University of California
Santa Barbara

Test Data Analytics: Exploration of Hidden Patterns for Test Cost Reduction and Silicon Characterization

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Chun-Kai Hsu

Committee in charge:

Professor Kwang-Ting Tim Cheng, Chair
Professor Malgorzata Marek-Sadowska
Professor Luke Theogarajan
Professor Xin Li, Carnegie Mellon University
Dr. Kenneth Butler, Texas Instruments

June 2016

The Dissertation of Chun-Kai Hsu is approved.

Professor Malgorzata Marek-Sadowska

Professor Luke Theogarajan

Professor Xin Li, Carnegie Mellon University

Dr. Kenneth Butler, Texas Instruments

Professor Kwang-Ting Tim Cheng, Committee Chair

March 2016

Test Data Analytics: Exploration of Hidden Patterns for Test Cost Reduction and Silicon Characterization

Copyright © 2016

by

Chun-Kai Hsu

Acknowledgements

Pursuing a Ph.D. is a long journey. I would not have reached this milestone without the generous support of my advisor, my committee, my friends, and my family. I will forever be grateful to Professor Kwang-Ting Cheng and his guidance. Professor Cheng is truly my role model of an academic professional. I acknowledge my committee members, Professor Malgorzata Marek-Sadowska, Professor Luke Theogarajan, Professor Xin Li, and Dr. Kenneth Butler, for their valuable feedback and insights on my research.

I am appreciative of Dr. Butler for his mentoring during my summer intern at Texas Instruments. I also want to thank John Carulli and Siddhartha Siddhartha from GlobalFoundries for their selfless share of industrial experience and knowledge. A special thank goes to Peter Sarson from ams AG. This dissertation would not have been completed without his aid.

I would like to acknowledge the financial support of the Gigascale Systems Research Center, the Semiconductor Research Corporation, and Taiwan Government.

As my labmate and the coauthor of my publications, I sincerely thank Fred Lin for his great helps to both my research and my life. I would also like to express my gratitude to Professor Chih-Tsun Huang and Professor Jing-Jia Liou from National Tsing Hua University. I graciously thank my labmates: Dr. Jen-Yi Wuu, Dr. Ming Gao, Dr. Peter Lisherness, Dr. Di-An Li, Dr. Yi-Chu Wang, Dr. Xin Yang, Dr. Amirali Ghofrani, Nicole Lesperance, Zhong Guan, Miguel Lastras, Rui Wu, Chong Huang, Ping-Lin Yang, Leilai Shao, and Yuyang Wang.

My life at Santa Barbara would have been difficult if not for the friendship

and company of my friends: Jacqueline Huang, Dr. Cheng-Ying Huang, Dr. Tony Lin, Dr. Chinhan Lin, Dr. Li-Hsien Sun, Dr. Chih-Chien Pan, Chien Sun, Peter Hsieh, Dr. Josephine Liu, Dr. Shao-Feng Hung, Kuochin Lien, and Ying-Chin Tseng. I would like to convey my special thanks to Te-Hsuan Chen from University of Michigan and Yu-Chieh Huang from National Chiao Tung University for the emotional support from thousands of miles away.

Finally, I am deeply grateful to my parents and my sister. I love my family and their words are always a great encouragement to me.

Curriculum Vitæ
Chun-Kai Hsu

Education

2016 (expected)	Ph.D. in Electrical and Computer Engineering, University of California, Santa Barbara, United States.
2008	M.S. in Electrical Engineering, National Tsin Hua University, Hsinchu, Taiwan.
2006	B.S. in Electrical Engineering, National Tsin Hua University, Hsinchu, Taiwan.

Experience

2011 – 2016	Graduate Research Assistant, University of California, Santa Barbara, United States.
2015	Graduate Intern, GlobalFoundries.
2012, 2013	Graduate Intern, Texas Instruments Inc.
2009 – 2011	Research Assistant, Industrial Technology Research Institute (ITRI), Taiwan.
2006 – 2008	Graduate Research Assistant, National Tsin Hua University, Taiwan.

Honors and Awards

ECE Dissertation Fellowship, University of California, Santa Barbara, 2015

Government Fellowship for Studying Abroad, Ministry of Education, Taiwan, 2011, 2012

Outstanding Student Scholarship, National Tsin Hua University, 2006, 2008

Publications

C.-K. Hsu, P. Sarson, and K.-T. Cheng, "Variation and Failure Characterization Through Pattern Classification of Test Data From Multiple Test Stages," submitted to *IEEE International Test Conference (ITC)*, 2016.

F. Lin, **C.-K. Hsu**, and K.-T. Cheng, "AdaTest: an Efficient Statistical Test Framework for Test Escape Screening," in *IEEE International Test Conference (ITC)*, 2015.

F. Lin, **C.-K. Hsu**, and, A. Busetto, and K.-T. Cheng, "Pairwise Proximity-Based Features for Test Escape Screening," in *IEEE International Conference on Computer-Aided Design (ICCAD)*, 2015.

F. Lin, **C.-K. Hsu**, and K.-T. Cheng, "Learning From Production Test Data: Correlation Exploration and Feature Engineering," *IEEE Asian Test Symposium (ATS)*, 2014.

F. Lin, **C.-K. Hsu**, and K.-T. Cheng, "Feature Engineering With Canonical Analysis for Effective Statistical Tests Screening Test Escapes," in *IEEE International Test Conference (ITC)*, 2014.

S. Zhang, F. Lin, **C.-K. Hsu**, K.-T. Cheng, and H. Wang, "Joint Virtual Probe: Joint Exploration of Multiple Test Items' spatial Patterns for Efficient Silicon

Characterization and Test Prediction," in *Design Automation and Test in Europe (DATE)*, 2014.

C.-K. Hsu, F. Lin, K.-T. Cheng, W. Zhang, X. Li, J. Carulli, and K. Butler, "Test Data Analytics — Exploring Spatial and Test-Item Correlations in Production Test Data," in *IEEE International Test Conference (ITC)*, 2013.

S.-S. Chen, **C.-K. Hsu**, H.-C. Shih, J.-C. Yeh and C.-W. Wu, "Processor and DRAM Integration by TSV-Based 3-D Stacking for Power-Aware SOCs," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2008.

C.-K. Hsu, L.-M. Denq, M.-Y. Wang, J.-J. Liou, C.-T. Huang, and C.-W. Wu, "Area and Test Cost Reduction for On-Chip Wireless Test Channels with System-Level Design Techniques," in *IEEE Asian Test Symposium (ATS)*, 2008.

Abstract

Test Data Analytics: Exploration of Hidden Patterns for Test Cost Reduction and Silicon Characterization

by

Chun-Kai Hsu

The manufacturing test process for a modern integrated circuit encounters excessively long test time and produces huge amount of test data. There is valuable information hidden in the test data about the device under test (DUT), far more than the binary go/no-go classification. Exploring the hidden correlations and patterns in the test data allows better understanding of the DUT and therefore leads to broad applications, such as test cost reduction and silicon characterization for discovering parametric variations and weak links in the manufacturing process.

The first part of this dissertation proposes a methodology with supporting statistical learning algorithms for test time and cost reduction through exploiting both spatial and inter-test-item correlations in the test data. The proposed algorithm identifies inter-test-item correlations for removing costly and unnecessary test items from a test program. An integrated method further reduces test time by taking into account spatial correlations of test data across a wafer and maximizing the number of test items whose values can be predicted without measurement. A case study of a high-volume industrial device demonstrates that some test items can be identified for removal from the test program without compromising test

quality and shows the significant reduction of test time.

In the second part of the dissertation, a framework for characterizing systematic variations and failures through exploring the hidden patterns of test data from multiple test stages is developed. The framework provides prediction of process variations with a fine resolution based on a limited number of probed process parameters, and extracts spatial patterns from both process parameters and production tests. A template matching technique exploits these spatial patterns to discover connections between process variations and failures detected by production tests. Experimental results demonstrate that the proposed framework reveals comprehensible and significant correlations in an industrial test dataset.

The third part of the dissertation describes a software toolbox dedicated to test data analytics developed in the course of this research. The toolbox provides flexible and scalable functions for parsing, processing, learning and display test data. The toolbox, which is released for non-commercial use, also provides examples and application programming interface for test data analysis.

Contents

Curriculum Vitae **vi**

Abstract **ix**

List of Figures **xiv**

List of Tables **xvi**

1 Introduction **1**
1.1 Proposed Methodologies 1
1.2 Hidden Patterns in Test Data 3
1.2.1 Spatial Correlation 4
1.2.2 Inter-Test-Item Correlation 5
1.2.3 Temporal Correlation 6
1.3 Literature Review 8
1.3.1 Techniques and Methodologies 8
1.3.2 Applications 10

2 Pattern Exploration **13**
2.1 Introduction 13
2.2 Correlations Among Test Items 15
2.2.1 Inter-Test-Item Correlation Model 16
2.2.2 Candidate Test Items for Removal From Test Program . . 18
2.2.3 The Group Lasso Regression Problem 20
2.2.4 The SOCP Problem 21
2.2.5 Weighted Group Lasso 22
2.3 Wafer-Lever Spatial Variation Prediction 23
2.3.1 Background: Virtual Probe 23
2.3.2 Joint Virtual Probe 27

2.3.3 Runtime of M-FOCUSS for JVP Estimation 33
2.3.4 Experimental Results 36
2.4 Summary 41

3 Test Time Reduction **42**
3.1 Introduction 42
3.2 Optimization for Test Time Reduction 43
3.2.1 Reflecting Item's Test Time 45
3.2.2 Weight Assignment for TTR Utilizing Both Spatial and Inter-Test-Item Correlations 45
3.3 Test Methodology Based on GL and WGL 46
3.3.1 Criteria for Classifying Predictability 47
3.3.2 Challenges of Random Defects, Prediction Error, and Process Stationarity 50
3.3.3 Test Procedure 52
3.4 Experimental Results 57
3.4.1 Inter-Test-Item Correlations Analysis 57
3.4.2 TTR With Test Time of Individual Item 64
3.4.3 Integrating Both Spatial and Inter-Test-Item Correlations 65
3.5 Summary 68

4 Silicon Characterization **70**
4.1 Introduction 70
4.2 Background: Biclustering and FABIA 74
4.3 Implementation 77
4.3.1 Application Flow 77
4.3.2 Spatial Modeling With a Greater Resolution 79
4.3.3 Pattern Extraction Using Biclustering 82
4.3.4 Template Matching 88
4.4 Experimental Results 89
4.4.1 WAT Data Analysis 89
4.4.2 Matches Between Process Parameters and Product Tests . 93
4.5 Summary 98

5 Test Data Analytics Toolbox **99**
5.1 Introduction 99
5.2 Implementation 100
5.2.1 Test Data Parsing 101
5.2.2 Test Data Selection 103
5.2.3 Learning Algorithms 104

5.3 Data Structure 104
5.3.1 `data` Namespace 104
5.3.2 `vp` Namespace 108
5.3.3 `gl` Namespace 109
5.4 Examples 111
5.4.1 VP Example 111
5.4.2 WGL Example 112

6 Conclusions 114

Bibliography 116

List of Figures

1.1 Test data correlations in three domains. 3
1.2 Spatial correlation. 5
1.3 Inter-test-item correlation. 6
1.4 Temporal correlation. 7

2.1 Spatial variation prediction. 24
2.2 Statistics of common near-zero DCT coefficients. 30
2.3 Runtime trends of JVP versus the problem scale. 34
2.4 Runtime comparison between JVP and VP. 35
2.5 Setting thresholds for classifying test items. 38

3.1 Predictability of a test item. 49
3.2 The flow of pre-test analysis. 54
3.3 The flow of test application. 56
3.4 Die map of measured values and predicted values. 58
3.5 Test items with the largest coefficient α. 58
3.6 Number of predictable test items versus penalty parameter λ. . . 59
3.7 Predictable test items versus the “predicting” test items. 61
3.8 Error boxplot of each predictable test item. 62
3.9 The predictability of test items. 63
3.10 Comparison of the GL and VP+WGL prediction errors. 67

4.1 A multiplicative model with three biclusters. 76
4.2 The application flow of the proposed framework. 78
4.3 The spatial relations between sites/shots and dies. 80
4.4 Color-coded predicted wafer maps of WAT parameters. 81
4.5 The procedure of extracting grayscale patterns. 84
4.6 The procedure of extracting binary patterns. 87
4.7 The mapping between site locations and die locations. 90

4.8 A diagram showing the factorized matrix $\mathbf{\Lambda}$. 91
4.9 Linear correlation matrix of twelve WAT items of Bicluster 1. . . 92
4.10 A grayscale pattern gallery. 94
4.11 Binary pattern examples. 94
4.12 A plot showing the significance of WAT parameters. 98

5.1 The flow of parsing test data from sources. 102
5.2 An example of performing the VP algorithm. 111
5.3 An example of performing the WGL algorithm. 113

List of Tables

2.1 Comparison of VP and JVP 40
2.2 Percentage of Test Items Classified as Predictable 40
2.3 Comparison in Test Application Phase 41

3.1 The Predictability Levels 48
3.2 Test Items Required to Cover All Random Defect Suspects 51
3.3 Number of Predictable Item vs. Margin from Spec Limits 62
3.4 Number of Escaped Dies for Various GL Models 64
3.5 Test Time Improvement by WGL 65
3.6 Intuitive Merge vs. Weighted Merge 66
3.7 Summary of Test Time Saving for Various Strategies 68

4.1 Matches Between WS Items and WAT Parameters 96

Chapter 1

Introduction

Efficient and effective testing and diagnosis could significantly improve product quality and manufacturing yield for modern integrated circuits (ICs). In order to screen out systematic and variation-induced failures, more test items have been added to test programs at different manufacturing stages, which result in excessively long test time and a huge amount of test data. There is valuable information hidden in the test data about the device under test (DUT), far more than the pass/fail judgment from each test item. Mining such hidden patterns and correlations could be useful for test time reduction [1], outlier prediction [2, 3, 4], and silicon characterization [5, 6].

1.1 Proposed Methodologies

This dissertation proposes statistical learning methodologies for test time reduction and silicon characterization through exploring hidden patterns in test data. Chapter 2 describes two techniques: *weighted group lasso* (WGL) and *joint*

virtual prove (JVP) that are developed for characterizing inter-test-item and spatial correlations, respectively. WGL identifies correlations among multiple test items for removing test items from the test program without compromising test quality. Moreover, WGL allows factoring in the distinct test times/costs of individual test items. As a result, WGL tends to remove more expensive test items from the test program. JVP concurrently captures wafer-level spatial correlations of multiple test items based on a small subset of measurements. In addition, JVP is computationally efficient and is applicable for real-time analysis due to jointly predicting the spatial patterns of multiple test items.

Chapter 3 introduces a methodology that exploits both spatial and inter-test-item correlations in the test data for test time and cost reduction. In general, test time and test cost are highly correlated due to the high capital cost and operating expenses of test equipment. For simplicity and consistency, in the rest of this dissertation such methods which can reduce both test time and test cost are referred as *test time reduction* (TTR) methods. The proposed TTR method maximizes the number of test items whose values can be predicted without measurement in a test program.

Chapter 4 presents a framework for characterizing systematic variations and failures through pattern classification of test data from different test stages. An unsupervised *biclustering* technique is utilized to extract spatial patterns from process parameters and production tests, respectively, by conducting both item-to-item and die-to-die correlations in the test data. A template matching technique then exploits these spatial patterns to reveal comprehensible correlations between process parameters and production failures.

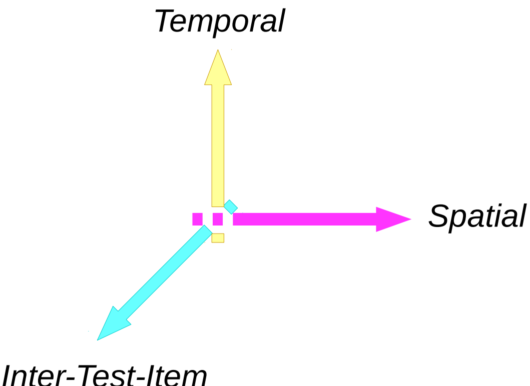


Figure 1.1: Test data correlations in three domains.

Chapter 5 demonstrates a MATLAB toolbox dedicated to test data analytics. In order to support research projects with test data from a wide range of sources, this modular and object oriented toolbox provides customizable parser for accessing test datasets in different formates. Moreover, the toolbox is integrated with several leaning algorithms and provides an universal application programming interface for every learning algorithm. The toolbox greatly enhances the efficiency for developing applications of test data analytics.

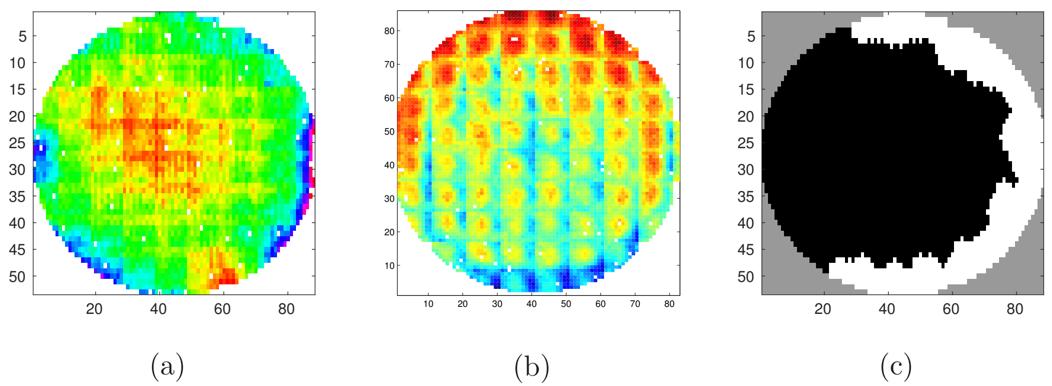
1.2 Hidden Patterns in Test Data

Patterns of test data are the components with discernible regularity hidden in a series of test measurements. Patterns are usually formed due to correlations caused by systematic variations or failures, such as the process variations during manufacturing and the measuring errors from automatic test equipment (ATE). As shown in Figure 1.1, test data correlations can be classified into three domains: spatial correlations, inter-test-item correlations, and temporal correlations [7].

1.2.1 Spatial Correlation

Spatial correlations, also known as die-to-die correlations, indicate that the measurement of a die is somehow correlated to the measurements of the other dies on the same wafer. Spatial correlations are visually observed as unique spatial patterns with respect to the amplitude of measurements of dies on a wafer map. One approach to interpret such patterns is color-coding the measurements of a wafer map into a two-dimensional image. In addition, the pass/fail outcomes of functional tests (i.e., thresholded performance measurements) form a binary wafer map, which is a special type of spatial patterns. Figure 1.2 shows three wafer maps with spatial patterns based on different tests. Figure 1.2a and Figure 1.2b are formed by test items with parametric measurements (dies with different measurements are denoted by pixels in different colors), and Figure 1.2c is based on a functional test with binary outcomes (pass and fail dies are denoted by white and black pixels, respectively). Such spatial patterns reveal the systematic variations across a wafer.

Studies that utilize spatial patterns/correlations have been proposed. In [8], Liu presented a method to construct spatial correlation models from test measurements using generalized least square fitting. Li *et al.* [9] and Kupp *et al.* [10] proposed techniques based on compressed sensing and Gaussian process, respectively, for predicting spatial variations from a small set of measurement data. In [11, 12, 13, 5, 14], these techniques were further improved towards more accurate prediction and less computation time.



(a) (b) (c)

Figure 1.2: Wafer maps with spatial correlations. (a) and (b) illustrate spatial patterns in two parametric test items while (c) shows spatial pattern in a functional test item. Note that the region outside of the wafer map in (c) is shown in gray for better visualization.

1.2.2 Inter-Test-Item Correlation

Correlations can also exist among different test items when, for example, the same test applied multiple times under different electrical or environmental settings, or different tests targeting the same functionality of a chip. Such correlated measurements result in numerical and abstract patterns that can be identified by several techniques, from simple linear regression to complex support vector machines. Moreover, a set of dies with strong inter-test-item correlations may be spatially clustered and form visually interpretable spatial patterns as shown in Figure 1.3, where three different test items that are performed on the same wafer exhibit similar spatial patterns. There exist potential systematic variations among these test items.

There have been known applications in production testing that utilize the inter-test-item correlations. An early Monte Carlo based approach proposed by Brockman and Director [15] analyzed the joint probability distributions of test

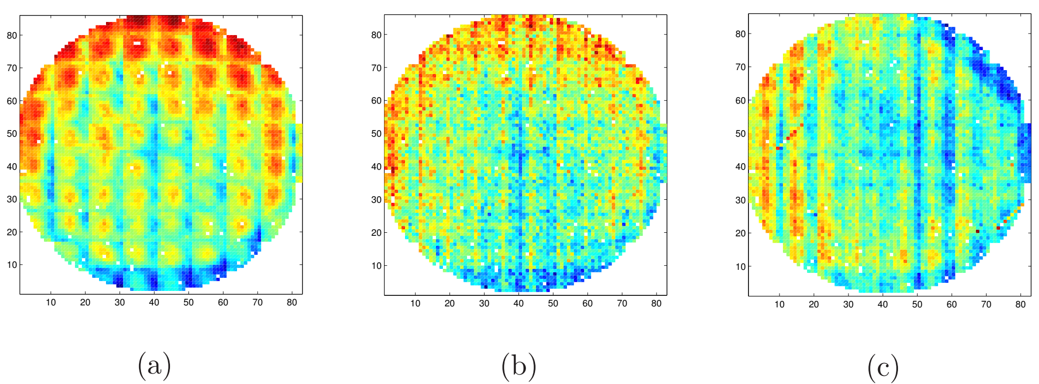


Figure 1.3: Wafer maps with inter-test-item correlations. (a), (b), and (c) are wafer maps of color-coded measurements from three different test items, respectively.

items for constructing a regression model of the untested performances. Chen and Orailoglu [16] examined the implication of inter-test-item correlations for test set minimization based on correlation graph model. Once inter-test-item correlations are identified, test items can be reordered or eliminated for more efficient defect screening and test time reduction [17, 18, 19, 20, 21, 1].

1.2.3 Temporal Correlation

Temporal correlations describe the variations across wafers and lots, i.e., measuring the same performance or process parameter at different times, or by different test equipments. Monitoring temporal correlations reveals the stability and robustness of the manufacturing and test processes. Taking into account temporal correlations can possibly further improve the accuracy and scalability of applications which solely rely on spatial or inter-test-item correlations. Figure 1.4 illustrates the temporal correlations between two test stages. Figure 1.4a and 1.4d are two similar patters extracted from two test dataset of the same product but

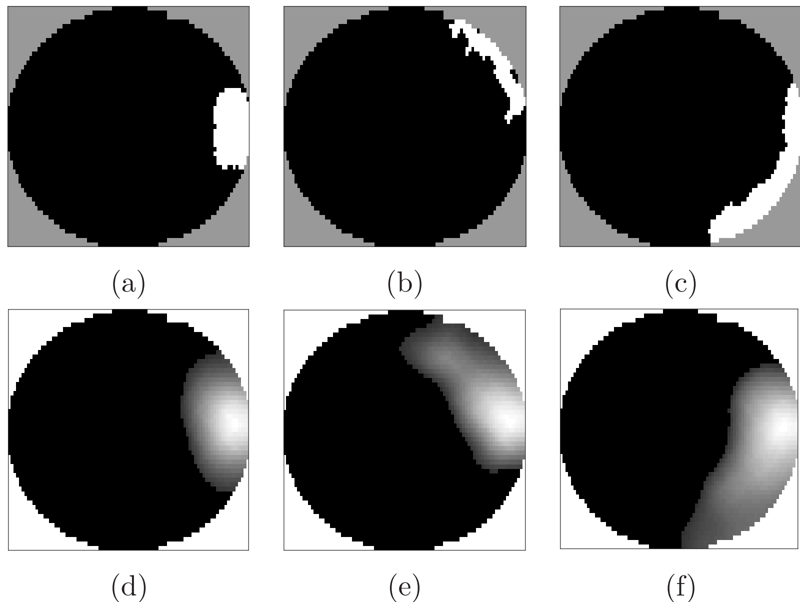


Figure 1.4: Wafer maps with temporal correlations. (a), (b), and (c) are binary patterns extracted from production test data. (d), (e), and (f) are grayscale patterns extracted from electronic test data. Details are described in Chapter 4.

at different test stages, respectively. The similarities between these two patterns in shape and location indicate the temporal correlations. The pair of Figure 1.4b and 1.4e, and the pair of Figure 1.4c and 1.4f are two other examples.

Some previous studies have been presented for discovering correlations between test data at different test stages. In [22], Devarakond *et al.* predicted electronic test measurements by production test measurements using regression analysis tools. In [23], Ahmadi *et al.* estimated the production yield by electronic test data through multivariate regression techniques. Bayesian model fusion framework is proposed to incorporate the knowledge of temporal correlations among wafers and lots [24, 25, 26, 27].

1.3 Literature Review

There have been many studies proposed within the scope of test data analytics. Selected works are summarized and classified in this section.

1.3.1 Techniques and Methodologies

The *alternate test* methodology is proposed to reduce the large number of specification tests through crafting new test input stimulus with test response that has maximum sensitivity to the specifications, and mapping the observed test response to multiple specifications at once [28]. In [29], Variyam *et al.* presented a fast transient testing methodology for predicting the performance parameters of analog circuits. Akbay and Chatterjee [30] explored a fault-based alternate test for reducing the complexity of ATE based on the abstractions of physical phenomena that cause specification violations. In [31], Voorakaranam *et al.* proposed a signature test methodology for test acceleration through directly tracking the ability of input test waveforms to predict the test specification values. Mannath *et al.* [32] demonstrated a methodology to replace a set of tests with structurally-based Built-in Self Tests. Ayaril *et al.* [33] presented an alternate test implementation based on model redundancy.

