.

1.2 On-Chip Self-Healing Methodology

On-chip self-healing methodology [41]-[46], [49]-[53], [58]-[60] has emerged as a promising avenue to address the aforementioned challenges. In self-healing circuit, tuning knobs (e.g. tunable current source) are employed which allow performance (e.g. voltage gain, power, etc.) flexibility. The key idea of on-chip self-healing is to actively measure and adaptively tune circuit performances in post-silicon on-chip. Compared to the post-manufacturing calibration work in [54]-[57], on-chip self-healing provides more flexibility. The performance tuning of on-chip self-healing is not restricted to be in the testing site, but can

be done anywhere. This provides additional tuning capacity when the environmental condition and/or the circuit condition changes in field.

In this sub-section, we provide an overview of on-chip self-healing methodology. In Section 1.2.1, we introduce essential components of self-healing circuit. In Section 1.2.2, we discuss the design flow of self-healing circuits.

1.2.1 Components of On-Chip Self-Healing

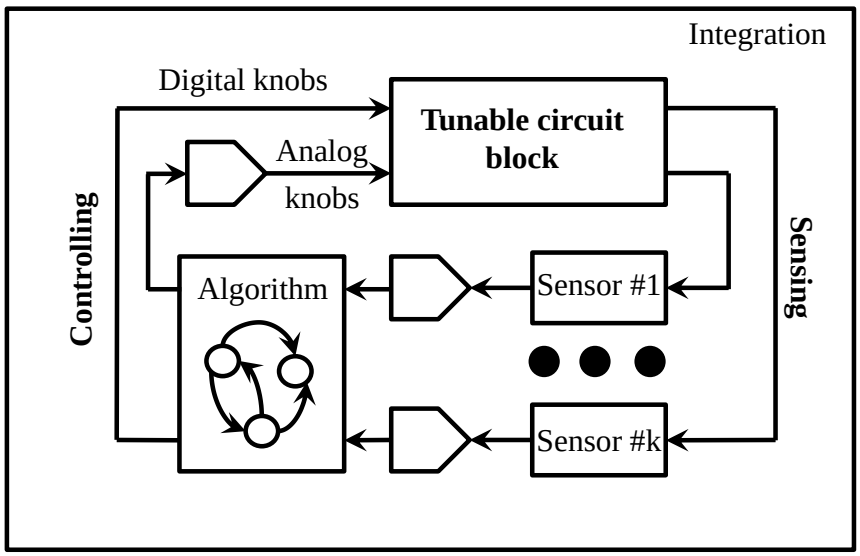


Figure 1-3. Three core components of on-chip self-healing are shown: tunable circuit block, sensing circuit and controlling circuit. The three components are seamlessly integrated on-chip to achieve self-healing.

Figure 1-3 shows three core components of the on-chip self-healing circuit [51], which need be seamlessly integrated on-chip:

- Tunable circuit block: Control knobs (e.g. tunable current mirror, tunable capacitor bank, etc.) are employed on-chip to allow flexibility in circuit performances (e.g. noise figure, gain, nonlinearity, etc.). The tunable circuit block can either be a single circuit component (e.g. low-noise amplifier) or a system consisted of multiple circuit components (e.g. the receiver path in a RF transceiver). The tunable circuit block serves as the main body of the self-healing circuit and usually consumes the major portion of area. The control knobs need to be carefully designed in the tunable circuit such that the tuning ranges of the circuit performances of interest are large enough to cover the process and

environmental condition variations. Different from traditional circuit block without control knobs, the tunable circuit block possesses multiple 'states' enabled by control knobs. Therefore, tunable circuit block has the potential of adaptively selecting the optimal state that suits best to a particular process corner or environmental condition. Such potential will be realized through the integration with sensing and controlling circuitry.

- Sensing circuit: Sensing circuit (e.g. voltage sensor, temperature sensor, frequency counter, etc.) is the circuit component that is able to capture the performance metrics of tunable circuit block. The design of sensing circuit is an important task in the self-healing design. The accuracy-cost trade-off must be carefully analyzed and optimized in order to achieve accurate performance sensing while not causing large over-head. For example, it is not reasonable to always design high accuracy sensing circuit, because they usually consume large circuit area and the yield impact may not high.
- Controlling circuit: Controlling circuit is a decision-making circuit component that obtains tunable circuit performance information from sensing circuit and optimally determines the tuning knob configuration for tunable circuit block. In our work, controlling circuit is consisted of microcontroller, I/O, SRAM [51]-[53]. The self-healing algorithm (e.g. exhaustive search) can be loaded in the controlling circuit and the self-healing operation procedure is performed exactly as directed by the self-healing algorithm. The self-healing algorithm can either be a component-level algorithm focusing on the healing of a particular circuit component (e.g. LNA) or a system-level algorithm targeting at system-level performance metrics (e.g. noise figure of a receiver path).

The eventual goal of the on-chip self-healing is to actively sense important circuit performances and adaptively tune the knob configurations in order to improve overall circuit performance (e.g. improving parametric yield, reducing power consumption, etc.).

1.2.2 Design Flow of On-Chip Self-Healing

Similar to traditional circuit without tuning knobs, self-healing circuit design also needs to go through several important design stages. The design flow of self-healing circuit is summarized in Figure 1-4. The flow spans four stages: (i) schematic design, (ii) layout design, (iii) chip manufacturing, and (iv) post-silicon tuning and testing. At each stage, simulation or measurement data are collected to tune and/or

validate the self-healing system in Figure 1-3.

The design starts from the schematic design where the topology, transistor sizing and other important design parameters are selected. Then the parasitics, routing and transistor placement are considered in the layout design. If the design fails to meet the performance specifications at any of the pre-silicon stage, the circuit design will be improved until the specifications are met. Once the design is fully verified in the pre-silicon stage, the chip will be manufactured. The post-silicon tuning is then done to improve the circuit performances in order to increase parametric yield or reduce power consumption. The post-silicon tuning involves close interaction between tunable circuit block, sensing, controlling circuitry, as well as the automatic testing equipment (ATE).

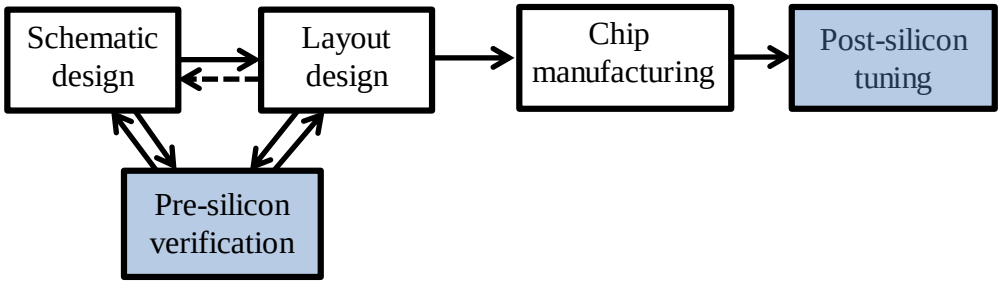


Figure 1-4. Design flow of self-healing circuit is shown, which is consisted of four stages: (i) schematic design, (ii) layout design, and (iii) chip manufacturing, and (iv) post-silicon tuning.

To reduce the overall cost per good chip, efficient analysis, design, validation and tuning methodology are required. In this thesis, we focus on three important problems related to self-healing circuit: pre-silicon validation, post-silicon tuning (as shown in the highlighted blocks in Figure 1-4) and cost analysis.

1.3 Pre-Silicon Validation of Self-Healing Circuit

In this section, we review the background of the pre-silicon validation of self-healing circuit. The goal of the pre-silicon validation is to verify the parametric yield considering the self-healing effect in pre-silicon. Parametric yield estimation is a fundamental task for analog/RF circuit and has been extensively studied in the literature [61]-[67]. The traditional parametric yield estimation can be classified into four broad categories: (i) Monte Carlo based approach [61]-[63], (ii) statistical blockade [64]-[66], (iii)

Bayesian inference based approach [67], and (iv) performance model based approach [68]-[69],

The Monte Carlo based approach is to repeatedly draw random samples from process parameter distributions and evaluate circuit performance of interest by circuit simulations (e.g. transistor-level simulations in Cadence). In the direct Monte Carlo approach [61], the pre-silicon parametric yield is estimated by calculating the portion of the number of simulated instances that pass all performance specifications N_{pass} out of the total number of simulated instances N_{total}:

$$Yield = \frac{N_{pass}}{N_{total}}. \tag{1.1}$$

Direct Monte Carlo approach is easy to implement and is not dependent on the dimensionality of the process parameters, because it directly operates in the performance space. However, the well-known limitation of the Monte Carlo approach is that it is very time consuming, since no structural insight is used. The Quasi-Monte Carlo method [63] is an extension of direct Monte Carlo. The key idea is to assign a sequence of deterministic Monte Carlo samples in parameter space to improve the convergence rate. This approach aims to uniformly sample the whole parameter space, which is able to better capture the low probability region and thus yield more efficient estimation. The Quasi-Monte Carlo approach is efficient when the parameter space is low-dimensional. However, as a trade-off, it suffers from the curse of dimensionality because it relies on the parameter space. The accuracy of Quasi-Monte Carlo can be even worse than direct Monte Carlo in high-dimensionality case. For self-healing circuits in advanced technology node, since the number of process parameters is high and a large number of knob configurations need to be simulated, the simulation cost of Monte Carlo approach would be very high. Therefore, Monte-Carlo approach is not suitable for pre-silicon validation task of self-healing circuit.

Statistical blockade [64]-[66] is proposed based on the observation that some Monte Carlo samples are not likely to fail, therefore simulating them would have very limited benefit. In particular, a classifier is constructed to block the Monte Carlo samples that are deemed to pass or fail. As such, the whole parameter space is classified into two types of regions. One region contains the parameter values that we are certain about the outcome of the simulation (i.e. pass or fail in performance specifications), which are blocked out from running simulations. The other region contains parameter values whose simulation outcomes are not clear. By simulation the second region, the outcomes of uncertain points are determined and classifier can

be further refined. The main strength of statistical blockade is its efficiency in low failure rate application. In such application, the majority of samples would pass the specifications, therefore avoid simulating those points can provide significant efficiency improvement over Monte Carlo approach. However, in analog/RF circuit, the failure rate is usually not as low as the rare failure rate application, due to the over-design consideration. Furthermore, since statistical blockade operates in the parameter space, the dimensionality problem would affect its effectiveness. In advanced technology nodes, the number of process parameters are usually high (e.g. above 1000), and the cost of training and refining the classifier can be very expensive. As a result, the benefit of applying statistical blockade on self-healing analog/RF circuit is very limited.

In light of the correlation between design stages, a Bayesian inference based approach [67] is proposed. The key observation that motivates the approach is that the performance probability distribution can be very similar in different design stages. For example, the performance cumulative distribution function (CDF) of the read path delay of SRAM in the schematic stage can be very similar to the CDF in post-layout stage. Such correlation can be explored among any stages in the design flow where the two stages follow a sequential order. The stage that appears earlier in the design flow is considered as the early stage while the stage appears later is considered as late stage. Mathematically, a Bayesian inference approach is proposed which encodes the early stage information into a prior distribution. The prior distribution is then combined with very few samples collected in the late stage (either by simulation or silicon measurements) through Bayesian inference [61]. The Bayesian inference would provide a posterior distribution that can be translated into the performance distribution in the late stage. Although Bayesian inference based approach is promising, it relies on the existence and accuracy of the early-stage model. In the case where an early-stage performance distribution does not exist or not reliable, the parametric yield estimation can be very expensive.

The performance modeling based approach [68]-[69] is consisted of two steps. First, performance model is constructed based on a number of simulation samples of the circuit. Performance model is a statistical model where circuit-level performance (e.g. phase noise) is approximated as an analytical function (e.g. polynomial) of device-level variations (e.g. ΔV_{TH}). As such, the circuit-level performance variations are linked to the process parameters variations. Next, based on the performance model, the probability distribution is predicted. Since the performance model usually has a special form (e.g. linear or

quadratic polynomials), the associated probability distribution is also structured. The statistical parameters (e.g. mean, variance, or high-order moments) can then be efficiently determined from performance model and the parametric yield can be estimated.

One key observation about the self-healing circuit is that the performances associated with different states are highly structured. In other words, the underlying statistical link between process parameters and circuit performances among different states are highly correlated. The performance modeling approach directly build model based on process parameters and therefore has the potential to captures such structure.

Performance modeling is the task to find model coefficients associated with performance model. It serves as a core component in the modeling, analyzing and optimization throughout the analog/RF circuit design flow. The performance modeling techniques have been extensively studied in the literature [70]-[79]. The state-of-the-art performance modeling technique is the sparse regression [74]-[80]. Sparse regression approach is motivated by the observation that only a small number of process variables contributes to the performance variation. Based on this assumption, sparse regression aims to find a small number of important basis functions and simply the model template. As such, only a small number of model coefficients that relates to important basis functions need to be solved. The modeling efficiency can be significantly improved by using sparse regression.

For self-healing circuit, this fundamental modeling problem can be extremely difficult with the following characteristics:

- High-dimensional variation space: A large number of device-level random variables must be used to model the process variations associated with a self-healing analog/RF system at advanced technology nodes. For example, a self-healing low-noise amplifier designed in a commercial 32nm process carries over 1000 device-level random variables.
- Large number of knob configurations: To maximize tuning range and resolution, a large number of knob configurations (i.e. ‘states’) should be designed. This, however, significantly increases the complexity of performance modeling, since a unique performance model is required to accurately capture the variability at each state.
- Correlated knob configurations: Since different states are associated with the same self-healing circuit, the performance models of the different states are correlated. In particular, the model template and

model coefficients are expected to be similar among these states.

It is important to note that most traditional approaches [74]-[80] are not fully equipped for performance modeling of self-healing circuits. Taking sparse regression [74]-[77] as an example, the sparsity of model coefficients is explored to improve efficiency. However, no correlation is considered between different states. In other words, performance modeling is done for each state independently and the performance modeling cost would quickly become intractable as the number of states increases. Several other performance modeling methods (e.g., simultaneous orthogonal matching pursuit [78], multi-mode multi-corner sparse regression [79], group lasso [80], etc.) have been proposed to exploit the aforementioned correlation information. However, these methods only take into account the shared model template among different states and they ignore the correlation of coefficient magnitude. In this regard, there is a strong need to develop an efficient performance modeling approach, which serves as a starting point of pre-silicon validation.

1.4 Post-Silicon Tuning of Self-Healing Circuit

Post-silicon tuning is another important step in the self-healing circuit design flow. The goal of post-silicon tuning is to find the optimal tuning knob configuration that gives desired circuit performances. Two important tasks need to be achieved to enable post-silicon tuning through on-chip self-healing: (i) on-chip performance sensing, and (ii) on-chip performance tuning. While the on-chip performance tuning is also an important topic [81]-[82], we restrict our discussion to on-chip performance sensing in this thesis.

To practically implement on-chip performance sensing, a large number of performance metrics must be measured accurately and inexpensively by on-chip sensors. Such a measurement task is not trivial because many analog and RF performance metrics (e.g. phase noise) cannot be easily measured by on-chip sensors [41]-[42]. For this reason, alternate test methodology, also called indirect performance sensing, has recently attracted great attention [43], [47], [50]-[51], where performance of interest (PoI) is not directly measured by an on-chip sensor. Instead, it is accurately predicted from a set of other performance metrics, referred to as the performance of measurement (PoMs) that are highly correlated with PoI and are easy to measure. Towards this goal, indirect sensor modeling is a critical task where the objective is to build a mathematical model to capture the correlation between PoI and PoMs so that PoI can be accurately

predicted from PoMs. The indirect sensor models, once trained, can be stored on an on-chip microcontroller and direct the control knob tuning in the controlling phase.

To provide a fully on-chip indirect performance sensing solution with low overhead, two important challenges must be addressed:

- High cost in on-chip hardware: The storage, computation and sensor inputs of indirect sensor models all require on-chip hardware resources. A complex model with many PoMs and model coefficients, though accurate, consumes massive hardware resources and thus introduce large overhead in self-healing [51].
- High cost in repeated calibration: Due to the process shift associated with manufacturing lines, the indirect sensor models must be repeatedly calibrated from time to time. During each calibration, PoMs and PoI need to be measured from a set of training chips, and then used for model regression. Such calibration procedure can lead to large overhead in self-healing if done inefficiently.

The aforementioned two challenges pose a strong need of developing an efficient on-chip indirect performance sensing methodology.

It should be noted that neural network [61] has been adopted as indirect sensor model in the on-chip self-healing work in [58]-[60]. Compared to the polynomial indirect sensor model in the proposed methodology, neural network model is able to achieve higher prediction accuracy when the polynomial model is insufficient to capture the performance variability. However, neural network consumes huge on-chip resources [58]-[60] and therefore has larger overhead than the proposed methodology. Such overhead can be a critical concern in the applications with limited on-chip resources (e.g. spy transmitter).

1.5 Cost Analysis

The design methodology of analog/RF circuits can be classified into three categories: (i) traditional design without tuning capacity (Traditional) [8]-[32], (ii) off-chip adaptive performance tuning (AT) [47]-[48], and (iii) on-chip self-healing (OH) [41]-[46], [49]-[53]. The Traditional method does not include tunable control knobs in the circuit design. For both AT and OH, tunable control knobs are designed to improve parametric yield. The performance calibration in AT is done off-chip, which relies on ATE measurements for all knob configurations. On the other hand, the performance calibration of OH is done

on-chip, which relies on microcontroller loaded with indirect sensor model. Although the three design methods have been proposed for many years, their trade-off has not been extensively studied in the literature. It is important for circuit designs to understand the critical cost factors when selecting an optimal design method. Toward this goal, we perform a cost analysis for Traditional, AT and OH. In this thesis, we limit the application to the mmWave transceiver product.

Cost per good chip is an important criterion that measures the profitability of product [99]. The cost per good chip essentially quantifies the overall capital investment in the manufacturing, testing and packaging cost throughout the good chip generation process. The manufacturing cost is the cost of fabricating a particular chip. It includes the material, equipment, maintenance, consumables and utilities cost during the manufacturing process. In this thesis for comparison purpose, we assume that given a particular manufacturing process (e.g. 32nm), the manufacturing cost per good chip is proportional to the die area of the circuit. In other words, if the circuit design has larger area, the manufacturing cost would be higher. In this regard, the OH approach would have the highest manufacturing cost among the three approaches, due to the overhead in controlling and sensing circuit. The Traditional approach, on the other hand, has the lowest manufacturing cost because it does not have either controlling or sensing circuit. In this thesis, we further assume that the packaging cost is approximately the same as the manufacturing cost [98].

The testing cost includes the cost of performing post-silicon performance tuning as well as validating the pass/fail of a particular die [91]-[99]. To combat the variability issue, tuning knobs are employed in the circuit blocks to allow flexibility. Once the circuits are manufactured, the control knobs must be adaptively tuned to maximally exploit the benefit of configurability. This procedure can be done either off-chip (i.e. AT) or on-chip (i.e. SH). AT essentially measure the performances for all knob configurations directly using ATE, while SH measure a small number of performances to fit an indirect sensor model and then tune the knob configurations using microcontroller on chip. Once the knob configuration is selected for each chip, AT and OH can be treated similarly as Traditional. In particular, the circuit performances are first verified through wafer probe test. Then the die is packaged and the final specification test is done for the packaged chips. It should be noted that for mmWave product, the packaging cost is very high, therefore there is a strong need to test the circuit in wafer probe level before

packaging. Given that ATE with mmWave measurement capability is usually very expensive, the testing cost is directly proportional to the total testing time using ATE. The Traditional approach would have smallest testing time, because it does not require any performance tuning. The AT approach, on the other hand, has the largest testing time, because all knob configurations need to be tested.

Compared to the Traditional approach, OH is able to achieve better parametric yield. However, the hardware cost overhead and testing cost overhead will be high. Compared to AT, OH will achieve lower parametric yield due to limited accuracy in on-chip performance sensors. However, the testing cost, especially the performance tuning cost, will be lower. Therefore, it is not clear which approach would provide lowest cost per good chip. In this regard, there is a strong need to perform a cost analysis considering important cost factors for Traditional, AT and OH approaches.

1.6 Thesis Contributions

In this thesis, we propose an efficient pre-silicon validation flow, efficient post-silicon tuning flow and a cost analysis for on-chip self-healing. The major technical contributions of this thesis are:

- We propose an efficient pre-silicon validation flow via performance modeling. In particular, we propose to build performance model of self-healing circuits based on Correlated Bayesian Model Fusion (C-BMF) algorithm. The proposed C-BMF method is motived by the fact that performance models associated with different states are correlated. Our key idea is to encode the correlation information for both model template and coefficient magnitude among different states by using a unified prior distribution. The prior distribution is then combined with a few simulation samples via Bayesian inference to efficiently determine the unknown model coefficients. Furthermore, we propose efficient and robust hyper-parameter inference technique to solve a large number of hyper-parameters involved in C-BMF. It consists of two major steps: (i) estimating the hyper-parameters to obtain an initial guess using a heuristic approach, and (ii) optimizing the hyper-parameter values using an expectation-maximization algorithm [61]. The performance model coefficients can then be determined based on the inferred hyper-parameters. Moreover, an efficient pre-silicon validation approach is proposed where the correlated performance modeling algorithm serves as a core component. Instead of simulating all the knob configurations from all chips, we generate pseudo samples based on

performance model. As such, the parametric yield estimation cost can be significantly reduced.

- We propose an efficient post-silicon tuning flow via indirect performance sensing. In particular, we propose a Bayesian Model Fusion (BMF) based indirect sensor model calibration technique. The key idea of BMF is to combine the old (i.e., before process shift) indirect sensor model with very few new (i.e., after process shift) measurement data to generate a new model that is aligned with the new process condition. Mathematically, the old model is encoded as prior knowledge, and a Bayesian inference is derived to optimally fit the new model by maximum-a-posteriori (MAP) estimation. As such, the indirect sensor model can be efficiently calibrated. As an extension of BMF, we further propose a novel indirect sensor modeling algorithm that is referred to as Co-Learning Bayesian Model Fusion (CL-BMF). CL-BMF takes advantage of the additional information collected from measurements to reduce the performance modeling cost. Different from the BMF which focus on the prior information of model coefficients (i.e. the coefficient side information) only, CL-BMF takes advantage of another new form of prior knowledge: the performance side information (PSI). In particular, CL-BMF combines the coefficient side information (CSI), the PSI and a small number of training samples through Bayesian inference based on a graphical model. Furthermore, a PSI selection technique is proposed to select most important performances to provide PSI. As a result, the indirect sensor model calibration cost can be further reduced.
- Due to the overhead involved in the on-chip self-healing, it is not clear to circuit designers whether the on-chip self-healing would really benefit the final cost of product. In this regard, we present a cost analysis on a mmWave self-healing transceiver to compare three different circuit designs methodologies: (i) traditional circuit design without tunable component, (ii) tunable circuit with off-chip performance calibration, and (iii) on-chip self-healing circuit. The cost per good chip is calculated which quantifies the overall capital investment in the manufacturing, testing and packaging cost throughout the good chip generation process for each design approach. The cost analysis can provide a guideline to circuit designer on whether or not to apply on-chip self-healing for a particular application.

It is important to note that the proposed C-BMF, BMF and CL-BMF algorithms are derived from rigorous statistical theory, and can be generally applied to other applications such as fast statistical analysis of rare failure events [62], probability distribution estimation [67], etc.

1.7 Thesis Organization

In this section, we briefly summarize the thesis organization. The thesis is composed of three core components: efficient pre-silicon validation, efficient post-silicon tuning and cost analysis.

In Chapter 2, we first develop a Correlated Bayesian Model Fusion (C-BMF) algorithm to efficient fit performance model for self-healing circuit via Bayesian inference. In the Bayesian inference, a large number of hyper-parameters need to be determined. Towards this goal, we further propose an efficient and robust hyper-parameter inference approach, which consists of two major steps: (i) estimating the hyper-parameters to obtain an initial guess using a heuristic approach, and (ii) optimizing the hyper-parameter values using an expectation-maximization algorithm. The proposed performance modeling approach is validated in two circuit examples designed in a commercial 32nm CMOS process. We further present the parametric yield estimation flow for self-healing circuit based on performance model. Instead of simulating all the knob configurations from all chips, we generate pseudo samples using performance model of self-healing circuit. As such, the parametric yield estimation cost can be significantly reduced. The efficacy of the proposed parametric yield estimation algorithm is validated by simulation data collected from two circuit examples designed in a commercial 32nm CMOS process.

In Chapter 3, we address the problem of efficient post-silicon tuning. Towards this goal, we propose a novel indirect performance sensing technique for on-chip self-healing of analog and RF circuits. In particular, we propose to use sparse regression to identify a set of important basis functions so that the overhead of on-chip self-healing can be minimized. We further propose an efficient indirect sensor model calibration approach based on Bayesian Model Fusion (BMF). In BMF algorithm, we encode the early stage information in the prior distribution. The prior distribution is then combined with a few samples in the late stage via Bayesian inference. As a result, the indirect sensor model can be updated with low cost. Experiments on silicon measurement data demonstrate that the efficacy of the proposed indirect performance sensing

The BMF algorithm essentially takes advantage of the side information of model coefficients (i.e. CSI) to facilitate the efficient indirect sensor model calibration. In Chapter 4, we extend BMF algorithm to further consider another type of side information: the PSI. CL-BMF optimally combines the following

information: (i) the CSI, (ii) the PSI, and (iii) a small number of training samples. Bayesian inference is constructed and represented as a graphical model, where the CSI and the PSI are encoded by the prior distribution and the likelihood function respectively. From the Bayesian inference, the unknown model coefficients can be accurately determined by maximizing the posterior distribution. Furthermore, a PSI selection technique is also proposed to select most important performances to provide PSI. As is demonstrated by our circuit examples designed in a commercial 32nm SOI CMOS process, the proposed CL-BMF method achieves significant cost reduction over other indirect sensor modeling techniques.

In Chapter 5, we present a cost analysis on a mmWave self-healing transceiver. The manufacturing cost, testing cost and packaging cost are taken into consideration to analyze the cost per good chip. In this cost analysis, three different circuit design methodologies are compared: (i) traditional circuit design without tunable component, (ii) tunable circuit with off-chip performance calibration, and (iii) proposed on-chip self-healing circuit. Based on our cost analysis for this particular example, the cost per good chip of self-healing circuit is significantly less than the other two circuit design methods.

Chapter 6 concludes the thesis with a high-level summary of the work, and discusses several future potential directions of research related to this work.

Chapter 2

Efficient Pre-Silicon Validation via Performance Modeling

2.1 Motivation

One key observation about the self-healing analog/RF IC is that the performances associated with different states (i.e. knob configurations) are highly structured. In other words, the underlying statistical link between process parameters and circuit performances among different states are highly correlated. The performance modeling approach directly build model based on process parameters and therefore has the potential to captures such structure. In this regard, we propose the pre-silicon validation approach that is consisted of two steps: (i) performance modeling of self-healing circuits, and (ii) parametric yield estimation based on performance model. In the proposed flow, the main challenge is the performance modeling step. Once the performance model is obtained, the parametric yield can be estimated straightforwardly. Furthermore, the performance model, once obtained, can be applied to other important applications for self-healing circuit such as corner extraction [84] and design optimization [25]-[36].

To start with, we mathematically formulate the performance modeling problem for self-healing circuits. Given a self-healing analog/RF circuit (e.g. a down-conversion mixer) with totally K states (i.e. knob configurations), the performance model of the k-th state can be represented by an analytical function (e.g., polynomial) of device-level variations (e.g., ΔV_{TH}):

$$y_k \approx f_k(\mathbf{x}) = \sum_{m=1}^{M} \alpha_{k,m} \cdot b_m(\mathbf{x}) \quad (k = 1, 2, \cdots, K) \tag{2.1}$$

where $\{y_k;\ k = 1, 2, \cdots, K\}$ contains the performance of interest (PoIs) associated with the K states, $\mathbf{x}$ is a vector representing the device-level variations, $f_k(\mathbf{x})$ denotes the performance model of interest which

establishes a mapping from $\mathbf{x}$ to y_k, $\{\alpha_{k,m}; m = 1, 2, ..., M\}$ contains the model coefficients of $f_k(\mathbf{x})$, $\{b_m(\mathbf{x}); m = 1, 2, ..., M\}$ contains the basis functions (e.g., linear or quadratic polynomials), and M denotes the total number of basis functions. Without loss of generality, we assume that the same set of basis functions $\{b_m(\mathbf{x}); m = 1, 2, ..., M\}$ are shared among all the states.

In order to determine the performance models in (1), we need to find the model coefficients $\{\alpha_{k,m}; k = 1, 2, \ldots, K; m = 1, 2, ..., M\}$. Towards this goal, the traditional least-squares fitting method first collects a number of sampling points of $\mathbf{x}$ and $\{y_k; k = 1, 2, \cdots, K\}$ for all K states, and then solves the model coefficients from the following optimization problems:

$$\min_{\boldsymbol{\alpha}_k} \quad \left\| \mathbf{y}_k - \mathbf{B}_k \cdot \boldsymbol{\alpha}_k \right\|_2^2 \quad (k = 1, 2, \cdots, K) \tag{2.2}$$

where

$$\mathbf{B}_k = \begin{bmatrix} b_1\left(\mathbf{x}_k^{(1)}\right) & b_2\left(\mathbf{x}_k^{(1)}\right) & \cdots & b_M\left(\mathbf{x}_k^{(1)}\right) \\ b_1\left(\mathbf{x}_k^{(2)}\right) & b_2\left(\mathbf{x}_k^{(2)}\right) & \cdots & b_M\left(\mathbf{x}_k^{(2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ b_1\left(\mathbf{x}_k^{(N)}\right) & b_2\left(\mathbf{x}_k^{(N)}\right) & & b_M\left(\mathbf{x}_k^{(N)}\right) \end{bmatrix} \quad (k = 1, 2, \cdots, K) \tag{2.3}$$

$$\boldsymbol{\alpha}_k = \begin{bmatrix} \alpha_{k,1} & \alpha_{k,2} & \cdots & \alpha_{k,M} \end{bmatrix}^T \quad (k = 1, 2, \cdots, K) \tag{2.4}$$

$$\mathbf{y}_k = \begin{bmatrix} y_k^{(1)} & y_k^{(2)} & \cdots & y_k^{(N)} \end{bmatrix}^T \quad (k = 1, 2, \cdots, K). \tag{2.5}$$

In (2.3)-(2.5), $\mathbf{x}_k^{(n)}$ is the value of $\mathbf{x}$ at the n-th sampling point of the k-th state, $y_k^{(n)}$ is the value of y_k at the n-th sampling point, $\|\bullet\|_2$ stands for the L_2-norm of a vector, and N represents the total number of sampling points collected for each state. Here we assume that the same number of samples is collected for each state, since we consider all states to be equally important. To avoid over-fitting, the number of sampling points (i.e., N) must be substantially greater than the number of unknown coefficients (i.e., M).

To reduce the number of samples required for performance modeling, the simultaneous orthogonal matching pursuit (S-OMP) algorithm [78] has been proposed in the literature. The key idea of S-OMP is to explore the sparse property and the correlation of model template between different states. As an extension of the orthogonal matching pursuit (OMP) algorithm [75], S-OMP naturally inherits the assumption of sparse regression. In other words, S-OMP exploits the fact that only a small number of basis functions in $\{b_m(\mathbf{x}); m = 1, 2, ..., M\}$ are important and, hence, should be used to approximate $\{y_k; k = 1, 2, \cdots, K\}$. More

importantly, S-OMP also explores the correlation of model template among $\{y_k;\ k = 1, 2, \cdots, K\}$. In particular, S-OMP assumes that different states possess the same set of important basis functions. This is usually a valid assumption in practice, because different states are associated with the same self-healing circuit.