Techniques based on *Bayesian inference* are developed to predict the performance of late stages based on the measurements of early stages, such as learning temporal correlations from test data. Lee *et al.* [34] presented a Bayesian learning framework for accurately modeling spatial delay correlations in statistical static timing analysis. In [19], Gotkhindikar *et al.* employed Bayesian statistics to model

the per test failure rates for an on-tester adaptive test scheme. Li *et al.* [24] proposed a Bayesian model fusion (BMF) framework to minimize simulation and/or measurement cost through statistically modeling the performance correlation between early design stages and late test stages. In [25], Zhang *et al.* demonstrated a wafer-level spatial variation modeling technique based on BMF. Ahmadi *et al.* [27] and Fang *et al.* [26] further utilized BFM for fab-to-fab yield forecasting and yield estimation of binary simulation/measurement outcome, respectively.

Several studies exploits the breakthroughs in *compressed sensing*, which is a signal processing technique for accurately predicting variations and reconstructing a signal from a small set of measurement data by solving an underdetermined linear system [35, 36, 37]. In [9], Li *et al.* proposed a technique, virtual probe (VP), to recover full-wafer spatial variation from a small set of dies in a wafer. Zhang *et al.* [11, 12] further reduced the number of sampled measurements required by VP. A TTR framework utilizing VP was presented by Chang *et al.* [38]. Chung *et al.* [39] and Gonçalves *et al.* [14] further improved the efficiency of VP through solving the compressed sensing problem by orthogonal matching pursuit and dual augmented Lagrangian method, respectively.

Multivariate analysis is used to analyze data in the sense that numerous observations or variables are obtained for each individual or unit studied [40]. In [41], O'Neill presented that outlier analysis using principal component can screen out defective parts. Bounceur *et al.* [42] employed the copulas theory for parametric test metrics estimation, such as estimating parametric defect level and yield loss. Akkouche *et al.* [21] demonstrated an approach for test ordering through performing multivariate parametric statistical modeling on a small set of functional

devices. In [43], Sumikawa *et al.* studied the potential of capturing customer returns with models constructed based on multivariate analysis of parametric wafer sort test measurements.

Test data are presented by a fixed number of features (i.e., test measurements) which can be binary, categorical or continuous. *Feature egineering* or *feature extraction* is developed for finding a good data representation through employing feature construction and feature selection. Feature construction converts raw data into a set of useful features while feature selection is performed to select relevant and informative features [44]. In [45], Krishnan and Kerkhoff exploited multivariate reliability classifier model with Mahalanobis distance as a feature set for the identification of outliers. Lin *et al.* [2], provided feature construction and feature reduction techniques based on canonical analyis for screening potential test escapes. In [4], Lin *et al.* proposed a new set of proximity-based features to expose the abnormalities of test escapes.

1.3.2 Applications

The purpose of *test compaction* is to reduce test time through identifying and eliminating information redundancy in tests. Statistical learning techniques are developed to predict pass/fail decision of circuits based on only a subset of tests. In [46], Biswas and Blanton proposed a statistical test compaction method based on decision trees for eliminating redundant tests from the complete specification-based test set of an integrated device. In [17], Biswas and Blanton also employed boolean minimization and optimized test covering to identify redundant tests for test compaction. Stratigopoulos *et al.* [18] exploited redundancy in the specifica-

tion tests of an RF device for test compaction based on a multi-objective genetic algorithm.

Adaptive test and *test reordering* schemes pursue effective test item ordering for detecting failures earlier and benefiting stop-on-fail test programs. In [16], Chen and Orailoglu proposed a test selection algorithm through capturing systematic process variations and leading to an early detection of faults. Yilmaz *et al.* [47, 48] presented a per-device adapting test list compaction method with additional defect screening mechanism, which utilizes on-line measurements to tailor an optimized test list. In [20], Yilmaz and Ozev proposed an adaptive approach for multi-site testing through incorporating device-to-device correlations of parallel neighbor devices.

When the defects on the wafer form spatial patterns/signatures, it usually indicates the identification of potential problems, such as process variations and mismatch between equipments. *Wafer clustering* and *wafer classification* group wafers with similar spatial signatures for exploration and diagnosis of systematic failures. In [49], Chen and Liu developed a system based on a neural-network architecture for recognizing the spatial patterns of clustered defects. Yuan *et al.* [50] proposed a model-based clustering technique with nearest-neighbor noise removal for identifying clustered defect patterns. Ooi *et al.* [51] presented an automation tool with cluster extraction algorithm based on image segmentation techniques, and defect-cluster recognition algorithm using an alternating decision tree classifier. In [52], Zhang *et al.* proposed a methodology for automatic clustering of wafer spatial signatures based on sparse regression and complete-link hierarchical clustering. Wu *et al.* [53] performed failure pattern recognition and wafer

map similarity ranking on large-scale test datasets through employing a reduced representation of wafer maps based on feature extraction.

Statistical learning approaches for *yield estimation* have been under development for high-volume production devices. In [54], Wang *et al.* proposed a BMF-based technique for parametric yield estimation through utilizing the simulation data from an early stage. Ahamdi *et al.* [23] utilized the correlations between electronic test and probe test measurements for yield estimation. Kang *et al.* [55] proposed prediction models using wafer map features to predict die-level failures in the final test through a random forest algorithm.

There have been studies focus on *outlier detection* by data mining and statistical analysis of the test data. The goals of such studies are to improve the outgoing product quality and reliability through identifying the abnormality in, but not limited to, dies, wafers, and lots. In [56], Butler *et al.* proposed statistical burn-in avoidance techniques based on fixed-limit analyses with parametric or non-parametric statistics. Fang *et al.* [57] demonstrated an outliers screening approach based on utilizing the measurements of neighboring dies and the measurements of different blocks within the target. In [58], O'Neill proposed the concept of statistical test that looks for outliers from the patterns of existing measurements without additional physical measurements. Nahar *et al.* [59] presented a statistical approach to utilize production wafer probe data for identifying at risk material early in the production process. Sumikawa *et al.* [60] and Chen *et al.* [61] utilized one-class support vector machines and decision tree classification, respectively, to predict systematic failures and customer returns.

Chapter 2

Pattern Exploration

2.1 Introduction

Process variations at very small process nodes cause significant deviations in device performance. In contrast to random defects, failures resulting from parametric variations exhibit much stronger correlations at both die and wafer levels. As described in Section 1.2, modeling such parametric variations and taking them into account in the design and test processes help increase design robustness and improve product yield.

In this chapter, two techniques: *weighted group lasso* (WGL) and *joint virtual probe* (JVP) are proposed for characterizing inter-test-item and spatial correlations, respectively. WGL, which is based on a statistical regression technique called group lasso [62, 63], is developed to capture the correlations among test items using the test data of training chips. Not only identifying correlated parametric test items in any given test program, WGL could also find correlations

between test items in different test phases, such as wafer probe tests and package tests. The correlated test items can be removed from explicit testing and their values can be predicted by the measured values of other test items of the same chip. If the die IDs are traceable, such identified correlations can be used to reduce test time by removing those nearly redundant package test items.

Different test items often incur different amounts of test time and cost. For test time reduction, it is preferable to predict more costly test items if such options exist. WGL allows factoring in the distinct test times/costs of individual test items. As a result, WGL tends to find a solution where more expensive test items are more favored than less expensive ones as candidates for removal from the test program.

In [9], Li *et al.* proposed the virtual probe (VP) technique based on compressed sensing [35, 36, 37]. VP is formulated as a linear inverse problem. Based on the observation that, for a VP predictable test item, the vast majority of DCT coefficients are near-zero, i.e., with high sparsity, VP can accurately capture the wafer-level spatial correlations of a test item from a small subset of measurements. The captured spatial correlations can then be used to predict the performance of other dies on the same wafer without measurement. VP predicts the spatial variations without training a model [9], which is thus applicable for real-time analysis during test application. However, the computation time becomes a critical factor for such real-time applications.

JVP is proposed for concurrently capturing spatial patterns of multiple test items. In contrast to VP's limitation of deriving spatial pattern for one test item at a time, JVP jointly predicts the spatial patterns of multiple test items, while

the pattern for each test item can be distinct (but with some degree of correlation). If a set of K test items have similar sparsity profiles, i.e., having some similarity in locations of near-zero DCT coefficients, these K test items can be combined together to re-formulate the K linear inverse problems into a single linear inverse problem. Existing optimization algorithms, such as MMV FOCal Under-determined System Solver (M-FOCUSS) [64], can be used to find a sparse solution for such a problem. When applied to a group of test items, which have sufficient similarity among them, JVP could achieve a higher accuracy than VP for each individual item's spatial pattern prediction. Furthermore, because of concurrent consideration of multiple items, JVP incurs significantly less computation time than VP for analysis.

The rest of this chapter is organized as the following. Section 2.2 describes the WGL method for learning inter-test-item correlations while different weights can be assigned to different test items in the formulations. Section 2.3 describes the fundamental assumption and mathematical formulation of the proposed JVP method with experimental results on two industrial datasets. Finally, we conclude in Section 2.4.

2.2 Correlations Among Test Items

There exist correlations among the measurement data for different test items taken from the same chip. One goal of our methodology is to learn such inter-test-item correlations from the test data of a set of training chips. Specifically, the objective is to identify the test items whose values can be predicted as a linear

combination of the measured values of other test items of the same chip.

In this chapter, we focus on parametric test item only, for which the test value of the chip-under-test is a real number. In the first subsection, we define the inter-test-item correlation and how to model it. In the second subsection, we discuss the use of the correlation for identifying *predictable* test items whose values can be predicted by the values of other test items of the same chip. Finally, we show the statistical regression method that can efficiently find such predictable test items.

2.2.1 Inter-Test-Item Correlation Model

A first-order linear correlation may exist among test items. If such a correlation exists, we can predict the values of some test items, without actual measurement, using linear combinations of the measured values of other test items. If no such correlation exists, all test items must be physically measured. We define the inter-test-item correlation for one test item as

$$\hat{\boldsymbol{f}}_k = \sum_{i=1}^{n} \alpha_{ki} \boldsymbol{f}_i + C_k \,, \tag{2.1}$$

where $\hat{\boldsymbol{f}}_k$, a vector, denotes the predicted values of the target (the kth test item) for a set of chips, $\boldsymbol{f}_i$, a vector too, denotes the measured values of the ith test item of the same set of chips, n is the number of test items, and C_k is an offset constant. An element in $\boldsymbol{f}_i$ and $\hat{\boldsymbol{f}}_k$ represents the predicted or measured value of a chip and the dimension of these vectors is the number of chips in the set. We assume that the statistical characteristics, such as the correlation, are stationary (i.e., not varying) over all chips.

The vector of measured values of the kth test item, $\boldsymbol{f}_k$, is also included in the right hand side of Equation (2.1). If $\hat{\boldsymbol{f}}_k$ is predictable based on the measured values of other test items, there exists an appropriate value for every α_{ki}, $i \neq k$, to form the model in Equation (2.1) where α_{kk} is equal to zero. On the other hand, if $\hat{\boldsymbol{f}}_k$ is unpredictable, one trivial solution is that all α's except α_{kk} are zero and α_{kk} is equal to one.

If n test items are considered at the same time, the correlations are represented by a set of linear equations, i.e., n equations based on Equation (2.1) for $k = 1, 2, \cdots, n$. Without loss of generality, we normalize each test item to be zero mean and unit variance. As a result, the correlation model can be represented as

$$\begin{cases} \hat{\boldsymbol{F}}_1 = \sum_{i=1}^{n} \alpha_{1i} \boldsymbol{F}_i \\ \hat{\boldsymbol{F}}_2 = \sum_{i=1}^{n} \alpha_{2i} \boldsymbol{F}_i \\ \quad \vdots \\ \hat{\boldsymbol{F}}_n = \sum_{i=1}^{n} \alpha_{ni} \boldsymbol{F}_i \, , \end{cases} \tag{2.2}$$

where $\hat{\boldsymbol{F}}_i$ and $\boldsymbol{F}_i$ denote the normalized predicted values and the normalized measured values for the ith test items, respectively. Note that we no longer need Cs for these normalized equations. Assuming that we derive this correlation based on d chips, these n vector equations correspond to a total of nd scalar equations.

The inter-test-item correlation model can therefore be encoded by a matrix

formed by all α's in (2.2) as:

$$\begin{bmatrix} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1n} \\ \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{n1} & \alpha_{n2} & \cdots & \alpha_{nn} \end{bmatrix}. \tag{2.3}$$

This model uses n^2 variables (α_{ij} for $i, j = 1, 2, \cdots, n$) to represent the correlations. Note that (2.3), representing the correlations derived from Equation (2.2), is different from the conventional correlation matrix defined in statistics for the test items in which the (i, j) element represents $\text{corr}(\boldsymbol{F}_i, \boldsymbol{F}_j)$. Given the measured test data of n test items of d chips, we have many choices of regression and learning methods to derive the matrix in (2.3). We will introduce an efficient way of solving this problem in the following subsections. An exemplar solution is the identity matrix where each test item is correlated to itself only.

2.2.2 Candidate Test Items for Removal From Test Program

Based on Equation (2.1), a zero coefficient indicates that measurement of the corresponding test item is not needed for deriving the target item's value. For example, if we have $\alpha_{k1} = 0$ and $k = 1$ in Equation (2.1), we can predict the first test item ($\hat{\boldsymbol{f}}_1$) without relying on the actual measurement of the same item ($\boldsymbol{f}_1$). Considering all prediction equations simultaneously, if all coefficients corresponding to a test item are all zeros, we can eliminate the test item from a

test program for actual measurement. For example, referring to Equation (2.2), if the condition:

$$\forall i \ni \mathbb{N} \ \wedge \ 1 \leq i \leq n, \ \alpha_{i1} = 0 \tag{2.4}$$

is true, we can conclude that every test item, including the first test item itself, can be derived without relying on actual measurement of the first test item ($\boldsymbol{f}_1$ or $\boldsymbol{F}_1$). Test item one is then a candidate for removal from the test program. In the following, we refer to such a test item as a *candidate* test item. The general condition for a candidate test item is

$$\forall i \ni \mathbb{N} \ \wedge \ 1 \leq i \leq n, \ \alpha_{ik} = 0 \ \Rightarrow \ \text{item } k \text{ is a candidate.} \tag{2.5}$$

In the correlation matrix (2.3), a column of zeros indicates that the corresponding test item is a candidate. Consider the following example:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.2 & 0 & 0.5 & 0 & 0.1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.6}$$

The second test item is a candidate test item because the second column consist of only zeros. In addition, no other candidate test item exists and we need to explicitly test each of them.

2.2.3 The Group Lasso Regression Problem

According to the model definition described in the previous subsection, we can formulate the problem of finding the correlation matrix (2.3) as a minimization problem which attempts to minimize the difference between predicted values and measured values. The minimization problem for finding a correlation model is formulated as

$$\arg\min_{\alpha} \sum_{i=1}^{n} \left\| \boldsymbol{F}_i - \hat{\boldsymbol{F}}_i \right\|_2^2 , \tag{2.7}$$

which is equivalent to

$$\arg\min_{\alpha} \sum_{i=1}^{n} \left\| \boldsymbol{F}_i - \sum_{j=1}^{n} \alpha_{ij} \boldsymbol{F}_j \right\|_2^2 , \tag{2.8}$$

where $\| \cdot \|_2$ denotes the L_2 norm.

For many regression applications, it is often desirable to find a sparse solution for (2.7) that has as many zero coefficients as possible. The lasso (least absolute shrinkage and selection operator) method [62] was designed to find sparse solutions by adding an L_1 norm penalty to (2.7), resulting in a revised minimization problem as

$$\arg\min_{\alpha} \sum_{i=1}^{n} \left\| \boldsymbol{F}_i - \hat{\boldsymbol{F}}_i \right\|_2^2 + \lambda^* \sum_{i=1}^{n} \sum_{j=1}^{n} |\alpha_{ij}| , \tag{2.9}$$

where λ^* is a penalty parameter to control the trade-off between prediction error and the sum of all absolute values of alphas. In lasso, increasing λ^* forces more coefficients in correlation matrix to approach zero.

However, minimizing the number of nonzero coefficients in the correlation matrix does not address our goal of maximizing the number of candidates, i.e., maxi-

mizing the number of test items meeting Condition (2.5). We therefore introduce *group lasso* (GL) [63] which attempts to find a sparse solution which maximizes the number of columns with all near-zero entries in the solution matrix. The main idea of GL is to group coefficients corresponding to the same test item together and revise the minimization problem as

$$\begin{aligned} \arg\min_{\alpha} \quad & \sum_{i=1}^{n} \left\| \boldsymbol{F}_i - \hat{\boldsymbol{F}}_i \right\|_2^2 \\ \text{subject to} \quad & \lambda \geq \sum_{g=1}^{n} \sqrt{\sum_{i=1}^{n} \alpha_{ig}^2} \,. \end{aligned} \tag{2.10}$$

The term $\sqrt{\sum_{i=1}^{n} \alpha_{ig}^2}$ combines all coefficients in the gth column of the correlation model (2.3) together to form a group. As λ decreases, GL attempts to find a solution with nonzero groups instead of just nonzero coefficients.

2.2.4 The SOCP Problem

Because of the quadratic terms in the optimization problem (2.10), we reformulate it as a second-order cone programming (SOCP) problem that can be solved efficiently by interior point methods [65]. The reformulated problem becomes

$$\begin{aligned} \underset{\alpha, u}{\text{minimize}} \quad & T \\ \text{subject to} \quad & T \geq \sqrt{\|\boldsymbol{u}_1\|_2^2 + \cdots + \|\boldsymbol{u}_n\|_2^2} \,, \\ & g_i \geq \sqrt{\alpha_{1i}^2 + \alpha_{2i}^2 + \cdots + \alpha_{ni}^2} \,, \quad 1 \leq i \leq n \,, \\ & \boldsymbol{u}_i = \boldsymbol{F}_i - \hat{\boldsymbol{F}}_i \,, \quad 1 \leq i \leq n \,, \\ & \lambda = g_1 + g_2 + \cdots + g_n \,, \end{aligned} \tag{2.11}$$

where $\boldsymbol{u}_i$ denotes the difference between predicted and measured values. All $\boldsymbol{F}_i$, $\hat{\boldsymbol{F}}_i$, and $\boldsymbol{u}_i$ are vectors with d dimensions. In general, a smaller λ would more likely result in a sparser solution, i.e., more near-all-zero columns in the correlation matrix (equivalent to having more candidate test items).

2.2.5 Weighted Group Lasso

In solving the optimization problem (2.10), GL treats every item equally and tends to find a solution with a maximum number of predictable test items. However, as different test items incur different test times, maximizing the number of predictable test items does not necessarily maximize the reduction of test time. Weighted group lasso (WGL), an extension of GL, is proposed to address this issue.

WGL, whose basic formulation is similar to that of GL, is expressed as

$$\begin{aligned} \arg\min_{\alpha} \quad & \sum_{i=1}^{n} \left\| \boldsymbol{F}_i - \hat{\boldsymbol{F}}_i \right\|_2^2 \\ \text{subject to} \quad & \lambda \geq \sum_{g=1}^{n} w_g \sqrt{\sum_{i=1}^{n} \alpha_{ig}^2}\,, \end{aligned} \tag{2.12}$$

where w_g denotes the weight of the gth test item (the gth group). A weight for the corresponding test item is therefore incorporated to reflect its actual test time. In

the SOCP form, WGL is formulated as

$$\begin{aligned}
\underset{\alpha,u}{\text{minimize}} \quad & T \\
\text{subject to} \quad & T \geq \sqrt{\|\boldsymbol{u}_1\|_2^2 + \cdots + \|\boldsymbol{u}_n\|_2^2}\,, \\
& g_i \geq \sqrt{\alpha_{1i}^2 + \alpha_{2i}^2 + \cdots + \alpha_{ni}^2}\,, \qquad 1 \leq i \leq n\,, \\
& \boldsymbol{u}_i = \boldsymbol{F}_i - \hat{\boldsymbol{F}}_i\,, \qquad 1 \leq i \leq n\,, \\
& \lambda = w_1 g_1 + w_2 g_2 + \cdots + w_n g_n\,.
\end{aligned} \tag{2.13}$$

Groups of α's with a larger weight will be more dominant in the constraint in (2.12) than the groups with a smaller weight. Therefore, WGL tends to find a solution that minimize the values of α's for heavier-weight groups. This results in a higher probability that a group with larger weight would have more near-zero α's, i.e., the corresponding item has a higher probability to be a candidate test item for removal from the test program.

2.3 Wafer-Lever Spatial Variation Prediction

2.3.1 Background: Virtual Probe

This subsection describes the statistical regression method, virtual probe, in more detail as it is integrated into several proposed methodologies of this dissertation. The essence of VP is to test only a subset of dies at selected locations on a wafer, transform the measurements into spatial frequency domain, and use a statistical algorithm to accurately recover the test values of the remaining dies [9, 11, 12, 38]. Figure 2.1 shows the concept of applying VP to a test item of

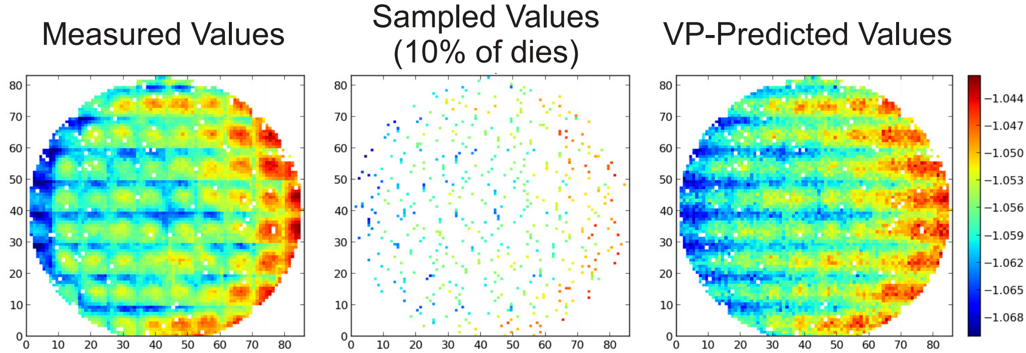


Figure 2.1: Measured values, sampled values, and VP-predicted values of a test item from an industrial product. The values are color-coded to produce the wafer maps.

an industrial product. In this example, the spatial model constructed from 10% randomly sampled dies accurately predict the test values of the remaining 90% dies on the same wafer.

The mathematical background of VP is briefly introduced as the following. Let $\{g(x, y);\ x = 1, 2, \ldots, P,\ y = 1, 2, \ldots, Q\}$ be a performance metric of the die at coordinate (x, y) on a size of $P \times Q$ wafer. The spatial variations of $g(x, y)$ can be represented by a two-dimensional linear transform in the frequency domain. In VP, the discrete cosine transform (DCT) is chosen for the transform. Let $\{G(u, v);\ u = 1, 2, \ldots, P,\quad v = 1, 2, \ldots, Q\}$ be the DCT coefficients after the transform, i.e., the coefficients of different frequencies in the spatial pattern.

The purpose of VP is to accurately recover $g(x, y)$ from a small number, M, of dies at the locations $\{(x_m, y_m);\ m = 1, 2, \ldots, M\}$, where $M \ll PQ$. Toward this goal, the linear equation is formulated:

$$A\boldsymbol{\eta} = \boldsymbol{b}\,, \tag{2.14}$$

where

$$A = \begin{bmatrix} A_{1,1,1} & A_{1,1,2} & \cdots & A_{1,P,Q} \\ A_{2,1,1} & A_{2,1,2} & \cdots & A_{2,P,Q} \\ \vdots & \vdots & \ddots & \vdots \\ A_{M,1,1} & A_{M,1,2} & \cdots & A_{M,P,Q} \end{bmatrix}, \tag{2.15}$$

$$A_{m,u,v} = \alpha_u \cdot \beta_v \cdot \cos\frac{\pi(2x_m - 1)(u-1)}{2P} \cdot \cos\frac{\pi(2y_m - 1)(v-1)}{2Q}, \tag{2.16}$$

$$\alpha_u = \begin{cases} \sqrt{1/P} & (u = 1) \\ \sqrt{2/P} & (2 \leq u \leq P), \end{cases} \tag{2.17}$$

$$\beta_v = \begin{cases} \sqrt{1/Q} & (v = 1) \\ \sqrt{2/Q} & (2 \leq v \leq Q), \end{cases} \tag{2.18}$$

$$\boldsymbol{\eta} = [G(1,1) \ \cdots \ G(P,Q)]^T, \tag{2.19}$$

$$\boldsymbol{b} = [g(x_1, y_1) \ \cdots \ g(x_M, y_M)]^T. \tag{2.20}$$

Once $\boldsymbol{\eta}$ is determined by solving Equation (2.14), the metric values $g(x, y)$ can be recovered by the inverse discrete cosine transform (IDCT).

It is, however, not trivial to solve Equation (2.14). Since $M \ll PQ$, i.e., the number of equations is significantly less than the number of unknowns, (2.14) is profoundly underdetermined. The solution of η is therefore not unique and additional constraints are required. To obtain a unique solution of $\boldsymbol{\eta}$, VP assumes $\boldsymbol{\eta}$ to be sparse [9]. That is, most of the DCT coefficients are close to zero, though the locations of the zeros are unknown. Maximum posterior estimation (MAP) is

used to statistically solve (2.14) by reformulating it to

$$\begin{aligned} \underset{\eta}{\text{minimize}} \quad & \|\boldsymbol{\eta}\|_1 \\ \text{subject to} \quad & A\boldsymbol{\eta} = \boldsymbol{b}\,, \end{aligned} \tag{2.21}$$

where $\|\boldsymbol{\eta}\|_1$ stands for the L_1-norm of $\boldsymbol{\eta}$. Equation (2.21) can be solved efficiently with linear programming [9].

The generated sparse solution finds the sparsest set of coefficients in the frequency domain that accurately picture the spatial pattern of the sampled dies. The sampled dies, however, are only a very small portion of all the dies on a wafer. Therefore the spatial pattern reconstructed from the sampled dies may not be sufficient if the measurement data exhibit a more random distribution. In other words, if the assumption of sparsity is not valid for a certain test item, finding the sparse solution is not sufficient to recover the spatial pattern of the test item. In [38], a test item was categorized as highly-predictable, predictable, and unpredictable in a pre-test analysis phase based on the number of samples required by VP for the test item to reconstruct the spatial pattern within a certain error bound.

To improve prediction accuracy, the random sampling scheme in VP was modified to iteratively sample the optimal location in Bayesian virtual probe [11]. The correlations among different wafers within the same lot were utilized to further reduce the number of sampled dies on each wafer without compromising the prediction accuracy in multi-wafer virtual probe [12].

2.3.2 Joint Virtual Probe

Consider K items to be tested for dies on a wafer and for each test item, only a subset of sampled dies are measured. We denote the measurement vector and DCT coefficient vector of the kth item as $\boldsymbol{b}^{(k)}$ and $\boldsymbol{\eta}^{(k)}$, respectively. If the locations of sampled dies on the wafer are the same for all K items, their transformation matrices, A's, will be identical. Thus the linear system in Equation (2.14) for these K items can be re-expressed as

$$AH = B\,, \tag{2.22}$$

where $H = [\boldsymbol{\eta}^{(1)}\ \boldsymbol{\eta}^{(2)}\ \cdots\ \boldsymbol{\eta}^{(K)}]$ and $B = [\boldsymbol{b}^{(1)}\ \boldsymbol{b}^{(2)}\ \cdots\ \boldsymbol{b}^{(K)}]$.

In the following, we first show the inter-test-item correlations based on some statistical results derived from industrial products. Next, we utilize such observed correlations for unique and joint estimation of the DCT coefficients for multiple test items. Finally, we discuss the runtime characteristics of JVP.

Inter-Test-Item Correlation

Different test items could be affected by similar process parameters and thus have correlations in their spatial patterns. Although the exact relationships between test measurements and the underlying hidden parameters are unknown, statistical analysis results on several industrial products confirm the existence of significant correlations among test items.

When applying principal component analysis (PCA) on the test data of several industrial products, we observed that the eigenvalues of the covariance matrix

decay fast, i.e. most eigenvalues are close to zero, which is a strong indication that a large set of test items are mainly determined by a much smaller set of hidden parameters.

The absolute linear correlation coefficients, which can be easily calculated from the measurements, can also evaluate the test items' pairwise similarity in the spatial domain:

$$|r_{k,l}| = \frac{\left|\sum_m \left(b_m^{(k)} - \bar{b}^{(k)}\right)\left(b_m^{(l)} - \bar{b}^{(l)}\right)\right|}{\sqrt{\sum_m \left(b_m^{(k)} - \bar{b}^{(k)}\right)^2}\sqrt{\sum_m \left(b_m^{(l)} - \bar{b}^{(l)}\right)^2}}, \tag{2.23}$$

where $|r_{k,l}|$ denotes the similarity between the kth and the lth test items while $b_m^{(k)}$ and $\bar{b}^{(k)}$ denote the measured value of the mth sampled die and the average value of all sampled dies, respectively, for the kth test item.

For an exemplar industrial product consisting of 277 parametric test items, we selected four groups of test items, with different distributions and averages of pairwise correlation coefficients within the respective group, for further analysis of their DCT coefficient vectors $\boldsymbol{\eta}$'s produced by VP based on complete measurements from all dies on the wafer. Each group has 53 items (and thus 53 DCT coefficient vectors).

Figure 2.2 shows the statistics of the DCT coefficients of these four groups, in other words, how many near-zero DCT coefficients share the same rows in H. When the correlation among test items is weak, i.e., the case shown in the top left figure, though each DCT coefficient vector $\boldsymbol{\eta}$ is sparse (i.e., a vast majority of the coefficients in each vector are near-zero), the locations of near-zero coefficients in these vectors are somewhat random. If there exist strong correlations among test

items, they tend to have their near-zero DCT coefficients appeared in common locations (i.e., common rows). For example, in the bottom right figure, 1039 coefficients are near-zero for more than 40 of the 53 vectors each of which consists of 2646 coefficients. When a coefficient in a DCT coefficient vector is near-zero, it means that the corresponding frequency does not exist in this test item's spatial pattern. If two sparse coefficient vectors have high similarity in locations of their near-zero coefficients, it implies that these two items' spatial patterns miss similar frequencies, indicating the similarity in their spatial patterns. The greater similarity in the locations of near-zero elements among the sparse column vectors in H, the greater similarity among the spatial patterns of the corresponding test items.

The results shown in Figure 2.2 confirms that for a highly correlated group of test items, the sparsity profiles of their DCT coefficient vectors should be quite similar, indicating the potential of developing a joint sparse model. Note that although correlated test items show similarity in the locations of near-zero DCT coefficients, their non-zero DCT coefficients at specific locations (i.e., the weights for frequencies that exist in the test items' spatial patterns) might be very different. That is, their spatial patterns could still be distinct.

Mathematical Formulation

Determining H in Equation (2.22) is a linear inverse problem which solves for multiple DCT coefficient vectors simultaneously. Similar to Equation (2.14), Equation (2.22) is also underdetermined. In order to find a unique estimation of H, we use a two-dimensional mixed norm, $J^{(p,q)}(H)$, as the optimization objective [66,

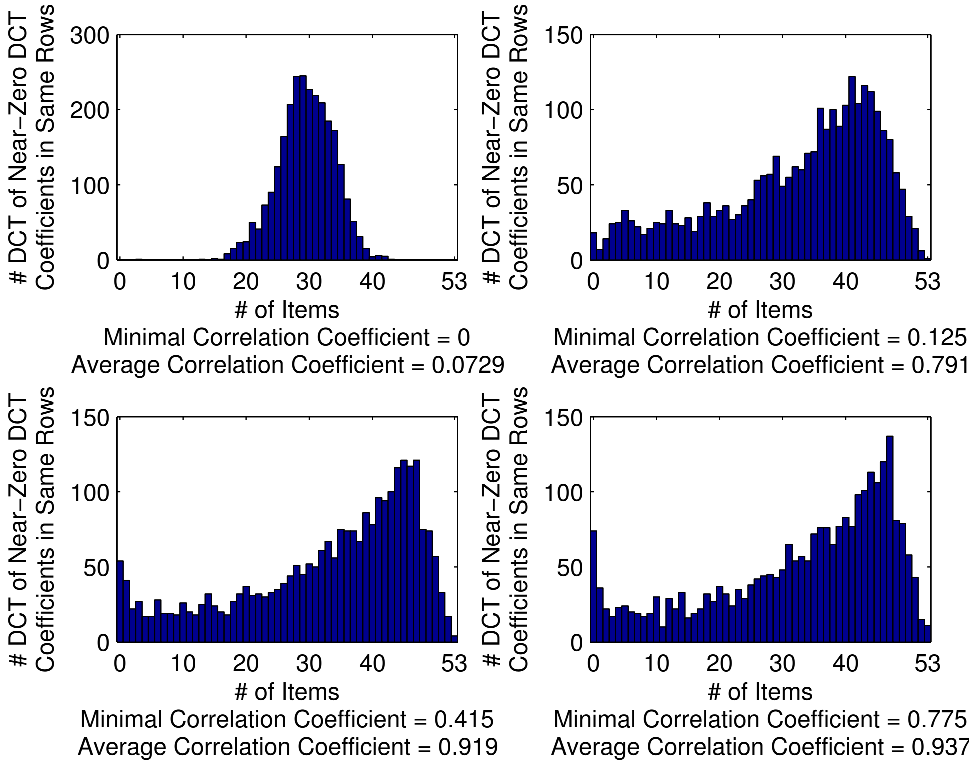


Figure 2.2: Statistics of common, near-zero DCT coefficients among four groups of test items with different degrees of inter-test-item correlations.

64, 67, 68]:

$$J^{(p,q)}(H) = \sum_{i=2}^{PQ} \left(\|\boldsymbol{\eta}_i\|_q\right)^p = \sum_{i=1}^{PQ} \left(\sum_{k=1}^{K} \left|\eta_i^{(k)}\right|^q\right)^{p/q} , \tag{2.24}$$

where $\boldsymbol{\eta}_i = \left[\eta_i^{(1)}\ \eta_i^{(2)}\ \cdots\ \eta_i^{(K)}\right]$ is the ith row of H, and p and q are user-defined parameters. It first calculates the l_q-norm of each row and then calculates the l_p-norm (without pth root) of the result vector of the row norms.

If we only assume that each column of H is sparse, as of VP, the optimization objective of (2.21) is equivalent to $J^{(1,1)}(H)$, which would result in a significantly sparse solution [67]. However, this solution, which only utilizes the spatial correlations, can be improved by utilizing inter-test-item correlations.

With the insight that correlations among a group of test items imply a similar sparsity profile among corresponding columns of H, we choose $p = 1, q = 2$, which effectively enforces both the column sparsity (i.e., considering spatial correlation) and the row similarity (i.e., considering inter-test-item correlation) [66, 64]. The estimation of H can therefore be expressed as:

$$\begin{aligned} \arg\min_{\boldsymbol{\eta}} \quad & J^{(1,2)}(H) = \sum_{i=1}^{PQ} \left(\sum_{k=1}^{K} \left|\eta_i^{(k)}\right|^2\right)^{1/2} \\ \text{subject to} \quad & AH = B\,. \end{aligned} \tag{2.25}$$

In the objective function, coefficients in each row of H are combined into an l_2-norm. By forcing a sparse distribution of these row norms, the solution tends to have more near-zero rows, which meets the desired characteristics of having a joint sparsity profile of H. However, since the formulation imposes little constraints on distributions of elements in the non-zero rows, the joint estimation can still produce unique sparsity profiles for different items.

JVP assumes that the involved items have some similarity in their sparsity profiles. If significant inter-test-item correlations don't exist among all test items, a preprocessing method might be needed to partition test items into groups, each of which JVP is applied to. On the other side, if the correlations are sufficiently strong for joint estimation, further partitioning could possibly reduce the accuracy, since a group with more items could potentially achieve better prediction accuracy [69].

We use MMV FOCal Underdetermined System Solver (M-FOCUSS) [64] to solve the optimization problem (2.25). M-FOCUSS is a gradient-based iterative algorithm, in which the tth iteration performs the following calculations based on H_{t-1}, the estimation of DCT coefficient matrix after $t-1$ iterations, to estimate H_t:

$$\begin{aligned} W_t &= \mathrm{diag}\left(\|\boldsymbol{\eta}_{i,t-1}\|_2^{1-p/2}\right) , \\ A_t &= AW_t , \\ H_t &= W_t A_t^H \left(A_t A_t^H\right)^{-1} B . \end{aligned} \tag{2.26}$$

The iterative process terminates when:

$$\frac{\|H_{t+1} - H_t\|_F}{\|H_t\|_F} < \delta , \tag{2.27}$$

where $\|\cdot\|_F$ donates Frobenius norm and δ is a user-specified parameter. As $J^{(1,2)}(H)$ is convex, this algorithm guarantees to converge to the globally minimized solution of (2.25).

2.3.3 Runtime of M-FOCUSS for JVP Estimation

The runtime of estimation of DCT coefficient vectors by M-FOCUSS is mainly determined by two factors:

1. The runtime of each iteration, which depends on the problem scale: the number of DCT coefficients PQ, the number of samples per item M, and the number of test item K. The theoretical time complexity per iteration is in the order of $O\left((PQ+M)M^2+(PQ)MK\right)$. For our application, the first part, $O\left((PQ+M)M^2\right)$, which is mainly the complexity for computation of a pseudo-inversion, dominates the runtime of each iteration.

2. The number of iterations, which strongly depends on the termination criterion shown in (2.27). A smaller δ will result in more iterations. It is also influenced by the problem scale. It is observed empirically that, for a fixed δ, fewer iterations are required for a larger M or a larger K.

Figure 2.3 shows the trends of JVP's runtime versus the problem scales, with a fixed sample ratio α while $M=\alpha PQ$. For a fixed α, the theoretical complexity can be simplified and expressed as $O\left((PQ)^3+(PQ)^2K\right)$. As long as PQ is relatively large (which is the case for our application), the runtime mainly depends on PQ (in our experiment, the growth with respect to PQ is closer to quadratic than cubic, primarily due to the implementation of the solver in MATLAB), while the linear runtime growth with item count K is relatively negligible.

Figure 2.4a and 2.4b shows the runtime trends of JVP and VP with respect to the number of test items, for three different sample sizes taken from the same production wafer with 625 dies. We use the same underlying solver, M-FOCUSS [64],

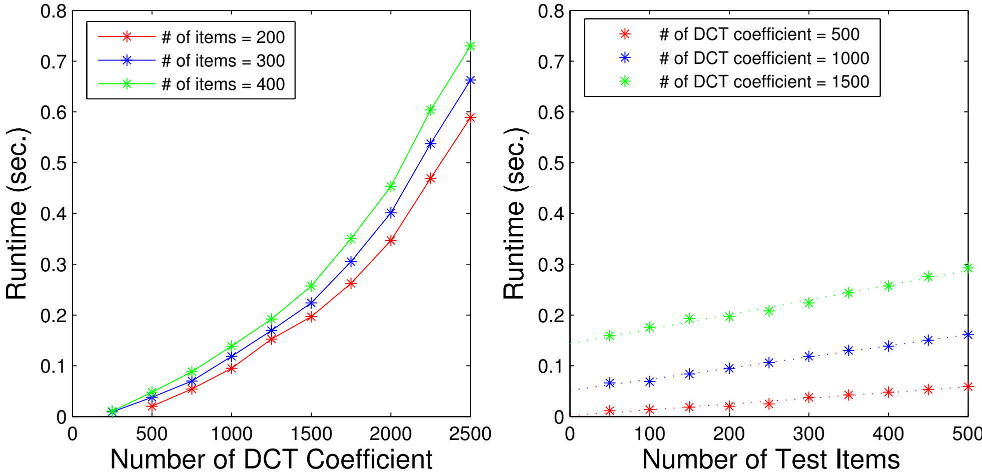
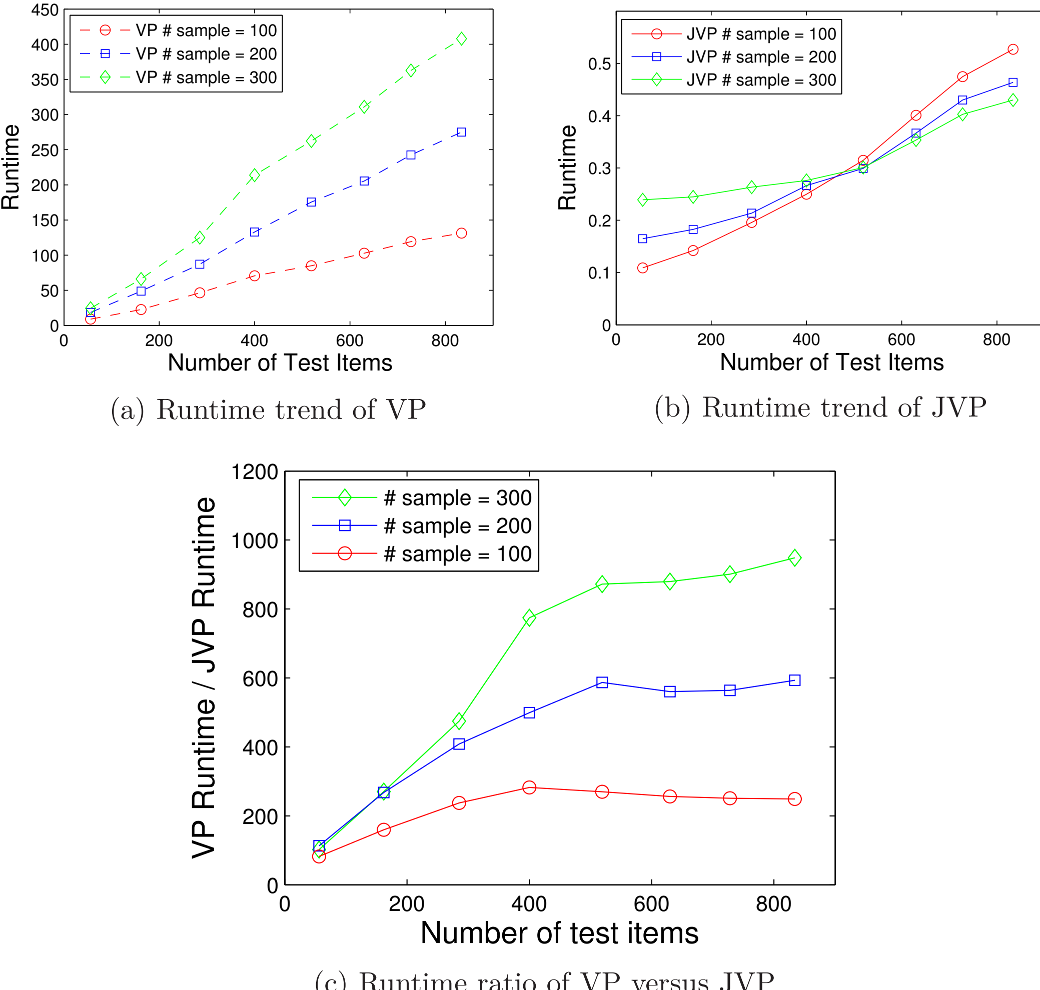


Figure 2.3: Runtime trends of JVP versus the problem scale.

for both VP and JVP. VP's runtime grows linearly with K with a non-trivial slope, because VP processes one item at a time. In contrast, JVP's runtime growth, while is also linear with K, has a significantly smaller slope. Note that for $K > \sim 500$, JVP's runtime reduces when the sample size increases from 100 to 300. This runtime trend is primarily due to faster convergence (i.e., fewer iterations required) when the sample size increases. The runtime ratio of VP versus JVP (i.e., the speedup achieved by JVP) shown in Figure 2.4c illustrates that JVP significantly outperforms VP, especially for larger K. For example, with a sample size of 100, JVP runs 256 times faster than VP for processing 630 test items. When the sample size increases to 300, the speedup achieved by JVP increases to 880X for processing 630 items. This is mainly due to the fact that, at a larger sample size, JVP incurs fewer iterations to converge than VP does.



(a) Runtime trend of VP

(b) Runtime trend of JVP

(c) Runtime ratio of VP versus JVP

Figure 2.4: Comparison of JVP's and VP's runtimes versus the number of test items K, for three sample sizes made from the same production wafer.

2.3.4 Experimental Results

Production test data from two different products were thoroughly analyzed for validating the proposed JVP technique. The production test data of these two products were first pre-processed to remove confidential information but information critical to this evaluation was maintained. Datasets 1 and 2 contains 277 and 985 parametric test items, respectively, and with 1043 and 625 dies per wafer respectively. For each dataset, we sampled 20% of the dies on a wafer for running VP and JVP. For a fair comparison, both VP and JVP use the same underlying solver M-FOCUSS [64]. For VP, this solver runs faster than the one used in [9, 38]. All experiments were conducted using MATLAB R2012b on an Intel Xeon Quad-core 3.60 GHz system.

Determining Predictability of a Test Item

VP and JVP tend to find a sparse representation for each test item, regardless of the validity of its sparsity assumption. It is thus necessary to evaluate test items' predictability in the pre-test analysis phase to determine if the test item can indeed be predicted with sufficient accuracy. Then in the test application phase, only those items classified as predictable are estimated.

Normalized error e_n, which is unbiased with respect to the data's mean and the degree and distribution of its deviation from the mean is defined as

$$\begin{aligned} e_n &= \mathrm{rms}\left(\left|\frac{\boldsymbol{P} - \mathrm{mean}(\boldsymbol{M})}{\mathrm{std}(\boldsymbol{M})} - \frac{\boldsymbol{M} - \mathrm{mean}(\boldsymbol{M})}{\mathrm{std}(\boldsymbol{M})}\right|\right) \\ &= \mathrm{rms}\left(\left|\frac{\boldsymbol{P} - \boldsymbol{M}}{\mathrm{std}(\boldsymbol{M})}\right|\right), \end{aligned} \tag{2.28}$$

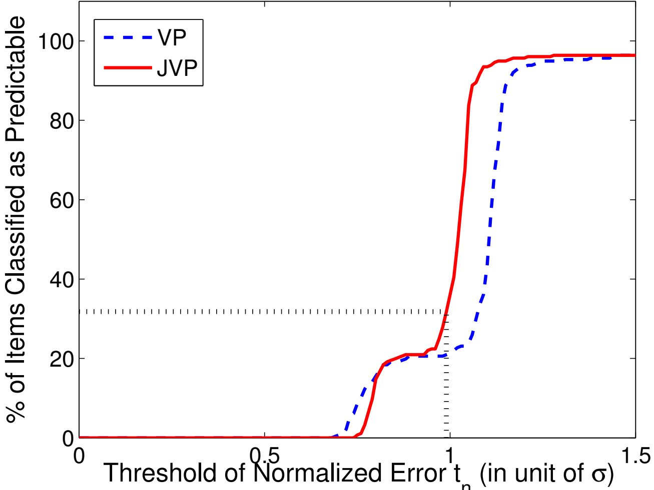
where $\boldsymbol{P}$ and $\boldsymbol{M}$ denotes the vectors of the predicted and the measured values of all dies for a test item, respectively. std$(\cdot)$ denotes the standard deviation, and rms$(\cdot)$ denotes the root mean square. Note that e_n is not an error percentage and its unit is one standard deviation (σ) of the test item's values of all dies on a wafer.

We use the normalized error e_n to classify a test item's predictability, because it reflects the accuracy of the captured spatial pattern without bias, as well as implies the validation of assumption of sparsity. If this error for the test data of the training wafer is lower than a given threshold t_n, the test item is classified as predictable, and, otherwise, it is unpredictable. Figure 2.5a shows the numbers of items classified as predictable versus t_n for both VP and JVP for Dataset 1. JVP produce slightly better results than VP.

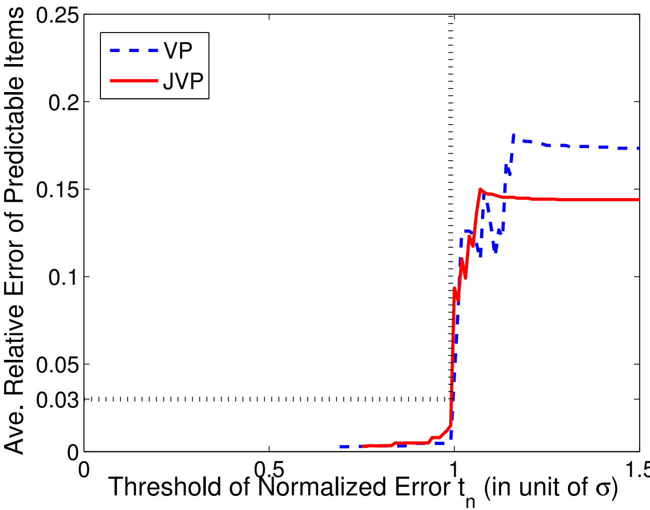
However, setting a proper threshold t_n, in unit of σ, is non-intuitive for a user. We therefore introduce a second metric, the average relative error e_r of a test item, which is the average of the prediction error normalized with respect to the measured values of the test item among all dies on the training wafer:

$$e_r = \text{mean}\left(\left|\frac{\boldsymbol{P} - \boldsymbol{M}}{\boldsymbol{M}}\right|\right) . \tag{2.29}$$

It would be more intuitive for a user to set a threshold on e_r, instead of e_n, to explore the trade-off between the average e_r among predictable items and the percentage of items classified as predictable. Figure 2.5b and 2.5a, which illustrate t_n versus percentage of items classified as predictable and t_n versus the average e_r's among predictable items, indicate how to explore the trade-offs. For example,



(a) Percentage of items classified as predictable versus the threshold of the normalized error t_n.



(b) Average of the average relative error e_r's among all predictable items versus the threshold of the normalized error t_n.

Figure 2.5: Setting thresholds for classifying test items.

once a threshold of e_r is set (e.g. 3%), the corresponding t_n can then be found in Figure 2.5b. In turn, the corresponding percentage of items classified as predictable can be found in Figure 2.5a. In this illustration, t_n is determined based on JVP's results (i.e., red curves in both figures).

Pre-Test Analysis Result

The goal of pre-test analysis is to identify test items which can be accurately predicted with a small subset of samples. To achieve this, the complete test data of a training wafer are analyzed. We repeatedly selected samples to run VP and JVP, and then calculated the prediction error for all test items.

Table 2.1 compares the results of VP and JVP. In comparison with VP, JVP shows significantly faster runtime (the last two rows). The second row shows the percentage of items classified as predictable by each method. As different methods produce different sets of predictable items , we then identify the intersection of predictable items $\boldsymbol{I}$, which contains those test items classified as predictable by both methods (the third row). The fourth row shows the average e_r for different methods based on $\boldsymbol{I}$ only. Note that JVP achieves a bit worse predictable accuracy than VP, because the exact prediction accuracy is not the main concern in pre-test analysis and all test items are processed by single run of JVP to fast identify predictable items.

Validation and Test Application

The above comparison of JVP and VP is for the training, pre-test analysis phase. The items classified as predictable in the pre-test analysis needs further

Table 2.1: Comparison of VP and JVP

	Dataset 1		Dataset 2	
Method	VP	JVP	VP	JVP
Test items classified as predictable	20.9%	31.8%	14.7%	29.1%
Test items predictable by both methods ($\boldsymbol{I}$)	20.9%		14.7%	
Average e_r of $\boldsymbol{I}$	0.49	0.49	0.96	0.97
Runtime (sec.)	186.64	0.92	72.32	0.50
Runtime improvement	–	202X	–	145X

Table 2.2: Percentage of Test Items Classified as Predictable

	Pre-Validation	**Post-Validation**
Dataset1	31.8%	26.7%
Dataset2	29.1%	27.1%

validation using the complete test data of another wafer. An item whose e_r exceeds a target threshold, 3% in our experiment, should be considered as unpredictable and thus removed from the final list of predictable items. Table 2.2 shows the percentage of predictable items, based on JVP, before and after this validation phase.

In the test application phase, only those validated predictable test items are analyzed and used for test prediction. Both prediction accuracy and runtime should be considered in the test application phase. The comparison of VP and JVP in the test application phase is shown in Table 2.3. JVP is 57X and 43X faster than VP for Datasets 1 and 2, respectively. And the average error among those predictable items is even reduced.