While S-OMP has been successfully applied to many practical applications (e.g. wafer spatial pattern analysis [78]), it is possible to explore other correlation information between states to further improve the efficiency of performance modeling for self-healing circuits. There is a strong need to develop a novel performance modeling approach that takes advantage of the correlation of coefficient magnitude. Once such performance model is built, the performance variation of self-healing circuits can be efficiently predicted, and thus the parametric yield can be estimated.

2.2 Prior Knowledge Definition

To start with, we consider the performance models of K states in a self-healing circuit as defined in (2.1). We rewrite their model coefficients in a concatenated form as:

$$\boldsymbol{\alpha} = \left[\boldsymbol{\beta}_1^{\,T} \quad \boldsymbol{\beta}_2^{\,T} \quad \cdots \quad \boldsymbol{\beta}_M^{\,T} \right]^T \tag{2.6}$$

where

$$\boldsymbol{\beta}_m = \left[\alpha_{1,m} \quad \alpha_{2,m} \quad \cdots \quad \alpha_{K,m} \right]^T \quad (m = 1, 2 \cdots, M) \tag{2.7}$$

$\{\alpha_{k,m};\ k = 1, 2, ..., K;\ m = 1, 2, ..., M\}$ are the model coefficients defined in (2.1), and $\boldsymbol{\alpha}$ is the combined column vector with size of $M \cdot K$. In (2.6), the model coefficients are organized such that the coefficients associated with the m-th basis function are grouped together in $\boldsymbol{\beta}_m$ (m = 1, 2, ..., M). Since the model coefficients in $\boldsymbol{\beta}_m$ are related to the same basis function, we assume that they are statistically correlated and define their prior distribution as:

$$pdf\left(\boldsymbol{\beta}_m\right) \sim N\left(\mathbf{0}, \lambda_m \cdot \mathbf{R}_m\right) \quad (m = 1, 2, \cdots, M). \tag{2.8}$$

In (2.8), $\{\lambda_m;\ m = 1, 2, ..., M\}$ are the hyper-parameters that control the sparsity of the basis functions, and $\{\mathbf{R}_m;\ m = 1, 2, ..., M\}$ are the positive definite matrices that quantify the correlation among model coefficients.

Studying (2.8) yields several important observations. First, the sparse property is naturally encoded in $\{\lambda_m;\ m = 1, 2, ..., M\}$. To understand this, we consider the m-th basis function where λ_m is zero. In this case, the variance of the zero-mean Gaussian distribution in (2.8) is also zero. As such, the coefficients $\boldsymbol{\beta}_m$ can only take zero values based on their prior distribution, thereby implying the sparsity of $\boldsymbol{\beta}_m$. Second, the correlation of model template is also encoded by the mathematical representation in (2.8). In particular, all the model coefficients in $\boldsymbol{\beta}_m$ share the same sparse pattern, because their prior distribution is controlled by the same hyper-parameter λ_m. Third, but most importantly, the correlation of coefficient magnitude is also encoded by this prior definition, as long as the covariance metrics $\{\mathbf{R}_m;\ m = 1, 2, ..., M\}$ are not diagonal. Here we further assume that:

$$\mathbf{R}_1 = \mathbf{R}_2 = \cdots \mathbf{R}_M = \mathbf{R} \tag{2.9}$$

in order to reduce the number of hyper-parameters and avoid overfitting.

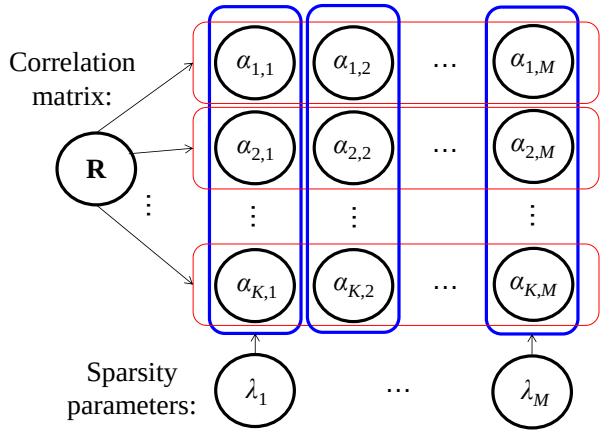


Figure 2-1. The prior knowledge for model coefficients is illustrated.

As shown in (2.1), the prior knowledge of sparsity and correlation is encoded by the hyper-parameters. To complete the definition of prior distribution for all model coefficients in $\boldsymbol{\alpha}$, we further assume that $\boldsymbol{\beta}_i$ and $\boldsymbol{\beta}_j$ are statistically independent for any $i \neq j$. Therefore, the joint distribution can be represented as:

$$pdf\left(\boldsymbol{\alpha}\right) \sim N\left(\mathbf{0}, \mathbf{A}\right) \tag{2.10}$$

where

$$\mathbf{A}=\begin{bmatrix}\lambda_1\cdot\mathbf{R} & & \\ & \ddots & \\ & & \lambda_M\cdot\mathbf{R}\end{bmatrix}. \tag{2.11}$$

The independence assumption in (2.10)-(2.11) simply indicates that we do not know any correlation information between different basis functions as our prior knowledge. Such correlation will be learned from sampling points, when the posterior distribution is calculated from Bayesian inference in Section 2.3.

2.3 Maximum-A-Posteriori Estimation

Once the prior distribution *pdf*(**α**) is defined in (2.10)-(2.11), we will combine *pdf*(**α**) with a number of samples collected from K states $\{(\mathbf{x}_k^{(n)}, y_k^{(n)})$; $n = 1, 2, \ldots, N$; $k = 1, 2, \ldots, K\}$ to solve the model coefficients **α** by maximum-a-posteriori (MAP) estimation. According to Bayes' theorem, the posterior distribution is proportional to the prior distribution *pdf*(**α**) multiplied by the likelihood function, which can be represented as [61]:

$$pdf\left(\boldsymbol{\alpha}\,|\,\mathbf{y}\right)\propto pdf\left(\boldsymbol{\alpha}\right)\cdot pdf\left(\mathbf{y}\,|\,\boldsymbol{\alpha}\right). \tag{2.12}$$

In (2.12), *pdf*(**y**|**α**) is the likelihood function, where

$$\mathbf{y}=\left[\,y_1^{(1)}\ \ \cdots\ \ y_K^{(1)}\ \ y_1^{(2)}\ \ \cdots\ \ y_K^{(2)}\ \ y_1^{(N)}\ \ \cdots\ \ y_K^{(N)}\,\right]^T \tag{2.13}$$

is a vector containing $N\cdot K$ sampling points collected from K states. The likelihood function *pdf*(**y**|**α**) represents the probability of observing these samples given the model coefficients **α**.

To derive the likelihood function, we assume that the error associated with the performance model $f_k(\mathbf{x})$ follows a zero-mean Gaussian distribution. We therefore rewrite (2.1) as:

$$y_k=\sum_{m=1}^{M}\alpha_{k,m}\cdot b_m\left(\mathbf{x}\right)+\varepsilon\quad\left(k=1,2,\cdots,K\right) \tag{2.14}$$

where ε represents the modeling error with the following distribution:

$$pdf\left(\varepsilon\right)=\frac{1}{\sqrt{2\pi}\cdot\sigma_0}\cdot\exp\left(-\frac{\varepsilon^2}{2\cdot\sigma_0^2}\right)\sim N\left(0,\sigma_0^2\right). \tag{2.15}$$

In (2.15) the standard deviation σ_0 controls the magnitude of the modeling error. Combining (2.14)-(2.15) yields the probability of observing a particular sampling point $(\mathbf{x}_k^{(n)}, y_k^{(n)})$:

$$pdf\left(y_k^{(n)}|\boldsymbol{\alpha}\right)\sim\frac{1}{\sqrt{2\pi}\cdot\sigma_0}\cdot\exp\left\{-\frac{1}{2\cdot\sigma_0^2}\cdot\left[y_k^{(n)}-\sum_{m=1}^{M}\alpha_{k,m}\cdot b_m\left(\mathbf{x}_k^{(n)}\right)\right]^2\right\}. \tag{2.16}$$

Given that all sampling points are independently generated, the joint likelihood function $pdf(\mathbf{y}|\boldsymbol{\alpha})$ can be written as:

$$pdf\left(\mathbf{y}\,|\,\boldsymbol{\alpha},\sigma_0\right)\sim N\left(\mathbf{D}\boldsymbol{\alpha},\sigma_0^2\mathbf{I}\right) \tag{2.17}$$

where $\mathbf{D}$ is a matrix with $N\cdot K$ rows and $M\cdot K$ columns. The matrix $\mathbf{D}$ can be obtained by permuting the rows and columns of the following block diagonal matrix:

$$\begin{bmatrix}\mathbf{B}_1 & & \\ & \ddots & \\ & & \mathbf{B}_K\end{bmatrix} \tag{2.18}$$

where $\{\mathbf{B}_k;\ k = 1, 2, \cdots, K\}$ are defined in (2.3). Combining (2.10), (2.12) and (2.17), the posterior distribution can be calculated as [61]:

$$pdf\left(\boldsymbol{\alpha}\,|\,\mathbf{y}\right)\sim N\left(\boldsymbol{\mu}_P,\boldsymbol{\Sigma}_p\right) \tag{2.19}$$

where

$$\boldsymbol{\Sigma}_p=\mathbf{A}-\mathbf{A}\mathbf{D}^T\left(\sigma_0^2\mathbf{I}+\mathbf{D}\mathbf{A}\mathbf{D}^T\right)^{-1}\mathbf{D}\mathbf{A} \tag{2.20}$$

$$\boldsymbol{\mu}_P=\sigma_0^{-2}\boldsymbol{\Sigma}_p\mathbf{D}^T\mathbf{y}\,. \tag{2.21}$$

The MAP solution of the model coefficients $\boldsymbol{\alpha}$ is:

$$\boldsymbol{\alpha}=\boldsymbol{\mu}_P=\sigma_0^{-2}\boldsymbol{\Sigma}_p\mathbf{D}^T\mathbf{y}\,. \tag{2.22}$$

While the proposed Bayesian inference has been illustrated, the hyper-parameters in the prior distribution (2.10) and the likelihood function (2.17) must be appropriately determined in order to solve the model coefficients $\boldsymbol{\alpha}$. In the next section, we will further discuss a statistical inference to find the optimal hyper-parameter values.

2.4 Hyper-Parameter Inference

The hyper-parameters of our proposed Bayesian inference are defined in Section 2.2 and Section 2.3: $\Omega = \{\lambda_1, \lambda_2, \ldots, \lambda_M, \mathbf{R}, \sigma_0\}$. To determine these hyper-parameters, one possible approach is to apply

cross-validation [61] which has been successfully used for various applications [74]-[79]. However, the cross-validation approach is not applicable to our Bayesian inference here. It is well-known that the computational complexity of cross-validation exponentially grows with the number of hyper-parameters, and thus is only suitable to handle small-size problems. Our Bayesian inference involves a large number of hyper-parameters, since the number of basis functions (i.e., M) can easily reach several thousand due to the large number of device-level process parameters at an advanced technology node. Hence, an alternative approach must be developed to efficiently determine these hyper-parameters in Ω.

Statistically, the hyper-parameters can be determined by maximizing the conditional probability of observing the data set, i.e., the marginal likelihood:

$$l(\Omega) = pdf(\mathbf{y} \mid \Omega) = \int pdf(\mathbf{y} \mid \boldsymbol{\alpha}, \Omega)\, pdf(\boldsymbol{\alpha} \mid \Omega) \cdot d\boldsymbol{\alpha}\,. \tag{2.23}$$

The integration in indicates that the effect of model coefficients $\boldsymbol{\alpha}$ is averaged out in the marginal likelihood. Given that the two terms in the integration depend on different sets of hyper-parameters in Ω, we can rewrite (2.23) as:

$$l(\Omega) = \int pdf(\mathbf{y} \mid \boldsymbol{\alpha}, \sigma_0)\, pdf(\boldsymbol{\alpha} \mid \lambda_1, ..., \lambda_M, \mathbf{R}) \cdot d\boldsymbol{\alpha}\,. \tag{2.24}$$

By combining (2.10)-(2.11) and (2.17) and taking the negative logarithm for (2.24), we have:

$$L(\Omega) = \mathbf{y}^T \left(\sigma_0^2 \mathbf{I} + \mathbf{D}\mathbf{A}\mathbf{D}^T\right)^{-1} \mathbf{y} + \log\left|\sigma_0^2 \mathbf{I} + \mathbf{D}\mathbf{A}\mathbf{D}^T\right|. \tag{2.25}$$

The hyper-parameters can be found by minimizing the cost function in (2.25). However, minimizing (2.25) is not trivial, because the cost function is not convex. Here we adopt an expectation-maximization (EM) algorithm to address this challenge. In particular, instead of directly minimizing (2.25), the EM algorithm approaches a local optimum of (2.25) by iteratively performing two operations [61], [83], known as the expectation step and the maximization step respectively.

In the expectation step, we first calculate the posterior distribution $pdf(\boldsymbol{\alpha}|\mathbf{y}, \Omega^{(\text{old})})$ according to (2.19)-(2.21) based on $\Omega^{(\text{old})}$, where $\Omega^{(\text{old})}$ denotes the hyper-parameters calculated in the last iteration. Next, we define the following quantity [61]:

$$Q\left(\Omega \mid \Omega^{(old)}\right) = \mathbf{E}_{\boldsymbol{\alpha}|\mathbf{y},\Omega^{(old)}}\left[\log p(\mathbf{y}, \boldsymbol{\alpha} \mid \Omega^{(old)})\right] \tag{2.26}$$

where the operator $\mathbf{E}$ represents the expectation of the log probability function with respect to the posterior distribution $pdf(\boldsymbol{\alpha}|\mathbf{y}, \Omega^{(\text{old})})$. Note that the expected value in (2.26) depends on the mean vector $\boldsymbol{\mu}_p$ and the

covariance matrix $\boldsymbol{\Sigma}_p$ of the posterior distribution in (2.19).

In the maximization step, we try to find Ω such that:

$$\Omega = \arg\max_{\Omega} Q\left(\Omega \mid \Omega^{(old)}\right). \tag{2.27}$$

The optimal solution of Ω corresponds to:

$$\frac{\partial}{\partial \Omega} Q\left(\Omega \mid \Omega^{(old)}\right) = 0. \tag{2.28}$$

Combining (2.26) and (2.28), we can find the optimal solution Ω of (2.27) as a function of $\Omega^{(\text{old})}$, resulting in the following rule for updating Ω [83]:

$$\lambda_m \leftarrow \frac{Tr\left[\mathbf{R}^{-1}\left(\boldsymbol{\Sigma}_p^m + \boldsymbol{\mu}_p^m \left(\boldsymbol{\mu}_p^m\right)^T\right)\right]}{K} \quad \left(m = 1, 2, \cdots, M\right) \tag{2.29}$$

$$\mathbf{R} \leftarrow \frac{1}{M}\sum_{m=1}^{M} \frac{\boldsymbol{\Sigma}_p^m + \boldsymbol{\mu}_p^m \left(\boldsymbol{\mu}_p^m\right)^T}{\lambda_m} \quad \left(m = 1, 2, \cdots, M\right) \tag{2.30}$$

$$\sigma_0 \leftarrow \sqrt{\frac{\left\|\mathbf{y} - \mathbf{D}\boldsymbol{\mu}_p\right\|_2^2 + Tr\left(\boldsymbol{\Sigma}_p \mathbf{D}^T \mathbf{D}\right)}{N}} \quad \left(m = 1, 2, \cdots, M\right) \tag{2.31}$$

where $\boldsymbol{\mu}_p{}^m$ corresponds to the m-th sub-vector in $\boldsymbol{\mu}_p$ with size of K, $\boldsymbol{\Sigma}_p{}^m$ corresponds to the m-th diagonal block in $\boldsymbol{\Sigma}_p$ with size of $K \times K$. The expectation step (2.19)-(2.21) and the maximization step (2.29)-(2.31) are performed iteratively until convergence is reached.

Given that the EM algorithm only converges to a local optimum, the initial guess of Ω must be carefully chosen. Here we propose a modified S-OMP algorithm to achieve this goal. The key idea of the proposed algorithm is to (i) reduce the hyper-parameter space by posing additional constraints on $\mathbf{R}$, (ii) apply a greedy algorithm for basis function selection, and (iii) apply cross-validation to solve this simplified problem within the reduced hyper-parameter space.

Following this idea, we first consider the parameterized covariance matrix $\mathbf{R}$ defined as:

$$\mathbf{R} = \begin{bmatrix} 1 & r_0 & \cdots & r_0^{K-1} \\ r_0 & 1 & & \\ & & \ddots & r_0 \\ r_0^{K-1} & & r_0 & 1 \end{bmatrix} \tag{2.32}$$

where $r_0 < 1$ is a non-negative hyper-parameter. The matrix in (2.32) has several implications. First, the

correlation becomes small when two states (i.e., their indexes) are substantially different. Second, the same decaying rate r_0 is used for all the states. This assumption often provides a good approximation, even though it is not highly accurate. However, since our goal here is to find a good initial guess of Ω, a reasonably accurate approximation should suffice. As the covariance matrix $\mathbf{R}$ depends on a single parameter r_0 in (2.32), we can now apply cross-validation to estimate its value with low computational cost. On the other hand, a greedy basis selection algorithm based on S-OMP [78] is used to infer the sparsity information that is encoded by the hyper-parameters $\{\lambda_1, \lambda_2, \ldots, \lambda_M\}$. In this way, we do not have to directly estimate $\{\lambda_1, \lambda_2, \ldots, \lambda_M\}$, which is computationally expensive.

At this point, the hyper-parameter space has been reduced to three variables: r_0, σ_0 and η, where η denotes the number of selected basis functions. These three hyper-parameters can be efficiently determined using cross-validation [61]. Namely, we start from a candidate set $\{(r_0^{(q)}, \sigma_0^{(q)}, \eta^{(q)}); q = 1, 2, \ldots, \omega\}$, where ω denotes the size of the set. At the q-th iteration step, we choose the hyper-parameter values $(r_0^{(q)}, \sigma_0^{(q)}, \eta^{(q)})$, estimate the model coefficients, and evaluate the modeling error. To solve the model coefficients with given $(r_0^{(q)}, \sigma_0^{(q)}, \eta^{(q)})$, we adopt a modified S-OMP algorithm to iteratively identify the important basis functions. Similar to S-OMP [78], a single most important basis function is greedily chosen at each iteration step by maximizing the correlation between the selected basis function and the modeling residual over all states. The model coefficients are then calculated from the Bayesian inference in (2.11), (2.20)-(2.22) and (2.32) based on $r_0^{(q)}$, $\sigma_0^{(q)}$ and the currently selected basis functions. Different from the traditional S-OMP method, we take into account the correlation of coefficient magnitude when solving the unknown model coefficients. The aforementioned iteration continues until the number of chosen basis functions reaches $\eta^{(q)}$. The modeling error can now be evaluated by cross-validation for the hyper-parameters $(r_0^{(q)}, \sigma_0^{(q)}, \eta^{(q)})$. Eventually, the optimal values of (r_0, σ_0, η) are chosen to minimize the modeling error, and they are used to initialize the hyper-parameters Ω for the iterative EM algorithm.

2.5 Algorithm Summary

Algorithm 1 summarizes the major steps of the correlated Bayesian model fusion (C-BMF) algorithm. It consists of two main components: (i) a modified S-OMP algorithm from Step 1 to Step 17 to

find the initial guess of the hyper-parameters Ω, and (ii) the EM algorithm from Step 18 to Step 20 to iteratively solve the optimal values of Ω where the model coefficients are updated at each iteration step.

Algorithm 1: Correlated Bayesian Model Fusion Algorithm

1. Start from a candidate set of hyper-parameters $\{(r_0^{(q)}, \sigma_0^{(q)}, \eta^{(q)}); q = 1, 2, \ldots, \omega\}$. Partition the sampling points into C groups with the same size for cross-validation.
2. Set the candidate index $q = 1$.
3. Set the cross-validation index $c = 1$.
4. Assign the c-th group as the testing set and all other groups as the training set.
5. Initialize the residual $Res_k = \mathbf{y}_k$, the basis vector set $\Theta = \{\}$, and the iteration index $p = 1$.
6. Calculate the inner product values $\{\xi_{k,m}; k = 1, 2, \ldots, K; m = 1, 2, \ldots, M\}$ between Res_k and all basis vectors $\mathbf{b}_{k,m}$ for each state, where $\mathbf{b}_{k,m}$ represents the m-th column of $\mathbf{B}_k$ in (2.3).
7. Select the index s based on the following criterion:

$$s = \arg\max_m \sum_{k=1}^{K} \left| \xi_{k,m} \right| . \tag{2.33}$$

8. Update $\Theta = \Theta \cup s$.
9. By considering the basis functions in Θ and $\lambda_m = 1$ where $m \in \Theta$, solve the model coefficients $\{\alpha_{k,m}; k = 1, 2, \ldots, K; m \in \Theta\}$ using (2.11), (2.20)-(2.22) with $r_0^{(q)}$, $\sigma_0^{(q)}$ and $\mathbf{R}$ defined in (2.32).
10. Calculate the residual:

$$Res_k = \mathbf{y}_k - \sum_{m \in \Theta} \alpha_{k,m} \cdot \mathbf{B}_{k,m} \quad (k = 1, 2, \cdots, K) . \tag{2.34}$$

11. If $p < \eta^{(q)}$, $p = p + 1$ and go to Step 6.
12. Calculate the modeling error $e_{q,c}$ using the testing set.
13. If $c < C$, $c = c + 1$ and go to Step 4.
14. Calculate $e_q = (e_{q,1} + e_{q,2} + ... + e_{q,C})/C$.
15. If $q < \omega$, $q = q + 1$ and go to Step 3.
16. Find the optimal r_0, σ_0, and η among the set $\{(r_0^{(q)}, \sigma_0^{(q)}, \eta^{(q)}); q = 1, 2, \ldots, \omega\}$ such that the modeling error e_q is minimized.
17. Initialize the matrix $\mathbf{R}$ using (2.32), $\lambda_m = 1$ where $m \in \Theta$, and $\lambda_m = 10^{-5}$ where $m \notin \Theta$.
18. Calculate $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ based on the expectation step using (2.19)-(2.21).

19. Update $\{\lambda_m; m = 1, 2, \ldots, M\}$, $\mathbf{R}$ and σ_0 based on the maximization step using (2.29)-(2.31).
20. If convergence is not reached, go to Step 18. Otherwise, stop iteration and calculate the model coefficients $\boldsymbol{\alpha}$ by using (2.22).

2.6 Performance Modeling Results

In this section, two circuit examples designed in a commercial 32nm CMOS process are used to demonstrate the efficacy of the proposed C-BMF algorithm. Our objective is to build the performance models for tunable circuits. For testing and comparison purposes, two different modeling algorithms are implemented: (i) S-OMP [78], and (ii) C-BMF. Here, S-OMP is chosen for comparison, as it is one of the state-of-the-art techniques in the literature.

In each example, a set of random samples are generated by transistor-level Monte Carlo simulations. The data set is partitioned into two groups, referred to as the training and testing sets respectively. The training set is used for coefficient fitting, while the testing set is used for model validation. All numerical experiments are run on a 2.53GHz Linux server with 64GB memory.

2.6.1 Low-Noise Amplifier

Figure 2-2 shows the simplified circuit schematic of a tunable low-noise amplifier (LNA) designed in a commercial 32nm CMOS process. In this example, there are totally 1264 independent random variables to model the device-level process variations, including both the inter-die variations and the random mismatches. The LNA is designed with 32 different knob configurations (i.e., states) controlled by a tunable current source. Our objective is to model three performance metrics, noise figure (NF), voltage gain (VG) and third-order intercept point (IIP3), as linear functions of the 1264 random variables for all 32 states. The modeling error is estimated by using a testing set with 50 samples per state.

Figure 2-3(a)-(c) show the performance modeling error for NF, VG and IIP3 respectively. Two important observations can be made here. First, for both S-OMP and C-BMF, the modeling error decreases when the number of samples increases. Second, with the same number of samples, C-BMF achieves significantly higher accuracy than S-OMP, because C-BMF takes into account the additional correlation

information of coefficient magnitude.

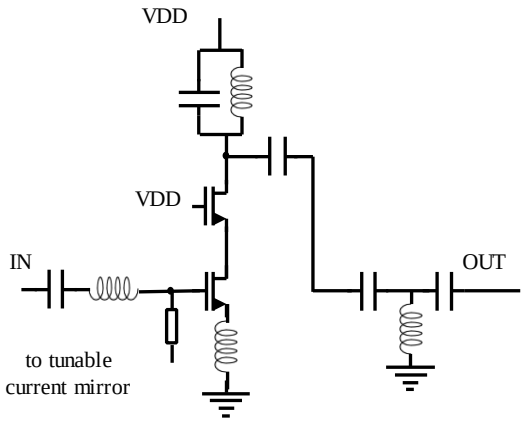


Figure 2-2. The schematic of a tunable LNA designed in a commercial 32nm CMOS process is shown.

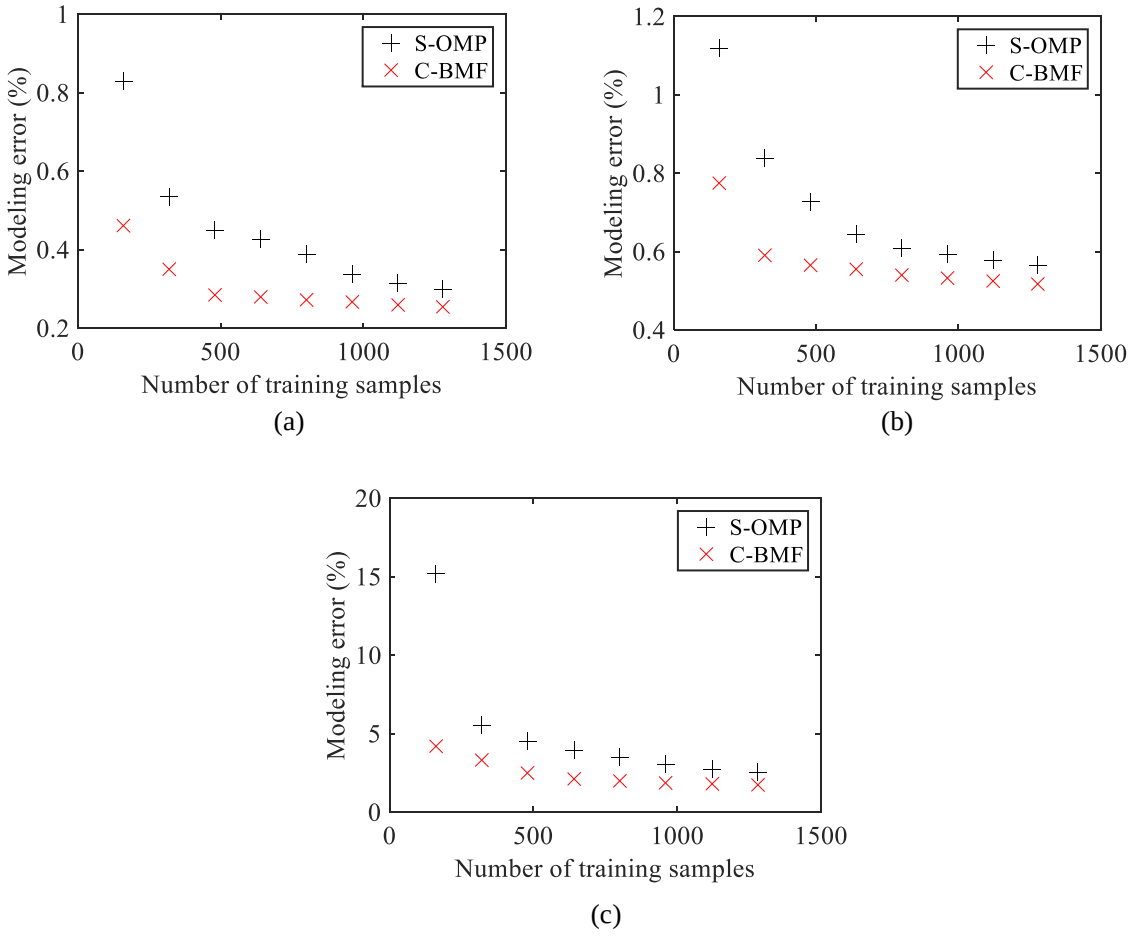


Figure 2-3. The performance modeling error for NF, VG, and IIP3 are compared for (i) S-OMP and (ii) C-BMF method.

Table 2-1 further compares the performance modeling error and cost for S-OMP and C-BMF. The overall modeling cost consists of two major components: (i) the circuit simulation cost for collecting training samples, and (ii) the model fitting cost for solving all model coefficients. As shown in Table 2-1, the overall modeling cost is dominated by the simulation cost in this example. C-BMF achieves more than 2× cost reduction over S-OMP without surrendering any accuracy.

Table 2-1. Performance modeling error and cost for LNA

Example	S-OMP	C-BMF
Number of training samples	1120	480
Modeling error for NF	0.316%	0.285%
Modeling error for VG	0.577%	0.566%
Modeling error for IIP3	2.738%	2.497%
Simulation cost (Hours)	2.72	1.16
Fitting cost (Sec.)	1.32	316.13
Overall modeling cost (Hours)	2.72	1.25

2.6.2 Down-conversion Mixer

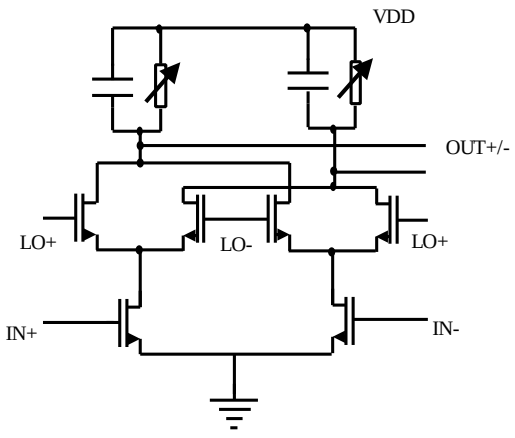


Figure 2-4. The schematic of a tunable Mixer designed in a commercial 32nm CMOS process is shown.

Table 2-2. Performance modeling error and cost for down-conversion mixer

	S-OMP	C-BMF
Number of samples	1120	480
Modeling error for NF	0.173%	0.166%
Modeling error for VG	2.758%	2.569%
Modeling error for I1dBCP	2.401%	2.340%
Simulation cost (Hours)	17.20	7.37
Fitting cost (Sec.)	1.39	407.10
Overall modeling cost (Hours)	17.20	7.48

Shown in Figure 2-4 is the simplified circuit schematic of a down-conversion mixer designed in a commercial 32nm CMOS process. In this example, there are totally 1303 independent random variables to model the device-level process variations, including both the inter-die variations and the random mismatches. The mixer is designed with 32 different states controlled by two tunable load resistors. The performances of interest include NF, VG, and input referred 1dB compression point (I1dBCP). The modeling error is estimated by using a test set with 50 samples per state.

Figure 2-5(a)-(c) show the performance modeling error for NF, VG and I1dBCP respectively. Note that C-BMF requires substantially less training samples than S-OMP to achieve the same modeling accuracy. In this example, C-BMF achieves more than 2× cost reduction over the traditional S-OMP approach, as shown in Table 2-2.