Table 2.3: Comparison in Test Application Phase

	Method	Ave. e_r	Runtime (sec.)	Improvement
Dataset 1	VP	0.58%	36.27	–
	JVP	0.54%	0.64	57X
Dataset 2	VP	0.77%	20.60	–
	JVP	0.70%	0.48	43X

2.4 Summary

In this chapter, we propose a weighted group lasso technique for building an inter-test-item correlation model among all test items based on a given test program. Learning from the manufacturing test data of training chips, WGL identifies test items which can be eliminated from measurement without compromising test quality.

In addition, we proposed a joint virtual probe technique which captures the spatial patterns in the test data for multiple test items jointly. JVP is formulated as a convex optimization problem for an under-determined linear inverse problem which can be solved by existing algorithms, such as the M-FOCUSS algorithm, and achieves a very significant speedup in comparison with the original VP. JVP benefits from the correlated data of other test items when estimating a test item's spatial pattern and thus could achieve better accuracy than VP as well.

Chapter 3

Test Time Reduction

3.1 Introduction

It is well known that, for some test items, there exist spatial correlations among dies on the same wafer. There also exist correlations among multiple measurements taken from the same chip (i.e., inter-test-item correlations). For test time reduction, it is preferable to predict more costly test items if such options exist. In this chapter, we propose a TTR methodology that integrate both VP* and WGL techniques to enable utilization of both spatial and inter-test-item correlations. We can run VP first to identify items that can be predicted without measurement (referred to as VP-predictable items) based on spatial correlations. It is then followed by running WGL for which those VP-predictable items are assigned a small weight and the other test items (i.e., spatially unpredictable items) are assigned a large weight. With such assignments, WGL, which identifies addi-

*We can use either JVP or VP for spatial pattern prediction. For simplicity and consistency, in the rest of this chapter we refer such methods as VP.

tional predictable items based on inter-test-item correlations, will find predicable items mainly from the pool of spatially unpredictable test items, thus maximizing the union of the predicable items derived from the spatial and inter-test-item correlations.

The proposed methodology offers the flexibility of exploring the trade-off between the number of removed test items and the prediction accuracy. We propose to use two predictability criteria, the bound of relative prediction error and the margin from the specification limits, to control the training process. We conducted experiments on a high volume industrial device and identified 47% of the test items to be candidates for sampling or elimination from the test list, out of 338 parametric test items, with a potential test time savings of 55% given our test time assumptions.

The rest of this chapter is organized as the following. Section 3.2 illustrates how to integrate WGL with the VP method. Section 3.3 discusses the criteria for classifying test items as predictable or not and the flow of our proposed methodology. Section 3.4 provides experimental results, and we conclude this chapter in Section 3.5.

3.2 Optimization for Test Time Reduction

In Chapter 2, we described two TTR methods, VP and WGL, which target spatial and inter-test-item correlations, respectively. In this section, we aim at addressing the following issues:

1. Taking into account the distinct time of each individual test item for overall

test time reduction. Different test items incur different test times. In finding predictable test items by TTR methods, it is preferred that the more costly test items are predicted because they contribute more to the total test time. Reflecting different test times of test items requires a scheme to assign different significance to differentiate test items. This problem also includes how to map the practical test times to reasonable parameters so the generated result is most improved in terms of TTR.

2. Considering both spatial and inter-test-item correlations in test data for overall test time reduction. As spatial patterns and inter-test-item correlations are two independent approaches, many TTR methods have been proposed targeting either of the two. It is natural to ask if there is a way to utilize both correlations simultaneously and expand the dimensions of test data analytics. For instance, having different sets of predictable test items from VP and WGL, we want to maximize the union of the two sets so that we find the largest number of predictable test items. If a test item has been identified as predictable in one method, the other method should tend to predict the other test items instead of the one already predicted by the first method.

In the following we discuss issues of assigning an appropriate weight for a test item in the WGL problem.

3.2.1 Reflecting Item's Test Time

Directly using the test time of the ith item as the weight w_i in (2.12) or (2.13) might lead to impractical solutions. For example, having groups with a very large weight in the constraint in (2.12) might cause undesired dominance of the constraint (which reflects the sparsity of the correlation matrix) over the cost function for minimization (which reflects the prediction error) as the optimization target. As a result, the solution might have unacceptably high prediction error or find very few predictable test items.

In our methodology, we use the normalized test times as the weights in the WGL definition. For instance, assuming that we have n test items and their test times are $t_1, t_2, \ldots, t_n$, respectively, we normalize all t's so that their normalized values are in the range of 0.5 to 1.5 and the mean is equal to 1. These normalized t's are then used as the weights, w, in (2.12) and (2.13).

3.2.2 Weight Assignment for TTR Utilizing Both Spatial and Inter-Test-Item Correlations

A straightforward TTR strategy to utilize both spatial and inter-test-item correlations is to run both VP and GL on all test items independently and the aggregate their results for TTR. Assume T_{VP} and T_{GL} are the sets of predictable test items identified by VP and GL, respectively. Then their union $T_I = T_{VP} \cup T_{GL}$ would be the set of total predictable test items that can be removed from measurement. However, such a strategy often produces sub-optimal results. It is desirable to minimize the overlap (i.e. intersection) of T_{VP} and T_{GL} and maximize

their union, which in turn maximizes the test time reduction. This optimized strategy can be implemented by running VP first, followed by running WGL. After identifying T_{VP}, we set a lower weight, w_l, for every test item in T_{VP} and a higher weight, w_h for test items not in T_{VP} before running WGL. Because WGL tends to minimize the α's for groups with a higher weight, the resulted predictable test items by WGL, T_{WGL}, would have minimum overlap with T_{VP}. As a result, the final set of predictable test items, $T_J = T_{VP} \cup T_{WGL}$, under this strategy would most likely be larger than the set, T_I, produced by the straightforward strategy.

3.3 Test Methodology Based on GL and WGL

In this section, we describe in detail the application of the inter-test-item correlation model of GL/WGL. The first issue to be addressed is to evaluate if the prediction accuracy of a candidate test item is sufficiently high and if the item can indeed be safely eliminated from the test program without compromising the test quality. Those candidate test items meeting a desired level of prediction accuracy (i.e., predictability) are referred to as *predictable test items* in the following.

We then discuss some practical issues of applying the proposed methods in production. Specifically, we discuss the issues of handling random defects, normalized prediction error, and the need of continuous cross-validation to monitor if the manufacturing process is sufficiently stationary.

Then we describe the two-stage test methodology of GL or WGL in detail: the pre-test analysis for learning the inter-test-item correlation model and the test application stage which utilizes the learned model for TTR.

3.3.1 Criteria for Classifying Predictability

Even if the correlations among the test items are weak, GL or WGL will still produce a correlation model. However, the prediction accuracy based on the model might not be accurate.

According to Section 2.2.2, a useful correlation model for TTR has one or multiple near-zero columns. The values of the corresponding candidate test items can be predicted by a combination of other test items. Since we use a penalty parameter, λ, in the minimization problem (2.11) to control the sparse level of α, each $\boldsymbol{u}$ of the solution found may not be minimal. That is, there is no guarantee on the prediction accuracy for the candidate test items derived from the correlation model. Hence we need to apply one more filtering step to the set of candidate test items: only a subset of them that meet some criteria and achieve a desired level of prediction accuracy will be selected as *predictable* test items.

We evaluate the predictability of a test item based on two criteria:

1. The maximum relative prediction error among a total of d chips in the training set:

$$e = \max(\{|(\hat{g}_i - g_i)/g_i| : i = 1, \cdots, d\}) , \tag{3.1}$$

   where $\hat{g}_i$ and g_i denote the predicted and the measured values of chip i in the training set.

2. The margin between specification limits of a test item and the range covering *most* of the training chips' predicted values. If we denote the 25%, 50%, and 75% points of the cumulative distribution function (CDF) of the predicted values of all training chips as Q_1, Q_2, and Q_3 respectively, the IQR, defined

Table 3.1: The Predictability Levels

Predictability level	Relative error	Margin from spec limits, percentage of $(L_h - L_l)$
High	$0\% \sim 5\%$	$> 35\%$
Medium	$5\% \sim 25\%$	$15\% \sim 35\%$
Low	$25\% \sim 100\%$	$< 15\%$

as the range of the middle fifty, would be equal to $Q_3 - Q_1$. The interquartile range method defines the range X from $Q_1 - 1.5IQR$ to $Q_3 + 1.5IQR$ as the range covering most of the data points for an arbitrary distribution (conceptually similar to the 3σ range for a normal distribution).

If we denote L_l and L_h as the low and high limits of a test item's specification range respectively and M as the desired margin between the specification limits and the range X defined above (i.e., $[Q_1 - 1.5IQR, Q_3 + 1.5IQR]$), the following is the second criterion used for classifying a test item as predictable:

$$L_l + M < X(\hat{g}_i, i = 1, \cdots, d) < L_h - M \,. \tag{3.2}$$

We define three levels of predictability, *high*, *medium*, and *low*, for each of the two criteria in Table 3.1. Based on the predictability levels, we classify each test item as either predictable or unpredictable, as shown in Figure 3.1. That is, a test item is considered predictable only if it has a high predictability level for at least one criterion and does not have a low predictability level for any criterion.

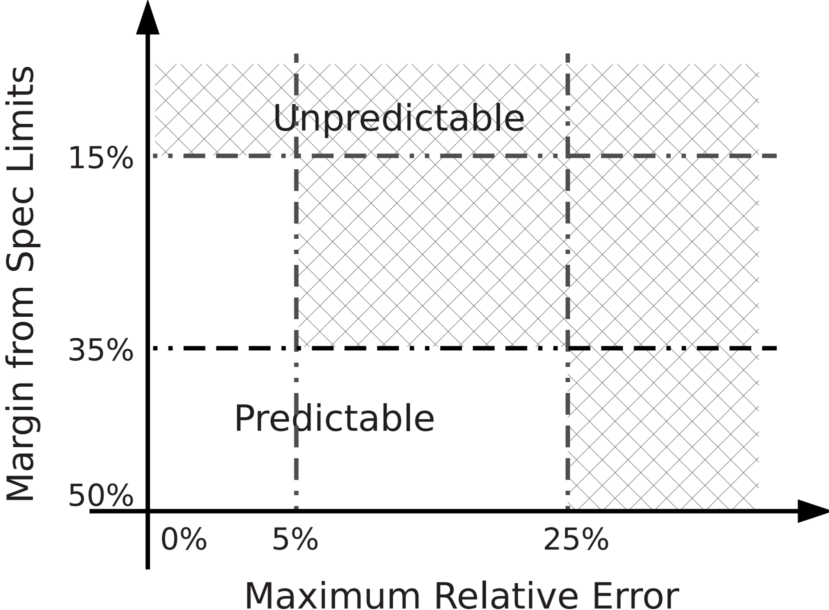


Figure 3.1: We classify each test item into one of two possible categories, predictable or unpredictable, based on joint consideration of two criteria. Predictability is evaluated through relative prediction error and the distribution of the prediction error.

3.3.2 Challenges of Random Defects, Prediction Error, and Process Stationarity

Selecting Test Items Targeting Random Defects

VP and GL are effective only for test items and chips that are affected by process and systematic variations. For chips with random defects, their values of some test items might not follow the correlations captured from the training chips. Therefore, even for predictable items, the values of such defective chips predicted by VP and GL/WGL might be inaccurate.

However, a defective chip with a random defect is usually more catastrophic (than systematic and variation-induced failures) and can often be detected by multiple test items. In addition, it has been observed that chips with random defects can often be detected by a small subset of test items, carefully selected from a test program consisting of a large number of test items. As an example, Table 3.2 shows the number of test items of a high-volume production chip that are required to cover all failed chips in the training set whose test items cannot be accurately predicted by VP and GL (i.e., most likely the suspects with random defects). Out of 338 test items in total, while the number of test items required to cover all random defect suspects increases as more wafers are considered, the test item count required is still relatively small in comparison with the total number of test items.

Based on this observation, we address random defects by selecting additional test items for explicit measurement, among those items classified as predictable based on the criteria discussed in Section 3.3.1. Specifically, in the model valida-

Table 3.2: Test Items Required to Cover All Random Defect Suspects

Number of training wafers	1	2	3	5	10	25
Number of test items to cover all random defect suspects	9	12	18	26	38	57

tion phase, if the number of chips escaping from a predictable test item is greater than a threshold (i.e., the prediction error is abnormally large for too non-trivial number of chips in the training set), we disqualify it as a predictable item and, instead, classify it as unpredictable and thus requiring explicit measurement.

Screening Test Items by Normalized Error

While using the relative prediction error in Equation (3.1) and setting an upper bound on e as one of the criteria for classifying the test items ensures the test quality will not be compromised, the use of the relative prediction error, however, is biased by the test item's mean and variation when evaluating the prediction quality. It is possible that a model in which a test item's distribution is not accurately captured, but still has a small relative prediction error, if the test item's mean is large and its variation is small. For such a case, though the distribution of the test values are not accurately captured, the errors, divided by their large mean, are sufficiently small to pass the error bound e.

To address this problem, the test items that pass the two criteria in Section 3.3.1 are further examined using their normalized prediction error. The normalized prediction error, without the bias of the test item's mean and variation, better reflects the accuracy of capturing the test item's distribution. Specifically, those test items with a normalized prediction error greater than a threshold will

be screened out and excluded from the final set of selected predictable test items.

Stationarity

There exist wafer-to-wafer and lot-to-lot variations. Therefore, in applying the model trained based on the test data of one wafer for testing of another wafer, it is necessary to perform additional validation to assure the correlation patterns of the chip/wafer under test are sufficiently close to the patterns exhibited in the training data.

The validation can be easily done by taking additional measurement for a small number of the chips for the predictable test items. If the statistics of the differences between the predicted values and measurements are significantly greater than those estimated from the training set, explicit measurements for all test items should be made for all dies in the wafer. The complete test data of the wafer will then go through further outlier analysis. If the analysis concludes that the wafer is an outlier, the original model will continue to be used. Otherwise, retraining based on the target wafer's new data is triggered and the retrained model will be used for further testing of other wafers. Through this continuous validation, the methodology can be adapted to address significant wafer-to-wafer and lot-to-lot variations.

3.3.3 Test Procedure

In this subsection, we summarize the procedures of both pre-analysis and test application stages.

Flow of Pre-Test Analysis

The input to the pre-test analysis procedure includes *a*) complete test data of a set of chips as the training set (including die locations and specification limits, measured values, and test time for each individual test item), *b*) criteria for predictability classification (as described in Section 3.3.1), and *c*) the preferred statistical regression methods (VP and/or WGL with a choice of weight assignment as discussed in Section 3.2).

We run VP first, if VP is chosen as a preferred method. Based on the options discussed Section 3.2, we use either the test times or test items' predictability classified by VP to determine the weights before running WGL. Next, we build an inter-test-item correlation model by solving the WGL problem defined in (2.13). After that, we determine the predictable items based on the criteria illustrated in Table 3.1 and Figure 3.1. In addition, extra test items are selected for measurement, based on the discussion in Section 3.3.2, to detect failed chips caused by random defects. Finally, we estimate test time saving, the yield loss, and the escape rate by comparing the predicted values with measured values of training chips. The pre-test analysis procedure is illustrated in Figure 3.2.

Test Application Flow

In the test application stage, we skip predictable test items from measurement. Based on the partial measurement results and the correlation model, the values of those predictable test items are calculated. We can apply this procedure, illustrated in Figure 3.3, to any wafer or any collection of chips for testing. We first perform all tests for those predetermined sample dies/chips in the tar-

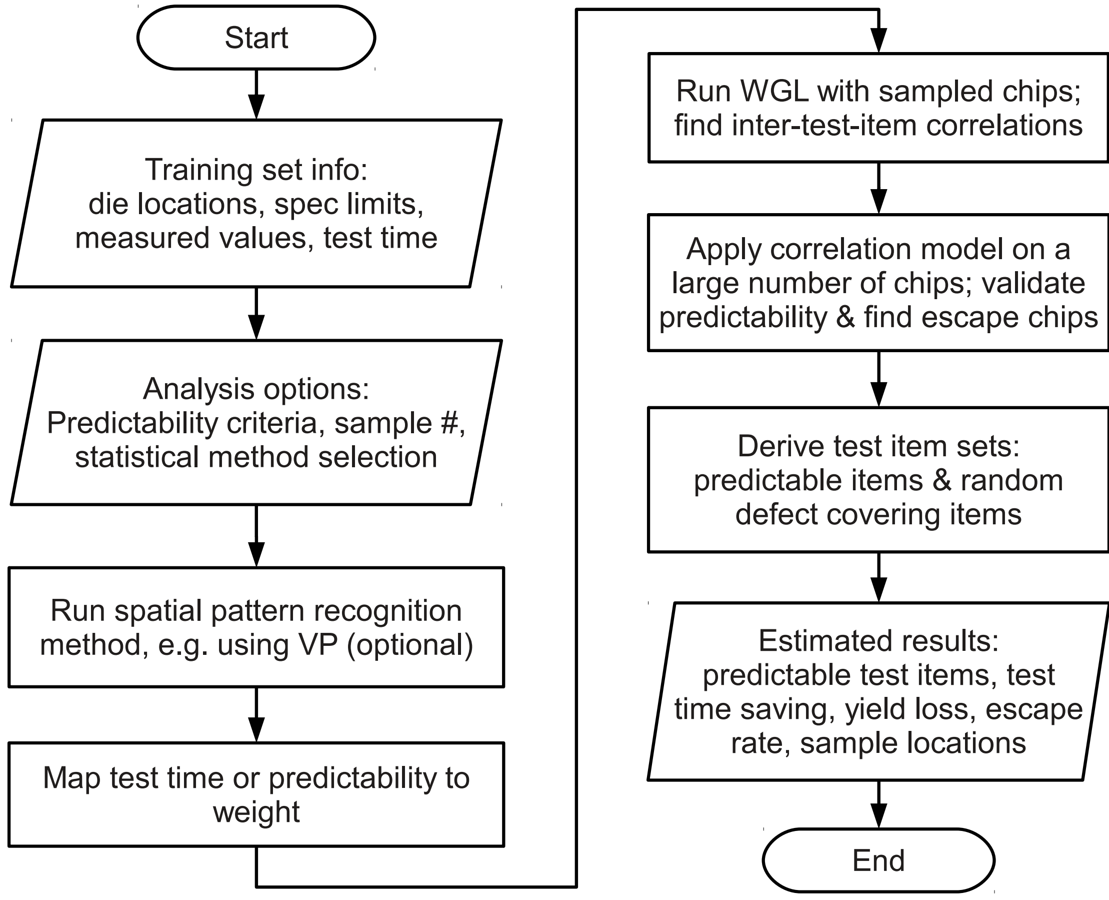


Figure 3.2: The flow of pre-test analysis.

get wafer or collection of chips. The measured values of these samples are used for three purposes: *a*) used by VP to calculate the predicted values of the other dies based on the spatial correlations, *b*) used by GL/WGL to calculate each test item's mean and standard deviation for chips on the target wafer which are needed for converting the normalized predicted values, defined in (2.1), to the predicted values, defined in (2.2), and *c*) used for checking the stationarity for the validity of inter-test-item correlation model on the target wafer (or a target collection of chips) based on the discussion in Section 3.3.2.

Then we perform the following two processes concurrently for the remaining chips that are not those predetermined samples: *a*) testing the unpredictable test items for chips, and *b*) using the statistical methods (VP and/or WGL) to predict the values of those predictable test items. After gathering all predicted values and measured values, we can apply the validation process described in Section 3.3.2. If the wafer (or the collection of chips for testing) passes the validation for stationarity, the test results are considered valid. Otherwise, we cannot trust the predicted values and thus have to test all test items for the measured values of all chips on the wafer for the outlier analysis. Several proposed methods, such as [70, 71], can be used to help identify outlier wafers. If the wafer is an outlier, we do not change the correlation model and continue the above procedure to the next wafer. Otherwise, we rerun the pre-test analysis to build a new correlation model before applying the procedure to the next wafer.

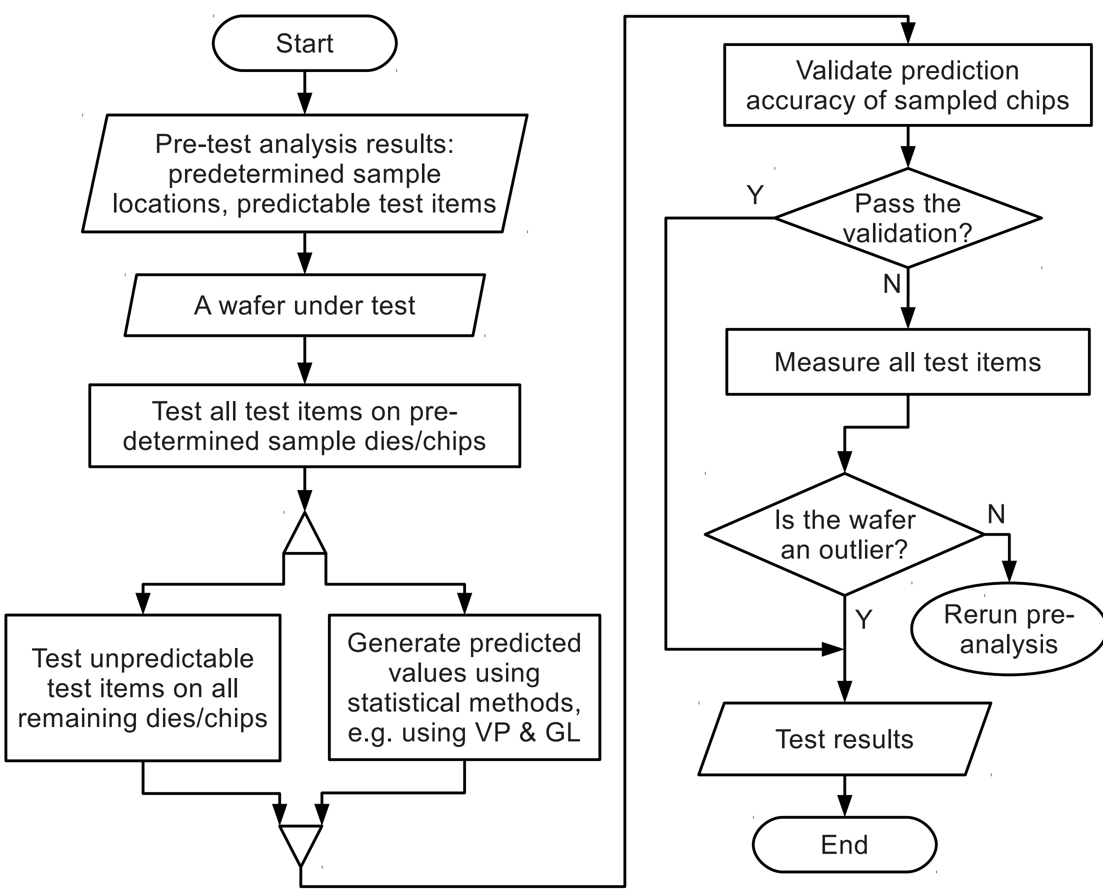


Figure 3.3: The flow of test application.

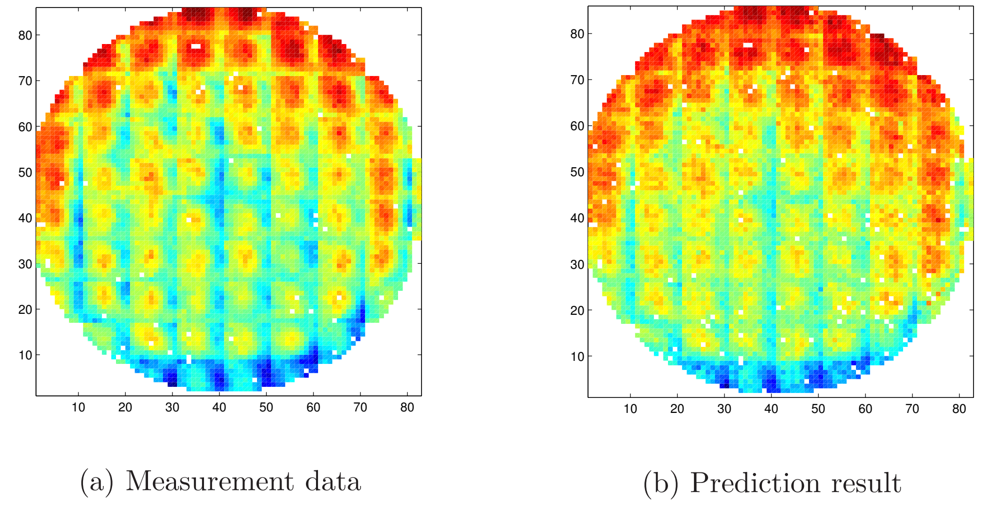
3.4 Experimental Results

We applied GL and WGL to the wafer sort data of a high-volume industrial device. There were 25 wafers per lot and 5500+ dies per wafer. For each lot, 500 randomly sampled dies (by Latin hypercube sampling method) on the first wafer were used for training, and the other 5000+ dies on the same wafer were used for validating the trained model, as discussed in Section 3.3.3. The test program we analyzed consists of hundreds of test items, approximately 70% of which are parametric in nature, and those tests are the ones to which we applied our proposed method.

3.4.1 Inter-Test-Item Correlations Analysis

The correlation matrix, which is the solution to the GL regression problem of (2.10), reveals the correlations among test items, as formulated in (2.2). Coefficient α_{ij} reflects the significance of $\boldsymbol{F}_j$ in predicting $\boldsymbol{F}_i$. A larger α_{ij} means that $\boldsymbol{F}_j$ contributes more in predicting $\boldsymbol{F}_i$; therefore, test item j has a stronger correlation with test item i. One example of the prediction is shown in Fig. 3.4 where the values of the test item are color-coded. The prediction result of the 87th test item shows very high consistency with the measurement data. Figure 3.5 shows the three test items with the largest coefficients α's in predicting test item 87 in Figure 3.4. It can be observed from the wafer maps that test item 35, with the largest α, has the strongest correlation with test item 87.

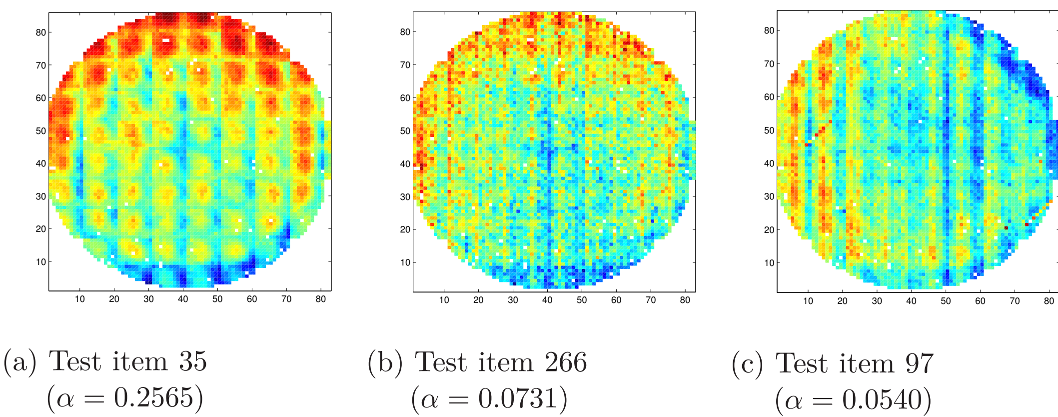
Figure 3.6 shows the number of predictable test items versus the penalty parameter λ for different relative error bounds e, defined in Equation (3.1). With a



(a) Measurement data

(b) Prediction result

Figure 3.4: Die map of measured values and predicted values of test item 87.



(a) Test item 35 ($\alpha = 0.2565$)

(b) Test item 266 ($\alpha = 0.0731$)

(c) Test item 97 ($\alpha = 0.0540$)

Figure 3.5: Test items with the largest coefficient α in predicting test item 87.

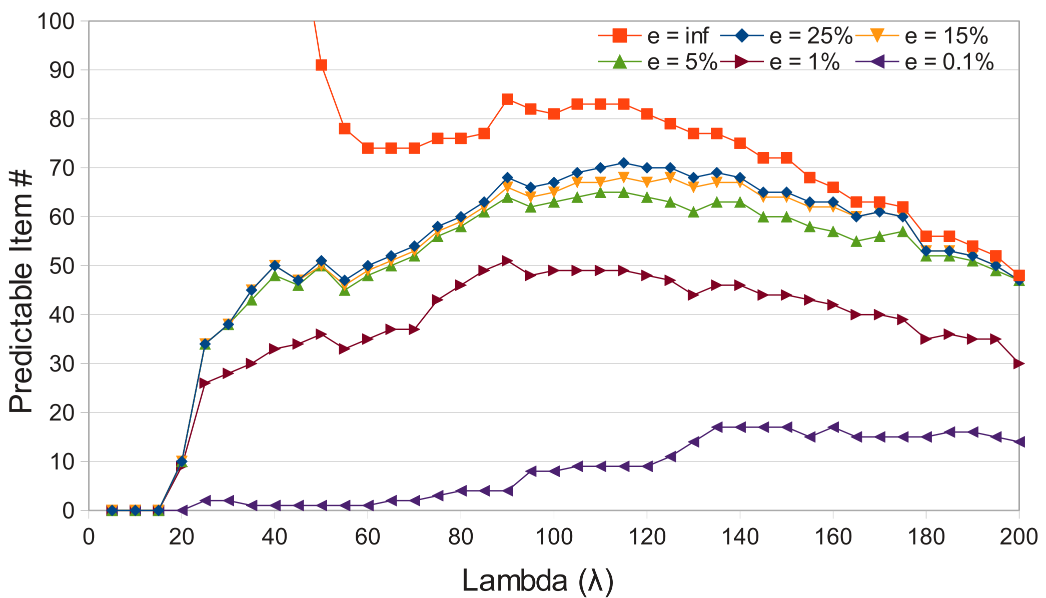


Figure 3.6: Number of predictable test items versus penalty parameter λ for six different error bounds e's.

fixed error bound, different λ values result in different numbers of predictable test items. For a given λ, the larger the error bound, the more the predictable test items. Furthermore, the λ value that maximizes the number of predictable test items varies for different e's. The optimal value of λ depends on the test data, including the number of test items, their measured values, and the number of dies for training.

For e being infinity, which completely ignores the constraint imposed by the bound of prediction error, all candidate test items will be classified as predictable test items. The curve of $e = \text{inf}$, thus showing the number of candidate test items, rises up rapidly when λ becomes smaller than 50. It approaches very close to the total number of test items when $\lambda \to 0$ because sparsity of the correlation matrix becomes the dominant emphasis while accuracy is ignored. As a general

trend, those candidate test items found at a low λ value, however, more likely have a larger prediction error and thus will more likely be screened out by the predictability criteria and considered unpredictable in our methodology. For e at 5%, the maximum number of test items identified as predictable is around 19% among the 338 test items we analyzed.

In addition to identifying predictable test items, the correlation matrix produced by GL also reveals the strength of correlations among test items. Figure 3.7 shows the relations between the predictable test items and the test items used to predict others, i.e., the "predicting" test items. In the scatter graphs, a point at coordinate (j, i) means that test item $\boldsymbol{f}_j$ is in the set of test items that are used for predicting test item $\boldsymbol{f}_i$. The orders of the predictable and predicting test items were rearranged so the figures show clusters of points. Showing points with $\alpha_{ij} \geq 0.01$, Figure 3.7a contains more points, some of which may not represent significant relations. Figure 3.7b, on the other hand, shows only relations with $\alpha_{ij} \geq 0.05$ and reveals only relations that are sufficiently strong.

If we have a cluster of points with x-coordinates in the range of $\boldsymbol{u}$ and y-coordinates in the range of $\boldsymbol{v}$, we can conclude that $\boldsymbol{f_u}$ predict $\boldsymbol{f_v}$, and therefore $\boldsymbol{f_u}$ and $\boldsymbol{f_v}$ have strong correlations. For example, in Figure 3.7b the first 23 reordered predictable test items are predicted by 7 of the first 8 predicting test items, implying that the 23 predictable test items and the 7 predicting test items are highly correlated.

We then applied the model for testing the other 24 wafers. The prediction errors of using the model produced by setting $e = 5\%$, the prediction errors of an exemplar are illustrated by a boxplot in Figure 3.8. The y-axis indicates

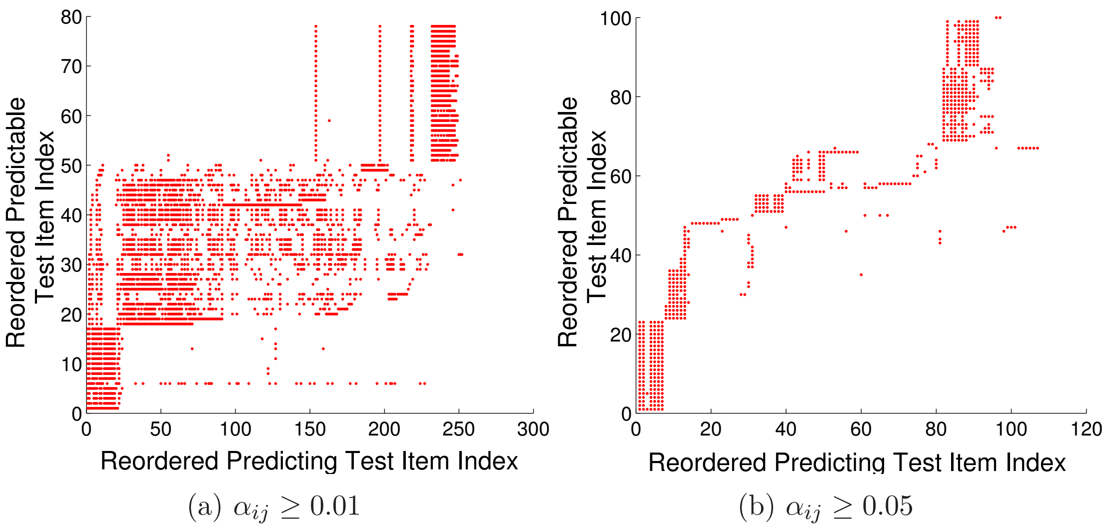


Figure 3.7: The relations between predictable test items and the "predicting" test items.

the relative prediction error of dies corresponding to each predictable test item (the x-axis). The ends of a whisker represent the lowest datum still within the lower quartile minus $1.5IQR$ (where IRQ is the interquartile range, the difference between the upper and lower quartiles), and the highest datum still within $1.5IQR$ plus the upper quartile. Any data not included between the whiskers are plotted as a cross. As illustrated in Figure 3.8, all prediction errors are within the 5% error bound e when applying the model obtained from the training wafer to another.

In our methodology, the margin from the spec limits is the other criteria for classifying test item predictability. Increasing the desired margin will reduce the number of test items classified as predictable as shown in Table 3.3. If the manufacturing process is relatively stable, we can set a smaller margin and the number of test items classified as predictable would be larger. For example, when the margin is set to 5%, 71 test items (i.e., 21% of total items) are classified as predictable.

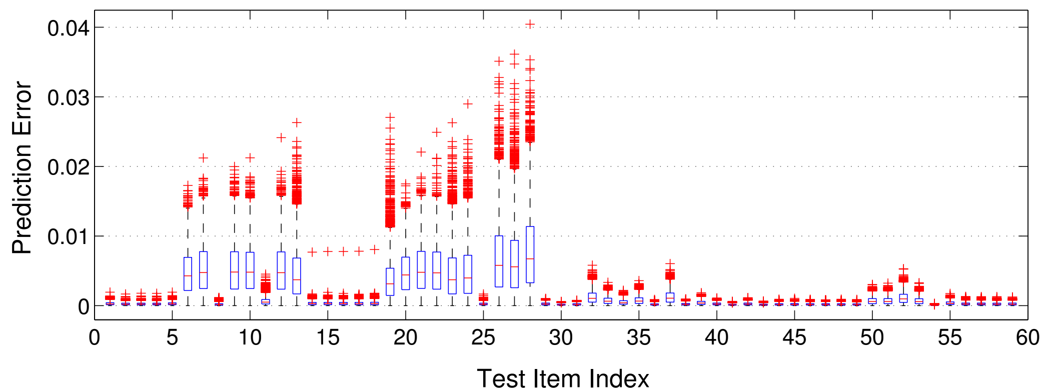


Figure 3.8: Error boxplot of each predictable test item.

Table 3.3: Number of Predictable Item vs. Margin from Spec Limits

Margin from spec limits	5%	15%	25%	35%	45%
Number of items as predictable	71	64	49	20	11

Figure 3.9 shows an example of GL's training result of one wafer for which each test item's prediction error and its margin from the spec limits are illustrated. Each dot in the figure denotes a candidate test item, in which the x-coordinate is the item's prediction error and the y-coordinate is one minus the margin from the spec limits (so the smaller the y value, the larger the margin and the more predictable the test item). We classify the test items in the unhatched area as predictable test items, as discussed in Section 3.3.1.

For the evaluation of the test application procedure described in Section 3.3.3, we used 500 dies in wafer No. 1 to train a model, and used all the remaining dies in the wafer for selection of predictable test items. When using the trained model for test application, we are concerned about test escapes: escaping faulty chips whose predicted values are mistakenly within the specification limits.

The test escapes of using various trained models are summarized in Table 3.4.

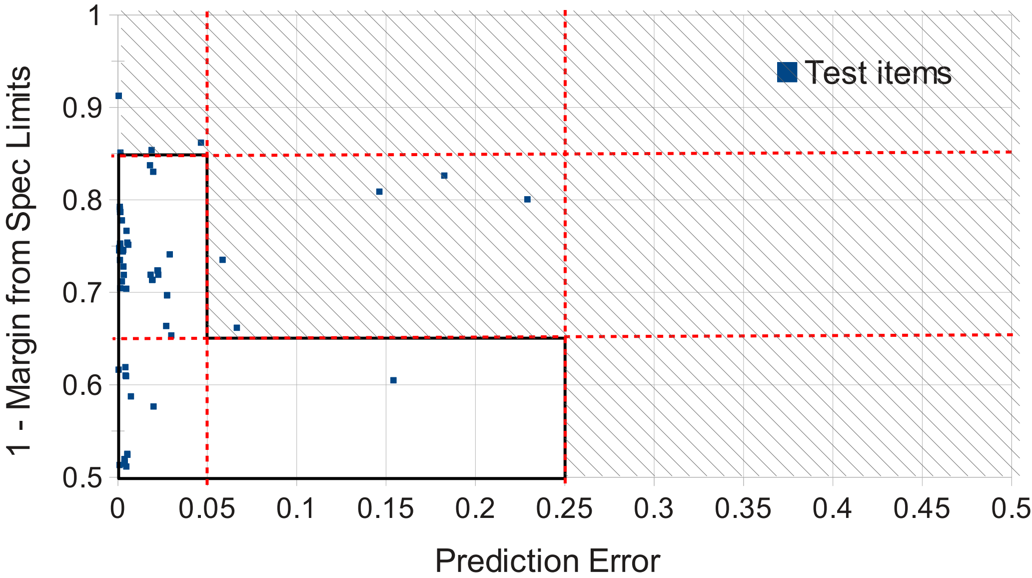


Figure 3.9: The predictability of test items based on test data of one wafer. Each dot denotes a candidate test item and the test items in unhatched/hatched area are defined as predictable/unpredictable test items.

We trained five different models. The first two models were generated by setting an error bound only, at $e = 5\%$ and $e = 25\%$ respectively. The third and the fourth models were generated by setting a lower bound on the margin from the spec limits only, at 15% and 35% respectively. The fifth model was generated using both criteria as showed in the unhatched area in Figure 3.9. Assuming that the test time of all test items is identical, the percentages of test time savings are 19.2, 21.0, 18.9, 5.9, and 17.5% respectively while the number of escaped dies, among the 130,950 dies tested, are 3, 14, 12, 1, and 1, respectively. Setting large error bound derives more predictable test items with, however, more escaped dies. On the other hand, setting tight margin results in fewer escaped dies with lower test time savings. The fifth model have a balance between test time savings and prediction quality.

Table 3.4: Number of Escaped Dies for Various GL Models

Error bound	Spec margin	# of escaped dies[a]	Predictable item #	Test time savings
5%	–	3	65	19.2%
25%	–	14	71	21.0%
–	15%	12	64	18.9%
–	35%	1	20	5.9%
$\sim 25\%^b$	$\sim 35\%^b$	1	59	17.5%

[a] 130,950 dies in total
[b] The unhatched area in Figure 3.9

3.4.2 TTR With Test Time of Individual Item

Enhanced from GL, WGL enables two applications for further test time reduction. First, it can take into account the distinct test times of individual test items. As the information of the actual test time and cost of each individual test item in the production test program is not available to us, we randomly generated two different sets of test times for our experiments to illustrate WGL's functionality and capability. In the first experiment, we used ten distinct test times, ranging from 0.1 to 3.5 unit, and randomly assigned 10% of the test items for each test time. In the second experiment, we also used ten distinct test times but with a larger range, ranging from 0.25 to 25 unit, and assigned them evenly to all test items.

The experimental results are shown in Table 3.5, where T_{GL} and T_{WGL} denote the predictable test items found by GL and WGL, respectively. For both cases, WGL can achieve further test time reduction by taking into account the test time of individual test item. In comparison with GL, WGL achieves additional 11%

Table 3.5: Test Time Improvement by WGL

Case	# of T_{GL}	Escaped dies #	Time savings	# of T_{WGL}	Escaped dies #	Time savings
1	59	1	17%	80	8	28%
2	59	1	17%	79	3	35%

and 18% test time savings than GL does for these two cases. Because the spread of test times for different test items in Case 2 is larger than that in Case 1, WGL achieves a greater test time saving in Case 2 even though it classifies one fewer test item as predictable than the first case.

3.4.3 Integrating Both Spatial and Inter-Test-Item Correlations

The second feature of WGL is the ability to exploit and integrate both spatial and inter-test-item correlations in test data. As described in Section 3.2.2, we can merge the results of VP (targeting spatial correlations) and GL (targeting inter-test-item correlations) in a straightforward way by taking the union of the predictable test items classified by each methods. Rather than directly taking the union of predictable test items, WGL can be used to optimize the number of total predictable test items. This is achieved by setting different weights to VP-predictable and VP-unpredictable test items before running WGL to explore the inter-test-item correlations.

The experimental results of comparing these two strategies are shown in Table 3.6, where T_{VP} denotes the predictable test items derived by VP and T_{GL} denotes either the predictable test items derived by GL for the first strategy (in

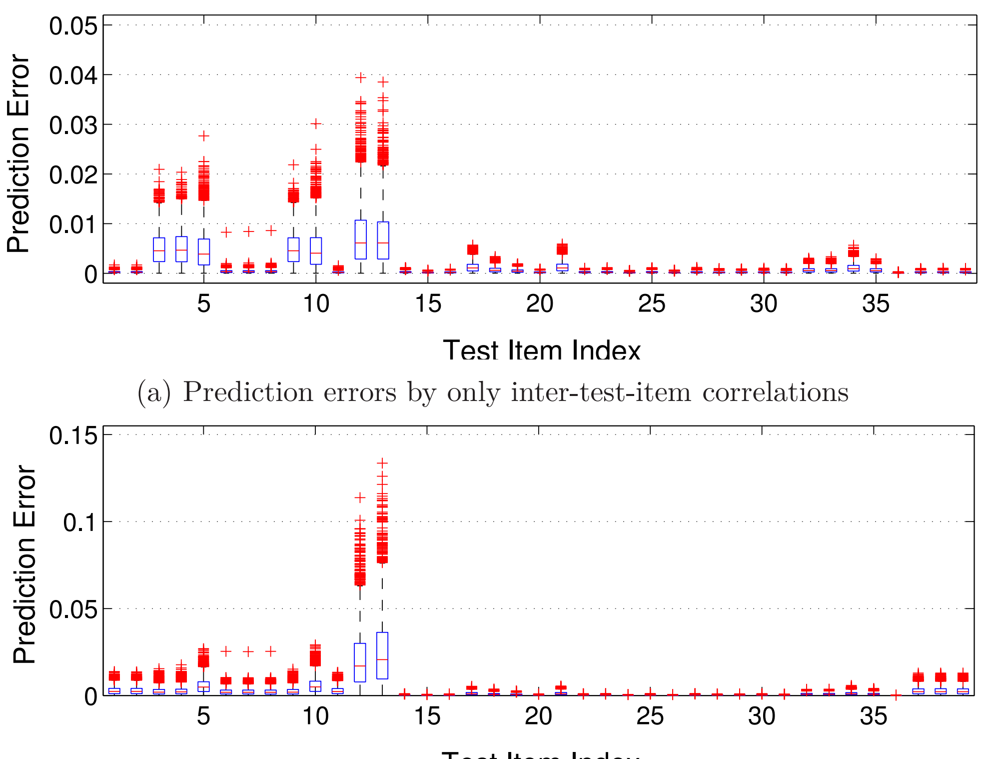
Table 3.6: Intuitive Merge vs. Weighted Merge

Method	$T_{VP} \cup T_{GL}$	T_{VP}	T_{GL}	$T_{VP} \cap T_{GL}$	Time savings
Straightforward	134	96	59	21 (36%)	39.6%
Weighted	153	96	65	8 (12%)	45.3%

the row indicated as "Straightforward") or by WGL for the second strategy (in the row indicated as "Weighted"). The values shown in the table are counts of test items in different sets. In this experiment, we assume all test items have an identical test time. The weighted strategy can save an additional 5.7% test time over the straightforward strategy for using both spatial and inter-test-item correlations. WGL successfully reduces the overlap between the predictable test items produced by VP and GL — the number of test items in their intersection reduces from 21 to 8.

Figure 3.10 attempts to compare the prediction errors of the same set of test items for using inter-test-item correlations only versus using both spatial and inter-test-item correlations. Figure 3.10a shows the prediction errors by using only inter-test-item correlation for prediction and Figure 3.10b shows the errors by using both spatial and inter-test-item predictions for prediction. When utilizing both spatial and inter-test-item correlations, VP was first run, whose results are then used as the input to WGL.

The fact that, under the VP+WGL strategy, some test items which are used for predicting other items in WGL were not actually tested but, instead, predicted using VP increase the overall prediction error. Test items 12 and 13 incurred a noticeable drop in accuracy. The reason for this is because these two items classified as predicable based on their relatively large margin from the spec limits,



(a) Prediction errors by only inter-test-item correlations

(b) Prediction errors by both spatial and inter-test-item correlations

Figure 3.10: Comparison of the GL and VP+WGL prediction errors for the same set of test items, which are the intersection of predictable test items derived by GL and by VP+WGL.

while they have a relatively large prediction error. As shown in Figure 3.10b, the prediction error of test items 12 and 13 are still within the 25% error bound used for classification.

Table 3.7 compares the results of using different statistical methods and strategies. In this comparison, the test time of each test item is based on the assumption of Case 2 in Table 3.5. Applying VP or GL only can achieve 31.5% and 15.7% test time savings, respectively. If we integrate VP and GL, which is equivalent to using VP and WGL with an assignment of the same weight to all test items, the test

Table 3.7: Summary of Test Time Saving for Various Strategies

Method	# of escaped dies[a]	# of predictable items	Test time savings[b]
VP only	4	96	31.5%
GL only	1	59	15.7%
VP+WGL with the same weight	5	136	41.8%
VP+WGL weighted by time	7	132	46.3%
VP+WGL weighted by VP	12	151	47.9%
VP+WGL weighted by time & VP	8	160	55.0%

[a] 130,950 dies in total
[b] Based on the test time assumption of Case 2 in Table 3.5

time saving increases to 41.8%. Integrating VP and WGL by assigning different weights to test items properly reflecting their test times can further improve the test time saving to 46.3%. If we integrate VP with WGL whose weights are based on VP's classification results, the improvement reaches to 47.9%. Finally, the last case shows that the integration of VP and WGL whose weights are jointly determined by both test items' test times and VP's classification results can achieve 55.0% improvement. However, the number of escaped dies might slightly increase in order to gain some of such improvements.

3.5 Summary

In this chapter, we propose a methodology to utilize inter-test-item correlations for test time reduction. We further improve the methodology to take into account

the test time of each individual test item for further reduction of production test time. Through integration with VP which can capture spatial correlations in test data, the methodology also allows exploiting and utilizing both spatial and inter-test-item correlations simultaneously for test time reduction. A case study of a high-volume industrial device shows that the proposed methodology can reduce the test time by 55.0% with only 8 escaped dies out of 130,950 tested dies in total.

Chapter 4

Silicon Characterization

4.1 Introduction

Efficient and effective testing and diagnosis could significantly improve product quality and manufacturing yield for complex designs. At different manufacturing stages, different test strategies are used to target different failures and variations. For example, to evaluate the quality and stability of the manufacturing process, wafer acceptance test (WAT, also known as wafer electrical test, WET, or e-test) is performed at the end of processing a wafer on process control monitors that are small devices located on the scribe lines. On the other hand, production test is conducted for each integrated circuit at a later stage to ensure the correctness and quality of each shipped product. As their objectives and test vehicles are different, the relationship and their implication between the test data from different test stages are not clear. As a result, analysis of test data has usually been restricted to data from one test stage only.

The same variations and/or defects can affect performances and characteristics measured at multiple test stages. For instance, variation to a process parameter can affect both Iddq test in the production test stage and the gate-oxide quality test in WAT. Therefore, characterizing systematic variations and failures based on test data from multiple test stages can potentially further improve process control monitoring, yield estimation, outlier wafer/die detection, and failure diagnosis. However, it is challenging to identify the connection between a bad die's syndromes derived from wafer probe test, especially for those caused by systematic failures, and its process parameters.

Representing the production test data in the form of color-coded wafer maps, two-dimensional spatial patterns, formed by either statistics of good/faulty dies or normalized measurement data, are considered as input to analysis of systematic failures. If similar spatial patterns (with respect to shape, size, and location) in wafer maps derived from test data are observed, there is a high probability that the corresponding faulty dies may experience similar systematic variations. Moreover, if the wafer maps derived from a process parameter and from a production test item exhibit a similar spatial pattern, then it should be strong evidence that the failures are strongly related to variations of a certain process step.

For the objective of characterizing systematic variations and failures between two different test stages, there are several challenges:

a) For the production test items with a binary (pass or fail) outcome, finding the correlations between such nonparametric test items and process parameters is a nontrivial task. Compared to parametric test items, nonparametric test items compress details into a single-bit test result, pass or fail, and thus

the test data has little information available for further analysis. That is, all failures have the same test syndrome, a fail symbol, but could be caused by different or the same type of variations and defects.

b) For WAT, there are limited number (e.g., five or nine) of sites within a wafer for measuring process parameters, while production tests are conducted for every die on the wafer. Furthermore, sites and dies are in different scales; a site usually covers multiple dies. Thus, data analytics between process parameters and production test measurements is limited by the resolution of the WAT data. Predicting process parameters for every die location in a complete wafer map based on the WAT data from the limited sites can help increase the accuracy for the temporal (i.e., inter-test-stage) analysis.

c) A correlation model hardly fits the test data with wafer-to-wafer and lot-to-lot variations, which are commonly seen in most products. A model tends to overfits test data that come from a single wafer or from a single lot. On the other hand, a model could underfit test data if data from all accessible wafers are used. In other words, a task of fitting a robust and flexible model for test data within a long time period is nontrivial.

d) Random variation and abnormal wafers degrade the accuracy of test data characterization, especially for the WAT data that have relatively few measurements for a wafer. Spatial patterns that are identified in more wafers or for more process parameters are more trustworthy than others that occur in fewer wafers of for fewer process parameters. Any proposed methods should be insensitive to outliers.

In this chapter, we propose a framework with several learning and statistical techniques to address these challenges. To overcome challenge a), we invoke pattern classification, which is well studied in the areas of both semiconductor manufacturing and computer vision, to build the connection between parametric process parameters with nonparametric production tests. Two-dimensional wafer maps could reveal some characteristics of systematic failures that binary vectors cannot due to inclusion of extra coordinate information. One example is to separate random defects from the test syndromes with systematic failures that follow certain two-dimensional spatial patterns.