2.7 Parametric Yield Estimation of Self-Healing Circuits

The main goal of pre-silicon validation is the parametric yield estimation. In earlier sections of this Chapter, we proposed an efficient performance modeling algorithm for self-healing circuit. In this section, we present the parametric yield estimation flow based on performance modeling. To start with, we consider

the traditional circuit without control knobs, the performance constraints can be expressed as the following standard form:

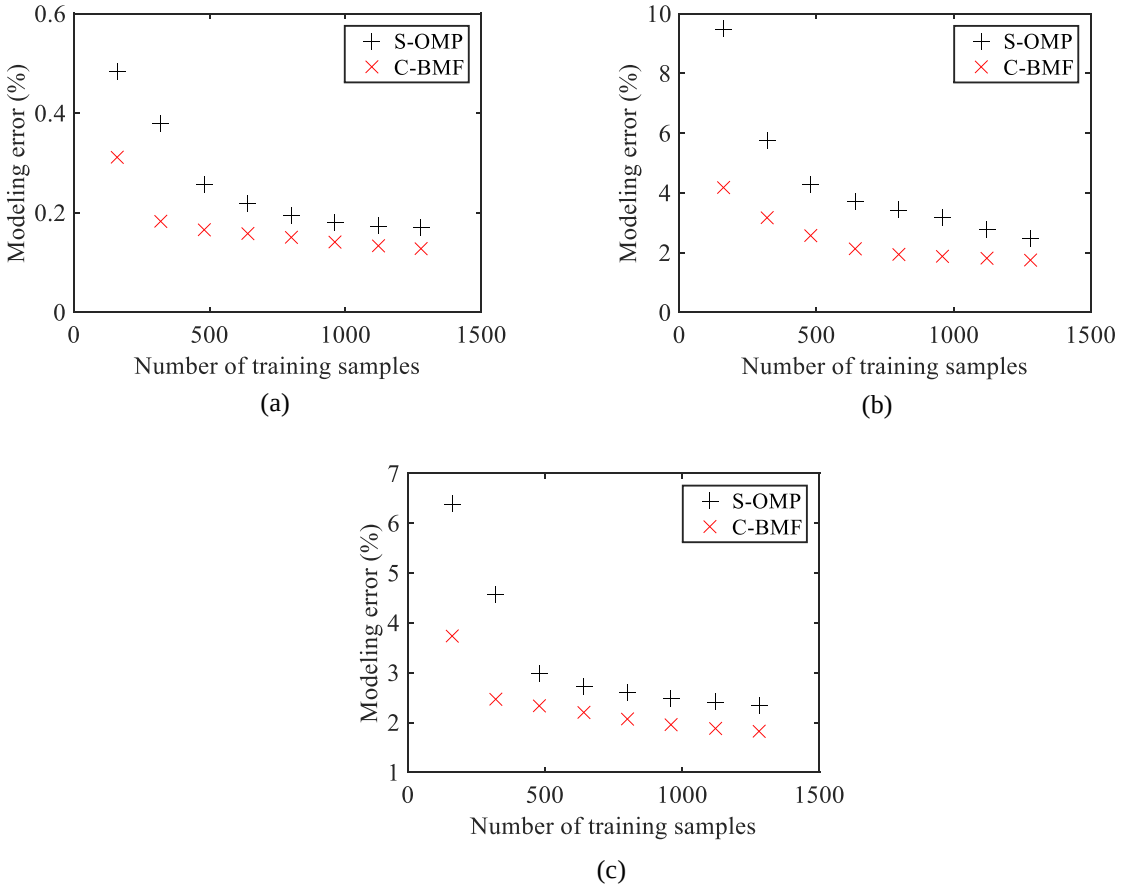


Figure 2-5. The performance modeling error for NF, VG, and I1dBCP are compared for (i) S-OMP and (ii) C-BMF method.

$$\left(f_1 \geq s_1\right) \wedge \left(f_2 \geq s_2\right) \wedge \cdots \wedge \left(f_R \geq s_R\right) \tag{2.35}$$

where $\{y_r;\ r = 1, 2, \cdots, R\}$ contains R performances with specifications $\{s_r;\ r = 1, 2, \cdots, R\}$. The parametric yield is the probability that all performances constraints are satisfied, which can be defined as:

$$Yield = P\left[\left(f_1 \geq s_1\right) \wedge \left(f_2 \geq s_2\right) \wedge \cdots \wedge \left(f_R \geq s_R\right)\right]. \tag{2.36}$$

The parametric yield definition of self-healing circuits takes a more complicated form because of the control knobs. The pass/fail of a self-healing chip is determined by the pass/fail of a set of control knobs as well as the self-healing algorithm. The performance constraints of self-healing circuits can be expressed as:

$$\left(f_{1,k(A,p)} \geq s_1\right) \wedge \left(f_{2,k(A,p)} \geq s_2\right) \wedge \cdots \wedge \left(f_{R,k(A,p)} \geq s_R\right) \tag{2.37}$$

and the parametric yield of self-healing circuits is defined accordingly as:

$$Yield = P\left[\left(f_{1,k(A,p)} \geq s_1\right) \wedge \left(f_{2,k(A,p)} \geq s_2\right) \wedge \cdots \wedge \left(f_{R,k(A,p)} \geq s_R\right)\right] \tag{2.38}$$

where $f_{r,k(A,p)}$ denotes the r-th performance value corresponding to the selected control knob configuration $k(A, p)$. Here notation $k(A, p)$ means that the selected control knob is a function of process corner (i.e. p) as well as self-healing algorithm (i.e. A). In other words, the control knob is adaptively selected according to the process corner and self-healing algorithm. Given a particular self-healing algorithm (e.g. exhaustively searching all knob configurations), $k(A, p)$ can be dependent on all possible knob configurations for this particular process corner p. Given the versatile feature of $k(A, p)$, we can imagine that the yield in (2.38) is very difficult to estimate.

The parametric yield of self-healing circuit in (2.38) can be estimated by a Monte Carlo method. By independently simulating N chips at N different process corners, the parametric yield can be expressed as:

$$Yield = \frac{N_{pass}}{N} = \frac{\sum_{n=1}^{N} Y_n}{N} \tag{2.39}$$

where

$$Y_n = \begin{cases} 1 & \text{if} \quad \left(f_{1,k(A,p_n),n} \geq s_1\right) \wedge \left(f_{2,k(A,p_n),n} \geq s_2\right) \wedge \cdots \wedge \left(f_{R,k(A,p_n),n} \geq s_R\right), \\ 0 & \text{otherwise} \end{cases} \tag{2.40}$$

N_{pass} denotes the number of chips with at least one control knob configuration passing all specifications, $\{Y_n;\ n = 1, 2, \cdots, N\}$ contains the binary variables indicating pass/fail of chips, $\{k(A, p_n);\ n = 1, 2, \cdots, N\}$ denotes the final knob configurations selected by the self-healing algorithm A at process corner p_n (i.e. chip n), and $f_{r,\ k(A,\ pn),n}$ represents the r-th performance associated with final knob selection in the n-th chip.

The traditional parametric yield estimation flow of self-healing circuit is shown in Figure 2-6. The flow begins with the indirect sensor model construction from a number of Monte Carlo samples based on random sampling. The indirect sensor model provides a mapping from PoM to PoI. Then, a number of chips are randomly generated in simulation. Given that self-healing algorithm is usually based on indirect sensor model prediction, all PoMs from all knob configurations are required to determine the final knob selection. This leads to totally $N \times K$ simulations for PoM. Once all PoM values are collected, the final knob

selection can be determined for all chips. The PoI associated with this particular knob configuration can then be simulated, and the pass/fail of the chip can be determined by (2.40). The parametric yield of the self-healing circuit can then be determined by (2.39).

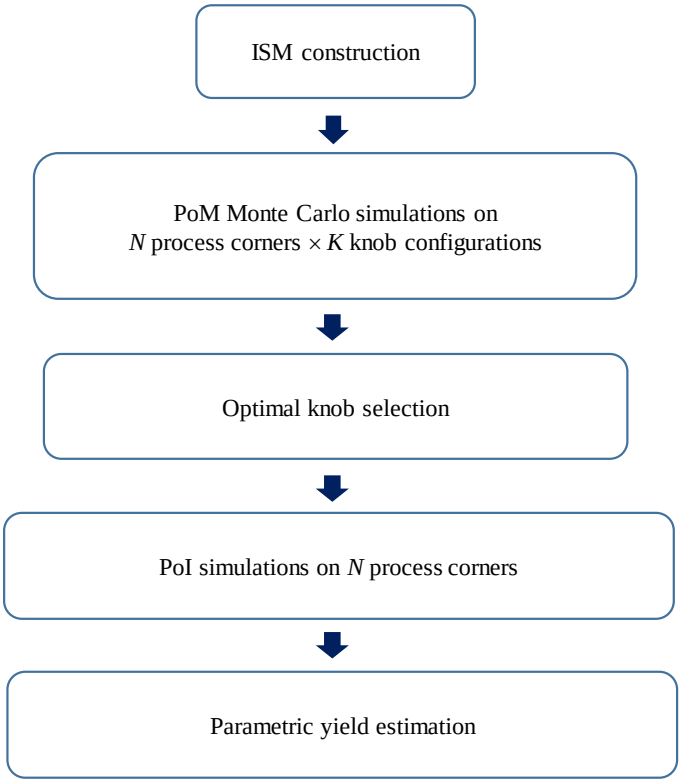


Figure 2-6. Traditional parametric yield estimation flow of self-healing circuit.

To efficiently estimate the parametric yield, we propose the flow in Figure 2-7. The main different between the proposed flow and the traditional flow is the PoM samples generation. In particular, instead of simulating PoMs in all chips, we first collect a number of samples to build performance models. Based on performance model, we generate PoM pseudo samples. As such, a significantly less number of samples are needed to find the optimal knob configurations for each chip. The parametric yield can then be efficiently estimated for self-healing circuit.

2.8 Yield Estimation Results

In this section, two circuit examples designed in a commercial 32nm CMOS process are used to demonstrate the efficacy of the proposed parametric yield estimation algorithm.

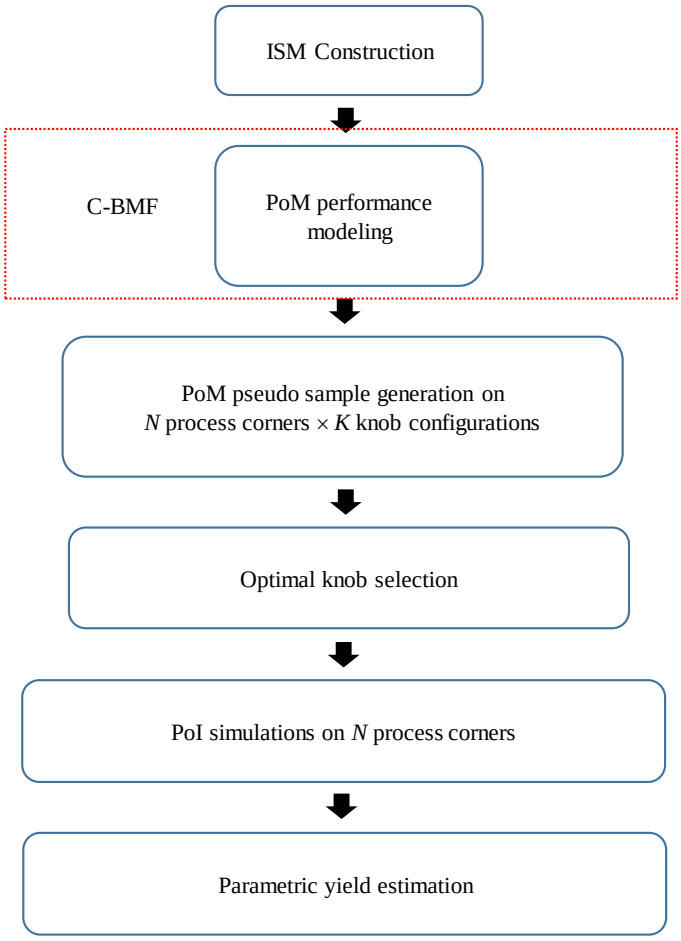


Figure 2-7. Proposed parametric yield estimation flow of self-healing circuit.

2.8.1 Low-Noise Amplifier

Show in Figure 2-8 is the schematic of a self-healing LNA. The performances of interest include noise figure (NF), voltage gain (VG) and third-order intercept point (IIP3). The performances of measurements include DC voltage of the input transistor, DC current of the LNA and the peak detector output. Cubic indirect sensor models are used to predict circuit performances of interest. More details about indirect sensor model fitting will be discussed in Chapter 4. The self-healing LNA possesses 32 different knob configurations enabled by tunable current mirror.

To demonstrate the efficacy of the proposed approach, we collect simulation data from 500 chips generated by Monte Carlo random sampling. We compare three different approaches: (i) traditional yield estimation approach as illustrated in Figure 2-6 (Traditional), (ii) performance modeling based yield

estimation approach in Figure 2-7 using S-OMP (S-OMP based yield estimation), and (iii) performance modeling based yield estimation approach in Figure 2-7 using C-BMF (C-BMF based yield estimation). The only difference between (ii) and (iii) is the difference in the performance modeling algorithm for PoM. To assess the robustness of the three approaches, we use bootstrapping [61] to repeatedly draw samples for 200 times and estimate the confidence interval. The bar in Figure 2-6 shows the 1% and 99% confidence interval. The x-axis shows the overall simulation time which considers all steps in Figure 2-6 or Figure 2-7.

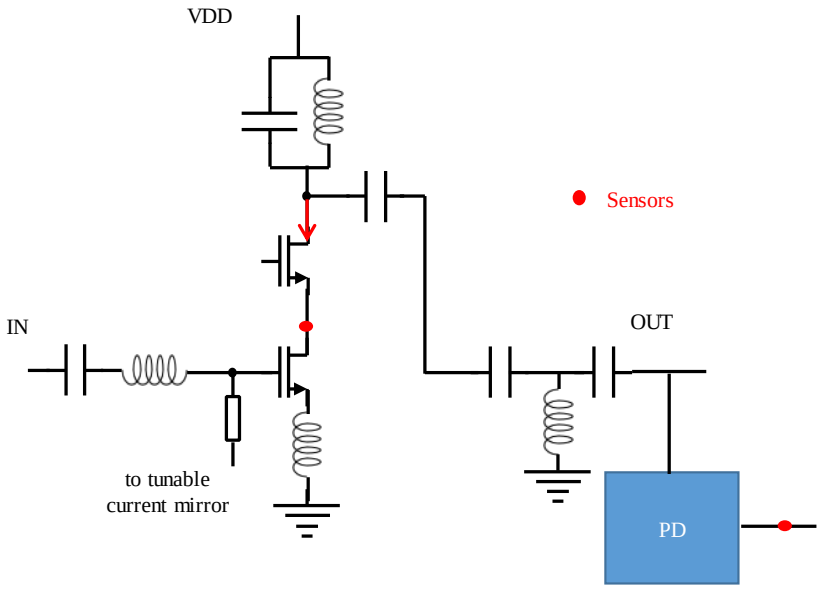


Figure 2-8. Schematic of a self-healing LNA.

Observing Figure 2-9 yields several important observations. First, the confidence interval shrinks with the increased number of samples collected for all three approaches. Second, the performance modeling based approaches requires significantly less samples than traditional method to achieve similar confidence interval. Third, the S-OMP approach is biased from the golden value, due to limited accuracy in PoM prediction. Fourth, the C-BMF approach is able to provide unbiased estimation with small confidence interval using only a small number of samples. Based on all the simulation samples, the golden parametric yield is 73.8%. Table 2-3 further compares the simulation time cost comparison between the traditional approaches and the proposed C-BMF based approach. To achieve a similar confidence interval, the proposed approach requires 5.8× less samples than the traditional approach.

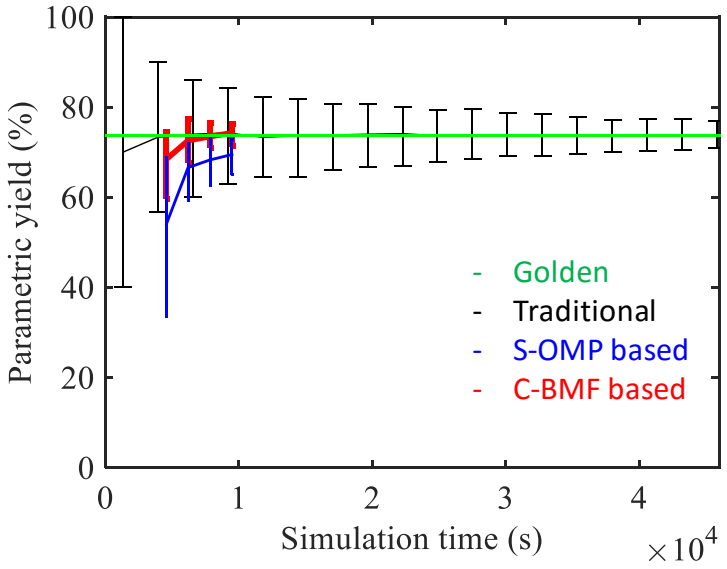


Figure 2-9. Comparison of three parametric yield estimation approaches: (i) Traditional, (ii) S-OMP based yield estimation, and (iii) C-BMF based yield estimation.

Table 2-3. Confidence interval comparison between proposed and traditional approaches

	C-BMF based yield estimation	Traditional
Confidence interval of yield (%)	[73.5-2.6, 73.5+2.9]	[73.7-2.9, 73.7+3.1]
Simulation time (s)	7843	45850

2.8.2 Down-Conversion Mixer

Show in Figure 2-10 is the schematic of a self-healing down-conversion mixer. The performances of interest include noise figure (NF), voltage gain (VG) and input referred 1dB compression point (I1dBCP). The performances of measurements include DC voltage of the input transistor, DC current of the mixer and the peak detector output. Cubic indirect sensor models are used to predict circuit performances of interest. The self-healing mixer possesses 32 different knob configurations enabled by tunable resistor.

To demonstrate the efficacy of the proposed approach, we collect simulation data from 250 chips generated by Monte Carlo random sampling. We compare three different approaches: (i) traditional yield estimation approach as illustrated in Figure 2-6 (Traditional), (ii) performance modeling based yield

estimation approach in Figure 2-7 using S-OMP (S-OMP based yield estimation), and (iii) performance modeling based yield estimation approach in Figure 2-7 using C-BMF (C-BMF based yield estimation). To assess the robustness of the three approaches, we use bootstrapping to repeatedly draw samples for 200 times and estimate the confidence interval. The bar in Figure 2-6 shows the 1% and 99% confidence interval. The x-axis shows the overall simulation time which considers all steps in Figure 2-6 or Figure 2-7.

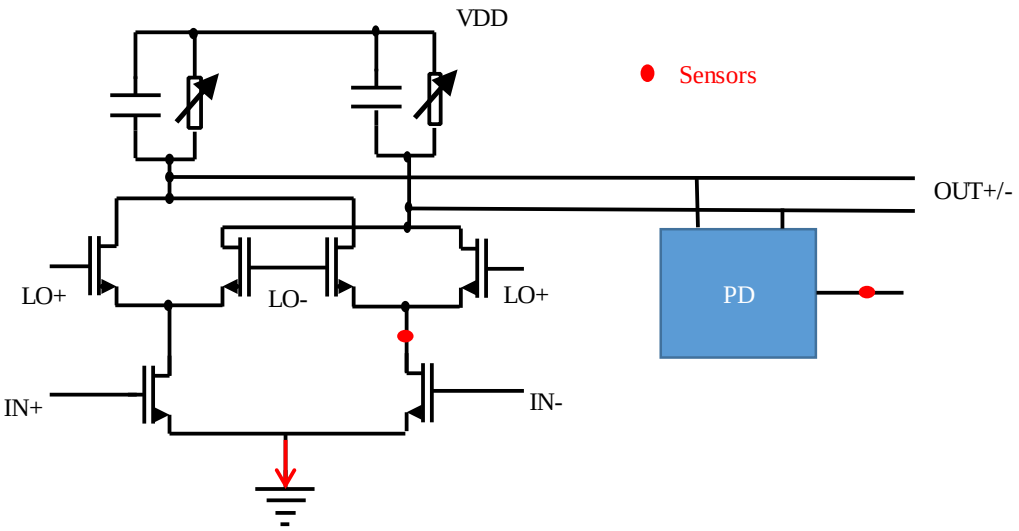


Figure 2-10. Schematic of a self-healing down-conversion mixer.

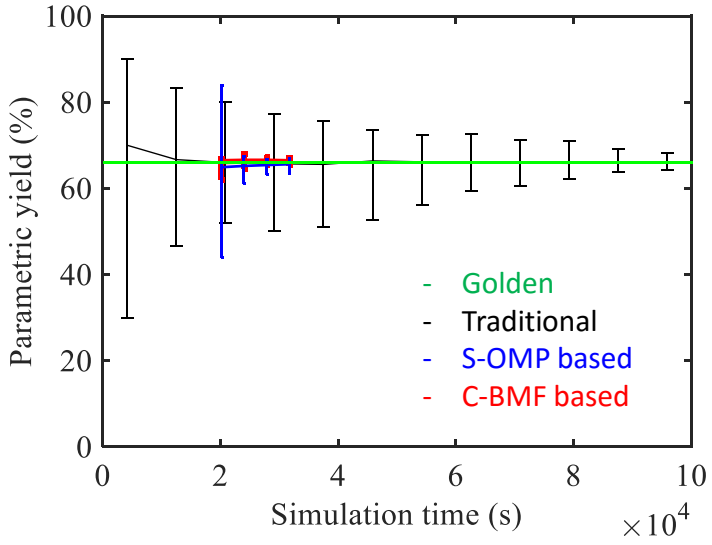


Figure 2-11. Comparison of three parametric yield estimation approaches: (i) Traditional, (ii) S-OMP based yield estimation, and (iii) C-BMF based yield estimation.

Observing Figure 2-11 yields several important observations. First, the confidence interval shrinks with the increased number of samples collected for all three approaches. Second, the performance modeling based approaches requires significantly less samples than traditional method to achieve similar confidence interval. Third, the C-BMF approach is able to provide accurate estimation with small confidence interval using only a small number of samples. Based on all the simulation samples, the golden parametric yield is 66.0%. Table 2-4 further compares the simulation time cost between the proposed and traditional approaches. To achieve a similar confidence interval, the proposed approach requires 3× less samples than the traditional approach.

Table 2-4. Confidence interval comparison between proposed and traditional approaches

	C-BMF based yield estimation	Traditional
Confidence interval of yield (%)	[66.2-0.9, 66.2+0.9]	[66.1-2.2, 66.1+1.7]
Simulation time (s)	31690	95910

2.9 Summary

In this Chapter, a novel C-BMF algorithm is proposed for efficient performance modeling of self-healing analog/RF IC. C-BMF encodes the correlation information for both model template and coefficient magnitude among different states in a prior distribution. Next, the prior distribution is combined with very few simulation samples to accurately solve the model coefficients. An efficient and robust hyper-parameter inference approach is also proposed to handle the large number of hyper-parameters involved in the Bayesian inference. As is demonstrated by two circuit examples designed in a commercial 32nm CMOS process, the proposed C-BMF method achieves more than 2× cost reduction compared to the state-of-the-art modeling technique.

Furthermore, an efficient pre-silicon validation approach is proposed where the correlated performance modeling algorithm serves as a core component. Instead of simulating all the knob configurations from all chips, we generate pseudo samples based on performance model. As such, the parametric yield estimation cost can be significantly reduced. Numerical results of a self-healing LNA and

a self-healing down-conversion mixer demonstrate that the propose approach is able to achieve up to 5.8× reduction in simulation cost. It should be noted that the proposed performance modeling algorithm for self-healing circuit can be widely applied to other important applications such as corner extraction [84] and design optimization [25]-[36].

Chapter 3

Efficient Post-Silicon Tuning via Indirect Performance Sensing

3.1 Motivation

One major challenge with on-chip self-healing is to efficiently implement on-chip sensors to accurately measure various analog and RF performance metrics. Towards this goal, indirect sensor modeling is a critical task where the objective is to build a mathematical model to capture the correlation between PoI and PoMs so that PoI can be accurately predicted from PoMs. To achieve this goal, PoMs and PoI are first measured from several training chips, and then indirect sensor models are constructed off-line based on these measurement data. Such indirect sensor models are eventually stored in an on-chip microcontroller for self-healing.

To describe an indirect senor, its model coefficients are stored in an on-chip microcontroller as fixed-point values. A complex model that is composed of many model terms would consume massive hardware resources, since a large number of model coefficients must be stored. Furthermore, during on-chip self-healing, an indirect sensor model is repeatedly evaluated to predict the corresponding PoI based on different PoMs and, therefore, a compact model could dramatically reduce the computational cost. Here, the computational cost accounts for on-chip multiplication and addition, and multiplication dominates the overall computational cost. For these reasons, an indirect sensor model should be compact in order to minimize the cost of on-chip self-healing.

Such a modeling task, however, is nontrivial since there is a tradeoff between the model complexity and the model accuracy. In general, it is likely that an over-simplified model will induce a large modeling error. Here, how to construct a compact indirect sensor model without sacrificing its modeling accuracy

remains an open question. In addition, these indirect sensor models must be repeatedly calibrated to accommodate the process shift associated with manufacturing lines. Such a model calibration issue has not been extensively studied yet. Hence, there is a strong need to develop a new methodology to facilitate efficient model calibration with low cost (i.e., requiring few additional measurement data). As such, the overhead of indirect performance sensing and, eventually, the overhead of on-chip self-healing can be minimized.

To address the aforementioned issues, a novel indirect performance sensing approach is proposed. The proposed method consists of two major steps: (i) pre-silicon indirect sensor modeling, and (ii) post-silicon indirect sensor calibration. In the first step, a compact indirect sensor model between PoMs and PoI is constructed based on pre-silicon simulation data by using sparse regression (SR) [74]-[77]. SR starts with a complicated model template (e.g., a high-order polynomial) that can accurately capture the correlation between PoMs and PoI. L1-norm regularization is then applied, resulting in a convex optimization problem which can be efficiently solved to determine the most important model terms in the template without sacrificing any modeling accuracy. Other model coefficients corresponding to the unimportant terms are simply set to zero, and are ignored in the final indirect sensor model. Intuitively, the unimportant model terms have negligible contribution for accurately predicting the value of PoI and, hence, can be discarded to minimize the self-healing cost.

Furthermore, in the second step, an indirect sensor model is repeatedly calibrated based on post-silicon measurement data. To perform efficient model calibration with low cost, a novel Bayesian model fusion (BMF) technique is proposed. The key idea of BMF is to combine the old (i.e., before process shift) indirect sensor model with very few new (i.e., after process shift) measurement data to generate a new model that is aligned with the new process condition. Mathematically, the old model is encoded as prior knowledge, and a Bayesian inference is derived to optimally fit the new model by maximum-a-posteriori (MAP) estimation.

Finally, an on-chip self-healing flow is developed where the indirect sensor models are extracted by the proposed technique. In this Chapter, we will present our proposed post-silicon tuning methodology in detail.

3.2 Indirect Performance Sensing

Without loss of generality, we denote PoI as f and PoMs as:

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_M \end{bmatrix}^T \tag{3.1}$$

where M stands for the number of performance metrics belonging to PoMs. The objective of indirect performance sensing is to accurately predict the PoI f from the PoMs $\mathbf{x}$ that are highly correlated with f and can be easily measured by on-chip sensors. Generating an indirect sensor model $f(\mathbf{x})$ consists of two major steps:

- Pre-silicon indirect sensor modeling aims to construct a compact model $f(\mathbf{x})$ that can accurately capture the correlation between the PoI f and the PoMs $\mathbf{x}$ based on pre-silicon simulation data.
- Post-silicon indirect sensor calibration aims to calibrate the indirect sensor model $f(\mathbf{x})$ based on post-silicon measurement data. Such model calibration must be repeatedly performed in order to accommodate the process shift associated with manufacturing lines.

We start with a generic and complicated model template (e.g., a high-order polynomial) to accurately capture the mapping between PoI and PoMs. The reason we choose a generic model is simply because we do not know the relation between PoI and PoMs in advance. Mathematically, we can write the model $f(\mathbf{x})$ as the linear combination of several basis functions:

$$f(\mathbf{x}) = \sum_{k=1}^{K} \alpha_k \cdot b_k(\mathbf{x}) \tag{3.2}$$

where $\{b_k(\mathbf{x}); k = 1, 2, \ldots, K\}$ are the basis functions (e.g., linear and quadratic polynomials), $\{\alpha_k; k = 1, 2, \ldots, K\}$ are the model coefficients, and K is the total number of basis functions.

Such a complicated model, though accurate, consumes considerable hardware resources to implement, as all model coefficients must be stored in an on-chip microcontroller to perform on-chip self-healing. To reduce the overhead of on-chip self-healing, we aim to select a small set of basis functions during pre-silicon modeling without surrendering any accuracy. Such a basis function selection task, however, is extremely challenging due to the tradeoff between the model complexity and the modeling error. In general, an over-simplified model is likely to have a large modeling error. SR [74]-[77] is applied to efficiently address the aforementioned basis function selection problem. More details about pre-silicon

indirect sensor modeling via SR will be discussed in Section 3.3.

Furthermore, at the post-silicon stage, the indirect sensor must be repeatedly calibrated to accommodate the process shift associated with manufacturing lines. Since post-silicon measurement is extremely expensive, sensor calibration must be accomplished with very few post-silicon measurement data to facilitate efficient generation of accurate indirect sensor models and, eventually, minimize the overhead of on-chip self-healing. To this end, a novel Bayesian model fusion (BMF) technique is proposed to keep the calibration cost affordable. The details about post-silicon indirect sensor calibration via BMF will be presented in next section.

3.3 Pre-Silicon Indirect Sensor Modeling via Sparse Regression

In this section, we aim to construct a compact indirect sensor model to accurately capture the relation between PoI and PoMs. Since the mapping from PoMs to PoI is not known in advance, we start with a generic and complicated model template consisting of a large number of basis functions (e.g., a high-order polynomial), as shown in (3.2). Our objective here is to automatically identify a small set of most important basis functions, and then determine their corresponding model coefficients based on pre-silicon simulation data.

To start with, we first collect a number of pre-silicon simulation samples $\{(\mathbf{x}^{(n)}, f^{(n)}); n = 1, \ldots, N\}$, where $\mathbf{x}^{(n)}$ and $f^{(n)}$ denote the values of $\mathbf{x}$ and f for the n-th sampling point respectively, and N denotes the total number of sampling points. Thus, a set of linear equations can be expressed as:

$$\mathbf{B}^T \cdot \boldsymbol{\alpha} = \mathbf{f} \tag{3.3}$$

where

$$\mathbf{B} = \begin{bmatrix} b_1\left(\mathbf{x}^{(1)}\right) & b_1\left(\mathbf{x}^{(2)}\right) & \cdots & b_1\left(\mathbf{x}^{(N)}\right) \\ b_2\left(\mathbf{x}^{(1)}\right) & b_2\left(\mathbf{x}^{(2)}\right) & \cdots & b_2\left(\mathbf{x}^{(N)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ b_K\left(\mathbf{x}^{(1)}\right) & b_K\left(\mathbf{x}^{(2)}\right) & \cdots & b_K\left(\mathbf{x}^{(N)}\right) \end{bmatrix} \tag{3.4}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_K \end{bmatrix}^T \tag{3.5}$$

$$\mathbf{f} = \begin{bmatrix} f^{(1)} & f^{(2)} & \cdots & f^{(N)} \end{bmatrix}^T \tag{3.6}$$

One simple approach to solve the model coefficients $\boldsymbol{\alpha}$ is to apply the traditional ordinary least squares (OLS) fitting method [61]. OLS determines the model coefficients $\boldsymbol{\alpha}$ by solving the following optimization problem:

$$\underset{\boldsymbol{\alpha}}{\text{minimize}} \quad \left\| \mathbf{B}^T \cdot \boldsymbol{\alpha} - \mathbf{f} \right\|_2^2 \tag{3.7}$$

where $\|\bullet\|_2$ denotes the L_2-norm of a vector. Intuitively, OLS intends to find a solution $\boldsymbol{\alpha}$ that can minimize the mean squared modeling error.