To address challenge b), we extend Virtual Probe (VP) [9] to predict process parameters in every location within a complete wafer map, based on WAT data at limit site locations. We improve VP for predicting process parameters in every location within a complete wafer map in the die scale.

We then address challenge c) using a *biclustering* technique. A biclustering technique classifies a two-dimensional test dataset into several biclusters. In each bicluster, the dies are similar to each other on the test items and vice versa. Hence, fitting a model using a subset of data (i.e., fitting based on the data in a bicluster) is more efficient and effective than using the entire dataset. Our experimental results show that some spatial patterns can particularly be recognized in a bicluster. Several biclustering algorithms have been published, such as FABIA [72]. As an advantage, FABIA also takes care of challenge d) by the algorithm itself excluding outliers from the identified biclusters. In other words, a random defect will not be classified as a systematic pattern.

We organize this chapter as follows. We define patterns of test data and

review the background of FABIA in Section 4.2. In Section 4.3, we detail the implementation of our proposed framework, including spatial modeling with a greater resolution, biclustering, and pattern classification. Then, we demonstrate the efficacy of our proposed methods using industrial data in Section 4.4. Finally, we conclude in Section 4.5.

4.2 Background: Biclustering and FABIA

Biclustering is widely used in bioinformatics field for extracting knowledge from gene expression measurements [73], and has also been generalized for handling two-dimensional dataset. Using semiconductor test data as an example, each row of a test measurement matrix corresponds to a die sample and each column corresponds to a test item. Performing clustering on columns (i.e., clustering similar test items) has a limitation that test items that are somewhat similar may only be similar on a subset of dies, but not all dies. Similarly, clustering rows (i.e., identifying similar dies) has a limitation that dies that are somewhat similar may be similar only for a subset, not all, of test items. Biclustering addresses these limitations and utilizes simultaneous clustering on the row and column dimensions of the test data matrix.

FABIA (Factor Analysis for BIcluster Acquisition) [72] is based on a multiplicative model that can be used to efficiently explore linear dependencies between dies and test items. This section illustrates how to perform biclustering on test data using FABIA. We briefly summarize the mathematical background of FABIA in the following. The test data are represented as a matrix $\boldsymbol{X} \in \mathbb{R}^{l \times n}$ where l

and n are the number of dies and test items, respectively. The element x_{jk} of $\boldsymbol{X}$ corresponds to the measurement of the kth test item in the jth die sample.

In a multiplicative mode, two vectors are similar if one is a multiple of the other, i.e., their correlation coefficient is one (or minus one). Based on this assumption, a bicluster is defined as a pair of a row set and a column set where the rows are similar to each other and the columns are also similar to each other. Such linear dependency is represented by $\boldsymbol{z}\boldsymbol{\lambda}^T$ where $\boldsymbol{\lambda}$ is a prototype column vector with nonzero elements for test items participating in a bicluster, and $\boldsymbol{z}$ is a column vector of factors by which the corresponding prototype columns are scaled. The nonzero elements of $\boldsymbol{z}$ denote the dies that participate in the same bicluster. As shown in Figure 4.1, a model with p biclusters is formulated by

$$\boldsymbol{X} = \sum_{i=1}^{p} \boldsymbol{z}_i \boldsymbol{\lambda}_i^T + \boldsymbol{\Upsilon} = \boldsymbol{Z}\boldsymbol{\Lambda} + \boldsymbol{\Upsilon}\,, \tag{4.1}$$

where $\boldsymbol{\Upsilon} \in \mathbb{R}^{l \times n}$ is additive noise; $\boldsymbol{Z} \in \mathbb{R}^{l \times p}$ and $\boldsymbol{\Lambda} \in \mathbb{R}^{p \times n}$ are the sparse factor matrix and the sparse prototype matrix, respectively. FABIA allows overlapping biclusters, but p should be defined explicitly.

FABIA formulates biclustering as a sparse matrix factorization problem. The measurements of the ith die ($\boldsymbol{x}_i$, the ith row of $\boldsymbol{X}$) can be interpreted by a factor analysis model:

$$\boldsymbol{x}_i = \sum_{j=1}^{p} \boldsymbol{z}_{ij} \boldsymbol{\lambda}_j^T + \boldsymbol{\epsilon}_i = \tilde{\boldsymbol{z}}_i \boldsymbol{\Lambda} + \boldsymbol{\epsilon}_i\,, \tag{4.2}$$

where $\boldsymbol{\epsilon}_i$ is the ith row of the noise matrix $\boldsymbol{\Upsilon}$ and $\tilde{\boldsymbol{z}}_i = (z_{i1}, \ldots, z_{ip})$ denotes the ith row of the factor matrix $\boldsymbol{Z}$. FABIA assumes that each $\tilde{\boldsymbol{z}}_i$ is $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ distributed. The unit covariance matrix indicates that the biclusters are not

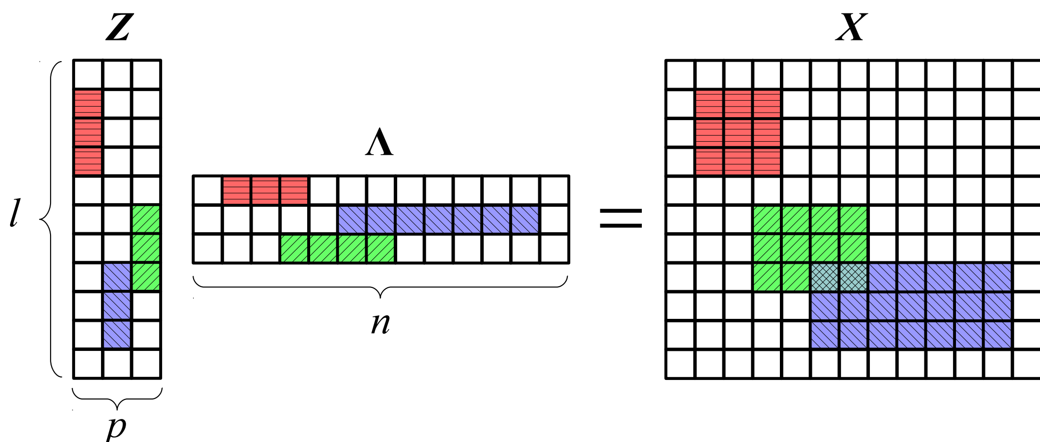


Figure 4.1: A multiplicative model with three biclusters. Each rectangular checkerboard represents a matrix. Colored/hatching rectangles are nonzero elements of matrices while white rectangles are zeros. The nonzero elements within a bicluster and within the corresponding prototype and factor vectors have the same color and hatching pattern. Note that the nonzero elements are adjacent to each other for visualization purposes only and additive error $\boldsymbol{\Upsilon}$ is not shown in this figure.

correlated, and, hence, one bicluster in the data will not be divided into dependent small biclusters. Another assumption is that each $\boldsymbol{\epsilon}_i$ is an independent Gaussian noise and is $\mathcal{N}(\mathbf{0}, \boldsymbol{\Psi})$ distributed where $\boldsymbol{\Psi} \in \mathbb{R}^{n \times n}$ is a diagonal covariance matrix.

FABIA then identifies the biclusters (i.e., selecting the model parameters $\boldsymbol{\Lambda}$ and $\boldsymbol{\Psi}$ that explain the data best) using *variational expectation maximization* while assuming component-wise independent Laplace priors for both $\tilde{\boldsymbol{z}}$'s and $\boldsymbol{\lambda}$'s. The implementation details are beyond the scope of this dissertation and can be found in [72].

4.3 Implementation

A proposed framework for characterizing systematic variations and failures between dataset from different test stages is detailed in this section.

4.3.1 Application Flow

Figure 4.2 shows the application flow of the proposed framework. The inputs are two different test datasets: one is process control test data, such as WAT data, which usually have relatively few samples due to the limited number of probe points on each wafer, and the other is production test data, such as WS data whose sample count is equal to the number of tested dies. Both datasets include multiple measurements (measured by a set of test items) for each sample. In addition, it is not required that the two datasets are from the same set of wafers although there should exist stronger correlations if they are. Based on these data, the method reports the potential correlated process parameters for the targeted production test items.

Two different methods are applied for extracting patterns from the two types of test data mentioned above. The first method, illustrated in the left part of Figure 4.2, is designed to extract grayscale patterns from WAT data. A grayscale pattern is defined as a wafer map in which each element is represented by a real number. VP, described in Section 2.3.1, is employed to predict a complete wafer map with a greater resolution based on the WAT data that only have limited samples for each wafer. Next, FABIA, introduced in Section 4.2, is employed to discover biclusters that reveal spatial patterns hidden in the VP-extrapolated

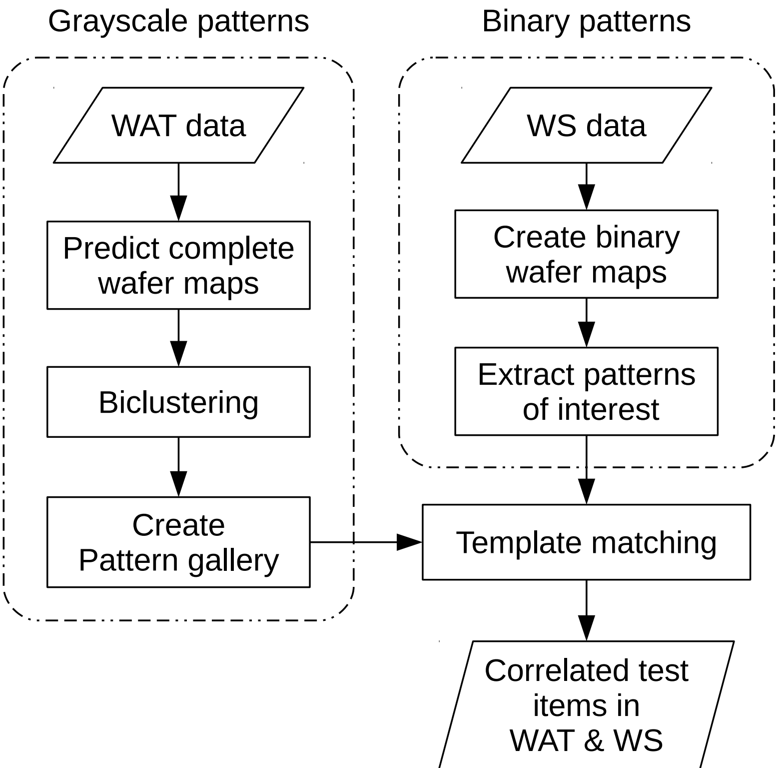


Figure 4.2: The application flow of the proposed framework.

WAT data. Such patterns are collected in a *pattern gallery* for further processing.

The second method, shown in the right part of Figure 4.2, is designed to extract binary patterns from the production test data, such as the WS data. Binary wafer maps, in which each element has a value of either zero or one indicating a pass or a fail, are derived from thresholded WS data based on predefined specifications. Next, binary patterns are extracted from the wafer map through clustering and classification methods. A template matching technique is then performed to compare the patterns of interest, derived from the WS data, with each pattern in the pattern gallery, extracted from the WAT data, to explore any potential correlations. The following subsections will detail each step of the framework.

4.3.2 Spatial Modeling With a Greater Resolution

As shown in Figure 4.3, each site (also known as a shot) is a rectangle area, which is usually of the same size as a lithographic photomask, composed of a number of identical dies. The process control monitors are a set of simple circuits located on the scribe lines between dies. The same set of monitors is duplicated for each site because of the exposure using the same photomask. Therefore, the measured process parameters at the process control monitors exhibit the process characteristics of a site, not just of a single die.

The VP algorithm described in 2.3.1 is utilized to model wafer-level spatial variations. Figure 4.4 shows examples of different strategies for performing VP based on WAT measurements at five sites to predict a complete wafer map. First, Figure 4.4b shows the predicted wafer map using five samples each of which has a size of a site as shown in Figure 4.4a. The predicted wafer map has a low resolution

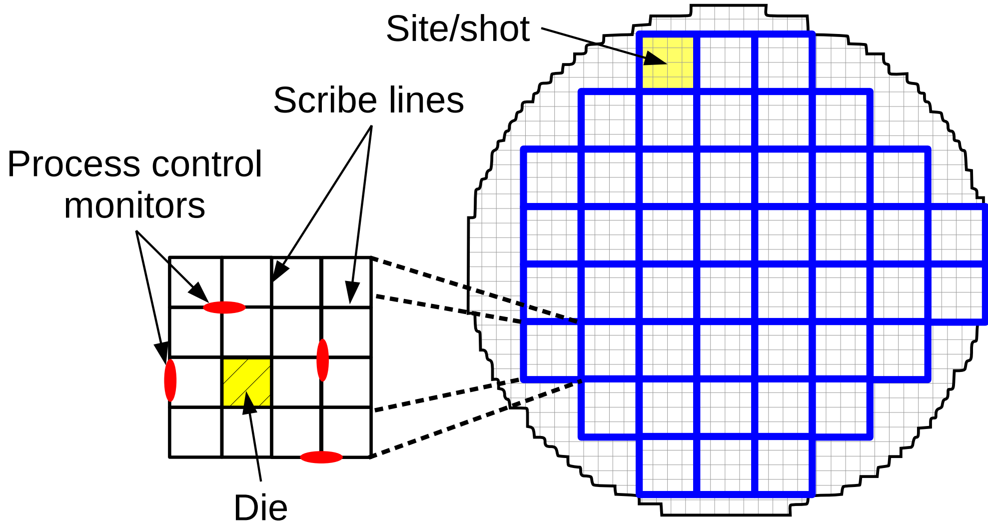


Figure 4.3: The spatial relations between sites/shots and dies. Process control monitors are denoted by red ellipses located on the scribe lines. Sites are denoted by rectangles with blue thick edges. Note that the dies close to wafer boundary form incomplete sites that are not emphasized in the figure and are incapable for probing process parameters.

and becomes pixelated due to the relatively large area of a site. In addition, the predicted values alter sharply from site to site. Second, if each measurement is used only to represent a single die at the center of the corresponding site, as shown in Figure 4.4c, VP derives a predicted wafer map that contains inaccurate high frequency patterns shown in Figure 4.4d because of an insufficient number of samples. The third strategy results in the best prediction through duplicating each WAT measurement for multiple die locations in a round shape within the site as shown in Figure 4.4e. Figure 4.4f shows a predicted wafer map that includes more details with a greater resolution.

As the number of samples is very limited, VP (or any other spatial modeling techniques) cannot guarantee the accuracy of the predicted wafer maps. To

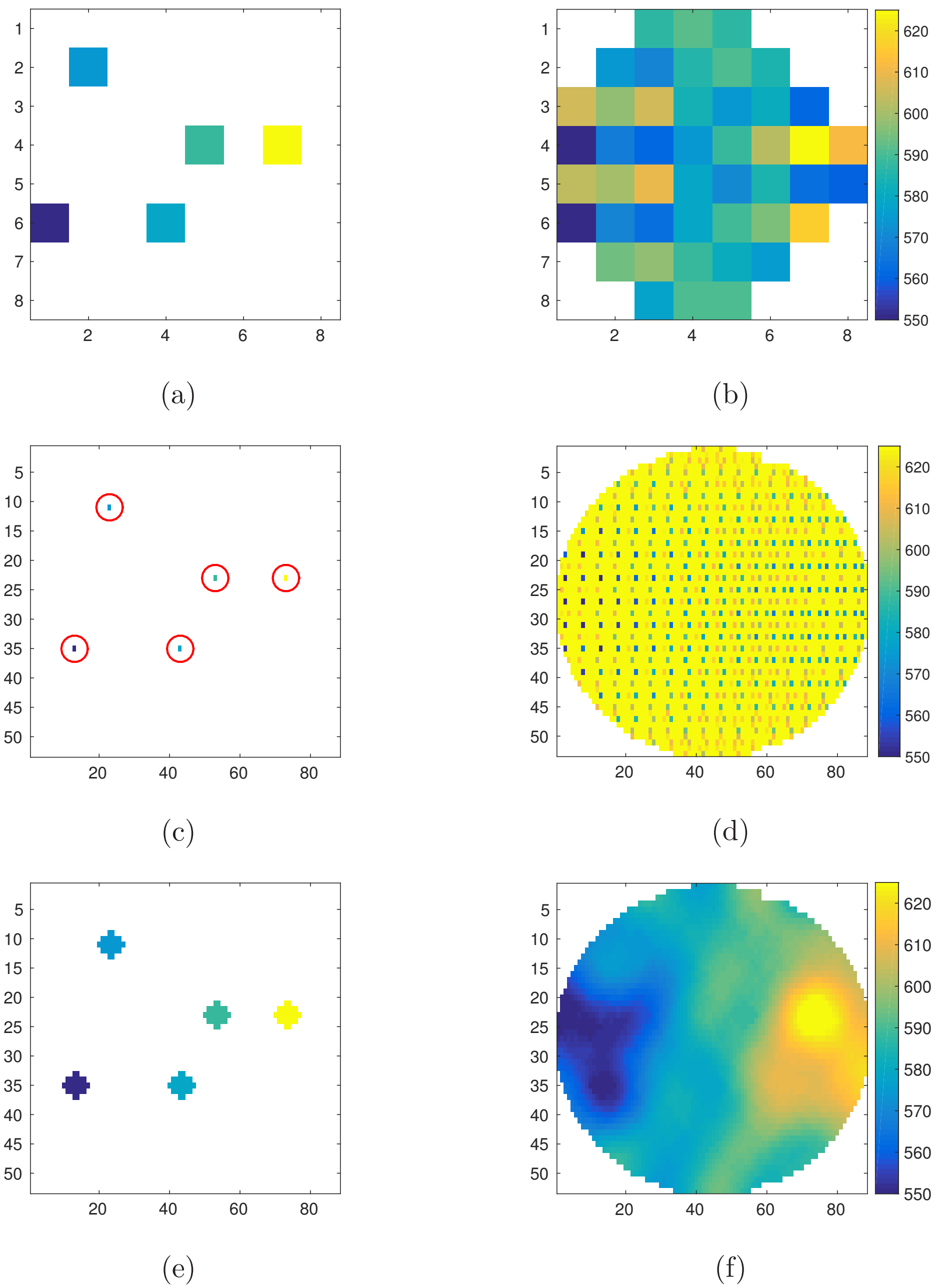


Figure 4.4: Color-coded predicted wafer maps of WAT using different sample strategies. (a), (c), and (e) show the samples, as a single site, a single die, and multiple dies, respectively, used by VP for prediction/extrapolation. (b), (d), and (f) show the corresponding predicted wafer maps.

maximize the accuracy, it is better that every measurement is used for training. However, there is no ground truth for verifying the accuracy of the predicted WAT wafer maps derived by VP as there are no probed measurements beyond those locations which are already used for VP prediction. On the other hand, cross-validation technique is not valid either due to the lack of probed WAT measurements for an extra independent validation data set.

Fortunately, it is observed that systematic variations and failures usually influence multiple wafers for most products in mass production. Therefore, if there exist strong correlations between process parameters and test item measurements that are influenced by systematic variations, they should be observed repeatedly in multiple wafers and lots, and thus the reliance on high accuracy of the extrapolated WAT wafer map should be significantly reduced. Hence, as long as a subset of (not necessary all) predicted wafer maps are reasonable accurate, out method should be able to reveal such systematic behaviors.

4.3.3 Pattern Extraction Using Biclustering

FABIA is the biclustering method used for extracting spatial patterns from multiple wafer maps. FABIA has two major inputs: a data matrix $\boldsymbol{X}$ and a limit p on the number of biclusters to be identified. Although p has a strong influence on the computation time (basically a cubic time complexity), a slightly large number is the best since FABIA will only identify as many biclusters as necessary to explain the data. The contents of $\boldsymbol{X}$ are dependent on the types of patterns for which we target to extract.

Grayscale patterns in Process Parameters

For a total of w wafers with n WAT test items used in analysis, we have wn predicted wafer maps derived from VP. An $l \times n$ matrix $\boldsymbol{X}$ is created using the predicted WAT measurements where $l = \sum_{i=1}^{w} l_i$ is the total number of dies and l_i is the number of dies in the ith wafer. A row of $\boldsymbol{X}$ consists of n WAT measurements of one die, and a column of $\boldsymbol{X}$ records measurements of one WAT item. Note that the WAT measurements are normalized with respect to each column. FABIA is then preformed on $\boldsymbol{X}$ to find biclusters that will be represented by sparse matrices $\boldsymbol{Z}$ and $\boldsymbol{\Lambda}$.

Figure 4.5 illustrates the procedure of extracting spatial patterns from the jth bicluster through analyzing $\boldsymbol{z}_j$, which is the jth column of $\boldsymbol{Z}$. Samples (dies) with larger absolute values in $\boldsymbol{z}_j$ indicate their more significant roles in the jth bicluster. Those factors in $\boldsymbol{z}_j$ that represent dies from the kth wafer can be grouped and shown as a wafer map $\boldsymbol{M}_k$ (for example, $\boldsymbol{M}_1$ consists of $z_{1j}, z_{2j}, \cdots, z_{l_1 j}$ in $\boldsymbol{z}_j$). Some of these wafer maps expose similar spatial patterns (e.g., $\boldsymbol{M}_1$ and $\boldsymbol{M}_w$ in Figure 4.5) while some wafers do not (e.g., $\boldsymbol{M}_2$). Note that each blank rectangle in $\boldsymbol{z}_j$ and, in turn, in $\boldsymbol{M}$s shown in Figure 4.5 has a zero value or a value smaller than a given threshold.

In practice, such wafer maps consist of errors, resulting from WAT measurement errors, modeling errors by FABIA, and prediction errors by VP. To enhance the accuracy of spatial patterns revealed by a bicluster and reduce the impact caused by these errors, we overlap all wafer maps resulting from $\boldsymbol{z}_j$ and derive the cumulative sum for each die location resulting in $\boldsymbol{M}_A = \sum_{i=1}^{w} \boldsymbol{M}_i$ as shown in Figure 4.5. This aggregated wafer map, $\boldsymbol{M}_A$, can be treated as an image in grayscale

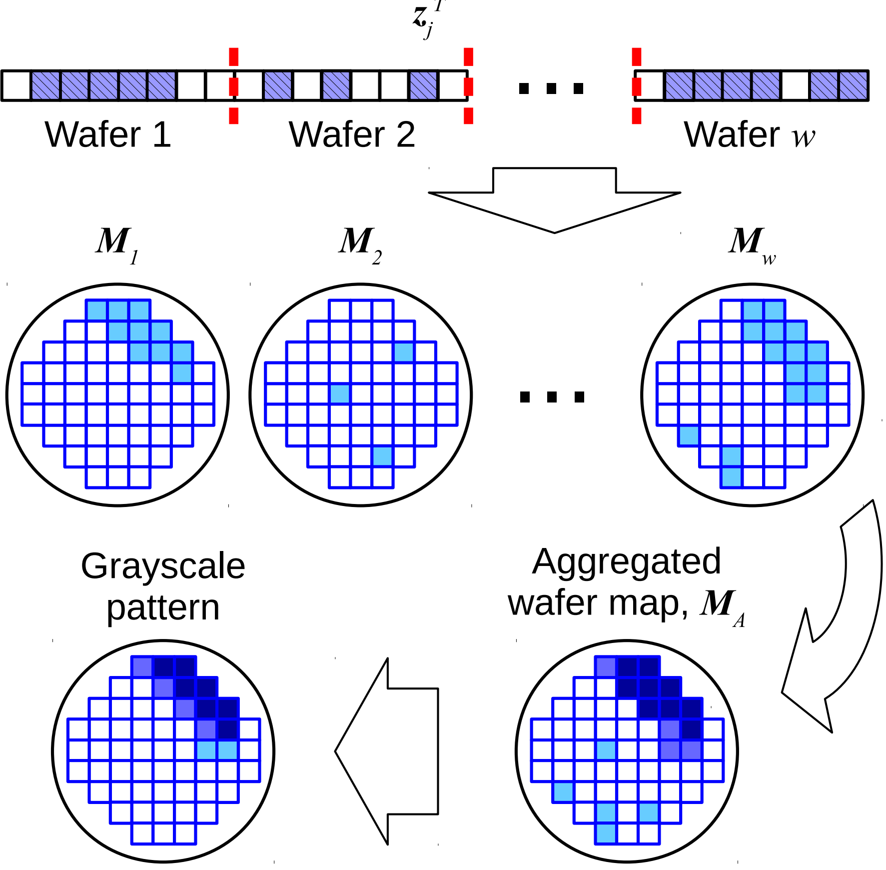


Figure 4.5: The procedure of extracting grayscale patterns from a bicluster that is derived by FABIA using WAT data.

and is then normalized to a range $[0, 1]$. Applying an appropriate threshold (e.g., 0.8) to eliminate errors from various sources results in the final grayscale pattern as shown in the lower-left wafer map of Figure 4.5.

Based on the factorization model defined in Section 4.2, the dies classified into the same bicluster are similar to each other on a subset of WAT items, i.e., these dies have linear inter-test-item correlations. In other words, the measurements of such correlated WAT items increase/decrease simultaneously. Moreover, the intensity of each pixel in a grayscale pattern is proportional to the number of dies that belong to the same bicluster at the location where each pixel is. FABIA do not take spatial locations of dies into account when processing the WAT data. Therefore, if the correlations observed in a bicluster exhibit some spatial pattern such as the grayscale pattern shown in Figure 4.5 instead of randomly spreading dots, it indicates a strong connection between a local systematic variation and the corresponding process parameters. We derive grayscale patterns from each bicluster and create a pattern gallery from all identified biclusters for further analysis.

Binary Patterns in Production Tests

For production test data, we focus on the pass/fail decisions based on the pre-defined specifications instead of actual measurements. Hence, the test signature of a die is a n-bit vector where n is the number of test items. For integrated circuit fabrication, classifying dies based on such test signatures is called binnig, i.e., a group of dies that fail on the same set of test items are assigned an identical bin number. The locations of the dies with the same bin number may form spatial

patterns that suggest potential systematic variations and failures, especially when the patterns are correlated with some process parameters.

To extract patterns from a target bin (e.g., bin k), the procedures are illustrated in Figure 4.6. Assume that there are w wafers used for analysis and the binning results (i.e., the locations of classified dies) are represented by binary wafer maps, $\boldsymbol{B}_1, \boldsymbol{B}_2, \cdots, \boldsymbol{B}_w$. Each die location of $\boldsymbol{B}$s is either zero (not a member of bin k, a blank rectangle) or one (a member of bin k, a solid rectangle). Such wafer maps of bin k can be represented by a $w \times c$ matrix $\boldsymbol{X}_k$ where c is the number of die locations of a wafer map. The ones in ith row of $\boldsymbol{X}_k$ denote the dies in ith wafer that are the members of bin k. The ones in a column of $\boldsymbol{X}_k$ denote the dies in different wafers but at the same die location (one out of c locations) with respect to a wafer map.

In the previous subsection, we have demonstrated that biclustering technique is effective for discovering clusters in a two-dimensional dataset with noise. Therefore, we apply FABIA on $\boldsymbol{X}_k$ to identify any patterns that exist in multiple wafers, i.e., a bicluster (which is a pair of a wafer set and a location set) in $\boldsymbol{X}_k$. Different from the case of processing WAT data, the factorized matrix $\boldsymbol{\Lambda}_k$ of $\boldsymbol{X}_k$ explicitly indicates the location of a pattern by its nonzero elements. As illustrated in the lower-right wafer maps of Figure 4.6, each row of $\boldsymbol{\Lambda}_k$ represents a wafer map with preliminary patterns.

Next, a sequence of image processing techniques are performed to enhance and isolate these preliminary patterns: *a*) dilating the image to delineate the outline of the objects of interest, *b*) filling interior gaps for solid objects, *c*) smoothing the objects to compensate the dilating executed before, and then *d*) detecting

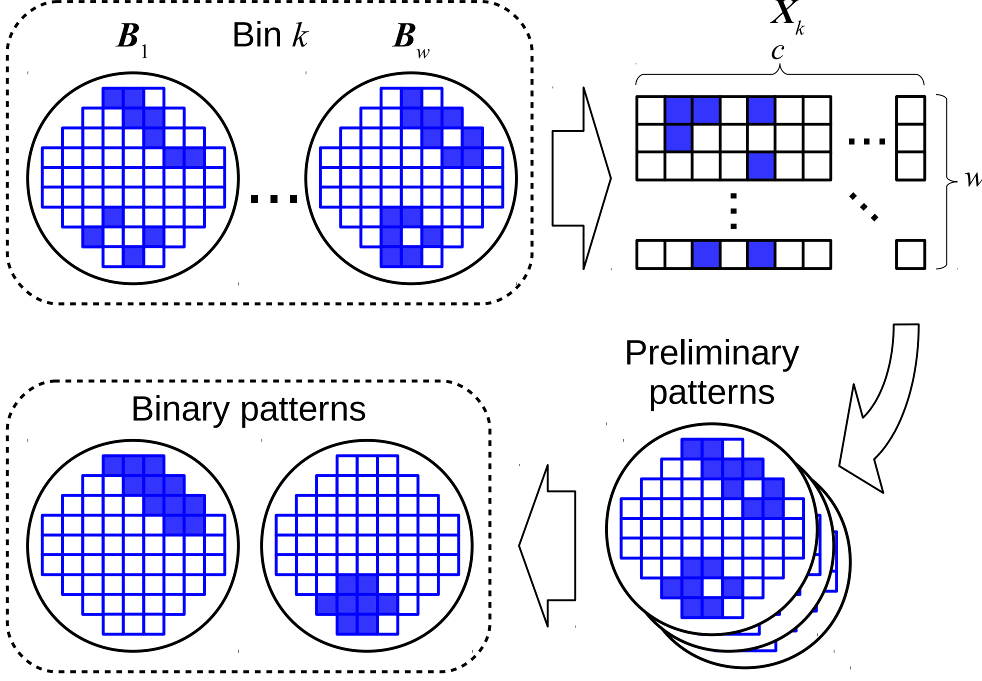


Figure 4.6: The procedure of extracting binary patterns based on the binning results.

connected components while excluding abnormal large and tiny objects. These steps result in the final binary patterns as shown in the lower-left of Figure 4.6.

4.3.4 Template Matching

The next step of exploring correlations between WS data and WAT data is formulated as a template matching problem and is solved by normalized cross-correlation (NCC) [74, 75, 76]. Cross correlation is studied in the area of computer vision to solve the problem of determining the location of a given pattern/template within an image based on the squared Euclidean distance measure. For better identifying similarity, NCC normalizes the image and the template and has the following advantages: *a*) invariant to the image energy (i.e., sum of squares of pixel intensity) under the window containing the template, *b*) having the range $[-1, 1]$ of similarity that is independent of the size of the template, and *c*) invariant to changes in image amplitude, such as brightness or contrast of the image.

The NCC similarity γ of a template t shifted to location (u, v) with respect to an image f is defined as

$$\gamma_{t,f}(u,v) = \frac{\sum_{x,y}\Big[f(x,y) - \bar{f}_{u,v}\Big]\Big[t(x-u,y-v) - \bar{t}\Big]}{\sqrt{\sum_{x,y}\Big[f(x,y) - \bar{f}_{u,v}\Big]^2 \sum_{x,y}\Big[t(x-u,y-v) - \bar{t}\Big]^2}}, \tag{4.3}$$

where $\bar{f}_{u,v}$ denotes the mean value of pixels of $f(x, y)$ within the region under the shifted t, and $\bar{t}$ denotes the mean value of all pixels in t.

Referring to the application flow in Section 4.3.1, a binary pattern and a grayscale pattern serve as the template t and the image f for Equation (4.3), respectively. Usually, NCC is performed for each possible offset and the offset

$(u_{\max}, v_{\max})$ resulting in the maximum similarity $\gamma_{\max}$ indicates the upper-left corner of the best matching region in the image for the given template. In our framework, templates (WS binary patterns) and images (WAT grayscale patterns) are both of a size of a wafer map. Therefore, we define the maximum similarity of a WS pattern t based on a WAT pattern gallery $\boldsymbol{F}$ as

$$\gamma_{t,\boldsymbol{F}} = \max(\{\gamma_{t,f_i}(0,0) : i = 1, 2, \ldots, |\boldsymbol{F}|, f_i \in \boldsymbol{F}\}), \tag{4.4}$$

where $\gamma_{t,f_i}(0,0)$ denotes the similarity between t and the ith pattern of $\boldsymbol{F}$ with no shift for t, and $|\boldsymbol{F}|$ denotes the number of patterns in the gallery.

4.4 Experimental Results

We evaluate the proposed framework using a test dataset from a non-volatile memory product using 200mm wafers processed on a industry standard 0.35μm mixed-signal technology from ams AG. The dataset includes WAT data and WS data for 300+ wafers with 3500+ dies per wafer. The WAT data have 124 process parameters that were probed from five out of 40 sites per wafer, and the WS data have 95 production test items that are measured for every die.

4.4.1 WAT Data Analysis

As shown in Figure 4.3, the WAT data consist of measurements from the process control monitors in five different sites of a wafer, and different sites were selected for different wafers. Before applying VP to derive complete wafer maps,

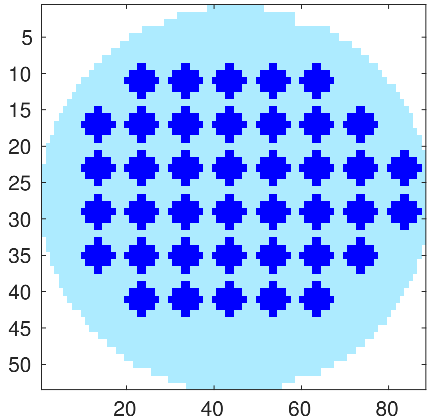


Figure 4.7: The mapping between site locations and die locations. Each WAT sample is replaced with 24 die samples in the same region.

we replaced each WAT sample with 24 die samples having identical measurements that are duplicated from the WAT sample. The locations of these 24 samples are within the region of the site where the WAT sample was probed. The mapping between site locations and die locations are shown in Figure 4.7. Each solid round in Figure 4.7 indicates 24 dies that are used to replace a site in the same region. As a result, VP has 120 samples per wafer as its input, instead of five samples per wafer in the original WAT data. An example of a predicted WAT wafer map is illustrated in Figure 4.4f. Even though this process parameter is measured at only five sites as shown in Figure 4.4e, the predicted wafer map clearly exposes an upward trend from left to right.

In the next step, we explored biclusters in the predicted WAT data using the FABIA algorithm [72, 77]. The VP-predicted WAT measurements formed a two-dimensional matrix $\boldsymbol{X}$ with 1.1M samples (dies) in rows and 124 WAT items in columns. The limit on the number of biclusters to be identified p was set to 12.

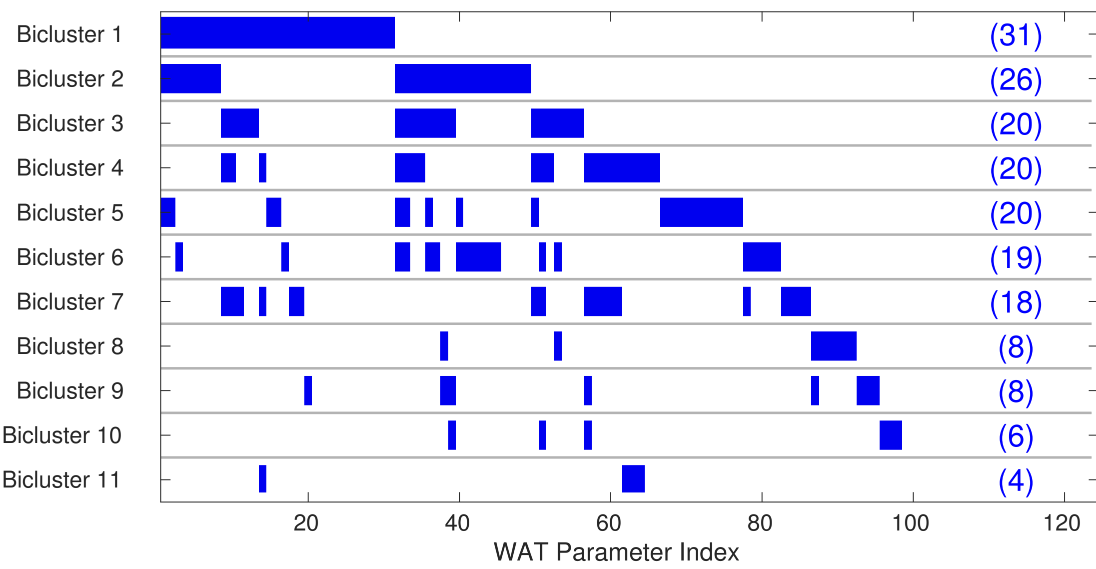


Figure 4.8: A diagram showing the factorized matrix $\mathbf{\Lambda}$. The solid bars denote nonzero elements of $\mathbf{\Lambda}$, i.e., the corresponding WAT items of which a bicluster is composed. The number of WAT items in the item set of a bicluster is given in parenthesis. Note that both bicluster axis and item axis in this diagram are reordered for better visualization.

Based on Equation (4.1), these settings resulted in a model with 11 biclusters. The factorized matrix $\mathbf{\Lambda}$ are illustrated in Figure 4.8. The diagram shows that an item set of a bicluster consists of four to 31 WAT items and the item sets of two different biclusters may have overlap. In addition, each item set has unique WAT items except the item set of Bicluster 11, which is a subset of Bicluster 4's. However, Bicluster 4 is not a subset of Bicluster 11 due to having a different set of dies from each other according to the factorized matrix $\boldsymbol{Z}$. The diagram also indicates that around one-fourth of items (30 out of 124) are linearly independent of any other items and are not clustered into any biclusters.

As mentioned in Section 4.1, systematic variations and failures may not occur globally and only affect a portion of dies in a wafer. That is, with a high

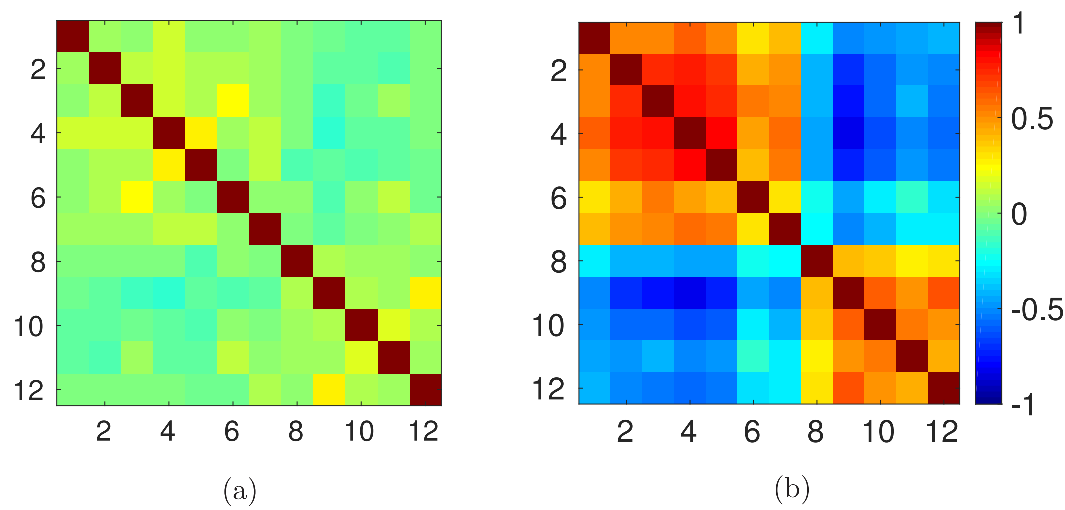


Figure 4.9: Linear correlation matrix of twelve WAT items of Bicluster 1. Each rectangle represents a color-coded correlation coefficient of two WAT items. (a) and (b) were derived based on S_1' and S_1, respectively.

probability, such variations appear in a small region of a wafer. Based on the same factorization model described above, assuming that S_1 denotes the die set of Bicluster 1, and S_1' denotes the complement of S_1, i.e., the die set that consists of dies not in S_1. S_1 and S_1' include 23% and 77% of the total number of dies, respectively. Figure 4.9 shows the color-coded correlation matrices of twelve WAT items in Bicluster 1 for these two die sets. Every correlation coefficient shown in Figure 4.9a except those on the diagonal is quite small and is in the range from -0.16 to 0.27. Hence, we can conclude no (or very weak) linear correlations among these WAT items based on S_1'. On the other hand, the correlations shown in Figure 4.9b are significantly stronger (-0.82 is the strongest coefficient) and become evident for the existence of local systematic variations in S_1.

4.4.2 Matches Between Process Parameters and Product Tests

Based on the procedure described in Section 4.3.3, grayscale patterns of WAT parameters were extracted from the factorized matrix $\boldsymbol{Z}$, which indicates the die set of each bicluster. Referring to Equation (4.2), a part of dies in a die set positively contribute to $\boldsymbol{X}$ while the rest of the dies contributing negatively. Therefore, each aggregated wafer map from $\boldsymbol{Z}$ revealed two patterns (one representing the positive part and the other representing the negative part of a bicluster) and formed a pattern gallery with 22 grayscale patterns as shown in Figure 4.10. Even though some patterns are similar, such as Figure 4.10(k) and Figure 4.10 (l), they were extracted from different biclusters with distinct test item sets and represent different correlations. These spatial patterns abstracted from 124 WAT parameters illustrate some potential systematic variations.

On the other hand, there are 95 test items in the WS data and 23 of them are nonparametric test items with only pass/fail outcomes. 95 test items are classified into 76 test groups and the test items in the same group target similar circuit functions. Based on the procedure described in Section 4.3.3, binary patterns were extracted from the test results of each test group (i.e., the union of the faulty dies detected by each test item in a test group). Figure 4.11 illustrates some binary pattern examples.

For several critical WS items that are related to the quality of memory circuits, the template matching technique, NCC, was performed to examine the most possible candidates in the grayscale pattern gallery. Table 4.1 lists five matches that were discovered in this industrial test dataset and were sorted by the NCC

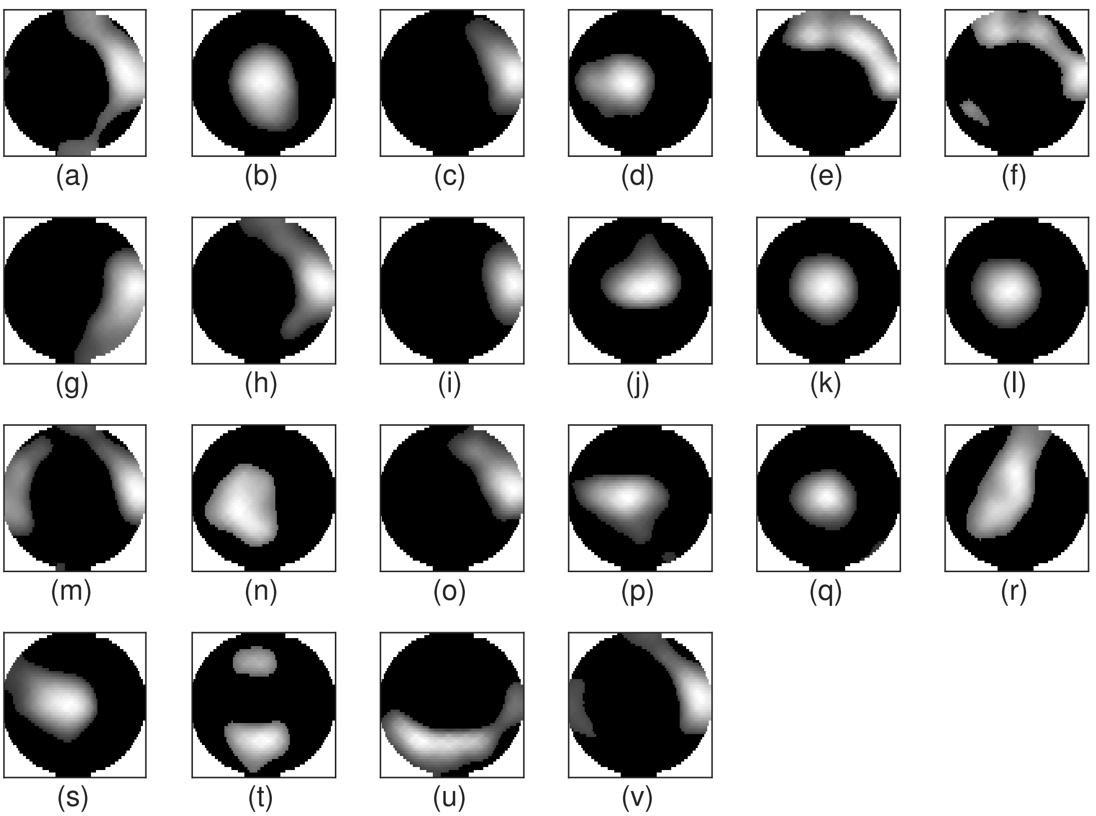


Figure 4.10: A grayscale pattern gallery. The intensity of each wafer map is expressed within a range from zero (black) to one (white). Note that the regions outside of a wafer map should be in black (zero values) and are shown as white regions for better visualization.

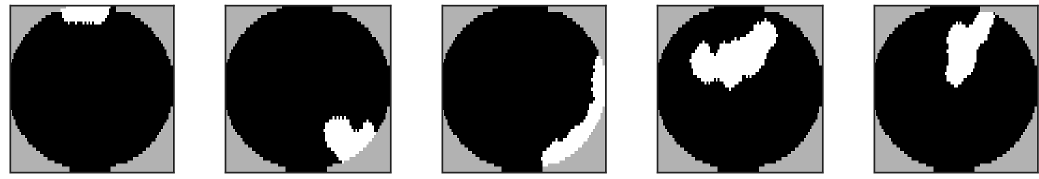


Figure 4.11: Binary pattern examples. Note that the regions outside of a wafer map should be in black (zero values) and are shown as gray regions for better visualization.

similarities. Each row in the Table 4.1 indicates the WS item set with the corresponding binary pattern, the WAT parameter set with the corresponding grayscale pattern, and the similarity between these two patterns of a discovered match. The last column reports the level of existing comprehensible correlations between the WAT parameters and the WS items, which are confirmed by the process engineer of ams AG.

Each Match shows a pattern for a WS item set and matches with the WAT pattern derived from the listed parameters. A direct correlation between the WS items and each single WAT parameter is not obvious in the majority of cases and is identified with difficulties. Most of the time several WAT parameters have to shift together to result in a WS fail. The block of the chips related to the WS items has to be analyzed taking the shifts of multiple WAT parameters into account to get a better understanding of a possible correlation or in the best case to get the confirmation of the correlation. However, the proposed framework successfully reveals such correlations.

The correlation between WAT parameters and WS items for Match 1 and Match 2 is comprehensible for the WAT parameters `CWET`, `CW1`, and `JET1` that correspond to the same element. This element is frequently used in the block of the chip responsible for the WS items listed. WAT parameters `QPMZ1` and `QPMZ3` might also contribute to the correlation although the contribution is not as obvious as for the other parameters mentioned above. The WS `SATrip_ICellD#` items can be influenced by a diode related to the WAT parameter `CW2`. The diode is used as a sensing element. If the leakage of the diode shifts, the corresponding WS items of Match 2 will also show a shift. A possible contribution to the correlation of all

Table 4.1: Matches Between WS Items and WAT Parameters

Match	WS Item(s)		WAT Parameters		γ	Comprehensible Correlations
	Name	Pattern	Name	Pattern		
1	Volts1stErase, TailErase/ProgramFunc, TailFunctionProg, BasicFunctional, EnduranceRead, RetentionRead/ICell		CWET, CW2, CW2I, CW1, CW1MEF, TWC, JET1, QPMZ1, QPMZ2, QPMZ3		0.78	High
2	SATrip_ICellD0, SATrip_ICellD1, ..., SATrip_ICellD15		CWET, CW2, CW2I, CW1, CW1MEF, TWC, JET1, QPMZ1, QPMZ2, QPMZ3		0.75	Median
3	Volts1stProgram		TMFBL1, WU1, WU2		0.58	Low
4	Volts1stProgram		CWHPY1, CWHPY2, HBNNB, JET2, LQ1, LQ2, UDP1, UPD2		0.53	High
5	MarginHighIRef, MarginLowIRef		SEJG1, SEJG2, SEJG3, XFGG1, XFGG2		0.47	None

other WAT parameters is not directly comprehensible.

The correlation between the WS item `Volt1stProgram` and the listed WAT parameters for Match 4 is comprehensible since the WS item and most of the WAT parameters are related to the memory element used in the circuitry. A correlation is obvious for most of the WAT parameters (especially for `CWHPY1` and `CWHPY2`) except for `JET2`, `LQ1`, and `UDP1`. Match 3 and Match 4 show that the same WS item is matched with different WAT parameters. Match 3 is related to the sensing of the memory element while Match 4 is directly related to the memory element. For Match 5, the similarity is already below 0.5 and the WS pattern is more off-center than the WAT pattern. There is no known possible correlations for Match 5.

The listed WAT parameters in Table 4.1 are further analyzed based on their significance in a bicluster. An element with a larger absolute value in $\boldsymbol{\lambda}$ of Equation 4.1 indicates a test item that plays a more significant role in a bicluster. In Match 1, `CW1` and `QPMZ1` are the two most significant parameters that exhibit the WAT grayscale pattern due to their large absolute λ values as shown in Figure 4.12. In Match 2, the parameter `CW2` with comprehensible correlations with WS items also has a relatively large λ. Moreover, `CWHPY1` and `CWHPY2` in Match 4, which are confirmed having comprehensible correlations with the WS `Volt1stProgram` item, are the two most significant parameters that exhibit the grayscale pattern. The interpretation of Figure 4.12 and the explanation of Table 4.1 are coherent.

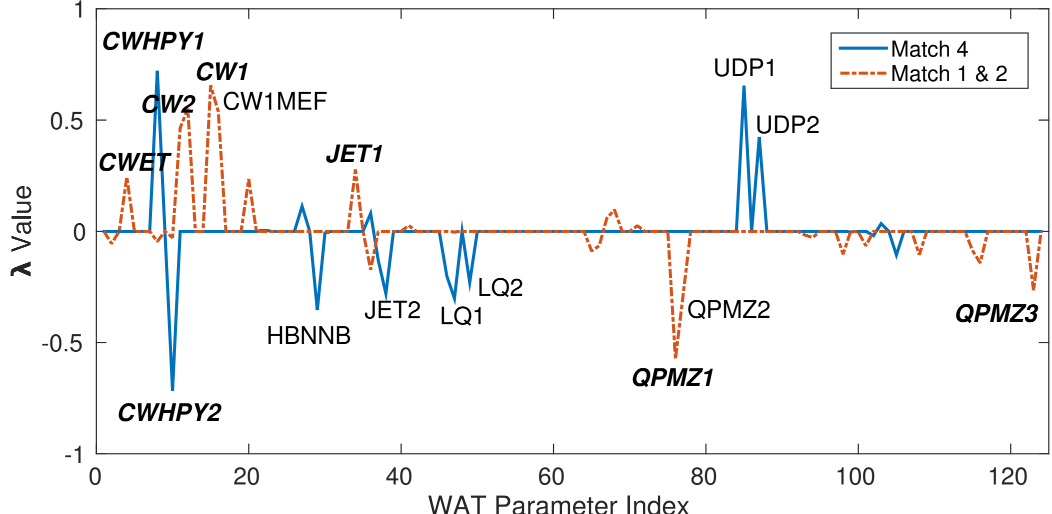


Figure 4.12: A plot showing the significance of WAT parameters in Match 1, Match 2, and Match 4. The parameters that are confirmed having comprehensible correlations with WS items are labeled in bold italic.