As mentioned at the beginning of this section, we aim to identify a small set of important basis functions from a large number of possible candidates. All other unimportant basis functions will be discarded due to their negligible contribution for accurately predicting the value of PoI. From this point of view, all model coefficients associated with these unimportant basis functions should be set to zero. Hence, identifying the most important basis functions is equivalent to finding a sparse solution $\boldsymbol{\alpha}$ for the linear equation in (3.7). The OLS formulation in (3.7) poses no constraint on the sparsity of $\boldsymbol{\alpha}$. In other words, the unconstrained optimization in (3.7) used by OLS cannot fit our need of basis function selection. Realizing this limitation of OLS, SR, instead, solves the following L_1-norm regularization problem:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \left\| \mathbf{B}^T \cdot \boldsymbol{\alpha} - \mathbf{f} \right\|_2^2 \\ \text{subject to} \quad & \left\| \boldsymbol{\alpha} \right\|_1 \le \lambda \end{aligned} \tag{3.8}$$

where $\|\bullet\|_1$ denotes the L_1-norm of a vector, and $\lambda > 0$ is a user-defined parameter. The formulation in (3.8) is a convex optimization problem and can be solved both efficiently (i.e., with low computational cost) and robustly (i.e., with guaranteed global optimum) [85].

There are several important properties associated with the optimization problem in (3.8). First, unlike the conventional OLS that minimizes the mean squared error only, the formulation in (3.8) minimizes the mean squared error subject to an L_1-norm constraint posed on the model coefficients $\boldsymbol{\alpha}$. It, in turn, promotes a sparse solution of $\boldsymbol{\alpha}$ that is desired by our application of basis function selection for on-chip self-healing.

Second, the parameter λ in (3.8) provides a tradeoff between the sparsity of the solution $\boldsymbol{\alpha}$ and the

modeling error. For instance, a large λ is likely to result in a small modeling error, but meanwhile it will increase the number of non-zeros in $\boldsymbol{\alpha}$. It is important to note that if the vector $\boldsymbol{\alpha}$ contains many non-zeros, a large number of model coefficients have to be stored in the on-chip microcontroller to predict the PoI and, hence, the cost of indirect performance sensing can be overly expensive. In practice, the value of λ must be appropriately set to accurately predict the PoI with a small set of basis functions. To find the optimal value of λ, we must accurately estimate the modeling error for different λ values. To avoid over-fitting, we cannot simply measure the modeling error from the set of sampling data that is used to calculate the model coefficients. Instead, modeling error must be measured from an independent data set.

To determine the modeling error for a given λ value, we adopt the idea of Q-fold cross-validation from the statistics community [61]. Namely, we partition the entire data set $\{(\mathbf{x}^{(n)}, f^{(n)}); n = 1, 2, \ldots, N\}$ into Q groups. Modeling error is estimated from Q independent runs. In each run, one of the Q groups is used to estimate the modeling error and all other groups are used to calculate the model coefficients by solving (3.8). Note that the training data for coefficient estimation and the testing data for error estimation are not overlapped. Hence, over-fitting can be easily detected. In addition, different groups should be selected for error estimation in different runs. As such, each run results in an error value e_q (q = 1, 2, ..., Q) that is measured from a unique group of the data set. The final modeling error is computed as the average of $\{e_q; q = 1, 2, ..., Q\}$, i.e., $e = (e_1 + e_2 + ... + e_Q)/Q$. More details about cross-validation can be found in [61].

So far, we only consider how to reduce the number of basis functions (i.e., the number of non-zeros in $\boldsymbol{\alpha}$) in order to save on-chip self-healing cost. Actually, different basis functions may involve different number of multiplications, and the computational cost to calculate each basis function when evaluating the indirect sensor can be quite different. For instance, $\alpha_1 \cdot x$ requires only one multiplication, while $\alpha_2 \cdot x^3$ needs three multiplications. To further reduce the computational cost, we can assign different weights for different coefficients (e.g., a small weight for α_1 while a large weight for α_2) in the constraint of (3.8). Intuitively, a coefficient with a larger weight is more likely to be set to zero in a weighted L_1-norm regularization. Because of the space limitation, the extended version of (3.8) to handle weighted $\boldsymbol{\alpha}$ is not mentioned here.

The aforementioned SR method can be efficiently applied to pre-silicon basis function selection and model coefficient estimation. However, the device models used for pre-silicon simulation are not perfectly

accurate and may differ from the post-silicon measurement results. For this reason, there is a strong need to further calibrate the proposed indirect sensor models based on post-silicon measurement data, as will be discussed in Section 3.4.

3.4 Post-Silicon Indirect Sensor Calibration via Bayesian Model Fusion

The objective of post-silicon indirect sensor calibration is to further correct the modeling error posed by pre-silicon simulation and also accommodate the process shift associated with manufacturing lines. One straightforward approach for sensor calibration is to collect a large amount of post-silicon measurement data and then completely re-fit the indirect sensor model. Such a simple approach, however, can be practically unaffordable, since post-silicon testing is time-consuming and, hence, it is overly expensive to collect a large set of post-silicon measurement data.

To address this cost issue, we propose a novel statistical framework, referred to as *Bayesian model fusion* (BMF), for efficient post-silicon sensor calibration. BMF relies on an important observation that even though the simulation and/or measurement data collected at multiple stages (e.g., pre-silicon vs. post-silicon) are not exactly identical, they are expected to be strongly correlated. Hence, it is possible to *borrow* the data from an early stage (e.g., pre-silicon) for sensor calibration at a late stage (e.g., post-silicon). As such, only few post-silicon data should be measured at the late stage and, hence, the cost of sensor calibration is substantially reduced.

More specifically, our indirect sensor models are initially fitted by using the early-stage (e.g., pre-silicon) data. Next, the early-stage sensor model is encoded as our *prior* knowledge. Finally, the indirect sensor model is further calibrated by applying Bayesian inference with very few late-stage (e.g., post-silicon) measurement data. Here, by "fusing" the early-stage and late-stage sensor models through Bayesian inference, the amount of required measurement data (hence, the measurement cost) can be substantially reduced at the late stage.

To fully understand the proposed BMF method, let us consider two different models: the early-stage model $f_E(\mathbf{x})$ and the late-stage model $f_L(\mathbf{x})$:

$$f_E(\mathbf{x}) = \sum_{k=1}^{K} \alpha_{E,k} \cdot b_k(\mathbf{x}) + \varepsilon_E \tag{3.9}$$

$$f_L(\mathbf{x}) = \sum_{k=1}^{K} \alpha_{L,k} \cdot b_k(\mathbf{x}) + \varepsilon_L \tag{3.10}$$

where $\{b_k(\mathbf{x}); k = 1, 2, \ldots, K\}$ are the basis functions selected by SR at the early stage, $\{\alpha_{E,k}; k = 1, 2, \ldots, K\}$ and $\{\alpha_{L,k}; k = 1, 2, \ldots, K\}$ contain the early-stage and late-stage model coefficients respectively, and ε_E and ε_L denote the modeling error associated with the early-stage and late-stage models respectively.

The early-stage model $f_E(\mathbf{x})$ in (3.9) is fitted by using the early-stage (e.g., pre-silicon) data. Hence, we assume that the early-stage model coefficients $\{\alpha_{E,k}; k = 1, 2, \ldots, K\}$ are already known, before fitting the late-stage model $f_L(\mathbf{x})$ in (3.10) based on the late-stage (e.g., post-silicon) measurement data. The objective of BMF is to accurately determine the late-stage model coefficients $\{\alpha_{L,k}; k = 1, 2, \ldots, K\}$ by combining the early-stage model coefficients $\{\alpha_{E,k}; k = 1, 2, \ldots, K\}$ with very few late-stage measurement data.

Our proposed BMF method consists of two major steps: (i) statistically extracting the prior knowledge from the early-stage model coefficients $\{\alpha_{E,k}; k = 1, 2, \ldots, K\}$ and encoding it as a prior distribution, and (ii) optimally determining the late-stage model coefficients $\{\alpha_{L,k}; k = 1, 2, \ldots, K\}$ by MAP estimation. In what follows, we will describe these two steps in detail.

3.4.1 Prior Knowledge Definition

Since the two models $f_E(\mathbf{x})$ and $f_L(\mathbf{x})$ in (3.9)-(3.10) both approximate the mathematical mapping from PoMs to PoI, we expect that the model coefficients $\{\alpha_{E,k}; k = 1, 2, \ldots, K\}$ and $\{\alpha_{L,k}; k = 1, 2, \ldots, K\}$ are similar. On the other hand, $f_E(\mathbf{x})$ and $f_L(\mathbf{x})$ cannot be exactly identical, since they represent the indirect sensor models at two different stages. To statistically encode the "common" information between $f_E(\mathbf{x})$ and $f_L(\mathbf{x})$, we define a Gaussian distribution as our prior distribution for each late-stage model coefficient $\alpha_{L,k}$:

$$pdf(\alpha_{L,k}) = \frac{1}{\sqrt{2\pi} \cdot \rho \cdot |\alpha_{E,k}|} \cdot \exp\left[-\frac{(\alpha_{L,k} - \alpha_{E,k})^2}{2 \cdot \rho^2 \cdot \alpha_{E,k}^2} \right] \quad k = 1, 2, \cdots, K \tag{3.11}$$

where $\alpha_{E,k}$ and $\rho^2 \cdot \alpha_{E,k}^2$ are the mean and variance of the Gaussian distribution respectively, and $\rho > 0$ is a

parameter that can be determined by cross-validation.

The prior distribution in (3.11) has a two-fold meaning. First, the Gaussian distribution $pdf(\alpha_{L,k})$ is peaked at its mean value $\alpha_{E,k}$, implying that the early-stage model coefficient $\alpha_{E,k}$ and the late-stage model coefficient $\alpha_{L,k}$ are likely to be similar. In other words, since the Gaussian distribution $pdf(\alpha_{L,k})$ exponentially decays with $(\alpha_{L,k} - \alpha_{E,k})^2$, it is unlikely to observe a late-stage coefficient $\alpha_{L,k}$ that is extremely different from the early-stage coefficient $\alpha_{E,k}$. Second, the standard deviation of the prior distribution $pdf(\alpha_{L,k})$ is proportional to $|\alpha_{E,k}|$. It means that the absolute difference between the late-stage coefficient $\alpha_{L,k}$ and the early-stage coefficient $\alpha_{E,k}$ can be large (or small), if the magnitude of the early-stage coefficient $|\alpha_{E,k}|$ is large (or small). Restating in words, each late-stage coefficient $\alpha_{L,k}$ has been provided with a relatively equal opportunity to deviate from the corresponding early-stage coefficient $\alpha_{E,k}$.

To complete the definition of the prior distribution for all late-stage model coefficients $\{\alpha_{L,k}; k = 1, 2, \ldots, K\}$, we further assume that these coefficients are statistically independent and their joint distribution is represented as:

$$pdf\left(\boldsymbol{\alpha}_L\right) = \prod_{k=1}^{K} pdf\left(\alpha_{L,k}\right) \tag{3.12}$$

where

$$\boldsymbol{\alpha}_L = \begin{bmatrix} \alpha_{L,1} & \alpha_{L,2} & \cdots & \alpha_{L,K} \end{bmatrix}^T \tag{3.13}$$

is a vector containing all late-stage coefficients $\{\alpha_{L,k}; k = 1, 2, \ldots, K\}$. Combining (3.11) and (3.12) yields:

$$pdf\left(\boldsymbol{\alpha}_L\right) = \frac{1}{\left(\sqrt{2\pi}\cdot\rho\right)^K \cdot \prod_{k=1}^{K}\left|\alpha_{E,k}\right|} \cdot \exp\left[-\frac{\left(\boldsymbol{\alpha}_L - \boldsymbol{\alpha}_E\right)^T \cdot \mathbf{A} \cdot \left(\boldsymbol{\alpha}_L - \boldsymbol{\alpha}_E\right)}{2\cdot\rho^2}\right] \tag{3.14}$$

where

$$\boldsymbol{\alpha}_E = \begin{bmatrix} \alpha_{E,1} & \alpha_{E,2} & \cdots & \alpha_{E,K} \end{bmatrix} \tag{3.15}$$

is a vector containing all early-stage coefficients $\{\alpha_{E,k}; k = 1, 2, \ldots, K\}$, and

$$\mathbf{A} = \mathrm{diag}\left(\frac{1}{\alpha_{E,1}^2} \quad \frac{1}{\alpha_{E,2}^2} \quad \cdots \quad \frac{1}{\alpha_{E,K}^2}\right) \tag{3.16}$$

denotes a diagonal matrix. The independence assumption in (3.12) simply implies that we do not know the

correlation information among these coefficients as our prior knowledge. The correlation information will be learned from the late-stage measurement data, when the posterior distribution is calculated by MAP estimation in the next sub-section.

3.4.2 Maximum-A-Posteriori Estimation

Once the prior distribution is defined, we collect a few (i.e., N) late-stage measurement data $\{(\mathbf{x}^{(n)}, f_L^{(n)});\ n = 1, 2, \ldots, N\}$, where $\mathbf{x}^{(n)}$ and $f_L^{(n)}$ are the values of $\mathbf{x}$ and $f_L(\mathbf{x})$ for the n-th data point respectively. These new measurement data can tell us additional information about the difference between early and late stages and, hence, help us to determine the late-stage coefficients $\boldsymbol{\alpha}_L$.

Based on Bayes' theorem [61], the uncertainties of the late-stage coefficients $\boldsymbol{\alpha}_L$ after knowing the data $\{(\mathbf{x}^{(n)}, f_L^{(n)});\ n = 1, 2, \ldots, N\}$ can be mathematically described by the following posterior distribution:

$$pdf\left(\boldsymbol{\alpha}_L \middle| \mathbf{X}, \mathbf{f}_L\right) \propto pdf\left(\boldsymbol{\alpha}_L\right) \cdot pdf\left(\mathbf{X}, \mathbf{f}_L \middle| \boldsymbol{\alpha}_L\right) \tag{3.17}$$

where

$$\mathbf{f}_L = \begin{bmatrix} f_L^{(1)} & f_L^{(2)} & \cdots & f_L^{(N)} \end{bmatrix}^T \tag{3.18}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)} & \mathbf{x}^{(2)} & \cdots & \mathbf{x}^{(N)} \end{bmatrix}^T \tag{3.19}$$

In (3.17), the prior distribution $pdf(\boldsymbol{\alpha}_L)$ is defined by (3.14). The conditional distribution $pdf(\mathbf{X}, \mathbf{f}_L \mid \boldsymbol{\alpha}_L)$ is referred to as the likelihood function. It measures the probability of observing the new data $\{(\mathbf{x}^{(n)}, f_L^{(n)});\ n = 1, 2, \ldots, N\}$.

To derive the likelihood function $pdf(\mathbf{X}, \mathbf{f}_L \mid \boldsymbol{\alpha}_L)$, we assume that the modeling error ε_L in (3.10) can be represented as a random variable with zero-mean Gaussian distribution:

$$pdf\left(\varepsilon_L\right) = \frac{1}{\sqrt{2\pi} \cdot \sigma_0} \cdot \exp\left[-\frac{\varepsilon_L^2}{2 \cdot \sigma_0^2}\right] \tag{3.20}$$

where the standard deviation σ_0 indicates the magnitude of the modeling error. Similar to the parameter ρ in (3.11), the value of σ_0 can be determined by cross-validation. Since the modeling error associated with the n-th data point $(\mathbf{x}^{(n)}, f_L^{(n)})$ is simply one sampling point of the random variable ε_L that follows the Gaussian

distribution in (3.20), the probability of observing the n-th data point $(\mathbf{x}^{(n)}, f_L^{(n)})$ is:

$$pdf\left[\mathbf{x}^{(n)}, f_L^{(n)}\middle|\boldsymbol{\alpha}_L\right] = \frac{1}{\sqrt{2\pi}\cdot\sigma_0}\cdot\exp\left\{-\frac{1}{2\cdot\sigma_0^2}\cdot\left[f_L^{(n)} - \sum_{k=1}^{K}\alpha_{L,k}\cdot b_k\left(\mathbf{x}^{(n)}\right)\right]^2\right\}. \tag{3.21}$$

Note that the likelihood function $pdf[\mathbf{x}^{(n)}, f_L^{(n)} \mid \boldsymbol{\alpha}_L]$ in (3.21) depends on the late-stage model coefficients $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$. Assuming that all data points $\{(\mathbf{x}^{(n)}, f_L^{(n)});\ n = 1, 2, \ldots, N\}$ are independently generated, we can write the likelihood function $pdf(\mathbf{X}, \mathbf{f}_L \mid \boldsymbol{\alpha}_L)$ as:

$$\begin{aligned} pdf\left(\mathbf{X},\mathbf{f}_L\middle|\boldsymbol{\alpha}_L\right) &= \prod_{n=1}^{N} pdf\left[\mathbf{x}^{(n)}, f_L^{(n)}\middle|\boldsymbol{\alpha}_L\right] = \frac{1}{\left(\sqrt{2\pi}\cdot\sigma_0\right)^N}\cdot \\ &\exp\left\{-\frac{1}{2\cdot\sigma_0^2}\cdot\sum_{n=1}^{N}\left[f_L^{(n)} - \sum_{k=1}^{K}\alpha_{L,k}\cdot b_k\left(\mathbf{x}^{(n)}\right)\right]^2\right\} \end{aligned} \tag{3.22}$$

Eq. (3.22) can be re-written as:

$$pdf\left(\mathbf{X},\mathbf{f}_L\middle|\boldsymbol{\alpha}_L\right) = \frac{1}{\left(\sqrt{2\pi}\cdot\sigma_0\right)^N}\cdot\exp\left\{-\frac{\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L - \mathbf{f}_L\right)^T\cdot\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L - \mathbf{f}_L\right)}{2\cdot\sigma_0^2}\right\} \tag{3.23}$$

where $\mathbf{B}$, $\boldsymbol{\alpha}_L$ and $\mathbf{f}_L$ are defined in (3.4), (3.13) and (3.18), respectively.

After the new data $\{(\mathbf{x}^{(n)}, f_L^{(n)});\ n = 1, 2, \ldots, N\}$ are available, the late-stage coefficients $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$ can be described by the probability density function $pdf(\boldsymbol{\alpha}_L \mid \mathbf{X}, \mathbf{f}_L)$ (i.e., the posterior distribution) in (3.17). Depending on the shape of the posterior distribution $pdf(\boldsymbol{\alpha}_L \mid \mathbf{X}, \mathbf{f}_L)$, the late-stage coefficients $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$ do not take all possible values with equal probability. If the posterior distribution $pdf(\boldsymbol{\alpha}_L \mid \mathbf{X}, \mathbf{f}_L)$ reaches its maximum value at $\{\alpha^*_{L,k};\ k = 1, 2, \ldots, K\}$, these values $\{\alpha^*_{L,k};\ k = 1, 2, \ldots, K\}$ are the optimal estimation of the late-stage coefficients, since these coefficient values are most likely to occur. Such a method is referred to as the MAP estimation in the literature [61].

The aforementioned MAP estimation can be formulated as an optimization problem:

$$\underset{\boldsymbol{\alpha}_L}{\text{maximize}} \quad pdf\left(\boldsymbol{\alpha}_L\middle|\mathbf{X},\mathbf{f}_L\right) \tag{3.24}$$

Substituting (3.17) into (3.24) yields:

$$\underset{\boldsymbol{\alpha}_L}{\text{maximize}} \quad pdf\left(\boldsymbol{\alpha}_L\right)\cdot pdf\left(\mathbf{X},\mathbf{f}_L\middle|\boldsymbol{\alpha}_L\right) \tag{3.25}$$

Combining (3.14), (3.23) and (3.25), we have:

$$\underset{\boldsymbol{\alpha}_L}{\text{maximize}} \quad \exp\left[\begin{array}{l} -\dfrac{\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right)^T\cdot\mathbf{A}\cdot\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right)}{2\cdot\rho^2}- \\ \dfrac{\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right)^T\cdot\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right)}{2\cdot\sigma_0^2} \end{array}\right] \tag{3.26}$$

Since the exponential function is monotonically increasing, Eq. (3.26) can be re-written as:

$$\underset{\boldsymbol{\alpha}_L}{\text{minimize}} \quad \begin{array}{l} \eta\cdot\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right)^T\cdot\mathbf{A}\cdot\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right) \\ +\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right)^T\cdot\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right) \end{array} \tag{3.27}$$

where

$$\eta=\sigma_0^2/\rho^2\,. \tag{3.28}$$

It is straightforward to prove that the cost function in (3.27) is convex [85]. Hence, its global optimum can be directly solved by applying the first-order optimality condition [85]:

$$\begin{array}{l} \dfrac{\partial}{\partial\boldsymbol{\alpha}_L}\left[\begin{array}{l} \eta\cdot\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right)^T\cdot\mathbf{A}\cdot\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right) \\ +\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right)^T\cdot\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right) \end{array}\right] \\ =2\cdot\eta\cdot\mathbf{A}\cdot\left(\boldsymbol{\alpha}_L-\boldsymbol{\alpha}_E\right)+2\cdot\mathbf{B}\cdot\left(\mathbf{B}^T\cdot\boldsymbol{\alpha}_L-\mathbf{f}_L\right)=\mathbf{0} \end{array}\,. \tag{3.29}$$

Solving the linear equation in (3.29) results in the optimal value of $\boldsymbol{\alpha}_L$:

$$\boldsymbol{\alpha}_L=\left(\eta\cdot\mathbf{A}+\mathbf{B}\cdot\mathbf{B}^T\right)^{-1}\cdot\left(\eta\cdot\mathbf{A}\cdot\boldsymbol{\alpha}_E+\mathbf{B}\cdot\mathbf{f}_L\right). \tag{3.30}$$

Studying (3.30), we observe that only the value of η is required to find the late-stage model coefficients $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$ and, hence, we only need to determine η, instead of the individual parameters σ_0 and ρ. In our work, the optimal value of η is determined by cross-validation.

Once the optimal η value is found, the late-stage model coefficients $\{\alpha_{L,k};\ k = 1, 2, ..., K\}$ are calculated from (3.30), and then an updated indirect sensor model $f_L(\mathbf{x})$ in (3.10) is generated to match the late-stage measurement data. Such a calibrated indirect sensor model is eventually stored in an on-chip microcontroller to facilitate efficient on-chip self-healing, as will be discussed in detail in the next section.

Finally, it is important to mention that the post-silicon indirect sensor calibration is performed off-chip and, hence, no hardware overhead is introduced. To further reduce the indirect sensor calibration cost (i.e., with very few number of post-silicon measurement data), we can calibrate the indirect sensor if and only if the new measurement data are not consistent with the old indirect sensor model. In practice, such

inconsistency can be detected by measuring a small number of dies from each wafer or lot to estimate the indirect sensing error. If the error is not sufficiently small, the indirect sensor model must be calibrated.

3.5 On-Chip Self-Healing Flow

In this section, we will further develop a practical on-chip self-healing flow based on our proposed indirect sensing approach. As mentioned earlier, the key idea of on-chip self-healing is to actively monitor the post-manufacturing circuit performance metrics and then adaptively adjust a number of tuning knobs (e.g., bias voltage) in order to meet the given performance specifications. In this work, we mathematically formulate the self-healing problem as a constrained optimization where one particular performance metric is minimized subject to a set of given performance constraints:

$$\begin{aligned} \underset{\mathbf{t}}{\text{minimize}} \quad & f(\mathbf{t}) \\ \text{subject to} \quad & g_p(\mathbf{t}) \geq s_p \quad (p = 1, 2, \cdots, P) \end{aligned} \tag{3.31}$$

where $\mathbf{t}$ denotes the set of tuning knobs, $f(\mathbf{t})$ denotes the performance metric that we aim to minimize, and $\{g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$ denote the other P performance metrics with the given specifications $\{s_p; p = 1, 2, \ldots, P\}$. Take mixer as an example. We aim to minimize the mixer power while keeping its gain and 1dB compression point larger than their specifications. In this case, $f(\mathbf{t})$ is the mixer power, $g_1(\mathbf{t})$ is the mixer gain, and $g_2(\mathbf{t})$ is the 1dB compression point.

There are two important clarifications we need to make for the optimization formulation in (3.31). First, the formulation in (3.31) is set up for a circuit where one performance metric $f(\mathbf{t})$ should be minimized while constraining all other performance metrics $\{g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$ to their lower bounds $\{s_p; p = 1, 2, \ldots, P\}$. For a circuit where a performance metric $f(\mathbf{t})$ should be maximized, the objective function in (3.31) can be simply modified to $-f(\mathbf{t})$. Similarly, for a circuit where the performance metrics $\{g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$ should be constrained to their upper bounds $\{s_p; p = 1, 2, \ldots, P\}$, the constraints in (3.31) can be adjusted as $-g_p(\mathbf{t}) \geq -s_p$ ($p = 1, 2, \ldots, P$). Second, not all the performance metrics $f(\mathbf{t})$ and $\{g_p(\mathbf{t}); p \in 1, 2, \ldots, P\}$ in our self-healing circuit can be directly measured by on-chip sensors. For the performance metrics that cannot be easily measured by on-chip sensors, the proposed indirect performance sensing technique is applied to efficiently and accurately predict their values.

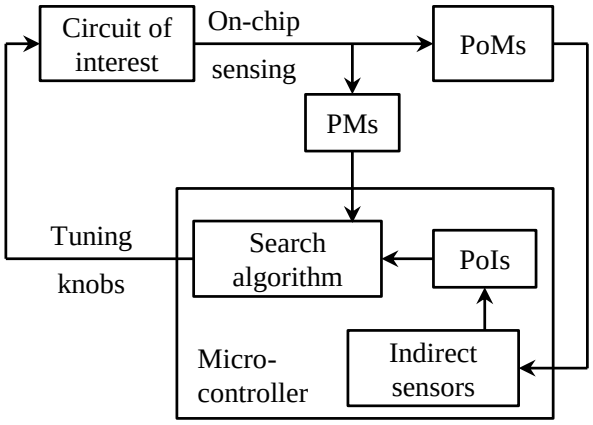


Figure 3-1. A simplified block diagram describes the on-chip self-healing flow.

To find the optimal solution **t*** in (3.31), we propose an on-chip self-healing flow shown in Figure 3-1 where the indirect sensors are modeled and calibrated by our proposed SR and BMF techniques described in previous sections. The indirect sensor models are stored and evaluated by a microcontroller for on-chip self-healing. The search algorithm starts with an initial guess $\mathbf{t}_0$. We set $\mathbf{t} = \mathbf{t}_0$ and all performance metrics $f(\mathbf{t})$ and $\{g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$ are measured either directly or indirectly. Here, we use the symbol PMs to represent the performance metrics that are directly measured by on-chip sensors, and the symbol PoIs to represent the performance metrics that are estimated by the proposed indirect sensors. In particular, to estimate the PoIs, the corresponding PoMs are first measured by on-chip sensors. Next, the indirect sensor models stored in the on-chip microcontroller are evaluated to predict the PoIs, as shown in Figure 3-1. Based on the performance values $\{f(\mathbf{t}), g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$, $\mathbf{t}$ is updated and the aforementioned process is repeated until the optimal solution $\mathbf{t}^*$ is found. Algorithm 2 summarizes the details of such an optimization flow with indirect performance sensing for on-chip self-healing. Once the optimal solution $\mathbf{t}^*$ is found, tuning knobs are adjusted to the values of $\mathbf{t}^*$ and the self-healing process is complete.

Algorithm 2: On-chip Self-healing Flow

1. Start with the constrained optimization problem in (3.31) and an initial guess $\mathbf{t}_0$.
2. Set $\mathbf{t} = \mathbf{t}_0$.
3. Measure $f(\mathbf{t})$ and $\{g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$ either directly or indirectly.
4. Based on the performance values $\{f(\mathbf{t}), g_p(\mathbf{t}); p = 1, 2, \ldots, P\}$, update $\mathbf{t}$.
5. If $\mathbf{t}$ is the optimal solution, stop iteration. Otherwise, go to Step 3.

For different circuits of interest with different performance metrics and tuning knobs, the search strategy of updating **t** in Step 4 of Algorithm 2 could be substantially different. For instance, if there is only a small number of (e.g., one or two) tuning knobs, we can apply a simple brute-force search algorithm to find the optimal solution of (3.31). Without loss of generality, we assume that the tuning knobs can take H possible values $\{\mathbf{t}_h;\ h = 1, 2, \ldots, H\}$. The initial value of **t** is set as $\mathbf{t}_1$ in the first iteration, and $\{f(\mathbf{t}_1),\ g_p(\mathbf{t}_1);\ p = 1, 2, \ldots, P\}$ are either directly or indirectly measured. Next, in the second iteration, **t** is updated to $\mathbf{t}_2$, and $\{f(\mathbf{t}_2),\ g_p(\mathbf{t}_2);\ p = 1, 2, \ldots, P\}$ are measured. Similarly, in the h-th iteration, $\{f(\mathbf{t}_h),\ g_p(\mathbf{t}_h);\ p = 1, 2, \ldots, P\}$ are measured. In the end, we have a large data set $\{f(\mathbf{t}_h),\ g_p(\mathbf{t}_h);\ p = 1, 2, \ldots, P,\ h = 1, 2, \ldots, H\}$. The optimal solution $\mathbf{t}^*$ for the optimization in (3.31) can be eventually determined based on the performance values $\{f(\mathbf{t}_h),\ g_p(\mathbf{t}_h);\ p = 1, 2, \ldots, P,\ h = 1, 2, \ldots, H\}$. Algorithm 3 summarizes the details of the aforementioned brute-force search algorithm.

Algorithm 3: Brute-force Search for On-chip Self-healing

1. Start with the constrained optimization problem in (3.31) and H possible values $\{\mathbf{t}_h;\ h = 1, 2, \ldots, H\}$ for the tuning knobs. Set $h = 1$.
2. Set the tuning knobs to $\mathbf{t} = \mathbf{t}_h$.
3. Measure $f(\mathbf{t})$ and $\{g_p(\mathbf{t});\ p = 1, 2, \ldots, P\}$ and set $f(\mathbf{t}_h) = f(\mathbf{t})$, and $\{g_p(\mathbf{t}_h) = g_p(\mathbf{t});\ p = 1, 2, \ldots, P\}$.
4. If $h < H$, $h = h + 1$ and go to Step 2. Otherwise, go to Step 5.
5. Based on the performance values $\{f(\mathbf{t}_h),\ g_p(\mathbf{t}_h);\ p = 1, 2, \ldots, P,\ h = 1, 2, \ldots, H\}$, determine the optimal solution $\mathbf{t}^*$ for the optimization in (3.31).

The brute-force search algorithm (i.e., Algorithm 3), though simple to implement, has no practical utility if we have a large number of tuning knobs. To understand the reason, let us consider the general case of U tuning knobs where the u-th tuning knob can take V_u possible values. In this case, we have $H = V_1 \times V_2 \times \ldots \times V_U$ possible values for **t** in (3.31). With the increasing number of tuning knobs, the total number of possible values for these tuning knobs (i.e., H) will dramatically increase, thereby making the brute-force search algorithm quickly intractable. In these cases, other efficient search algorithms (e.g., interior point method [85]) must be applied to solve the optimization in (3.31) for on-chip self-healing.

Before ending this section, it is important to discuss the design overhead of on-chip self-healing that requires a number of additional circuitries (e.g., on-chip sensors, on-chip microcontroller, etc.), as shown in

Figure 3-1. There are several important clarifications we need to make here. First, many analog and RF circuit blocks on the same chip may require self-healing, and they can possibly share the same on-chip sensors and microcontroller. Second, for a typical system-on-chip (SoC) application, the microcontroller is needed for other computing tasks during the normal operation. In other words, the microcontroller is not added for on-chip self-healing only. For these reasons, the design overhead of on-chip self-healing is fairly small, or even negligible, in many application scenarios.

3.6 Case Study

In this section, a 25GHz differential Colpitts VCO designed in a 32nm CMOS SOI process is used to validate the proposed on-chip self-healing flow based on off-line data analysis. Figure 3-2 shows the simplified schematic of the VCO. It consists of a cross-coupled differential pair connected to two common-gate Colpitts oscillators. The capacitor at the output is tunable so that the VCO frequency can be centered at different frequency bands. The bias voltage V_b is controlled by a DAC for self-healing. More details about the VCO design can be found in [86].

For the VCO shown in Figure 3-2, since we only have one tuning knob (i.e., the bias voltage V_b), the simple brute-force search algorithm described in Algorithm 2 is applied for self-healing. In this example, phase noise is an important performance of interest, and its specifications derived from the system requirement for four different center frequencies are shown in Table 3-1. If the phase noise value of a VCO is smaller than the given specification at all four frequencies shown in Table 3-1, this VCO is considered as "PASS". Otherwise, we consider it as "FAIL". The objective of self-healing is to find the optimal bias voltage to minimize the phase noise.

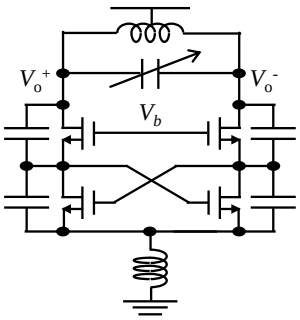


Figure 3-2. A Simplified circuit schematic is shown for a Colpitts VCO.

Accurately measuring the phase noise at 25GHz is not trivial. Hence, an indirect sensor is used for on-chip phase noise measurement (i.e., phase noise is considered as a PoI in Figure 3-1). According to Leeson's model [87], oscillation frequency (x_1), oscillation amplitude (x_2), and bias current (x_3), all of which are easy to measure using fully integrated sub-circuits, have strong correlation with phase noise and, hence, are first chosen as PoMs. The sensors used to measure these three PoMs are listed in Table 3-2. An on-chip current sensor to measure bias current will be integrated in our future work. More details about how to measure these three PoMs can be found in [50]. In addition, the tuning knob V_b (x_4) is considered as another PoM. Since V_b is directly controlled by a DAC, the digitized value of V_b is known, and no measurement is required. In total, four PoMs (i.e., $\mathrm{x} = [x_1\ x_2\ x_3\ x_4]^T$) are chosen as the indirect sensor inputs, and are summarized in Table 3-3. Next, a quadratic model template with four input variables (i.e., x_1, x_2, x_3 and x_4) and, hence, fifteen polynomial terms in total is used to build the indirect sensor for phase noise. With all fifteen polynomial terms, the average modeling error is 0.36dBc/Hz. SR is then applied to simplify the quadratic model template. Nine polynomial terms are eventually selected by SR, as summarized in Table 3-4. The average modeling error of the simplified quadratic model is 0.41dBc/Hz. The degradation of the modeling accuracy is negligible (0.05dBc/Hz only). The accuracy of the simplified quadratic model with nine polynomial terms can be further demonstrated by the scatter plot between the actual phase noise and the predicted phase noise shown in Figure 3-3.

Next, we collect four PoMs and phase noise at all possible bias voltages from a silicon wafer that contains 61 functional VCOs. The VCOs that are not functioning in this wafer are not considered here. These data are further used to calibrate the indirect phase noise sensor to improve its accuracy. Without self-healing, the parametric yield achieved by using a fixed bias voltage for all the VCOs on the wafer is summarized in Table 3-5. Here, bias code denotes a digitized bias voltage, and parametric yield is defined as the ratio between the number of functional VCOs that can meet all four given phase noise specifications shown in Table 3-1 and the total number of functional VCOs. From Table 3-5, we can see that the best parametric yield achieved by using a fixed bias voltage (i.e., bias code is 4) is only 11.48%. If other bias voltages are selected during the design, the parametric yield is even worse, which is almost zero for this wafer. It, in turn, serves as an excellent design case to demonstrate the importance of self-healing.

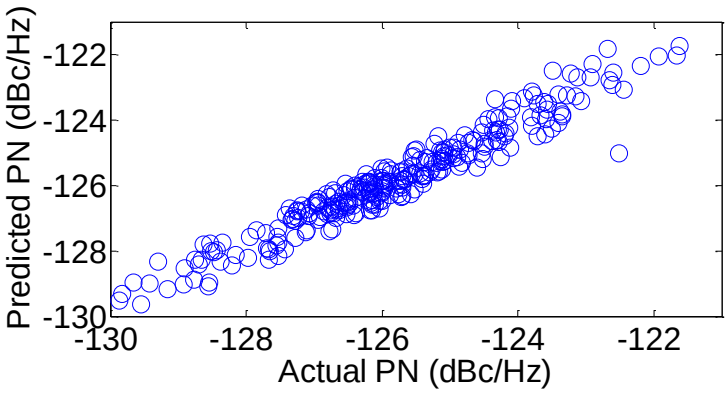


Figure 3-3. Scatter plot is shown for the actual phase noise and the predicted phase noise based on the simplified quadratic model.

Table 3-1. Frequencies and corresponding phase noise specifications

Frequency (GHz)	26.2	24.6	23.3	22.2
PN specification (dBc/Hz)	-123.5	-123.5	-123.5	-123.5

Table 3-2. PoMs and measurement sensors

PoM	x_1	x_2	x_3
Measurement Sensor	On-chip counter	On-chip peak detector + on-chip 6 bit ADC	Off-chip current Sensor

Table 3-3. PoI and PoMs of indirect phase noise sensor

	Performance Metric
PoI	Phase Noise
PoMs	Oscillation Frequency (x_1)
	Oscillation Amplitude (x_2)
	Bias Current (x_3)
	Bias Voltage (x_4)

Table 3-4. Basis functions selected for indirect phase noise sensor

Index	Term	Index	Term	Index	Term
1	x_2	4	x_1^2	7	$x_3 \bullet x_4$
2	x_3	5	$x_1 \bullet x_2$	8	x_4^2
3	x_4	6	$x_1 \bullet x_4$	9	const
Index	Term	Index	Term	Index	Term
1	x_2	4	x_1^2	7	$x_3 \bullet x_4$

Table 3-5. Parametric yield of the wafer by using a fixed bias voltage

Bias Code	1	2	3	4	5	6
Parametric Yield	0	0	1.64%	11.48%	3.28%	0

For testing and comparison purposes, three different self-healing methods are implemented:

- **Ideal**: The optimal bias voltage is determined by directly measuring the phase noise with an off-chip tester for all bias voltages. As a result, no indirect phase noise sensor is needed, and all the off-chip measurement data from the wafer will be used. This approach is not considered as on-chip self-healing; however, it provides the upper bound of the yield improvement that can be achieved by self-healing.
- **OLS**: The traditional OLS method is applied to fit the indirect phase noise sensor based on a number of measured VCOs from the wafer. Next, the indirect sensor is applied to self-heal all the VCOs on the wafer.
- **BMF**: The indirect phase noise sensor learned by SR is considered as the early-stage model. Next, the proposed BMF algorithm is applied to calibrate the early-stage model and generate a late-stage model based on a few measured VCOs from the wafer. The late-stage model is then applied to self-heal all the VCOs on the wafer.

Figure 3-4 shows the parametric yield of the wafer achieved by three different self-healing methods given different number of measured VCOs from the wafer. Table 3-6 further summarizes the measurement

cost for self-healing. Studying Figure 3-4 and Table 3-6 reveals several important observations. First, BMF requires substantially less number of measured VCOs to build the indirect phase noise sensor than the traditional OLS method. In this example, BMF needs to measure one VCO only, while OLS requires measuring four VCOs (4×) to achieve a similar yield.

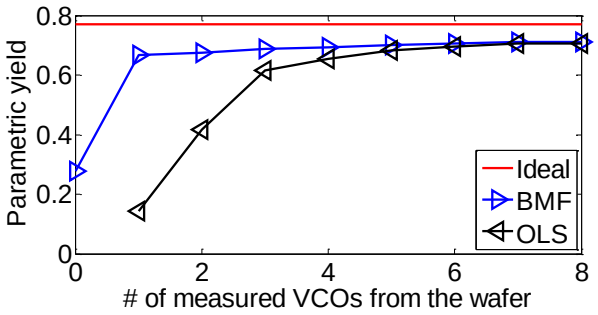


Figure 3-4. Post-self-healing parametric yield of the wafer is shown as a function of the number of measured VCOs from the wafer.

Second, studying the BMF results in Figure 3-4, we notice that if no measurement data is collected from the wafer (i.e., the number of measured VCOs from the wafer is zero) and the self-healing is performed with the indirect sensor fitted from the early-stage data by SR, the post-self-healing parametric yield is only 27.87%. Once a single VCO is measured from the wafer, the indirect phase noise sensor is calibrated by BMF and the post-self-healing parametric yield is increased to 66.80%. It, in turn, demonstrates that the aforementioned model calibration is a critical step for yield enhancement.

Table 3-6. Measurement cost and parametric yield by self-healing

Self-healing method	# of measured VCOs	Parametric yield
Ideal	61	77.05%
OLS	4	65.30%
BMF	1	66.80%

Finally, it is important to note that the post-self-healing parametric yield of BMF is close to that of the “ideal” case. It, in turn, implies that the modeling error of our proposed BMF method is fairly small,

even if only a single VCO is measured from the wafer.

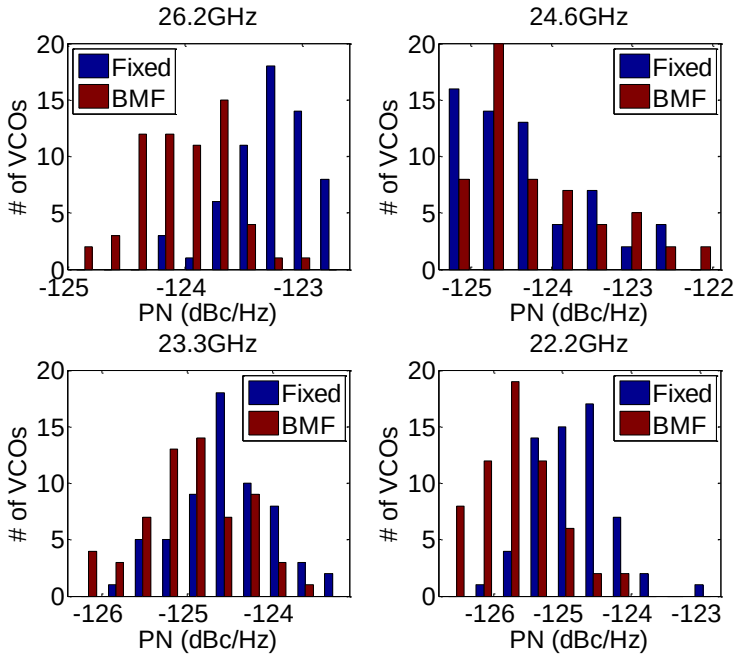


Figure 3-5. Histogram of the measured phase noise values from all the VCOs on the wafer. Blue bars represent the results from Fixed where bias code is 4, and red bars represent the results from BMF where a single measured VCO is used from the wafer.

Before ending this section, we compare the phase noise values from the proposed self-healing flow to those from the fixed bias voltage method (Fixed) to study why the proposed flow can achieve a much better parametric yield than Fixed. Figure 3-5 shows the histogram of the measured phase noise values from all the VCOs at different frequencies. The blue bars show the results from Fixed where bias code is 4, and the red bars show the results from our proposed BMF technique with a single measured VCO from the wafer. From Figure 3-5, we have several observations. First, both BMF and Fixed get larger phase noise values at higher frequencies, which is consistent with our expectation. Hence, the phase noise specification at 26.2GHz is the most difficult one to meet among all four phase noise specifications. Second, the proposed BMF technique can get much smaller phase noise values than Fixed at 26.2GHz, which is the reason that BMF achieves a much better parametric yield than Fixed.

3.7 Summary

In this Chapter, we propose a novel indirect performance sensing technique for on-chip self-healing of analog and RF circuits. In particular, a set of important basis functions are first identified by SR so that the overhead of on-chip self-healing can be minimized. Next, the indirect sensors are repeatedly calibrated by BMF to accommodate the process shift associated with manufacturing lines. The indirect sensors are eventually stored in an on-chip microcontroller to facilitate efficient on-chip self-healing. The proposed indirect performance sensing and on-chip self-healing methodology is validated by a 25GHz differential Colpitts VCO designed in a 32nm CMOS SOI process. Our silicon measurement data show that the parametric yield of the VCO is significantly improved after applying self-healing. In our future work, we will further extend the proposed indirect performance sensing and on-chip self-healing methodology to other large-scale circuits such as phase-locked loop (PLL).

Chapter 4

Co-Learning Bayesian Model Fusion for Indirect Sensor Modeling

4.1 Motivation

Indirect sensor modeling is one type of performance modeling where the circuit performance is fitted as an analytical function of circuit operation points. To efficiently fit and calibration indirect sensor model, two performance modeling techniques have been proposed in Chapter 4. Sparse regression (SR) is used to efficiently determine model coefficients by reducing the model complexity. Bayesian model fusion (BMF) is developed, which borrows the data generated from an early stage to facilitate efficient performance modeling at a late stage. However, to further reduce the modeling cost, there is a strong need to rethink the fundamental strategy for performance modeling.

The aforementioned performance modeling approaches attempt to take advantage of the *side information* to reduce the number of required samples and, hence, the modeling cost. Here we define the side information as the information that is *not* present in the original performance modeling problem, but is indirectly associated with it. Taking sparse regression [74]-[79] as an example, side information refers to the prior knowledge that the model coefficients are sparse (i.e., most coefficients are close to zero). For BMF that is proposed in Chapter 4, side information indicates that the early-stage and late-stage models are similar and, hence, their model coefficients are close. It is important to note that these conventional approaches only focus on the side information related to model coefficients, referred to as the *coefficient side information* (CSI).

In this Chapter we further improve the efficacy of performance modeling by considering other side information. Towards this goal, we propose a novel technique that is referred to as *Co-Learning Bayesian*

Model Fusion (CL-BMF). CL-BMF considers two different models that predict the same circuit performance. We assume that the complexity of these two models is different: a model with low/high complexity must be trained from a small/large number of samples. The key idea of CL-BMF is to pass the knowledge from the low-complexity model to the high-complexity model (i.e. *co-learning*) to reduce its training cost. In other words, once the low-complexity model is accurately built based on a small number of training samples, the high-complexity model can be trained by borrowing the information from the low-complexity model, instead of relying on the expensive training data only. In our work, the performance model of interest is treated as the high-complexity model, while the low-complexity model provides the *performance side information* (PSI) to reduce the training cost of the high-complexity model.

Mathematically, the proposed CL-BMF method is derived from the Bayesian inference that can be represented as a graphical model [61]. To build a performance model, CL-BMF combines the following information: (i) the PSI which enables co-learning, (ii) the CSI which provides prior knowledge, and (iii) a small number of training samples collected by pre-silicon simulation or post-silicon measurement. Once the Bayesian inference is constructed, the unknown model coefficients can be accurately determined by maximizing the posterior distribution.

For an analog/RF circuit (e.g. a voltage controlled oscillator), the indirect sensor model is an analytical function (e.g. polynomial) of device-level operation point (e.g. DC bias current):

$$y \approx f_1(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m \cdot b_m(\mathbf{x}) \tag{4.1}$$

where y denotes the *performance of interest* (PoI), $\mathbf{x}$ is a vector representing the device-level operation point, f_1 denotes the performance model of interest which establishes a mapping from $\mathbf{x}$ to y, $\{\alpha_m;\ m = 1, 2, ..., M\}$ contains the model coefficients, $\{b_m(\mathbf{x});\ m = 1, 2, ..., M\}$ contains the basis functions (e.g., linear and quadratic polynomials), and M denotes the total number of basis functions.

To determine the performance model in (4.1), the model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ must be solved. Towards this goal, the traditional least-squares fitting method first collects a number of sampling points of $\mathbf{x}$ and y, and then solves the model coefficients from the following optimization problem:

$$\min_{\boldsymbol{\alpha}} \quad \left\| \mathbf{y} - \mathbf{B} \cdot \boldsymbol{\alpha} \right\|_2^2 \tag{4.2}$$

where

$$\mathbf{B}=\begin{bmatrix} b_1\left(\mathbf{x}^{(1)}\right) & b_2\left(\mathbf{x}^{(1)}\right) & \cdots & b_M\left(\mathbf{x}^{(1)}\right) \\ b_1\left(\mathbf{x}^{(2)}\right) & b_2\left(\mathbf{x}^{(2)}\right) & \cdots & b_M\left(\mathbf{x}^{(2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ b_1\left(\mathbf{x}^{(K)}\right) & b_2\left(\mathbf{x}^{(K)}\right) & & b_M\left(\mathbf{x}^{(K)}\right) \end{bmatrix} \tag{4.3}$$

$$\boldsymbol{\alpha}=\begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_M \end{bmatrix}^T \tag{4.4}$$

$$\mathbf{y}=\begin{bmatrix} y^{(1)} & y^{(2)} & \cdots & y^{(K)} \end{bmatrix}^T. \tag{4.5}$$

In (4.3)-(4.5), $\mathbf{x}^{(k)}$ and $y^{(k)}$ are the values of $\mathbf{x}$ and y at the k-th sampling point respectively, K represents the total number of sampling points, and $\|\bullet\|_2$ stands for the L_2-norm of a vector. To avoid over-fitting, the number of sampling points (i.e., K) must be substantially greater than the number of unknown coefficients (i.e., M).

To reduce the number of required sampling points, several advanced performance modeling approaches [53], [74]-[79] have recently been proposed. These approaches take advantage of the CSI (i.e. the extra prior knowledge related to the model coefficients). Taking SR as an example [74]-[79], the model coefficients are solved from the following convex optimization problem:

$$\min_{\boldsymbol{\alpha}} \quad \left\|\mathbf{y}-\mathbf{B}\cdot\boldsymbol{\alpha}\right\|_2^2+\lambda_1\left\|\boldsymbol{\alpha}\right\|_1+\lambda_2\left\|\boldsymbol{\alpha}\right\|_2^2 \tag{4.6}$$

where $\|\bullet\|_1$ stands for the L_1-norm of a vector. In , the CSI is encoded by the regularization terms $\|\boldsymbol{\alpha}\|_1$ and $\|\boldsymbol{\alpha}\|_2$, and λ_1 and λ_2 denote two parameters controlling the regularization. Specifically, the CSI here is the prior knowledge that the model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ should be sparse and do not take any extremely large values.

On the other hand, consider BMF [53] as another example. The late-stage model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ are efficiently solved by borrowing the early-stage model information. The performance modeling problem is then formulated as a maximum-a-posteriori (MAP) estimation:

$$\min_{\boldsymbol{\alpha}} \quad \left\|\mathbf{y}-\mathbf{B}\cdot\boldsymbol{\alpha}\right\|_2^2+\lambda\cdot\boldsymbol{\alpha}^T\cdot\mathbf{D}\cdot\boldsymbol{\alpha} \tag{4.7}$$

where λ denotes a parameter controlling the regularization term, and $\mathbf{D}$ stands for a diagonal matrix learned from the early-stage model. In (4.7), the CSI is encoded in the second term which indicates the similarity

between the early-stage and late-stage models.

While the CSI has been successfully applied to many practical applications, it is possible to exploit other side information to further reduce the performance modeling cost. Motivated by this observation, we will propose a new CL-BMF technique that takes into account the PSI.

4.2 Performance Side Information for Co-Learning

For a given performance modeling task, obtaining the PoI value y at each sampling point is often expensive (e.g., by running a transistor-level simulation). Therefore, to reduce the overall performance modeling cost, it is extremely important to reduce the number of required PoI samples for model fitting. Towards this goal, we propose a co-learning method to generate *pseudo samples* of PoI without running actual circuit simulation or performing physical silicon measurement. These pseudo samples are considered as the PSI in our work.

In particular, we consider a vector of performance metrics $\mathbf{z}$ that satisfies the following three criteria: (i) $\mathbf{z}$ is inexpensive to simulate or measure, (ii) $\mathbf{z}$ can be used to accurately predict the PoI y, and (iii) $\mathbf{z}$ is low-dimensional and, hence, the following model has low complexity and can be accurately learned from a small number of training samples:

$$y \approx f_2(\mathbf{z}) = \sum_{t=1}^{T} \beta_t \cdot c_t(\mathbf{z}) \tag{4.8}$$

where $\{\beta_t;\ t = 1, 2, ..., T\}$ contains the model coefficients, $\{c_t(\mathbf{z});\ t = 1, 2, ..., T\}$ contains the basis functions, and T denotes the total number of basis functions. To simplify our discussion, we further assume that the vector $\mathbf{x}$ in (4.1) and the vector $\mathbf{z}$ in (4.8) do not share any common element. Namely, the i-th element x_i of $\mathbf{x}$ is not identical to the j-th element z_j of $\mathbf{z}$ for any i and j.

The key idea of co-learning is to amalgamate the PSI when fitting $f_1(\mathbf{x})$. Since a much smaller training set is needed to fit $f_2(\mathbf{z})$ than $f_1(\mathbf{x})$, we first learn $f_2(\mathbf{z})$ based on a small number of training samples of y and $\mathbf{z}$. Next, $f_1(\mathbf{x})$ can be trained using the PSI generated from $f_2(\mathbf{z})$, instead of relying on the expensive simulation or measurement samples only.

To elaborate, we consider the co-learning procedure shown in Figure 4-1. First, a small number of *physical samples* $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}, y^{(r)});\ r = 1, 2, ..., R\}$ are collected by simulation or measurement and they are

used to fit the low-complexity model $f_2(\mathbf{z})$ accurately. Next, a large number of samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}); r = R + 1, R + 2, ..., K\}$ are collected, and a set of pseudo samples $\{(\mathbf{x}^{(r)}, f_2(\mathbf{z}^{(r)})); r = R + 1, R + 2, ..., K\}$ are generated for the PoI, as shown by the dashed box in Figure 4-1. Note that since $\mathbf{z}$ is inexpensive to simulation or measure, these pseudo samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}, f_2(\mathbf{z}^{(r)})); r = R + 1, R + 2, ..., K\}$ can be obtained with low cost. Finally, the high-complexity model $f_1(\mathbf{x})$ is fitted by using all the samples, including both the physical samples $\{(\mathbf{x}^{(r)}, y^{(r)}); r = 1, 2, ..., R\}$ and the pseudo samples $\{(\mathbf{x}^{(r)}, f_2(\mathbf{z}^{(r)})); r = R + 1, R + 2, ..., K\}$. By taking advantage of the extra pseudo samples, the aforementioned co-learning is expected to result in a more accurate model $f_1(\mathbf{x})$ than the traditional approach that uses the physical samples $\{(\mathbf{x}^{(r)}, y^{(r)}); r = 1, 2, ..., R\}$ only. Alternatively speaking, applying co-learning can reduce the number of required physical samples and, hence, the overall modeling cost for $f_1(\mathbf{x})$.

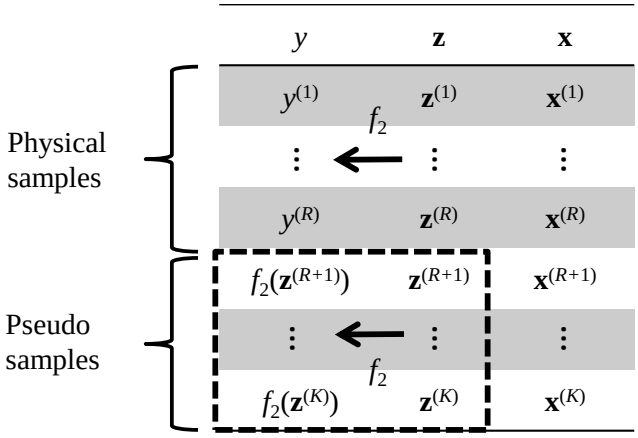


Figure 4-1. The co-learning procedure is illustrated. The low-complexity model $f_2(\mathbf{z})$ is first fitted using a small number of physical samples that are collected by simulation or measurement. Next, a set of pseudo samples are generated for the PoI, as shown by the dashed box. Finally, the high-complexity model $f_1(\mathbf{x})$ is fitted by using all the samples, including both the physical samples and the pseudo samples.

Note that the low-complexity model $f_2(\mathbf{z})$ may not be highly accurate in practice. Hence, directly using the pseudo samples $\{(\mathbf{x}^{(r)}, f_2(\mathbf{z}^{(r)})); r = R + 1, R + 2, ..., K\}$ to fit the high-complexity model $f_1(\mathbf{x})$ may result in large modeling error. It is crucial to develop a statistical framework that can appropriately incorporate the PSI into our regression modeling process. To this end, a Bayesian inference will be constructed and represented by graphical model. We will discuss the proposed Bayesian interface, including the likelihood models of physical and pseudo samples respectively, in the following sub-sections.

4.3 Likelihood Model of Physical Samples

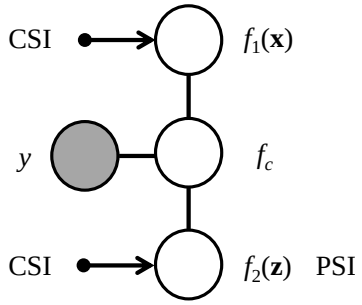


Figure 4-2. A graphical model is shown to statistically model the likelihood of physical samples.

As shown in Figure 4-2, a graphical model is constructed to statistically model the likelihood of physical samples. In the graphical model, each node represents a random quantity, and each directed/undirected edge represents a unidirectional/non-directional dependency. The small solid circle represents the prior information as our CSI. The filled node indicates that the node has been observed (i.e. the physical samples of y have been collected). In the co-learning process, we expect that the two models f_1 and f_2 are consistent with each other because they are predicting the same PoI. In this regard, we explicitly define a consensus function f_c to represent the true value of PoI predicted by f_1 and f_2.

According to the graphical model in Figure 4-2, the joint probability density function (PDF) of f_1, f_2, f_c and y can be represented as:

$$pdf(f_1, f_2, f_c, y) \propto \exp\left[-\frac{(f_1 - f_c)^2}{2\sigma_1^2}\right] \cdot \exp\left[-\frac{(f_2 - f_c)^2}{2\sigma_2^2}\right] \cdot \exp\left[-\frac{(y - f_c)^2}{2\sigma_c^2}\right] \cdot \qquad (4.9)$$

where σ_1, σ_2 and σ_c are three parameters. By integrating over the consensus function f_c, we have:

$$pdf(f_1, f_2, y) \propto \exp\left[-\lambda_1 \cdot (y - f_1)^2\right] \cdot \exp\left[-\lambda_2 \cdot (y - f_2)^2\right] \cdot \exp\left[-\lambda_3 \cdot (f_1 - f_2)^2\right] \cdot \qquad (4.10)$$

where λ_1, λ_2 and λ_3 are three constants depending on σ_1, σ_2 and σ_c.

Given a number of independent physical samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}, y^{(r)}); r = 1, 2, ..., R\}$ collected by pre-silicon simulation or post-silicon measurement, the joint PDF for all these physical samples can be represented as:

$$pdf(\mathbf{f}_{P,1},\mathbf{f}_{P,2},\mathbf{y}_P) \propto \exp\left[-\lambda_1 \cdot \left\|\mathbf{y}_P - \mathbf{f}_{P,1}\right\|_2^2\right] \cdot \exp\left[-\lambda_2 \cdot \left\|\mathbf{y}_P - \mathbf{f}_{P,2}\right\|_2^2\right] \cdot \exp\left[-\lambda_3 \cdot \left\|\mathbf{f}_{P,1} - \mathbf{f}_{P,2}\right\|_2^2\right] \tag{4.11}$$

where

$$\mathbf{y}_P = \left[y^{(1)} \quad y^{(2)} \quad \cdots \quad y^{(R)} \right]^T \tag{4.12}$$

$$\mathbf{f}_{P,1} = \left[f_1(\mathbf{x}^{(1)}) \quad f_1(\mathbf{x}^{(2)}) \quad \cdots \quad f_1(\mathbf{x}^{(R)}) \right]^T \tag{4.13}$$

$$\mathbf{f}_{P,2} = \left[f_2(\mathbf{z}^{(1)}) \quad f_2(\mathbf{z}^{(2)}) \quad \cdots \quad f_2(\mathbf{z}^{(R)}) \right]^T . \tag{4.14}$$

In (4.13)-(4.14), $f_1(\mathbf{x})$ and $f_2(\mathbf{z})$ are defined by (4.1) and (4.8), respectively. Hence, $\mathbf{f}_{p,1}$ and $\mathbf{f}_{p,2}$ can be re-written as:

$$\mathbf{f}_{P,1} = \mathbf{B}_P \cdot \boldsymbol{\alpha} \tag{4.15}$$

$$\mathbf{f}_{P,2} = \mathbf{C}_P \cdot \boldsymbol{\beta} \tag{4.16}$$

where

$$\boldsymbol{\beta} = \left[\beta_1 \quad \beta_2 \quad \cdots \quad \beta_T \right]^T \tag{4.17}$$

$$\mathbf{B}_P = \begin{bmatrix} b_1\left(\mathbf{x}^{(1)}\right) & b_2\left(\mathbf{x}^{(1)}\right) & \cdots & b_M\left(\mathbf{x}^{(1)}\right) \\ b_1\left(\mathbf{x}^{(2)}\right) & b_2\left(\mathbf{x}^{(2)}\right) & \cdots & b_M\left(\mathbf{x}^{(2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ b_1\left(\mathbf{x}^{(R)}\right) & b_2\left(\mathbf{x}^{(R)}\right) & & b_M\left(\mathbf{x}^{(R)}\right) \end{bmatrix} \tag{4.18}$$

$$\mathbf{C}_P = \begin{bmatrix} c_1\left(\mathbf{z}^{(1)}\right) & c_2\left(\mathbf{z}^{(1)}\right) & \cdots & c_T\left(\mathbf{z}^{(1)}\right) \\ c_1\left(\mathbf{z}^{(2)}\right) & c_2\left(\mathbf{z}^{(2)}\right) & \cdots & c_T\left(\mathbf{z}^{(2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ c_1\left(\mathbf{z}^{(R)}\right) & c_2\left(\mathbf{z}^{(R)}\right) & & c_T\left(\mathbf{z}^{(R)}\right) \end{bmatrix}. \tag{4.19}$$

Substituting (4.15)-(4.19) into the joint PDF in (4.11) yields the likelihood function:

$$pdf(\mathbf{f}_{P,1},\mathbf{f}_{P,2},\mathbf{y}_P \mid \boldsymbol{\alpha},\boldsymbol{\beta}) \propto \exp\left[-\lambda_1 \cdot \left\|\mathbf{y}_P - \mathbf{B}_P \cdot \boldsymbol{\alpha}\right\|_2^2\right] \cdot \exp\left[-\lambda_2 \cdot \left\|\mathbf{y}_P - \mathbf{C}_P \cdot \boldsymbol{\beta}\right\|_2^2\right] \cdot \exp\left[-\lambda_3 \cdot \left\|\mathbf{B}_P \cdot \boldsymbol{\alpha} - \mathbf{C}_P \cdot \boldsymbol{\beta}\right\|_2^2\right] . \tag{4.20}$$

The likelihood function in (4.20) consists of three L_2-norm terms. The first term represents the difference between the physical samples of y and the approximated function $f_1(\mathbf{x})$, the second term represents the

difference between the physical samples of y and the approximated function $f_2(\mathbf{z})$, and the third term represents the difference between $f_1(\mathbf{x})$ and $f_2(\mathbf{z})$. The aforementioned likelihood function will be further used to solve the Bayesian inference in Section 4.5.

4.4 Likelihood Model of Pseudo Samples

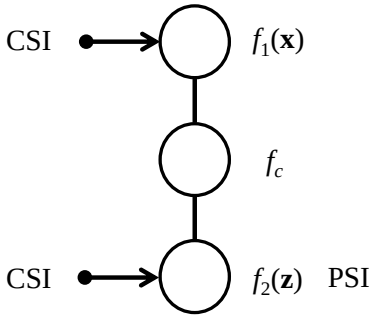


Figure 4-3. A graphical model is shown to statistically model the likelihood of pseudo samples.

As shown in Figure 4-3, a graphical model is constructed to statistically model the likelihood of pseudo samples. According to the graphical model in Figure 4-3, the joint PDF of f_1, f_2 and f_c can be represented as:

$$pdf(f_1, f_2, f_c) \propto \exp\left[-\frac{(f_1 - f_c)^2}{2\sigma_1^2}\right] \cdot \exp\left[-\frac{(f_2 - f_c)^2}{2\sigma_2^2}\right]. \tag{4.21}$$

By integrating over the consensus function f_c, we have:

$$pdf(f_1, f_2) \propto \exp\left[-\lambda_4 \cdot (f_1 - f_2)^2\right] \tag{4.22}$$

where λ_4 is a constant depending on σ_1 and σ_2. Given a number of independent pseudo samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}); r = R + 1, R + 2, ..., K\}$, the joint PDF for all these pseudo samples can be represented as:

$$pdf(\mathbf{f}_{S,1}, \mathbf{f}_{S,2}) \propto \exp\left[-\lambda_4 \cdot \left\|\mathbf{f}_{S,1} - \mathbf{f}_{S,2}\right\|_2^2\right] \tag{4.23}$$

where

$$\mathbf{f}_{S,1} = \left[f_1(\mathbf{x}^{(R+1)}) \quad f_1(\mathbf{x}^{(R+2)}) \quad \cdots \quad f_1(\mathbf{x}^{(K)}) \right]^T \tag{4.24}$$

$$\mathbf{f}_{S,2} = \left[f_2(\mathbf{z}^{(R+1)}) \quad f_2(\mathbf{z}^{(R+2)}) \quad \cdots \quad f_2(\mathbf{z}^{(K)}) \right]^T \tag{4.25}$$

In (4.24)-(4.25), $f_1(\mathbf{x})$ and $f_2(\mathbf{z})$ are defined by (4.1) and (4.8), respectively. Hence, $\mathbf{f}_{s,1}$ and $\mathbf{f}_{s,2}$ can be re-written as:

$$\mathbf{f}_{S,1} = \mathbf{B}_S \cdot \boldsymbol{\alpha} \tag{4.26}$$

$$\mathbf{f}_{S,2} = \mathbf{C}_S \cdot \boldsymbol{\beta} \tag{4.27}$$

where

$$\mathbf{B}_S = \begin{bmatrix} b_1\left(\mathbf{x}^{(R+1)}\right) & b_2\left(\mathbf{x}^{(R+1)}\right) & \cdots & b_M\left(\mathbf{x}^{(R+1)}\right) \\ b_1\left(\mathbf{x}^{(R+2)}\right) & b_2\left(\mathbf{x}^{(R+2)}\right) & \cdots & b_M\left(\mathbf{x}^{(R+2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ b_1\left(\mathbf{x}^{(K)}\right) & b_2\left(\mathbf{x}^{(K)}\right) & & b_M\left(\mathbf{x}^{(K)}\right) \end{bmatrix} \tag{4.28}$$

$$\mathbf{C}_S = \begin{bmatrix} c_1\left(\mathbf{z}^{(R+1)}\right) & c_2\left(\mathbf{z}^{(R+1)}\right) & \cdots & c_T\left(\mathbf{z}^{(R+1)}\right) \\ c_1\left(\mathbf{z}^{(R+2)}\right) & c_2\left(\mathbf{z}^{(R+2)}\right) & \cdots & c_T\left(\mathbf{z}^{(R+2)}\right) \\ \vdots & \vdots & \vdots & \vdots \\ c_1\left(\mathbf{z}^{(K)}\right) & c_2\left(\mathbf{z}^{(K)}\right) & & c_T\left(\mathbf{z}^{(K)}\right) \end{bmatrix}. \tag{4.29}$$

Substituting (4.26)-(4.27) into the joint PDF in (4.23) yields the likelihood function:

$$pdf\left(\mathbf{f}_{S,1}, \mathbf{f}_{S,2} \mid \boldsymbol{\alpha}, \boldsymbol{\beta}\right) \propto \exp\left[-\lambda_4 \cdot \left\|\mathbf{B}_S \cdot \boldsymbol{\alpha} - \mathbf{C}_S \cdot \boldsymbol{\beta}\right\|_2^2\right]. \tag{4.30}$$

The likelihood function in (4.30) contains only one L_2-norm term, which represents the difference between $f_1(\mathbf{x})$ and $f_2(\mathbf{z})$. It encodes our prior knowledge that the function values approximated by $f_1(\mathbf{x})$ and $f_2(\mathbf{z})$ are "likely" to be similar. The likelihood function in (4.30) will be combined with the likelihood function in (4.20) to solve the Bayesian inference in Section 4.5.

4.5 Bayesian Inference for Co-Learning

Based on the likelihood function defined in Section 4.3 and 4.4, the model coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ can be optimally determined by MAP estimation through Bayesian inference [61]. Given that the physical and pseudo samples are independently generated, the likelihood function in (4.20) and (4.30) can be combined:

$$
\begin{aligned}
&pdf\left(\mathbf{f}_{P,1},\mathbf{f}_{P,2},\mathbf{y}_P,\mathbf{f}_{S,1},\mathbf{f}_{S,2}\,|\,\boldsymbol{\alpha},\boldsymbol{\beta}\right)\propto\exp\left[-\lambda_1\cdot\left\|\mathbf{y}_P-\mathbf{B}_P\cdot\boldsymbol{\alpha}\right\|_2^2\right]\\
&\cdot\exp\left[-\lambda_2\cdot\left\|\mathbf{y}_P-\mathbf{C}_P\cdot\boldsymbol{\beta}\right\|_2^2\right]\cdot\exp\left[-\lambda_3\cdot\left\|\mathbf{B}_P\cdot\boldsymbol{\alpha}-\mathbf{C}_P\cdot\boldsymbol{\beta}\right\|_2^2\right]\\
&\cdot\exp\left[-\lambda_4\cdot\left\|\mathbf{B}_S\cdot\boldsymbol{\alpha}-\mathbf{C}_S\cdot\boldsymbol{\beta}\right\|_2^2\right]
\end{aligned}
\quad . \tag{4.31}
$$

We statistically represent the CSI of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as a PDF that is referred to as the prior distribution. Here, we define the prior distribution of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in a general form:

$$
pdf\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right). \tag{4.32}
$$

In (4.32), the prior knowledge is encoded in the PDF template (e.g., Gaussian distribution, Laplace distribution, etc.) and/or the PDF parameters (e.g. mean and standard deviation for a Gaussian distribution). More details about the prior definition will be discussed in Section xxx.

Once the prior distribution $pdf(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is specified, we will combine $pdf(\boldsymbol{\alpha}, \boldsymbol{\beta})$ with the physical samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}, y^{(r)}); r = 1, 2, ..., R\}$ and the pseudo samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}); r = R + 1, R + 2, ..., K\}$ to solve the model coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by MAP estimation [61]. The key idea of MAP estimation is to find the optimal values of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ to maximize the posterior distribution $pdf(\boldsymbol{\alpha}, \boldsymbol{\beta} \mid \mathbf{f}_{p,1}, \mathbf{f}_{p,2}, \mathbf{y}_p, \mathbf{f}_{s,1}, \mathbf{f}_{s,2})$. Intuitively, the posterior distribution indicates the remaining uncertainty of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, after we observe all physical and pseudo samples. Hence, MAP attempts to find the optimal $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ that are most likely to occur.

Based on Bayes' theorem, the posterior distribution can be represented as:

$$
\begin{aligned}
&pdf\left(\boldsymbol{\alpha},\boldsymbol{\beta}\,|\,\mathbf{f}_{P,1},\mathbf{f}_{P,2},\mathbf{y}_P,\mathbf{f}_{S,1},\mathbf{f}_{S,2}\right)\propto\\
&\quad pdf\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right)\cdot pdf\left(\mathbf{f}_{P,1},\mathbf{f}_{P,2},\mathbf{y}_P,\mathbf{f}_{S,1},\mathbf{f}_{S,2}\,|\,\boldsymbol{\alpha},\boldsymbol{\beta}\right)
\end{aligned}
\quad . \tag{4.33}
$$

Mathematically, the MAP solution can be found by solving the following optimization problem:

$$
\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad pdf\left(\boldsymbol{\alpha},\boldsymbol{\beta}\,|\,\mathbf{f}_{P,1},\mathbf{f}_{P,2},\mathbf{y}_P,\mathbf{f}_{S,1},\mathbf{f}_{S,2}\right)\cdot \tag{4.34}
$$

Combining (4.31)-(4.34) and taking the logarithm for the posterior distribution, we can convert (4.34) to the following equivalent optimization problem:

$$
\begin{aligned}
\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & -\log\left[pdf\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right)\right]+\lambda_1\cdot\left\|\mathbf{y}_P-\mathbf{B}_P\cdot\boldsymbol{\alpha}\right\|_2^2\\
&+\lambda_2\cdot\left\|\mathbf{y}_P-\mathbf{C}_P\cdot\boldsymbol{\beta}\right\|_2^2+\lambda_3\cdot\left\|\mathbf{B}_P\cdot\boldsymbol{\alpha}-\mathbf{C}_P\cdot\boldsymbol{\beta}\right\|_2^2\\
&+\lambda_4\cdot\left\|\mathbf{B}_S\cdot\boldsymbol{\alpha}-\mathbf{C}_S\cdot\boldsymbol{\beta}\right\|_2^2
\end{aligned}
\quad . \tag{4.35}
$$

In (4.35), the model coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are solved together. The first term in the cost function represents the prior knowledge which encodes the CSI. The second term penalizes the difference between the physical

samples of y and the approximated function $f_1(\mathbf{x})$. The first two terms resemble the traditional regression approaches as described in (4.6) and (4.7). The third term penalizes the difference between the physical samples of y and the approximated function $f_2(\mathbf{z})$, and the fourth and fifth terms penalize the inconsistency between $f_1(\mathbf{x})$ and $f_2(\mathbf{z})$ (i.e. the PSI) for physical and pseudo samples respectively.

While the basic idea of likelihood modeling and Bayesian inference for co-learning is illustrated in this section, several implementation issues must be carefully considered in order to make CL-BMF of practical utility. These implementation details will be further discussed in the next section.

4.6 Implementation Details

In this section, we further discuss several important implementation issues for CL-BMF, including (i) prior definition, and (ii) cross-validation.

4.6.1 Prior Definition

The prior knowledge (i.e. the CSI) of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is statistically encoded by the prior distribution $pdf(\boldsymbol{\alpha}, \boldsymbol{\beta})$ in (4.32). In this sub-section, we consider two special cases: (i) the Laplace prior distribution, and (ii) the Gaussian prior distribution. These prior distributions have been extensively used in the literature [53], [74]-[79]. It should be noted that the proposed CL-BMF framework can be easily extended to accommodate other prior distributions [88]-[90], even though the details of these possible extensions are not included in this thesis.

1) Laplace prior distribution: The Laplace prior distribution is defined as:

$$pdf(\boldsymbol{\alpha},\boldsymbol{\beta}) \propto \exp\left[-\kappa \cdot \left\| \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix} \right\|_1\right] = \exp\left[-\kappa \cdot \|\boldsymbol{\alpha}\|_1\right] \cdot \exp\left[-\kappa \cdot \|\boldsymbol{\beta}\|_1\right] \tag{4.36}$$

where κ is a hyper-parameter and $\|\bullet\|_1$ represents the L_1-norm of a vector. The Laplace prior distribution attempts to promote sparsity for the model coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. In other words, by applying the Laplace prior distribution, the MAP solution of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is likely to be sparse. Combining and (4.36), we obtain the following optimization problem:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad \|\boldsymbol{\alpha}\|_1 + \|\boldsymbol{\beta}\|_1 + \frac{\lambda_1}{\kappa}\cdot\|\mathbf{y}_P - \mathbf{B}_P\cdot\boldsymbol{\alpha}\|_2^2 + \frac{\lambda_2}{\kappa}\cdot\|\mathbf{y}_P - \mathbf{C}_P\cdot\boldsymbol{\beta}\|_2^2 + \frac{\lambda_3}{\kappa}\cdot\|\mathbf{B}_P\cdot\boldsymbol{\alpha} - \mathbf{C}_P\cdot\boldsymbol{\beta}\|_2^2 + \frac{\lambda_4}{\kappa}\cdot\|\mathbf{B}_S\cdot\boldsymbol{\alpha} - \mathbf{C}_S\cdot\boldsymbol{\beta}\|_2^2 \tag{4.37}$$

It is straightforward to verify that the optimization problem in (4.37) is convex and, hence, can be efficiently solved to find its global optimum.

2) Gaussian prior distribution: In the case where the CSI contains the magnitude information of the model coefficients from an early stage, a Gaussian prior distribution can be used. To this end, we consider the following models from the early-stage:

$$y \approx f_{E,1}(\mathbf{x}) = \sum_{m=1}^{M} \alpha_{E,m}\cdot b_m(\mathbf{x}) \tag{4.38}$$

$$y \approx f_{E,2}(\mathbf{z}) = \sum_{t=1}^{T} \beta_{E,t}\cdot c_t(\mathbf{z}) \tag{4.39}$$

where $f_{E,1}$ and $f_{E,2}$ denote the early-stage versions of f_1 and f_2, $\{\alpha_{E,m};\ m = 1, 2, ..., M\}$ contains the model coefficients of $f_{E,1}$, and $\{\beta_{E,t};\ t = 1, 2, ..., T\}$ contains the model coefficients of $f_{E,2}$. We expect that the model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ and $\{\beta_t;\ t = 1, 2, ..., T\}$ are close to $\{\alpha_{E,m};\ m = 1, 2, ..., M\}$ and $\{\beta_{E,t};\ t = 1, 2, ..., T\}$ respectively, since they are associated with the same PoI. Following this assumption, the joint prior distribution of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ can be defined as a multivariate Gaussian distribution:

$$pdf(\boldsymbol{\alpha},\boldsymbol{\beta}) \propto \exp\left[-\frac{1}{2}\cdot\left(\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}\right)^T\cdot\mathbf{D}\cdot\left(\begin{bmatrix}\boldsymbol{\alpha}\\ \boldsymbol{\beta}\end{bmatrix}\right)\right] \tag{4.40}$$

where

$$\mathbf{D} = \begin{bmatrix}\mathbf{D}_{\boldsymbol{\alpha}} & \mathbf{0}\\ \mathbf{0} & \mathbf{D}_{\boldsymbol{\beta}}\end{bmatrix} \tag{4.41}$$

$$\mathbf{D}_{\boldsymbol{\alpha}} = \mathrm{diag}\left(\alpha_{E,1}^{-2}, \alpha_{E,2}^{-2}, \cdots, \alpha_{E,M}^{-2}\right) \tag{4.42}$$

$$\mathbf{D}_{\boldsymbol{\beta}} = \mathrm{diag}\left(\beta_{E,1}^{-2}, \beta_{E,2}^{-2}, \cdots, \beta_{E,T}^{-2}\right). \tag{4.43}$$

In (4.42)-(4.43), diag(•) represents the operator to construct a diagonal matrix and $\mathbf{D}$ is the diagonal covariance matrix of the prior distribution encoding our CSI. Combining (4.35) and (4.40), we obtain the

following optimization problem:

$$\begin{aligned}\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \boldsymbol{\alpha}^T \cdot \mathbf{D}_{\boldsymbol{\alpha}} \cdot \boldsymbol{\alpha} + \boldsymbol{\beta}^T \cdot \mathbf{D}_{\boldsymbol{\beta}} \cdot \boldsymbol{\beta} + \lambda_1 \cdot \left\| \mathbf{y}_P - \mathbf{B}_P \cdot \boldsymbol{\alpha} \right\|_2^2 \\ & + \lambda_2 \cdot \left\| \mathbf{y}_P - \mathbf{C}_P \cdot \boldsymbol{\beta} \right\|_2^2 + \lambda_3 \cdot \left\| \mathbf{B}_P \cdot \boldsymbol{\alpha} - \mathbf{C}_P \cdot \boldsymbol{\beta} \right\|_2^2 \cdot \\ & + \lambda_4 \cdot \left\| \mathbf{B}_S \cdot \boldsymbol{\alpha} - \mathbf{C}_S \cdot \boldsymbol{\beta} \right\|_2^2 \end{aligned} \tag{4.44}$$

Eq. (4.44) represents a convex quadratic programing problem. Its globally optimal solution can be analytically solved by using the first-order optimality condition [85].

4.6.2 Cross-Validation

To solve the model coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ in (4.37) or (4.44), the parameters λ_1, λ_2, λ_3, λ_4 and κ must be carefully determined. First, we consider the optimization problem in (4.44) with four parameters λ_1, λ_2, λ_3 and λ_4. Note that these four parameters are not independent because they can be uniquely determined by σ_1, σ_2 and σ_c from (4.9) and (4.21). Here we assume that the simulation or measurement noise is negligible for y. Namely, the consensus function f_c (i.e. the true value of PoI) is equal to y (i.e. the observed value of PoI). In this case, the parameter σ_c can be set to a value that is close to zero and only two other parameters σ_1 and σ_2 should be determined. Second, we consider the optimization problem in (4.37) with five parameters λ_1, λ_2, λ_3, λ_4 and κ. It is straightforward to note that the parameter κ can be absorbed into λ_1, λ_2, λ_3 and λ_4. Therefore, we only need to determine two parameters $\kappa \cdot \sigma_1$ and $\kappa \cdot \sigma_2$ eventually.

It is important to find the optimal values of the aforementioned two parameters so that the modeling error is minimized. Towards this goal, we use N-fold cross-validation to estimate the modeling error for different parameter values. In particular, we partition the entire data set into N groups. Modeling error is estimated from N independent runs using the physical samples. In each run, one of the N groups is used to estimate the modeling error and all other groups are used to calculate the model coefficients. In addition, different groups should be selected for error estimation in different runs. As such, each run gives an error value e_n (n = 1, 2, ..., N) that is estimated from a unique group of data points. The final modeling error is computed as the average of $\{e_n; n = 1, 2, ..., N\}$, i.e., $e = (e_1 + e_2 + ... + e_N)/N$.

4.6.3 Side Information Selection

To exploit the benefit of PSI, it is important to appropriately identify a vector of performance metrics **z** that satisfies the criteria listed in Section 4.2. In practice, these performance metrics can be selected by an analog/RF designer according to his design experience. However, in some cases, the performance metrics **z** is not very clear, or a large number of performance metrics can be candidates of **z** and it is not possible to include all candidates. To address this issue, we propose to use sparse regression discussed in Section 3.3 to automatically select the important performance metrics in **z** from a set of candidates **w**. In particular, we are interested in the sparse solution of the following regression problem:

$$y \approx f_2(\mathbf{w}) = \sum_{l=1}^{L} \beta_l \cdot c_l(\mathbf{w}), \tag{4.45}$$

where $\{c_l(\mathbf{w}); l = 1, \ldots, L\}$ contains the basis function candidates.

Towards this goal, we first collect a number of pre-silicon simulation samples $\{(\mathbf{w}^{(n)}, y^{(n)}); n = 1, \ldots, N\}$, where **w** represent a candidate set where **z** is chosen from, $\mathbf{w}^{(n)}$ and $y^{(n)}$ denote the values of **w** and y for the n-th sampling point respectively, and N denotes the total number of sampling points. Based on these sampling points, a set of linear equations can be expressed as:

$$\mathbf{C} \cdot \boldsymbol{\beta} = \mathbf{f}. \tag{4.46}$$

where

$$\mathbf{C} = \begin{bmatrix} c_1(\mathbf{w}^{(1)}) & c_2(\mathbf{w}^{(1)}) & \cdots & c_L(\mathbf{w}^{(1)}) \\ c_1(\mathbf{w}^{(2)}) & c_2(\mathbf{w}^{(2)}) & \cdots & c_L(\mathbf{w}^{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ c_1(\mathbf{w}^{(N)}) & c_2(\mathbf{w}^{(N)}) & \cdots & c_L(\mathbf{w}^{(N)}) \end{bmatrix}. \tag{4.47}$$

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 & \beta_2 & \cdots & \beta_L \end{bmatrix}^T. \tag{4.48}$$

$$\mathbf{f} = \begin{bmatrix} f^{(1)} & f^{(2)} & \cdots & f^{(N)} \end{bmatrix}^T. \tag{4.49}$$

To find the important basis functions, we propose to use SR which solves the following L_1-norm regularization problem:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{minimize}} \quad & \left\| \mathbf{C} \cdot \boldsymbol{\beta} - \mathbf{f} \right\|_2^2 \\ \text{subject to} \quad & \left\| \boldsymbol{\beta} \right\|_1 \le \lambda \end{aligned} \tag{4.50}$$

where $\|\bullet\|_1$ denotes the L_1-norm of a vector, and $\lambda > 0$ is a user-defined parameter. The formulation in is a convex optimization problem and can be solved both efficiently (i.e., with low computational cost) and robustly (i.e., with guaranteed global optimum). When λ decreases, the number of coefficients with non-zero magnitude will decrease. The final set of important basis functions can be selected according to the requirement on the model accuracy and model complexity for a particular application.

4.6.4 Summary

Algorithm 4 summarizes the major steps of the proposed CL-BMF method. It consists of two major components: (i) prior distribution definition, and (ii) MAP estimation. CL-BMF can automatically assign the appropriate weights to the PSI, the CSI and the physical samples by optimally tuning the parameters λ_1, λ_2, λ_3, λ_4 and κ in (4.37) or (4.44) based on cross-validation. If the PSI or CSI is not accurate, cross-validation can automatically assign a small weight value to the inaccurate side information. As such, the performance model $f_1(\mathbf{x})$ will not be distorted due to the inaccurate side information.

Algorithm 4: Co-Learning Bayesian Model Fusion (CL-BMF)

1. Select vector $\mathbf{z}$ based on design expertise or sparse regression.
2. According to the CSI, define the prior distribution for the model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ and $\{\beta_t;\ t = 1, 2, ..., T\}$ by (4.36) or (4.40).
3. Collect the physical samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)}, y^{(r)});\ r = 1, 2, ..., R\}$ and the pseudo samples $\{(\mathbf{x}^{(r)}, \mathbf{z}^{(r)});\ r = R + 1, R + 2, ..., K\}$.
4. Solve the model coefficients $\{\alpha_m;\ m = 1, 2, ..., M\}$ and $\{\beta_t;\ t = 1, 2, ..., T\}$ based on the optimization problem in (4.37) or (4.44) where the parameters λ_1, λ_2, λ_3, λ_4 and κ are determined by cross-validation.

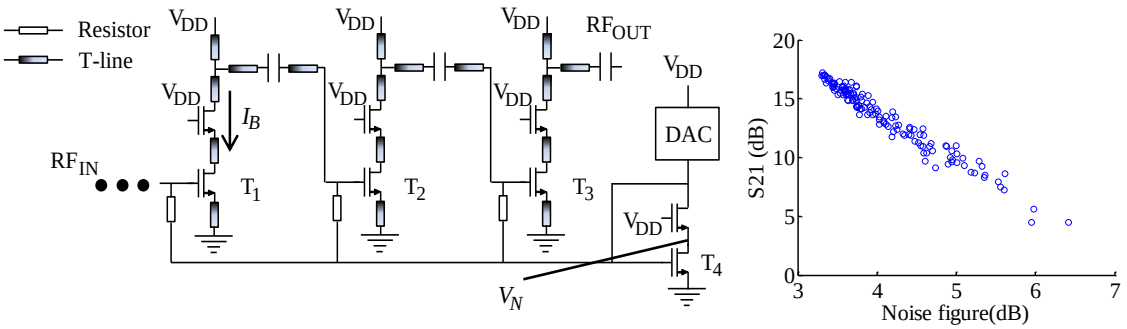
4.7 Numerical Results

In this section, we demonstrate the efficacy of CL-BMF by several circuit examples designed in a

commercial 32nm SOI CMOS process. Our objective is to build indirect sensor models for post-silicon tuning of these circuits. For testing and comparison purposes, two different performance modeling techniques are implemented: (i) Bayesian model fusion using Gaussian prior (BMF), and (ii) CL-BMF. Here, BMF method is chosen for comparison, since they are among the state-of-the-art techniques in the literature.

In our experiments, a two-level cross-validation is used. In the inner loop, the first-level cross-validation is used to determine the parameters λ_1, λ_2, λ_3, λ_4 and κ. In the outer loop, the second-level cross-validation is applied to estimate the modeling error. All numerical experiments are run on a 2.53GHz Linux server with 64GB memory.

4.7.1 Low-Noise Amplifier



(a) (b)

Figure 4-4. The simplified circuit schematic is shown in (a) for a three-stage 60GHz low-noise amplifier (LNA) designed in a commercial 32nm SOI CMOS process. The scatter plot of noise figure vs. S21 is shown in (b).

In this sub-section, we consider a tunable 60GHz low-noise amplifier (LNA) designed in a commercial 32nm SOI CMOS process. The simplified circuit schematic of the LNA is shown in Figure 4-4(a). It consists of three stages and their bias currents are tunable. Our objective is to efficiently measure the circuit performance metrics by low-cost on-chip sensors and then adaptively tune the bias currents to maximize the performance and/or reduce the power of the LNA.

Towards this goal, indirect performance sensing has been proposed in the literature to estimate the

performance metrics that cannot be easily measured by on-chip sensor, as we discussed in Chapter 4. In this example, we consider the noise figure (NF) of the LNA as our PoI. Instead of directly measuring the NF, the key idea of indirect performance sensing is to predict its value by using a set of other performance metrics that are referred to as the performances of measurements (i.e. PoMs). The choices of PoMs must satisfy the following two criteria. First, the PoMs should be highly correlated with the NF. Second, the PoMs can be easily measured by low-cost on-chip sensors.

To predict the NF of the LNA in Figure 4-4, we choose three performance metrics as the PoMs based on our design knowledge: (i) the bias current of the first stage (I_B), (ii) the drain voltage of the transistor T_4 (V_N), and (iii) the environmental temperature (*Temp*). Once the PoMs are chosen, we need to further build a performance model to approximate the NF as a polynomial function of the PoMs.

Table 4-1. Side information for performance modeling of LNA

	Description	Distribution	BMF	CL-BMF
CSI	Magnitude	Gaussian prior	Yes	Yes
PSI	S21	Gaussian likelihood	No	Yes

Table 4-1 summarizes the side information for the aforementioned performance modeling problem. At the pre-silicon stage, a performance model is fitted based on the simulation data generated by random sampling. The magnitude of the coefficients of this pre-silicon model is considered as our CSI, which is encoded by a Gaussian prior distribution.

At the post-silicon stage, both the NF and the PoMs (i.e. I_B, V_N and *Temp*) are measured from a number of chips by wafer probe test. Here, we have to measure the NF by wafer probe, because it is a high-frequency performance metric at 60GHz and, hence, cannot be measured at the package level. In addition, each chip must be measured individually at such a high frequency, implying that the measurement cost is prohibitively high for this LNA example.

Table 4-2. Performance modeling error and cost for LNA

	BMF	CL-BMF
Modeling error (dB)	0.28	0.28
Number of measured chips for NF	5	1
Measurement time for NF (Sec.)	250	50
Number of measured chips for S21	0	13
Measurement time for S21 (Sec.)	0	0.0039
Alignment time for wafer probe (Sec.)	0.63	1.63
Measurement cost (Sec.)	250.63	51.63
Fitting cost (Sec.)	0.011	0.241
Overall modeling cost (Sec.)	250.6	51.9

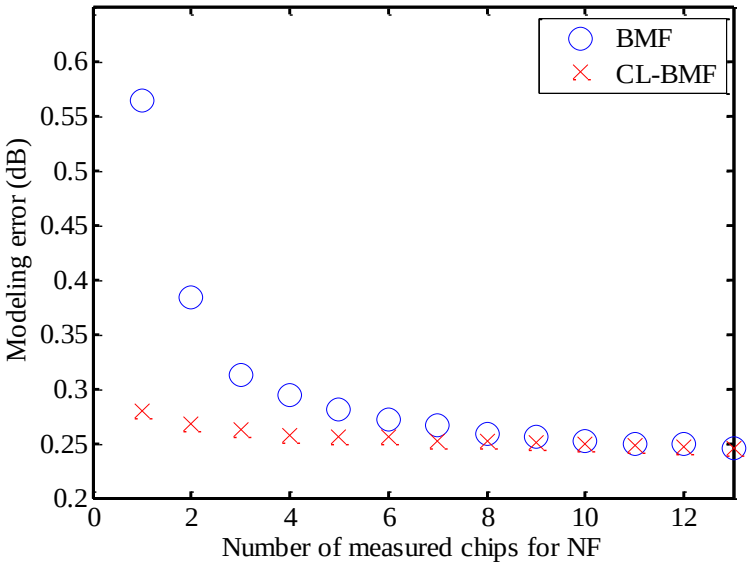


Figure 4-5. The performance modeling error of noise figure (NF) is shown as a function of the number of measured chips for NF.

For testing and comparison purposes, we build the performance models for NF by two different approaches: (i) BMF, and (ii) CL-BMF. When CL-BMF is applied, we consider the S21 as our alternative

performance metric to define the PSI, because it is strongly correlated with the NF according to our design experience and measuring the S21 is significantly less expensive than the NF for high-frequency wafer probe test. In our experiment, the S21 values measured from 13 chips are used to build the likelihood function for PSI. The correlation between S21 and NF is shown in Figure 4-4(b). In this example, S21 is strongly correlated with NF because the transistors have large sizes and S11 is not sensitive to process variations.

Figure 4-5 and Table 4-2 compare the modeling error and cost for BMF and CL-BMF. In this example, the overall modeling consists of four major parts: (i) the measurement cost for NF, (ii) the measurement cost for S21, (iii) the measurement cost for probe alignment when testing each die, and (iv) the fitting cost for solving the unknown model coefficients. As shown in Table 4-2, the overall modeling cost is dominated by the measurement cost for NF. CL-BMF requires less measurement data for NF and, hence, has lower modeling cost than BMF. In this example, CL-BMF achieves 5× speed-up over BMF in terms of the overall modeling cost.

4.7.2 Down-Conversion Mixer

In this sub-section, we consider a down-conversion mixer example designed in a commercial 32nm CMOS process. The current of this mixer is tunable with 13 different knob configurations. In this example, input referred 1dB compression point (I1dBCP) is considered as PoI. The DC drain voltage of input transistor and peak detector measurements at -40dBm, -20dBm and -4dBm are considered as PoM. The goal here is to efficiently fit ISM for I1dBCP. The I1dBCP is an expensive measurement, because the input power of the mixer needs to be swept to extract accurate I1dBCP, as shown in Figure 4-6. In this circuit example, the input power sweeping is performed as -40:0.5:0dBm. In other words, 81 power gain measurements are required to accurately extract I1dBCP.

We use CL-BMF to facilitate the efficient modeling. In this example, the power gain (PG) from all possible input power range can potentially serve as $\mathbf{z}$ which provides PSI. However, if all power gain measurements are considered as $\mathbf{z}$, the cost of $\mathbf{z}$ would be the same as the cost of I1dBCP. Therefore, a subset of power gain measurements must be selected which is capable of accurately predicting I1dBCP.

Towards this goal, we apply the side information selection technique as described in (4.45)-(4.50).

In particular, we consider all power gain simulations with different input power as **w**. The sparse regression is then applied to optimally find a subset of **w** that can well predict I1dBCP. Five different power gain measurements with different input power are selected as **z**, including -40dBm, -20dBm, -4dBm, -2dBm and 0dBm. Given that we do not have the silicon measurement data of down-conversion mixer, we use simulation data to validate the CL-BMF for ISM fitting. The **z** simulation data from 40 chips are collected to provide performance side information.

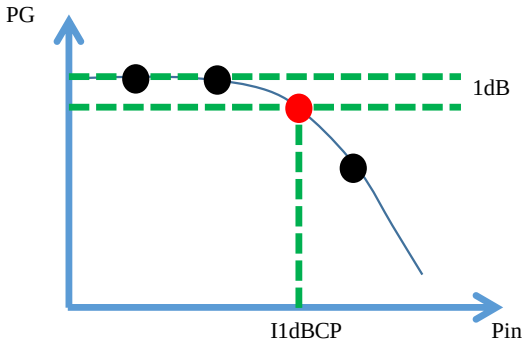


Figure 4-6. I1dBCP measurement procedure is shown. In order to accurately extract I1dBCP, the input power needs to be increased step by step until power gain is compressed by 1dB.

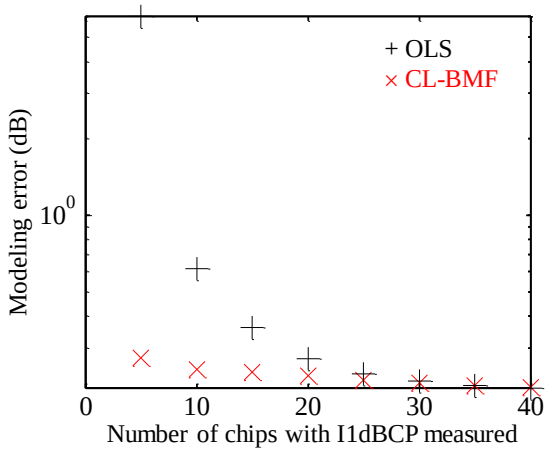


Figure 4-7. Indirect sensor modeling error is compared for OLS and CL-BMF.

In Figure 4-7, the indirect sensor modeling error of (i) OLS and (ii) CL-BMF are compared. Here OLS is selected for comparison because only model template of ISM is known. Studying Figure 4-7 yields several observations: (i) with the number of samples increases, the modeling error of both methods

decreases, (ii) with the same number of samples, CL-BMF achieves significantly less modeling error than OLS.

Table 4-3 further compares the indirect sensor modeling error and cost for OLS and CL-BMF. The overall modeling cost consists of two major components: (i) the measurement cost for I1dBCP, and (ii) the measurement cost for PG. The overall modeling cost is dominated by the measurement cost of I1dBCP. CL-BMF requires significantly less I1dBCP and, hence, has much lower modeling cost.

Table 4-3. Indirect sensor modeling cost of OLS and CL-BMF (normalized)

	OLS	CL-BMF
Error (dB)	0.27	0.27
I1dBCP measurements	260	65
PG measurements	0	2400
Total number of PG measurements	20800	7600

4.8 Summary

In this Chapter, a novel CL-BMF algorithm is proposed for efficient indirect sensor modeling of self-healing circuits. CL-BMF optimally combines the following information: (i) the CSI, (ii) the PSI, and (iii) a small number of training samples. Bayesian inference is constructed and represented as a graphical model, where the CSI and the PSI are encoded by the prior distribution and the likelihood function respectively. From the Bayesian inference, the unknown model coefficients can be accurately determined by maximizing the posterior distribution. The side information technique is also discussed. As is demonstrated by our circuit examples designed in a commercial 32nm SOI CMOS process, the proposed CL-BMF method achieves up to 5× cost reduction over other state-of-the-art indirect sensor modeling techniques. It should be noted that CL-BMF is also applicable to other performance modeling tasks, since it is formulated from a rigid mathematical framework.

Chapter 5

Cost Analysis of On-Chip Self-Healing

5.1 Motivation

Although the idea of on-chip self-healing has been proposed for many years, its benefit on the overall product cost has not been studied in the literature. Compared to the traditional design without tuning capacity, the on-chip self-healing is able to achieve better parametric yield. However, the hardware cost overhead and testing cost overhead will be high. Compared to the tunable design with off-chip performance tuning, the on-chip self-healing will achieve lower parametric yield due to limited accuracy in on-chip performance sensors. However, the testing cost, especially the performance tuning cost, will be lower. Therefore, it is still unclear to circuit designs when on-chip self-healing is appropriate in the cost perspective.

The rest of this Chapter is organized as follows. First, we briefly review the three different circuit design methodologies: (i) traditional circuit design without tuning capacity, (ii) off-chip adaptive performance tuning, and (iii) on-chip self-healing. The trade-off of the three methods are analyzed. Next, we consider the cost per good chip [99] as the judging criterion for overall product cost. The cost per good chip is calculated as the overall capital investment in the manufacturing, testing and packaging cost throughout the good chip generation process. According to [98], the cost percentage of manufacturing, testing and packaging can be around 40%, 40% and 30% respectively. Therefore, the three components must be carefully analyzed to yield accurate cost estimation. The cost per good chip is established from several important parameters such as parametric yield, ATE cost, testing time, etc. Finally, we study a mmWave transceiver test case to compare three different design methodologies. This cost analysis can provide a guideline to circuit designers on whether or not to apply on-chip self-healing for a particular application.

5.2 Three Design Methodologies

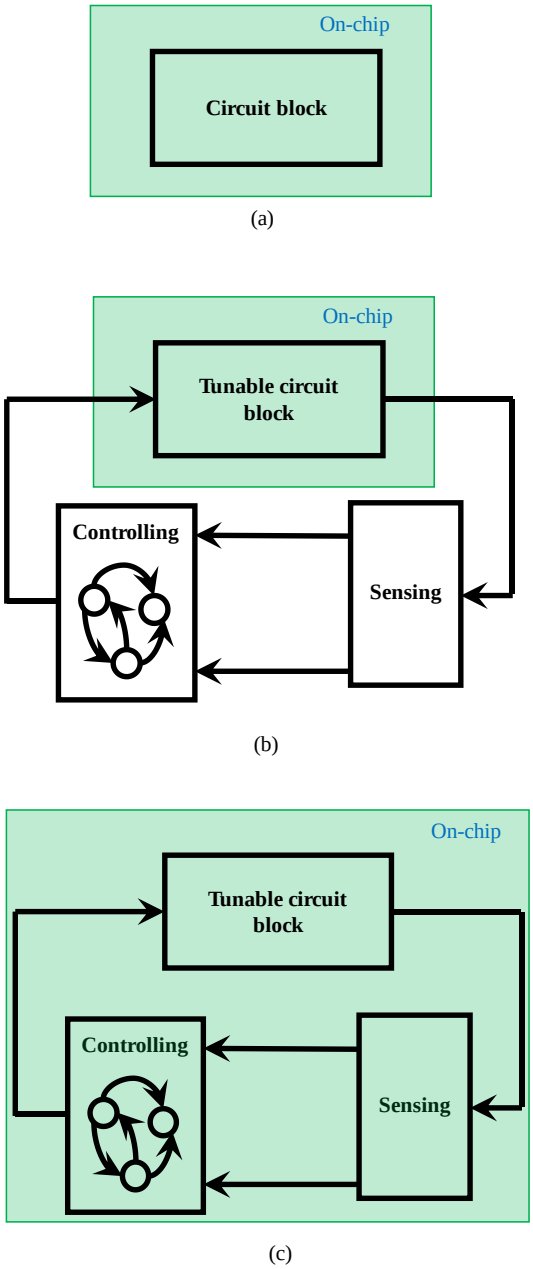


Figure 5-1. Circuit components of three different design methodologies are shown: (a) traditional design without tuning capacity (Traditional), (b) off-chip adaptive performance tuning (AT), and (c) on-chip self-healing (OH).

As shown in Figure 5-1(a)-(c), the design methodology of analog/RF circuits can be classified into three broad categories: (i) traditional design without tuning capacity (Traditional) [8]-[32], (ii) off-chip adaptive performance tuning (AT) [47]-[48], and (iii) on-chip self-healing (OH) [41]-[46], [49]-[53]. The features of the three design methodologies are listed as follows:

- Traditional: In Traditional design, no control knobs, sensing or controlling circuitry are designed in the circuit block, as shown in Figure 5-1(a). The Traditional design approach has been extensively studied in the literature [8]-[32]. Numerous design optimization methods have been proposed, as discussed in Section 1.1. However, due to the large-scale process variations in advanced technology nodes, the Traditional has become increasingly difficult. In particular, the design margin has continuously shrinking and the feasible design region has become smaller and smaller. It is more and more challenging to design a circuit with aggressive specification while maintaining high parametric yield. The resulting parametric yield loss will significantly degrade profitability of circuit products, as will be seen from the case study in Section 5.4.

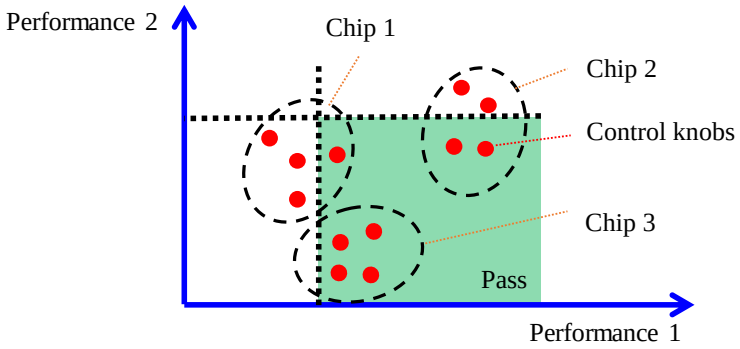


Figure 5-2. Adaptive feature of tunable circuit block is shown. Each black dashed ellipsoid denotes one manufactured chip. Each red dot represents a particular knob configuration.

- AT: In AT, the performance sensing and controlling is done off-chip, as shown in Figure 5-1(b). Given one knob configuration, the circuit performance of interest will be measured using off-chip ATE. Based on the ATE measurements, the knob configuration is adaptively tuned. This procedure is done until a certain criterion is met (e.g. performances are optimized or power is minimized while performances pass all specifications). AT is expected to achieve superior parametric yield compared to Traditional, due to its adaptive feature. Figure 5-2 further illustrates the adaptive feature of tunable

circuit block. In this example, only two performances are of interest, which are shown as x-axis and y-axis respectively. The green region represents the passing region. The three ellipsoids represent three instances of manufactured chips. Due to process variations, the performances of three chips are deviated from each other. Each red dot denotes a particular knob configuration in the tunable circuit. As we can see, by selecting different knob configurations, the manufactured chips are able to flexibly locate at any of the four locations enabled by knob configurations. As such, the parametric yield can be significantly improved compared to Traditional.

- OH: The performance sensing and calibration of OH are done on-chip, as shown in Figure 5-1(c). The on-chip self-healing is consisted of two steps. The first step is to collect a few ATE samples across a number of knob configuration in several chips. Indirect sensor models are then fitted using those ATE samples. The indirect sensor models are then loaded in on-chip microcontroller. As discussed in Section 3.2, the circuit performances of interest can be inexpensively predicted by indirect sensor models. The second step of OH is to perform on-chip performance tuning. Based on indirect sensor predictions, the controlling block will sweep a number of knob configurations until certain criterion is met. One major difference between OH and AT is that OH employs indirect performance sensor while AT uses off-chip ATE measurements. As such, OH is less accurate in performance sensing than AT, due to the limited accuracy in indirect sensor models. However, on the other hand, the testing cost related to OH is much lower than AT because OH requires much less ATE measurements to fit indirect sensor models.

5.3 Cost per Good Chip

The cost per good chip is defined as the overall capital investment in the manufacturing, testing and packaging cost throughout the good chip generation process [99]. It should be noted that the capital investment is not done once at the beginning of the manufacturing. Instead, it is done in several steps in the manufacturing and testing flow. For example, the packaging is only performed to the dies that pass the wafer probe test. In this regard, it is important to first understand the overall manufacturing and testing flow to analyze the cost.

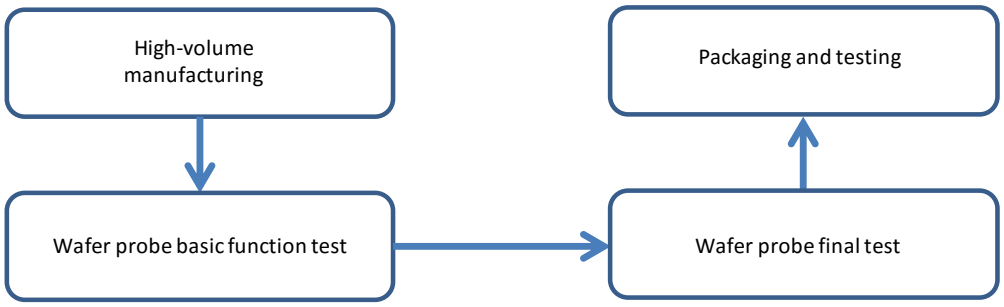


Figure 5-3. The manufacturing and testing flow of traditional design is shown.

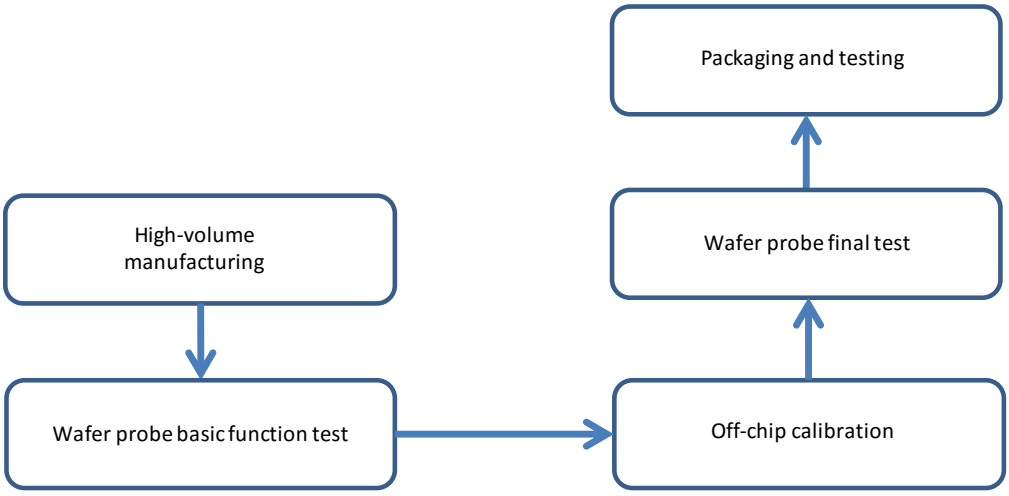


Figure 5-4. The manufacturing and testing flow of off-chip adaptive performance tuning (AT) is shown.

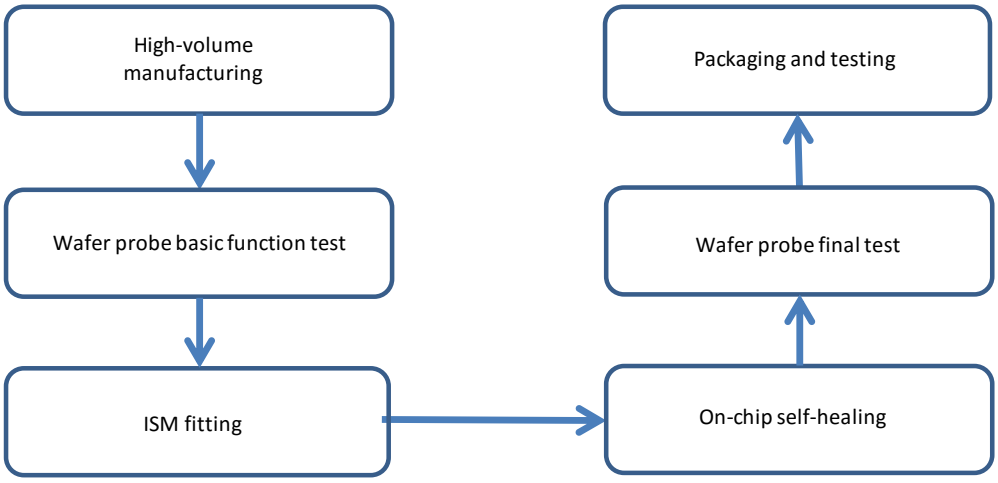


Figure 5-5. The manufacturing and testing flow of on-chip self-healing (OH) is shown.

The manufacturing and testing flow [91]-[99] of the three approaches are summarized in Figure 5-3-Figure 5-5. Here we consider a high-volume production scenario for mmWave circuit. The blocks represent

the important steps in the flow, while the arrows represent sequential relations between steps. For Traditional, as shown in Figure 5-3, the dies are first manufactured. Then a basic function test for supply voltage and bias current is performed. The bad dies detected in this step is discarded. Due to the high packaging cost, the dies are further screened in a wafer probe final test. Finally, the dies are packaged and tested.

Figure 5-4 shows the testing flow of AT. The dies are first manufactured and go through basic function test. Then, the passing dies from basic function test will be calibrated off-chip to adaptively select the optimal knob configuration in order to provide desired circuit performance. After this step, the dies are tested in wafer probe. The passing dies after wafer probe test are then packaged and tested.

Figure 5-5 shows the testing flow of OH. Similar to AT, the dies are first manufactured and go through basic function test. Then a few ATE samples are collected across a number of knob configuration in several dies. Indirect sensor models are fitted using those ATE samples and then loaded in on-chip microcontroller. The circuit performances of interest can be inexpensively predicted by indirect sensor models. Next, on-chip performance tuning is performed. Based on indirect sensor predictions, the controlling block will sweep a number of knob configurations until certain criterion is met. The criterion is usually defined according to circuit performances (e.g. minimizing power given specification constraints, minimizing noise figure, etc.). Once the on-chip performance tuning is finished, the dies are tested in wafer probe. The passing dies after wafer probe test are then packaged and tested.

In the whole manufacturing and testing flow, three tests are performed: (i) basic function test, (ii) wafer probe final test, and (iii) packaged test. In each test, dies/chips are tested to check whether particular criteria are met. For example, in the basic function test, the supply voltage and bias current are checked. The dies/chips that do not meet the criteria during the test are discarded. We define the following yields to denote the portion of good dies/chips in each test:

$$\text{Yield}_1 = \frac{\#\text{of good dies after basic test}}{\#\text{of manufactured dies}} \tag{5.1}$$

$$\text{Yield}_2 = \frac{\#\text{of good dies after wafer probe test}}{\#\text{of good dies after basic test}} \tag{5.2}$$

$$\text{Yield}_3 = \frac{\#\text{of final good chips}}{\#\text{of good dies after wafer probe test}}, \tag{5.3}$$

where Yield_1 denotes the ratio of the number of good dies after basic test and the total number of manufactured dies, Yield_2 denotes the ratio of the number of good dies after wafer probe test and the number of good dies after basic function test, and Yield_3 denotes the ratio of the number of final good chips and the number of good dies after wafer probe test. It should be noted that the overall yield of the manufacturing, testing and packaging process can be calculated as:

$$\begin{aligned}\text{Yield}_{\text{all}} &= \text{Yield}_1 \times \text{Yield}_2 \times \text{Yield}_3 \\ &= \frac{\#\text{of final good chips}}{\#\text{of manufactured dies}}\end{aligned} . \tag{5.4}$$

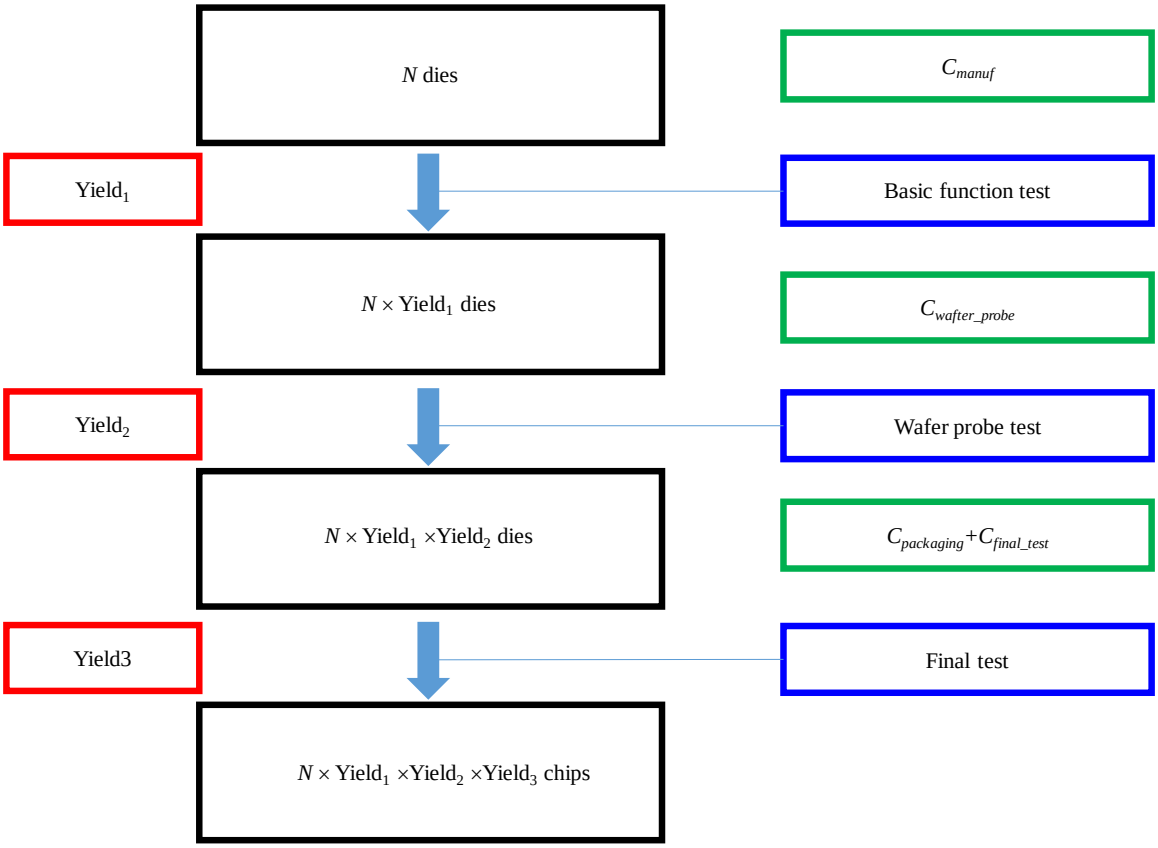


Figure 5-6. The manufacturing and testing procedure is shown where different types of costs (shown in green box) are applied on different number of dies/chips (shown in black box) due to the chip drop after each test (shown in blue box).

Figure 5-6 shows the overall manufacturing and testing procedure where the green box denotes the cost associated with each step, the blue box denotes the test that screens out a portion of dies/chips, the red box represents the yield associated with each test as denoted in (5.1)-(5.3), and the black box denotes the

number of dies/chips left after testing steps.

To calculate the cost per good chip, we start from manufacturing N dies. The total cost associated with this step is:

$$\text{Cost}_1 = C_{manuf} \times N, \tag{5.5}$$

where C_{manuf} denotes the manufacturing cost per die in high-volume production. After manufacturing, the basic function test is performed. Some dies fail to meet the specifications and are therefore discarded. This leaves $N \times \text{Yield}_1$ dies. For AT and OH, the remaining dies will be tuned in post-silicon using wafer probe test (directly or indirectly). And a final probe final test will be performed to determine whether to proceed to the packaging step. Given that totally $N \times \text{Yield}_1$ dies are engaged in the wafer probe test, this step involves the cost:

$$\text{Cost}_2 = C_{wafer\_probe} \times N \times \text{Yield}_1, \tag{5.6}$$

where C_{wafer_probe} denotes the wafer probe test cost per die. This cost includes both the post-silicon tuning cost and the wafer probe final test cost. The post-silicon tuning cost here would be adaptive performance tuning cost for AT or indirect sensor modeling cost for OH. The wafer probe test further screens out a portion of chips which leads to $N \times \text{Yield}_1 \times \text{Yield}_2$ remaining dies. These remaining dies are packaged and finally tested. The cost of this step would be:

$$\text{Cost}_3 = \left(C_{packaging} + C_{final\_test}\right) \times N \times \text{Yield}_1 \times \text{Yield}_2, \tag{5.7}$$

where $C_{packaging}$ represents the packaging cost and C_{final_test} denotes the cost for final sign-off tests after packaging. The total cost involved in the manufacturing and testing procedure can be calculated from (5.5)-(5.7) as:

$$\begin{aligned} &\text{Total Cost} = \text{Cost}_1 + \text{Cost}_2 + \text{Cost}_3 \\ &= C_{manuf} \times N + C_{wafer\_probe} \times N \times \text{Yield}_1 \\ &+ \left(C_{packaging} + C_{final\_test}\right) \times N \times \text{Yield}_1 \times \text{Yield}_2 \end{aligned} . \tag{5.8}$$

The final test further screens out a portion of chips and the final number of good (passing) chips is:

$$N_{pass} = N \times \text{Yield}_1 \times \text{Yield}_2 \times \text{Yield}_3 . \tag{5.9}$$

The cost per good chip can then be calculated as:

$$\begin{aligned}\text{Cost per good chip} &= \frac{\text{Total Cost}}{N_{pass}} = \frac{C_{manuf} \times N + C_{wafer\_probe} \times N \times \text{Yield}_1}{N \times \text{Yield}_1 \times \text{Yield}_2 \times \text{Yield}_3} \\ &+ \frac{\left(C_{packaging} + C_{final\_test}\right) \times N \times \text{Yield}_1 \times \text{Yield}_2}{N \times \text{Yield}_1 \times \text{Yield}_2 \times \text{Yield}_3} \\ &= \frac{C_{manuf}}{\text{Yield}_1 \times \text{Yield}_2 \times \text{Yield}_3} + \frac{C_{wafer\_probe}}{\text{Yield}_2 \times \text{Yield}_3} + \frac{\left(C_{packaging} + C_{final\_test}\right)}{\text{Yield}_3}\end{aligned} . \quad (5.10)$$

In (5.10), the cost in later stage is divided by less yield terms because the investment at a particular stage is only done to the chips that are currently good. For example, the packaging is done only on the chips that has passed the wafer probe test, but has nothing to do with the chips that was discarded in the basic function test.

For a particular chip, the manufacturing and packaging cost can be easily calculated based on wafer/package expense. The testing cost, however, needs to be calculated in a different way [99]. For mmWave product, expensive automatic testing equipment (ATE) are required for testing. Since ATE will depreciate in a period of time, every second during the usage of ATE has non-negligible cost. For example, as shown in Table 5-1, the cost of acquiring ATE can be $1 million, and it will depreciate within 3~5 years. The utilization of ATE (which is the percentage of ATE operating time over the depreciation time) can be 80%. An example of usage cost per second calculation for this ATE would be:

$$\begin{aligned}\text{Usage Cost per Sec} &= \frac{\text{ATE cost}}{\text{Lifetime} \times \text{Utilization}} \\ &= \frac{1000000}{3 \times 365 \times 24 \times 3600 \times 0.8} = 0.013(\$)\end{aligned} . \quad (5.11)$$

In other words, if the test of a particular mmWave product takes 2 minutes using this ATE, the testing cost is as high as $1.56.

Table 5-1. mmWave ATE parameters

	Explanations	Number
ATE cost	ATE acquisition cost	$ 1 million
Lifetime	Depreciation time	3~5 years
Utilization	ATE operating time/depreciation time	80%

5.4 Case Study

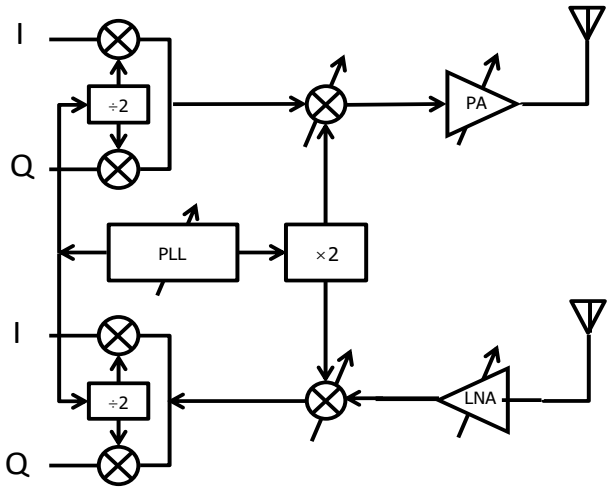


Figure 5-7.A mmWave transceiver schematic is shown.

In this section, we perform a cost analysis on a mmWave transceiver as shown in Figure 5-7. In this example, the LNA, power amplifier (PA), down-conversion mixer, up-conversion mixer and phase-locked loop (PLL) are all tunable. To compare the cost per good chip for OH, AT and Traditional, we first consider the manufacturing cost and packaging cost as shown in Table 5-2 [100]-[101]. The OH and AT requires on-chip DAC that controls each of the tunable circuit blocks. Totally 5 DACs are needed which takes 0.2mm^2 area [52]. To actively sense performances/calibrate ISM, two pads are placed at the output of PLL and down-conversion mixer, respectively. Furthermore, in OH, the microcontroller (uC) and sensors take additional 0.95mm^2 area [51]. As a result, the manufacturing cost of OH is the highest among the three approaches due to the hardware overhead. We further assume that the packaging cost is proportional to the manufacturing cost.

In Table 5-3, the parametric yield of circuit design based on Traditional, AT and OH are compared. The last column indicates whether the yield value is obtained from simulation, silicon measurement, or assumption. We assume that basic function test finds 3% of chips failing for Traditional. Since basic function testing mainly screens out failing chips due to short/open defects, we consider that such yield loss is proportional to the area. As such, the OH and AT has 3.4% and 3% yield loss in Yield1 respectively, as shown in the 2nd row. Then the block-level parametric yield values are estimated based on an industrial

32nm self-healing transceiver design. Due to the configurability feature, OH and AT are able to achieve significantly higher $Yield_2$ than Traditional. Then a 3% yield loss due to packaging and final test is assumed for all three approaches.

Table 5-2. Manufacturing cost and packaging cost comparison of Traditional, AT and OH.

	OH	AT	Traditional
28nm 300mm^2 wafer cost	5400 $	5400 $	5400 $
Area	70650mm^2	70650mm^2	70650mm^2
Cost per mm^2	0.0764 $/$mm^2$	0.0764 $/$mm^2$	0.0764 $/$mm^2$
Size of transceiver chip	7.65mm^2 (uC, sensors) (5 DACs, 2 pads)	6.70mm^2 (5 DACs, 2 pads)	6.50mm^2
Fabrication cost per transceiver	0.58 $	0.51 $	0.50 $
Packaging cost per transceiver	0.58 $	0.51 $	0.50 $

Table 5-3. Parametric yield of circuit design based on Traditional, AT and OH.

	OH	AT	Traditional	Comment
$Yield_1$	96.6%	97.0%	97.0%	Area based
LNA+Down-mixer	90.0%	90.0%	90.0%	Simulation
Up-mixer+PA	99.0%	99.0%	90.5%	Simulation
PLL	95.0%	95.0%	37.0%	Silicon
Frequency doubler	99.0%	99.0%	99.0%	Simulation
Others (Filter, I/Q mixer)	99.0%	99.0%	99.0%	Assumed
$Yield_2$	80.1%	80.5%	28.7%	Calculated
Packaging and final test ($Yield_3$)	97.0%	97.0%	97.0%	Assumed

It should be noted that some yield values presented in Table 5-3 are based on simulations. Therefore Table 5-3 only provides an upper bound on final yield estimation. Furthermore, the design cost for hiring design experts for configurable/self-healing circuit design is not considered. In what follows, we compare the cost per good chip using three different design approaches based on parameters described in Table 5-1-Table 5-3. The cost per second of ATE is assumed to be 0.010$. For Traditional, we assume that wafer probe final test takes 15s, and the test after packaging takes additional 15s. Therefore, the cost per good chip can be calculated using (5.10) as:

$$\begin{aligned}&\text{Cost per good chip}\\&=\frac{0.5}{0.97\times 0.287\times 0.97}+\frac{0.010\times 15}{0.287\times 0.97}+\frac{0.5+0.010\times 15}{0.97}\\&=1.85+0.54+0.67=3.06(\$)\end{aligned} \tag{5.12}$$

In (5.12), the three terms denote manufacturing, wafer probe tuning/testing, and packaging and final testing cost respectively.

For AT, we assume that wafer probe final test takes 15s, and the test after packaging takes additional 15s similar to Traditional. Furthermore, the off-chip performance tuning takes 75s to sweep 200 different knob configurations. The 200 knob configurations include 64 for up-conversion mixer and PA, 64 for LNA and down-conversion mixer, 32 for PLL and 40 for system-level tuning. The cost per good chip can be calculated using (5.10) as:

$$\begin{aligned}&\text{Cost per good chip}\\&=\frac{0.51}{0.97\times 0.805\times 0.97}+\frac{0.010\times 90}{0.805\times 0.97}+\frac{0.5+0.010\times 15}{0.97}\\&=0.67+1.15+0.67=2.49(\$)\end{aligned} \tag{5.13}$$

For OH, again we assume that wafer probe final test takes 15s, and the test after packaging takes additional 15s. Furthermore, the ISM construction requires 1000 off-chip ATE testing point from each measured die. The 1000 ATE testing points include 300 for up-conversion mixer and PA, 300 for LNA and down-conversion mixer, and 400 for PLL. The 1000 ATE testing points take 468.75s to finish. We further assume that the wafer is consisted of 9000 dies, and only 30 dies are needed to fit the ISM by using BMF approach. The cost per good chip can then be calculated using (5.10) as:

Cost per good chip

$$= \frac{0.58}{0.966 \times 0.801 \times 0.97} + \frac{0.010 \times 468.75 \times \frac{30}{9000} + 0.010 \times 15}{0.801 \times 0.97} + \frac{0.5+0.010 \times 15}{0.97} \qquad (5.14)$$
$$= 0.772 + 0.213 + 0.670 = 1.66(\$)$$

Comparing (5.12)-(5.14) yields several important observations. First, Traditional has very low parametric yield, and therefore the cost per good chip is very high. Second, OH has much lower performance tuning cost than AT (i.e. the second term in the cost calculation), because the ISM calibration cost is shared among all chips on the same wafer. Third, the overall cost per good chip of OH is the lowest among the three approaches, because it is able to achieve high parametric yield as well as low testing cost.

5.5 Summary

In this Chapter, we perform an overall cost analysis for different design methodologies considering manufacturing cost, testing cost and packaging cost. As is demonstrated in our test case, OH is able to achieve superior cost per good chip compared to AT and Traditional. The main benefit of OH is its capability of achieving high parametric yield and meanwhile maintaining a low testing cost. It should be noted that the cost analysis result should be case dependent, because the parametric yield values and testing parameters will change from design to design. However, we hope that our cost analysis could provide a guideline for circuit designers to determine the optimal design methodology.

Chapter 6

Thesis Summary & Future Work

6.1 Summary

The aggressive scaling of integrated circuits leads to large-scale process variations that cannot be easily reduced by foundries. Process variations manifest themselves as the uncertainties associated with the geometrical and electrical parameters of semiconductor devices. These device-level variations significantly impact the parametric yield of analog/RF circuits and lays a fundamental challenge on robust design techniques.

While the traditional robust design techniques are not sufficient to maintain high parametric yield with process variations, on-chip self-healing methodology has emerged as a promising methodology to address this challenge. In self-healing circuit, tuning knobs are designed which allow performance flexibility. The key idea of on-chip self-healing is to actively measure and adaptively tune circuit performances in post-silicon on-chip. As such, the parametric yield of circuit can be significantly improved.

Pre-silicon validation, post-silicon tuning and cost analysis are three important topics in self-healing analog/RF IC design flow. In pre-silicon, the self-healing circuit design must be verified by parametric yield analysis before moving to the manufacturing process. Once the circuit is manufactured, post-silicon tuning is required to exploit the benefit of configurability and improve parametric yield. The main challenge of both pre-silicon validation and post-silicon tuning is how to maintain low overhead in computation, hardware and testing cost. On the other hand, cost analysis of self-healing circuit is also a critical task to justify its benefit on the overall product cost over other design methodologies.

One important task in efficient pre-silicon validation is the efficient performance modeling. Towards this goal, a novel C-BMF algorithm is proposed. C-BMF encodes the correlation information for both model template and coefficient magnitude among different states in a prior distribution. Next, the prior

distribution is combined with very few simulation samples to accurately solve the model coefficients. An efficient and robust hyper-parameter inference approach is also proposed to handle the large number of hyper-parameters involved in the Bayesian inference. As is demonstrated by two circuit examples designed in a commercial 32nm CMOS process, the proposed C-BMF method achieves significant cost reduction compared to the state-of-the-art modeling technique.

Based on the C-BMF algorithm, an efficient pre-silicon validation flow is further proposed. Instead of simulating all the knob configurations from all chips, we generate pseudo samples based on the performance model. As such, the parametric yield estimation cost can be significantly reduced. Numerical results of a self-healing LNA and a self-healing down-conversion mixer demonstrates the efficacy of the proposed flow.

On the other hand, in post-silicon tuning, a large number of performance metrics must be measured accurately and inexpensively by on-chip sensors. We propose a novel indirect performance sensing technique to achieve such a goal. In particular, a set of important basis functions are first identified by SR so that the overhead of on-chip self-healing can be minimized. Next, the indirect sensor models are repeatedly calibrated by BMF to accommodate the process shift associated with manufacturing lines. The proposed indirect performance sensing and on-chip self-healing methodology is validated by a 25GHz differential Colpitts VCO designed in a 32nm CMOS SOI process. Our silicon measurement data show that the parametric yield of the VCO is significantly improved after applying self-healing.

As an extension of BMF, we propose a novel CL-BMF algorithm to further facilitate efficient indirect sensor modeling of self-healing circuits. CL-BMF optimally combines the following information: (i) the CSI, (ii) the PSI, and (iii) a small number of training samples. Bayesian inference is constructed and represented as a graphical model, where the CSI and the PSI are encoded by the prior distribution and the likelihood function respectively. From the Bayesian inference, the unknown model coefficients can be accurately determined by maximizing the posterior distribution. The side information selection technique is also discussed. Our circuit examples designed in a commercial 32nm SOI CMOS process demonstrate that the proposed CL-BMF method achieves significant cost reduction over other state-of-the-art indirect sensor modeling techniques.

At last, we perform a cost analysis for different design methodologies considering manufacturing

cost, testing cost and packaging cost. As is demonstrated in our mmWave transceiver test case, on-chip self-healing is able to achieve superior cost per good chip compared to off-chip performance tuning and traditional design without tuning knobs. The main benefit of on-chip self-healing is its capability of achieving high parametric yield and meanwhile maintaining a low testing cost. We hope this cost analysis could provide a guideline for circuit designers to determine the optimal design method.

6.2 Future Work

There are multiple directions that can be explored to extend this work to further benefit the manufacturing, design and testing community.

First, the proposed C-BMF performance modeling approach relies on the correlation of coefficient magnitude as well as model template. However, it is possible that such correlation presents within different clusters of states, instead of all states. For example, in the case where the bias current tuning range of LNA is very high, LNA may enter a very bad DC operating point with extreme bias current values. In such case, the performance model with bad DC operating point would be very different from the performance model with normal DC operating point. To handle the clustering of the states, an accurate clustering algorithm needs to be developed which is capable of identifying clusters with similar performance models.

Second, based on the performance model of self-healing circuit, corner extraction is another important application that can be further explored. With the proposed parametric yield estimation algorithm, designers are able to know the circuit yield and determine whether to sign-off the design. However, estimating the yield value only does not meet the needs of circuit designers. If a circuit fails the yield specification, it is important for the CAD tool to provide additional information for design debug. Towards this goal, worst-case corner extraction aims to identify the unique process condition at which a given circuit fails to work. With the extracted worst-case corners, designers can simulate their circuit at these particular corners and improve circuit performances accordingly.

Third, the proposed CL-BMF framework is based on polynomial models, which is in the category of parametric model. The predictive model has two broad categories: the parametric model and non-parametric model. The parametric model is the model that finds the model parameters based on training samples, and then make new predictions based on model parameters only. In other words, the training

samples are not used during the prediction. The non-parametric model, on the other hand, find the model parameters based on training sample and then make the model prediction using both model parameters and training samples. In the case where the predictive model is highly nonlinear, the non-parametric model usually provides better prediction than parametric model. With IC technology scaling, the performance models become increasingly nonlinear. Therefore, there is a strong need to further extend CL-BMF to non-parametric model.

Finally, the indirect performance sensing methodology can be utilized for testing cost reduction. The indirect sensor model essentially provides a prediction of circuit performances of interest using low cost on-chip sensors. In this thesis, indirect sensor model is used for efficient on-chip performance sensing for self-healing. It is also possible to extend the usage of indirect sensor model to testing, given its predictive feature. For example, indirect sensor model can be used to screen out fail chips before the wafer probe final test. As such, the overall testing cost can be reduced if the indirect sensor model is accurate enough. One challenge with this technique would be to properly handle the uncertainty associated with indirect sensor models.

Bibliography

[1] G. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, 1965.

[2] J. Fréchet, H. Ito and C. Willson "Positive and negative working resist compositions with acid-generating photoinitiator and polymer with acid-labile groups pendant from polymer backbone" *U.S. patent 4491628*, 1985.

[3] K. Jain, C. Willson, "High resolution optical lithography method and apparatus having excimer laser light source and stimulated Raman shifting", *US patent 4458994 A*, 1984.

[4] J. Lai, *Mechanics, mechanisms, and modeling of the chemical mechanical polishing process*, Ph.D. Dissertation, Massachusetts Institute of Technology, 2000.

[5] D. Clark, "Intel Rechisels the tablet on Moore's law," *Wall Street Journal Digits Tech News and Analysis*. 2015.

[6] M. Orshansky, S. Nassif, and D. Boning, *Design for Manufacturability and Statistical Design: A Constructive Approach*, Springer, 2007.

[7] X. Li, J. Le, L. Pileggi, *Statistical Performance Modeling and Optimization*, Now Publishers, 2007.

[8] J. Fishburn and A. Dunlop, "TILOS: a posynomial programming approach to transistor sizing," *IEEE International Conference on Computer Aided Design*, pp. 326 – 328, 1985.

[9] K. Francken and G. Gielen, "A high-level simulation and synthesis environment for $\Delta\Sigma$ modulators," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 8, pp. 1049 – 1061, 2003.

[10] H. Graeb, S. Zizala, J. Eckmueller and K. Antreich, "The sizing rules method for analog integrated circuit design," *IEEE International Conference of Computer Aided Design*, pp. 343 – 349, 2001.

[11] M. Hershenson, "Design of pipeline analog-to-digital converters via geometric programming," *IEEE International Conference of Computer Aided Design*, pp. 317 – 324, 2002.

[12] M. Hershenson, A. Hajimiri, S. Mohan, S. Boyd and T. Lee, “Design and optimization of LC oscillators,” *IEEE International Conference of Computer Aided Design*, pp. 65 – 69, 1999.

[13] M. Hershenson, S. Mohan, S. Boyd and T. Lee, “Optimization of inductor circuits via geometric programming,” *IEEE Design Automation Conference*, pp. 994 – 998, 1999.

[14] K. Kasamsetty, M. Ketkar and S. Sapatnekar, “A new class of convex functions for delay modeling and its application to the transistor sizing problem,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 7, pp. 779 – 788, 2000.

[15] M. Krasnicki, R. Phelps, J. Hellums, M. McClung, R. Rutenbar and L. Carley, “ASP: A practical simulation-based methodology for the synthesis of custom analog circuits,” *IEEE International Conference of Computer Aided Design*, pp. 350 – 357, 2001.

[16] M. Krasnicki, R. Phelps, R. Rutenbar and L. Carley, “MAELSTROM: efficient simulation-based synthesis for custom analog cells,” *IEEE Design Automation Conference*, pp. 945 – 950, 1999.

[17] P. Mandal and V. Visvanathan, “CMOS Op-Amp sizing using a geometric programming formulation,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 1, pp. 22 – 38, 2001.

[18] W. Nye, D. Riley, A. Sangiovanni-Vincentelli and A. Tits, “DELIGHT.SPICE: an optimization-based system for the design of integrated circuits,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 7, no. 4, pp. 501 – 519, 1988.

[19] R. Phelps, M. Krasnicki, R. Rutenbar, L. Carley and J. Hellums, “Anaconda: simulation-based synthesis of analog circuits via stochastic pattern search,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 6, pp. 703 – 717, 2000.

[20] G. Plas, G. Debyser, F. Leyn, K. Lampaert, J. Vandenbussche, W. S. G. Gielen, P. Veselinovic and D. Leenaerts, “AMGIE - a synthesis environment for CMOS analog integrated circuits,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 9, pp. 1037 – 1058, 2001.

[21] S. Sapatnekar, V. Rao, P. Vaidya and S. Kang, “An exact solution to the transistor sizing problem for CMOS circuits using convex optimization,” *IEEE Trans. Computer-Aided Design of Integrated*

Circuits and Systems, vol. 12, no. 11, pp. 1621 – 1634, 1993.

[22] G. Stehr, M. Pronath, F. Schenkel, H. Graeb and K. Antreich, "Initial sizing of analog integrated circuits by centering within topology-given implicit specifications," *IEEE International Conference of Computer Aided Design*, pp. 241 – 246, 2003.

[23] G. Stehr, M. Pronath, F. Schenkel, H. Graeb and K. Antreich, "Initial sizing of analog integrated circuits by centering within topology-given implicit specifications," *IEEE International Conference of Computer Aided Design*, pp. 241 – 246, 2003.

[24] Y. Wang, M. Orshansky and C. Caramanis, "Enabling efficient analog synthesis by coupling sparse regression and polynomial optimization," *IEEE Design Automation Conference*, pp. 1 – 6, 2014.

[25] S. Director, P. Feldmann and K. Krishna, "Statistical integrated circuit design," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 3, pp. 193 – 202, 1993.

[26] P. Feldman and S. Director, "Integrated circuit quality optimization using surface integrals," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 12, pp. 1868 – 1878, 1993.

[27] F. Schenkel, M. Pronath, S. Zizala, R. Schwencker, H. Graeb and K. Antreich, "Mismatch analysis and direct yield optimization by spec-wise linearization and feasibility-guided search," *IEEE Design Automation Conference*, pp. 858 – 863, 2001.

[28] Z. Wang and S. Director, "An efficient yield optimization method using a two step linear approximation of circuit performance," *IEEE European Design and Test Conference*, pp. 567 – 571, 1994.

[29] H. Abdel-Malek and A. Hassan, "The ellipsoidal technique for design centering and region approximation," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, no. 8, pp. 1006 – 1014, 1991.

[30] K. Antreich, H. Graeb and C. Wieser, "Circuit analysis and optimization driven by worst-case distances," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 13, no. 1, pp. 57 – 71, 1994.

[31] A. Seifi, K. Ponnambalam and J. Vlach, "A unified approach to statistical design centering of

integrated circuits with correlated parameters," *IEEE Trans. Circuits and Systems – I*, vol. 46, no. 1, pp. 190 – 196, 1999.

[32] G. Stehr, M. Pronath, F. Schenkel, H. Graeb and K. Antreich, "Initial sizing of analog integrated circuits by centering within topology-given implicit specifications," *IEEE International Conference of Computer Aided Design*, pp. 241 – 246, 2003.

[33] G. Debyser and G. Gielen, "Efficient analog circuit synthesis with simultaneous yield and robustness optimization," *IEEE International Conference of Computer Aided Design*, pp. 308 – 311, 1998.

[34] A. Dharchoudhury and S. Kang, "Worst-case analysis and optimization of VLSI circuit performances," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 4, pp. 481 – 492, 1995.

[35] X. Li, P. Gopalakrishnan, Y. Xu and L. Pileggi, "Robust analog/RF circuit design with projection-based posynomial modeling," *IEEE International Conference on Computer Aided Design*, pp. 855 – 862, 2004.

[36] X. Li and L. Pileggi, "Efficient parametric yield extraction for multiple correlated non-Normal performance distributions of analog/RF circuits," *IEEE Design Automation Conference*, pp. 928 – 933, 2007.

[37] X. Li, J. Wang, L. Pileggi, T. Chen, and W. Chiang, "Performance-centering optimization for system-level analog design exploration," *IEEE International Conference on Computer Aided Design*, pp. 422 – 429, 2005.

[38] S. Sen, "Channel-adaptive zero-margin & process-adaptive self-healing communication circuits/systems," *IEEE International Conference on Computer Aided Design*, pp. 80 – 85, 2014.

[39] M. Fell, "Roadmap for the emerging Internet of Things - its Impact, architecture and future Governance," *Carré & Strauss*, 2014.

[40] S. Vongsingthong and S. Smanchat, "Internet of Things: a review of applications & technologies" *Suranaree Journal of Science and Technology*, 2014.

[41] W. Khalil, B. Bakkaloglu and S. Kiaei, "A self-calibrated on-chip phase-noise measurement circuit

with 75 dBc single-tone sensitivity at 100 kHz offset," *IEEE J. Solid-State Circuits*, vol. 42, no. 12, pp. 2758 – 2765, 2007.

[42] A. Jose, A. Jenkins and S. Reynolds, "On-chip spectrum analyzer for analog built-in self-test," *IEEE VLSI Test Symp.*, pp. 131 – 136, 2005.

[43] S. Yaldiz, V. Calayir, X. Li, L. Pileggi, A. Natarajan, M. Ferriss and J. Tierno, "Indirect phase noise sensing for self-healing voltage controlled oscillators," *IEEE Custom Integrated Circuits Conference*, pp. 1 – 4, 2011.

[44] P. Variyam, S. Cherubal and A. Chatterjee, "Prediction of analog performance parameters using fast transient testing," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 3, pp. 349 - 361, 2002.

[45] D. Han, S. S. Akbay, S. Bhattacharya and A. Chatterjee, "On-chip self-calibration of RF circuits using specification-driven built-in self test (S-BIST)," *IEEE Int. On-Line Testing Symp.*, pp. 106 – 111, 2005.

[46] D. Han, B. S. Kim and A. Chatterjee, "DSP-driven self-tuning of RF circuits for process-induced performance variability," *IEEE Trans. Very Large Scale Integration System*, vol. 18, no. 2, pp. 305 - 314, 2010.

[47] V. Nataranjan, S. Sen, A. Banerjee and A. Chatterjee, "Analog signature-driven postmanufacture multidimensional tuning of RF systems," *IEEE Design & Test Magazine*, vol. 27, no. 6, pp. 6 - 17, 2010.

[48] N. Kupp, H. Huang, P. Drineas, and Y. Makris, "Post-production performance calibration in analog/RF devices," *IEEE International Test Conference*, pp. 1 – 10, 2010.

[49] C. Chien, A. Tang, F. Hsiao, and M. Chang, "Dual-control self-healing architecture for high performance radio SoC's," *IEEE Design & Test Magazine*, vol. 29, no. 6, pp. 40 - 51, 2012.

[50] B. Sadhu, M. A. Ferriss, A. S. Natarajan, S. Yaldiz, J. Plouchart, A. V. Rylyakov, A. Valdes-Garcia, B. D. Parker, A. Babakhani, S. Reynolds, X. Li, L. Pileggi, R. Harjani, J. A. Tierno, and D. Friedman, "A linearized low-phase-noise VCO-based 25-GHz PLL with autonomic biasing," *IEEE J. Solid-State Circuits*, vol. 48, no. 5, pp. 1138 - 1150, 2013.

[51] J.-O. Plouchart, F. Wang, B. Parker, B. Sadhu, A. Valdes-Garcia and D. Friedman, "Adaptive circuit

design methodology and test applied to millimeter-wave circuits," *IEEE Design & Test Magazine*, vol. 31, no. 6, pp. 8 – 18, 2014.

[52] J.-O. Plouchart, F. Wang, A. Balteanu, B. Parker, M. Sanduleanu, M. Yeck, V. Chen, W. Woods, B. Sadhu, A. Valdes-Garcia, X. Li, D. Friedman, "A 18mW, 3.3dB NF, 60GHz LNA in 32nm SOI CMOS Technology with Autonomic NF Calibration," *IEEE Radio Frequency Integrated Circuits Symposium*, 2015.

[53] S. Sun, F. Wang, S. Yaldiz, X. Li, L. Pileggi, A. Natarajan, M. Ferriss, J.-O. Plouchart, B. Sadhu, B. Parker, A. Valdes-Garcia, M. Sanduleanu, J. Tierno and D. Friedman, "Indirect performance sensing for on-chip self-healing of analog and RF circuits," *IEEE Trans. on Circuits and Systems - I*, vol. 61, no. 8, pp. 2243 - 2252, 2014.

[54] R. Senguttuvan and A. Chatterjee, "Alternate diagnostic testing and compensation of transmitter performance using response detection," *Proceedings of 25th IEEE VLSI Test Symposium*, pp.395-400, 2007.

[55] G. Srinivasan and A. Chatterjee, "Fourier spectrum-based signature test: A genetic CAD toolbox for reliable RF testing using low-performance test resources," *IEEE Asian Test Symposium*, pp. 139-142, 2007.

[56] K. Huang, N. Kupp, C. Xanthopoulos, J. M. Carulli Jr., Y. Makris, "Low-cost analog/RF IC testing through combined intra- and inter-die correlation models," *Special Issue on Speeding up Analog Integration and Test for Mixed-signal SOCs of the IEEE Design & Test of Computers*, vol. 32, no. 1, pp. 53-60, 2015.

[57] N. Kupp, H. Huang, P. Drineas, Y. Makris, "Improving analog and RF device yield through performance calibration," *IEEE Design and Test of Computers*, vol. 28, no.3, pp. 64 - 75, 2011.

[58] D. Maliuk and Y. Makris, "An experimentation platform for on-chip integration of analog neural networks, a pathway to trusted and robust analog/RF ICs," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 26, no. 8, 2015.

[59] D. Maliuk and Y. Makris, "An analog non-volatile neural network platform for prototyping RF BISt solutions," *IEEE European Design and Test Conference*, pp. 1 - 6, 2014.

[60] D. Banerjee, S. Sen, A. Chatterjee, "Self learning analog/mixed-signal/RF systems: dynamic

adaptation to workload and environmental uncertainties," *IEEE International Conference on Computer Aided Design*, pp. 59 – 64, 2015.

[61] C. Bishop, *Pattern Recognition and Machine Learning*. Prentice Hall, 2007.

[62] S. Sun, *Fast statistical analysis of rare failure events for SRAM circuits in high dimensional variation space*, Ph.D. Dissertation, Carnegie Mellon University, 2015.

[63] A. Singhee, R. Rutenbar, "From finance to flip flops: a study of fast quasi-Monte Carlo methods from computational finance applied to statistical circuit analysis," *IEEE International Symposium on Quality Electronic Design*, pp. 685 – 692, 2007.

[64] W. Wu, W. Xu, R. Krishnan, Y.-L. Chen, L. He, "REscope: High-dimensional statistical circuit simulation towards full failure region coverage," *IEEE Design Automation Conference*, pp. 1 – 6, 2014.

[65] A. Singhee and R. Rutenbar, "Statistical blockade: very fast statistical simulation and modeling of rare circuit events and its application to memory design," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 8, 2008.

[66] F. Gong, H. Yu, Y. Shi and L. He, "Variability-aware parametric yield estimation for analog/mixed-signal circuits: concepts, algorithms, and challenges," *IEEE Design & Test Magazine*, vol. 31, no. 4, pp. 6 – 15, 2014.

[67] X. Li, W. Zhang, F. Wang, S. Sun and C. Gu, "Efficient parametric yield estimation of analog/mixed-signal circuits via Bayesian model fusion," *IEEE International Conference on Computer Aided Design*, pp. 627 – 634, 2012.

[68] X. Li, J. Le, P. Gopalakrishnan and L. Pileggi, "Asymptotic probability extraction for nonnormal performance distributions," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 1, pp. 16 – 37, 2007.

[69] X. Li, Y. Zhan and L. Pileggi, "Quadratic statistical *MAX* approximation for parametric yield estimation of analog/RF integrated circuits," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 831 – 843, 2008.

[70] A. Singhee and R. Rutenbar, "Beyond low-order statistical response surfaces: latent variable

regression for efficient, highly nonlinear fitting," *IEEE Design Automation Conference*, pp. 256 – 261, 2007.

[71] A. Mitev, M. Marefat, D. Ma, J. Wang, "Principle Hessian direction based parameter reduction for interconnect networks with process variation," *IEEE International Conference on Computer Aided Design*, pp. 632-637, 2007.

[72] T. McConaghy and G. Gielen, "Template-free symbolic performance modeling of analog circuits via canonical-form functions and genetic programming," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, no. 8, pp. 1162-1175, 2009.

[73] X. Li and H. Liu, "Statistical regression for efficient high-dimensional modeling of analog and mixed signal performance variations," *IEEE Design Automation Conference*, pp. 38-43, 2008.

[74] X. Li, "Finding deterministic solution from underdetermined equation: large-scale performance modeling by least angle regression," *IEEE Design Automation Conference*, pp. 364-369, 2009.

[75] X. Li, "Finding deterministic solution from underdetermined equation: large-scale performance variability modeling of analog/RF circuits," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 11, pp. 1661-1668, 2010.

[76] X. Li, W. Zhang and F. Wang, "Large-scale statistical performance modeling of analog and mixed-signal circuits," *IEEE Custom Integrated Circuits Conference*, pp. 1-8, 2012.

[77] T. McConaghy, "High-dimensional statistical modeling and analysis of custom integrated circuits," *Custom Integrated Circuits Conference*, pp. 1-8, 2011.

[78] W. Zhang, K. Balakrishnan, X. Li, D. Boning, S. Saxena, A. Strojwas and R. Rutenbar, "Efficient spatial pattern analysis for variation decomposition via robust sparse regression," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 7, pp. 1072-1085, 2013.

[79] W. Zhang, T.-H. Chen, M.-Y. Ting and X. Li, "Toward efficient large-scale performance modeling of integrated circuits via multi-mode/multi-corner sparse regression," *IEEE Design Automation Conference*, pp. 897-902, 2010.

[80] X. Liu, S. Sun, P. Zhou, X. Li and H. Qian, "A statistical methodology for noise sensor placement and full-chip voltage map generation," *IEEE Design Automation Conference*, pp. 1-6, 2015.

[81] M. Jun, J. Tao, Y.-C. Wang, S. Yin, R. Negi, X. Li, T. Mukherjee and L. Pileggi, "Environment-

adaptable efficient optimization for programming of reconfigurable radio frequency (RF) receivers," *IEEE Military Communications Conference*, pp. 1459-1465, 2014.

[82] J. Tao, Y.-C. Wang, M. Jun, X. Li, R. Negi, T. Mukherjee and L. Pileggi, "Toward efficient programming of reconfigurable radio frequency (RF) receivers," *IEEE/ACM Asia and South Pacific Design Automation Conference*, pp. 256-261, 2014.

[83] Z. Zhang and B. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 912-926, 2011.

[84] H. Zhang et al., "Efficient design-specific worst-case corner extraction for integrated circuits," *IEEE Design Automation Conference*, pp. 386-389, 2009.

[85] S. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2009.

[86] J. Plouchart, M. A. Ferriss, A. S. Natarajan, A. Valdes-Garcia, B. Sadhu, A. V. Rylyakov, B. D. Parker, M. Beakes, A. Babakhani, S. Yaldiz, L. Pileggi, R. Harjani, S. Reynolds, J. A. Tierno, and D. Friedman, "A 23.5GHz PLL with an adaptively biased VCO in 32nm SOI-CMOS," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 60, no. 8, pp. 2009-2017, 2013.

[87] D. Leeson, "A simple model of feedback oscillator noise spectrum," *Proc. IEEE*, vol. 54, no. 2, pp. 329-330, 1966.

[88] C. Fang, F. Yang, X. Zeng and X. Li, "BMF-BD: Bayesian model fusion on Bernoulli distribution for efficient yield estimation of integrated circuits," *IEEE Design Automation Conference*, pp. 1 – 6, 2014.

[89] C. Fang, Q. Huang, F. Yang, X. Zeng, X. Li and C. Gu, "Efficient bit error rate estimation for high-speed link by Bayesian model fusion," *IEEE Design Automation and Test Conference in Europe*, pp. 1024 - 1029, 2015.

[90] Q. Huang, C. Fang, F. Yang, X. Zeng and X. Li, "Efficient multivariate moment estimation via Bayesian model fusion for analog and mixed-signal circuits," *IEEE Design Automation Conference*, pp. 1 – 6, 2015.

[91] A. Halder, *Efficient Alternate Test Generation for RF Transceiver Archietecures*, Ph.D. Dissertation, Georgia Institute of Technology, 2006.

[92] D. Brown, J. Ferrario, R. Wolf, J. Li, J. Bhagat and M. Samani, “RF testing on a mixed signal tester,” *Journal of Electronic Testing: Theory and Applications*, vol. 23, pp. 85–94, 2007.

[93] H.-G. Stratigopoulos, P. Drineas, M. Slamani and Y. Makris, “RF specificationtest compaction using learning machines,” *IEEE Trans. on Very Large Scale Integration Systems*, vol. 18, no. 6, 2010.

[94] D. Giopoulos, *Advances in Electronic Testing: Challenges and Methodologies*, Springer, 2006.

[95] E. Erdogan, *Efficient Test Methods for RF Transceivers*, Ph.D. Dissertation, Duke University, 2010.

[96] J. Ferrari, R. Wolf, S. Moss and M. Slamani, “A low-cost test solution for wireless phone RFICs,” *IEEE Communication Magazine*, vol. 41, no. 6, 2003.

[97] S. Bahukdumbi, *Wafer-level testing and test planning for integrated circuits*, Ph.D. Dissertation, Duke University, 2008.

[98] V. Natarajan, *Self-healing RF SoCs: low cost built-in test and control driven simultaneous tuing of multiple performance metrics*, Ph.D. Dissertation, Georgia Institute of Technology, 2010.

[99] K. Schaub and J. Kelly, *Production testing of RF and system-on-a-chip devices for wireless communications*, Artech House, 2004.

[100] S. Pinel, S Sarkar, P. Sen, B. Perumana, D. Yeh, D. Dawn and J. Laskar, “A 90nm CMOS 60GHz Radio,” *IEEE International Solid-State Circuits Conference*, 2008.

[101] H. Jones, “Economic impact of the technology choices at 28nm/20nm,” *International Business Strategy white paper*, 2012.