4.5 Summary

In this chapter, we propose a framework for characterizing systematic variations and failures through exploring the hidden patterns of test data from different test stages. The framework utilizes the spatial patterns extracted from both process parameters with a limited number of probed measurements and production tests with binary outcomes. The proposed framework is performed on an industrial test dataset and has successfully revealed some comprehensible correlations between WAT parameters and WS items. The results of such silicon characterization can be used to discover parametric variations and weak links in the manufacturing process.

Chapter 5

Test Data Analytics Toolbox

5.1 Introduction

There are two major requirements for a software tool to manage very-large-scale integration (VLSI) test data. VLSI test data have a unique data hierarchy: a product consists of lots; a lot consists of wafers; a wafer consists of dies. Moreover, test data have unique properties of internal connection: a die is tested at multiple test stages; a test stage performs multiple test items; a measurement from a test item is evaluated by specification limits. Therefore, the first requirement of a tool designed for analyzing such test datasets is to provide functions of data parsing, recording, selection, and management dedicated to the unique hierarchy and properties.

The other requirement is the flexibility for the integration of multiple research projects. Different projects may utilize test data from different sources and may conduct different learning algorithms. For example, the test time reduction

methodology described in Chapter 3 exploits two statistical learning algorithms for exploring spatial correlations and inter-test-item correlations, respectively. In addition, the silicon characterization framework proposed in Chapter 4 utilizes test data from two datasets, which are in different formats and are from different sources.

There are no opensource or public tools can fulfill these two requirements. Therefore, we build our own toolbox that is dedicated to test data analytics developed in the course of this research. The toolbox has the following features: *a*) a modular and object oriented design using MATLAB and Python, *b*) scalable and configurable parser for reading test data stored in different formats, *c*) data pre-processing functions, such as normalization and outlier removal, *d*) data processing functions, such as sampling, splitting, and concatenation, and *e*) various learning algorithms, such as variation modeling, clustering, and classification.

The rest of this chapter is organized as the following. Section 5.2 presents the modular architecture of the toolbox. Section 5.3 details the data structure of the toolbox, and we provides examples in Section 5.4.

5.2 Implementation

This section presents the main functions provided by the toolbox with implementation details.

5.2.1 Test Data Parsing

Figure 5.1 illustrates the process flow for parsing test data from the specified sources. The typical sources of test data are files and each file stores the test data from a single wafer. After receiving a request, the process searches for the corresponding saved `Archive` from the local disk storage first. An `Archive` is an object used to store parsed test data and is described in Section 5.3.1. If the test data of interest have been parsed before, an `Archive` might already be saved on the disk based on the user's configurations.

The process then invokes functions of a configured parser module, which is inherited from the base parser module, to read test data from the source file if the saved `Archive` is not existed. The base parser module includes several basic but configurable functions to parse test data in pre-defined formats, such as standard test data format (STDF) or comma-separated values (CSV). The user can configure the parser or even add any enhanced functions to the inherited parser module for accessing specific data format. For instance, a user can add functions to access test data from an online database instead of local files.

The next step, the process creates a new `Archive` for storing and tracking the requested test data. Based on the configurations provided by a user, this newly created `Archive` may be saved to the disk for speeding up the next requisition. Finally, the process returns the parsed test data through `Archive` objects. Note that an `Archive` is used for the test data that belong to one single wafer at one test stage.

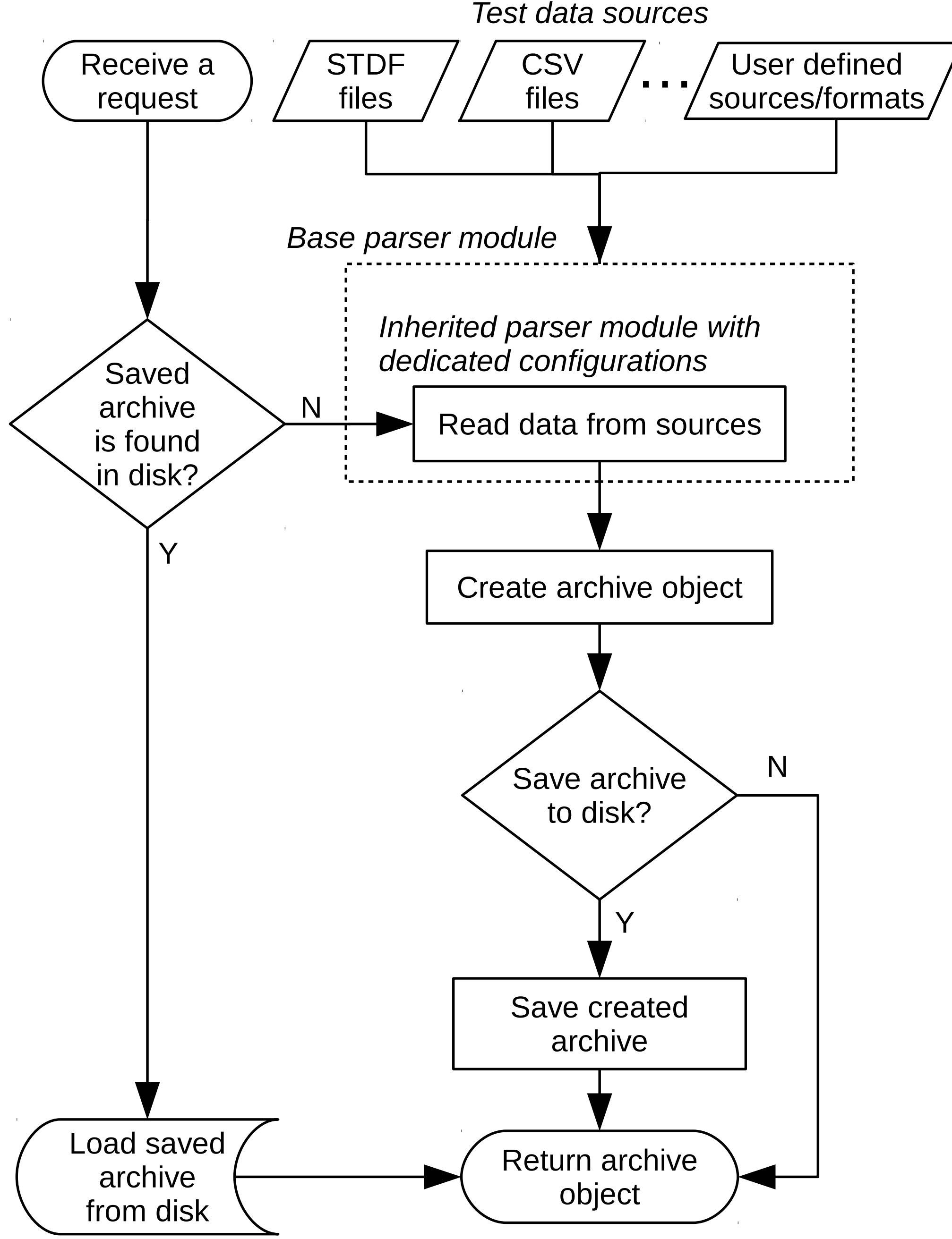


Figure 5.1: The flow of parsing test data from sources.

5.2.2 Test Data Selection

In this toolbox, any properties related to a source of test data are recorded by a `Source` object, which is detailed in Section 5.3.1. A user have to access test data through the selection functions of a `Source` that is linked with a particular parser (for the specified test data formats and sources). The toolbox tries to satisfies most of the scenarios in which a user will select test data. The supported scenarios are listed below.

a) Selecting test data based on different hierarchy. A user can retrieve a subset of dies based on lots and/or wafers and/or die locations specified by a user defined list.

b) Selecting test data based on different die types. A user can retrieve a subset of dies based on the specified die types: pass, fail, missing value (dies with missing measurements), invalid (dies with corrupt or damaged measurements), or a user defined list.

c) Selecting test data by sampling. A user can retrieve a subset of dies randomly sampled from a wafer based on a fixed number or a percentage.

d) Selecting test data based on different test item types. A user can retrieve a subset of test measurements based on the specified test item types: parametric, nonparametric/functional, valid (test items with valid spec limits as the principles for pass/fail decisions), or a user defined list.

e) A combination of the above four scenarios. For example, a user can retrieve a subset of test data that consists of 10% randomly sampled fail dies

from every wafers in the first and second lots with measurements of every parametric test items.

5.2.3 Learning Algorithms

The toolbox comes with various build-in learning algorithms, such as the VP and JVP algorithms that are described in Chapter 2; the GL and WGL algorithms that are described in Chapter 3. The toolbox can also be integrated with other projects and tools, such as LIBSVM [78] and MOSEK [79]. The toolbox provides a universal application programming interface (API) for every supported learning algorithm as applicable. A user can invokes the same set of methods, such as `train()`, `validate()`, `predict()`, `show()`, and `reset()` to utilize different learning algorithms.

5.3 Data Structure

5.3.1 `data` Namespace

`data` namespace includes objects for parsing, recording, selecting, and managing test data.

`Archive` Class

`data.Archive` is the object for storing and tracking the parsed test data of a single wafer. A `data.Archive` can be stored on disk in binary format and be loaded into memory while being requested by other objects or functions. `data.Archive` maintains a list of dies in a wafer with the following attributes

for each die: original measured values of each test item, normalized measured values of each test item, location with respect to a wafer map, test head touch down sequence, test head index (for multi-head test), bin number, and type (pass, fail, etc.).

Member functions:

`get_pass()` returns the indexes of pass dies.

`get_fail()` returns the indexes of fail dies.

`get_wafer(itemIndex)` returns a two-dimensional wafer map with test measurements of the test item specified by `itemIndex`.

`Item` Class

`data.Item` records properties related to a test item. It has the following attributes: title, spec limits, being a parametric test item or a function test item, test group, and test order.

Member functions:

`get_limit()` returns both the upper spec limit and the lower spec limit.

`Sample` Class

`data.Sample` records the indexes of dies sampled from a particular wafer, and the algorithm and configurations used for sampling. `data.Sample` also has attributes for controlling the sampling process. For example, `sampleByStep` indicates that dies are sampled based on probing step number (only applicable while

using multi-head tester), and `sampleGoodOnly` indicates that only good (pass) dies are sampled.

`Package` Class

`data.Package` is the object for storing a set of selected dies with measurements of selected test items. The possible scenarios for selecting test data are discussed in Section 5.2.2. `data.Package` also records the indexes of `data.Archive`'s of the selected dies.

Member functions:

`append(sample, archive)` appends the dies in `archive` based on the recoded indexes in `sample`.

`replace(start, value)` replaces measurements with `value` starting from the `start`-th die.

`remove(list)` removes a set of stored dies based on the indexes specified by `dList`.

`verticalCat(package)` vertically concatenates data in `package` to itself.

`horizontalCat(package)` horizontally concatenates data in `package` to itself.

`subset(dList, iList)` returns a new `data.Package` based on a set of dies specified by `dList` with the measurements of the test items specified by `iList`.

`Parser` Class

`data.Parser` read test data in a particular format defined by the provider of the test data. The details of parsing test data form the specified sources are discussed in Section 5.2.1. `data.Parser` has various attributes that can be configured for different formates of test data.

Member functions:

`load_data(file)` loads test data from the source specified by `file` and returns a `data.Archive`.

`load_item()` loads attributes of each test item and returns a set of `data.Item`'s where a `data.Item` records the attributes of a test item.

`Source` Class

`data.Source` is linked with a particular parser for the specified test data formats and sources. `data.Source` hides the details of accessing test data in different formats from the user through providing a universal API. The main object of `data.Source` is to support various scenarios of selecting test data based on a combination of multiple user specified configurations.

Member functions:

`load_data(filePtr)` returns a saved or a newly created `data.Archive` for the test data source specified by `filePtr` based on the description in Section 5.2.1.

`update(filePtr)` sets several attributes related to test items, such as test item type and spec limits, based on a subset of the test data specified by `filePtr`.

`examine_die(archive)` sets the type of each die stored in `archive` based on user defined configurations and spec limits.

`retrieve(filePtr, sampleNum, dieType, itemType)` returns a subset of test data by a `data.Package` based on the four arguments. The details are discussed in Section 5.2.2.

`generate_sample(aIndex, num, dieType)` samples dies with types specified by `dieType` from wafers (or `data.Archive`'s) specified by `aIndex` and samples only `num` dies per wafer.

`find_test_item(name)` finds all test items with `name` in the title.

5.3.2 `vp` Namespace

`vp` namespace includes objects and functions of the VP and JVP algorithms described in Section 2.3.1 and Section 2.3.2.

`Config` Class

`vp.Config` provides several configurations. `maxTolerance` and `maxIterNum` control the termination conditions of the M-FOCUSS solver used by VP and JVP. `sampleNum` and `clusterNum` define the number of sampled dies and the number of jointed test items, respectively.

`Model` Class

`vp.Model` records the prediction results of VP/JVP, such as `predictable` and `predictError`, which represent the predictability and the prediction error of a parametric test item, respectively.

`Learner` Class

`vp.Learner` is the main object for providing methods of the VP/JVP algorithm.

Member functions:

`train(inPackage)` explores the predictability of each parametric test item based on the given test data, `inPackage`.

`solve_vp(sample)` constructs a complete wafer map based on the sampled values specified by `sample`.

`predict(inPackage)` predicts errors and test escapes for the given test data, `inPackage`, based on the training results.

5.3.3 `gl` Namespace

`gl` namespace includes objects and functions of GL and WGL algorithms described in Section 2.2.5.

`Config` Class

`gl.Config` provides several configurations. `useWeight` controls the weight assignment of WGL function. `lambda` defines the λ value in Equation (2.13).

`Model` Class

`gl.Model` records the prediction results. `predictable` and `predictError` represent the predictability and the prediction error of a parametric test item, respectively. `alpha` records the explored linear correlations among test items.

`Learner` Class

`gl.Learner` is the main object for providing methods of GL/WGL algorithm.

Member functions:

`train(inPackage)` explores the predictability of each parametric test item based on the given test data, `inPackage`.

`solve_gl(inPackage, lambda)` solves the group lasso regression problem based on the given test data and λ, which are specified by `inPackage` and `lambda`, respectively.

`predict(inPackage)` predicts errors and test escapes for the given test data, `inPackage`, based on the training results.

5.4 Examples

5.4.1 VP Example

Figure 5.2 demonstrates an example of predicting spatial variations using the VP algorithm. This example includes five steps.

a) Specify the source and format of the target test data.

b) Select 500 dies for each wafer from the specified wafers to form a training set and a test set.

c) Perform the VP algorithm on the training set to find spatial predictable test items.

d) Perform the VP algorithm on the test set to simulate the test application.

e) Display the prediction results.

```
source = data.Source(parser, config);

trainSet = source.retrieve(trainFile, 500);
testSet = source.retrieve(testFile, 500);

vpLearner = vp.Learnel(config);
vpLearner.train(trainSet);
vpLearner.predict(testSet);

vpLearner.show();
```

Figure 5.2: An example of performing the VP algorithm.

5.4.2 WGL Example

Figure 5.3 demonstrates an example for performing TTR using the WGL algorithm through utilizing both spatial and inter-test-item correlations. This example includes eight steps.

a) Specify the source and format of the target test data.

b) Perform VP to derive spatial predictable test items.

c) Select only good dies from the specified wafers to form a training set.

d) Select all dies from the specified wafers to form a test set.

e) Assign the VP predictability as the weight of the WGL algorithm.

f) Perform the WGL algorithm on the training set to find predictable test items based on both spatial and inter-test-item correlations.

g) Simulate the test application.

h) Display the prediction results.

```
source = data.Source(parser, config);

trainSet = source.retrieve(trainFile, 500);
vpLearner = vp.Learnel(config);
vpLearner.train(trainSet);

trainSet = source.retrieve(trainFile, [], {'good'});
testSet = source.retrieve(testFile, [], {'all'});

config.useWeight = vpLearner.predictability;
wglLearner = gl.Learnel(config);

wglLearner.train(trainSet);
wglLearner.predict(testSet);

wglLearner.show();
```

Figure 5.3: An example of performing the WGL algorithm.

Chapter 6

Conclusions

This dissertation documents the efforts of defining and solving some problems for test data analytics. This chapter summarizes the key findings and contributions.

In the scope of exploring hidden patterns from test data, a new spatial variation prediction technique, Joint Virtual Probe, is proposed for jointly deriving spatial patterns of multiple test items. By simultaneously handling a large group of test items, JVP significantly reduces the overall runtime. And the prediction accuracy can also be improved due to JVP's implicit use of inter-test-item correlations in predicting spatial patterns. For investigating inter-test-item correlation, Weight Group Lasso, which can identify correlations among test items allows taking into account the distinct test time of each individual test item in the formulation as a weighted optimization problem. As a result, its solution would favor more costly test items for removal from the test program. Moreover, biclustering techniques are utilized to exploring patterns in the subsets of test data through conducting both item-to-item and die-to-die correlations.

The discovery of patterns and correlations hidden in the test data could help reduce test time and provide silicon characterization. A TTR methodology is proposed with supporting statistical regression tools that can exploit and utilize both spatial and inter-test-item correlations in test data. After learning the correlations in test data, some test items whose values can be predicted without measurement are identified for removal from the test program. A framework for characterizing systematic variations and failures through exploring the hidden spatial patterns of test data from multiple test stages has also been developed. A template matching technique exploits such spatial patterns to reveal the correlation between process variations and production failures.

A toolbox dedicated to test data analytics has been built for supporting the tasks mentioned above. The toolbox provides universal interface for various learning algorithms and test data from different sources. The toolbox will be released to public for non-commercial use.

Bibliography

[1] C.-K. Hsu, F. Lin, K.-T. Cheng, W. Zhang, X. Li, J. M. Carulli Jr., and K. M. Butler, *Test data analytics — exploring spatial and test-item correlations in production test data*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Sept., 2013.

[2] F. Lin, C.-K. Hsu, and K.-T. Cheng, *Feature engineering with canonical analysis for effective statistical tests screening test escapes*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct., 2014.

[3] F. Lin, C.-K. Hsu, and K.-T. Cheng, *AdaTest: an efficient statistical test framework for test escape screening*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct., 2015.

[4] F. Lin, C.-K. Hsu, A. G. Busetto, and K.-T. Cheng, *Pairwise proximity-based features for test escape screening*, in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design (ICCAD)*, pp. 300–306, Nov., 2015.

[5] S. Zhang, F. Lin, C.-K. Hsu, K.-T. Cheng, and H. Wang, *Joint virtual probe: Joint exploration of multiple test items' spatial patterns for efficient silicon characterization and test prediction*, in *Proc. Design, Automation, and Test in Europe (DATE)*, Mar., 2014.

[6] C.-K. Hsu, P. Sarson, and K.-T. Cheng, *Variation and Failure Characterization Through Pattern Classification of Test Data From Multiple Test Stages*. To be submitted to *IEEE Int'l Test Conf. (ITC)*, 2006.

[7] F. Lin, C.-K. Hsu, and K.-T. Cheng, *Learning from production test data: Correlation exploration and feature engineering*, in *IEEE Asian Test Symp. (ATS)*, pp. 236–241, Nov., 2014.

[8] F. Liu, *A general framework for spatial correlation modeling in VLSI design*, in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, June, 2007.

[9] X. Li, R. R. Rutenbar, and R. D. Blanton, *Virtual probe: A statistically optimal framework for minimum-cost silicon characterization of nanoscale integrated circuits*, in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design (ICCAD)*, Nov., 2009.

[10] N. Kupp, K. Huang, J. M. Carulli Jr., and Y. Makris, *Spatial correlation modeling for probe test cost reduction in RF devices*, in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design (ICCAD)*, pp. 23–29, Nov., 2012.

[11] W. Zhang, X. Li, and R. A. Rutenbar, *Bayesian virtual probe: Minimizing variation characterization cost for nanoscale IC technologies via Bayesian inference*, in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, June, 2010.

[12] W. Zhang, X. Li, E. Acar, F. Liu, and R. A. Rutenbar, *Multi-wafer virtual probe: Minimum-cost variation characterization by exploring wafer-to-wafer correlation*, in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design (ICCAD)*, pp. 47–54, Nov., 2010.

[13] K. Huang, N. Kupp, J. M. Carulli, and Y. Makris, *Handling discontinuous effects in modeling spatial correlation of wafer-level analog/RF tests*, in *Proc. Design, Automation, and Test in Europe (DATE)*, pp. 553–558, Mar., 2013.

[14] H. Gonçalves, X. Li, M. Correia, V. Tavares, J. Carulli Jr., and K. Butler, *A fast spatial variation modeling algorithm for efficient test cost reduction of analog/RF circuits*, in *Proc. Design, Automation, and Test in Europe (DATE)*, pp. 1042–1047, Mar., 2015.

[15] J. B. Brockman and S. W. Director, *Predictive subset testing: optimizing IC parametric performance testing for quality, cost, and yield*, *IEEE Trans. on Semiconductor Manufacturing* **2** (Aug., 1989) 140–113.

[16] M. Chen and A. Orailoglu, *Test cost minimization through adaptive test development*, in *Proc. IEEE Int'l Conf. on Computer Design (ICCD)*, Oct., 2008.

[17] S. Biswas and R. D. S. Blanton, *Test compaction for mixed-signal circuits using pass-fail test data*, in *Proc. IEEE VLSI Test Symp. (VTS)*, pp. 299–308, Apr., 2008.

[18] H.-G. Stratigopoulos, P. Drineas, M. Slamani, and Y. Makris, *RF specification test compaction using learning machines*, *IEEE Trans. on Very Large Scale Integration (VLSI) Systems* **18** (June, 2010) 998–1002.

[19] K. R. Gotkhindikar, W. Daasch, K. M. Butler, J. M. Carulli Jr., and A. Nahar, *Die-level adaptive test: Real-time test reordering and elimination*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Sept., 2011.

[20] E. Yilmaz and S. Ozev, *Adaptive multi-site testing for analog/mixed-signal circuits incorporating neighborhood information*, in *IEEE European Test Symp. (ETS)*, May, 2012.

[21] N. Akkouche, S. Mir, E. Simeu, and M. Slamani, *Analog/RF test ordering in the early stages of production testing*, in *Proc. IEEE VLSI Test Symp. (VTS)*, Apr., 2012.

[22] S. Devarakond, J. McCoy, A. Nahar, J. M. Carulli Jr., S. Bhattacharya, and A. Chatterjee, *Predicting die-level process variations from wafer test data for analog devices: A feasibility study*, in *Latin American Test Workshop (LATW)*, Apr., 2013.

[23] A. Ahmadi, K. Huang, A. Nahar, B. Orr, M. Pas, J. M. Carulli Jr., and Y. Makris, *Yield prognosis for fab-to-fab product migration*, in *Proc. IEEE VLSI Test Symp. (VTS)*, Apr., 2015.

[24] X. Li, W. Zhang, F. Wang, S. Sun, and C. Gu, *Efficient parametric yield estimation of analog/mixed-signal circuits via Bayesian model fusion*, in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design (ICCAD)*, pp. 627–634, Nov., 2012.

[25] S. Zhang, X. Li, R. D. Blanton, J. Machado da Silva, J. M. Carulli Jr., and K. M. Butler, *Bayesian model fusion: Enabling test cost reduction of analog/RF circuits via wafer-level spatial variation modeling*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct., 2014.

[26] C. Fang, F. Yang, X. Zeng, and X. Li, *BMF-BD: Bayesian model fusion on bernoulli distribution for efficient yield estimation of integrated circuits*, in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, June, 2014.

[27] A. Ahmadi, H.-G. Stratigopoulos, A. Nahar, B. Orr, M. Pas, and Y. Makris, *Yield forecasting in fab-to-fab production migration based on bayesian model fusion*, in *Proc. IEEE/ACM Int'l Conf. on Computer-Aided Design (ICCAD)*, pp. 9–14, Nov., 2015.

[28] L.-T. Wang, C.-W. Wu, and X. Wen, *VLSI Test Principles and Architectures.* Morgan Kaufmann, 2006.

[29] P. N. Variyam, S. Cherubal, and A. Chatterjee, *Prediction of analog performance parameters using fast transient testing*, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems* **21** (Mar., 2002) 349–361.

[30] S. S. Akbay and A. Chatterjee, *Fault-based alternate test of RF components*, in *Proc. IEEE Int'l Conf. on Computer Design (ICCD)*, pp. 518–525, Oct., 2007.

[31] R. Voorakaranam, S. S. Akbay, S. Bhattacharya, S. Cherubal, and A. Chatterjee, *Signature testing of analog and RF circuits: Algorithms and methodology*, *IEEE Trans. on Circuits and Systems* **54** (May, 2007) 1018–1031.

[32] D. Mannath, D. Webster, V. Montano-Martinez, D. Cohen, S. Kush, T. Ganesan, and A. Sontakke, *Structural approach for built-in tests in RF devices*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov., 2010.

[33] H. Ayari, F. Azais, S. Bernard, M. Comte, V. Kerzerho, O. Potin, and M. Renovell, *Making predictive analog/RF alternate test strategy independent of training set size*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov., 2012.

[34] B. Lee, L.-C. Wang, and M. Abadir, *Refined statistical static timing analysis through learning spatial delay correlations*, in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, July, 2006.

[35] E. J. Candès, J. Romberg, and T. Tao, *Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information*, *IEEE Trans. on Information Theory* **52** (Feb., 2006) 489–509.

[36] D. L. Donoho, *Compressed sensing*, *IEEE Trans. on Information Theory* **52** (Apr., 2006) 1289–1306.

[37] E. J. Candès, *Compressive sampling*, *Proc. the Int'l Congress of Mathematicians* **3** (Aug., 2006) 1433–1452.

[38] H.-M. Chang, K.-T. Cheng, W. Zhang, X. Li, and K. M. Butler, *Test cost reduction through performance prediction using virtual probe*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Sept., 2011.

[39] J. Chung, Y. Kim, and J. S. Yang, *3-D probe: Low-cost variation modeling using intertest-item correlations*, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems* **33** (Dec., 2014) 2005–2009.

[40] A. Afifi, S. May, and V. Clark, *Computer-Aided Multivariate Analysis*. Taylor & Francis, fourth ed., 2003.

[41] P. M. O'Neill, *Production multivariate outlier detection using principal components*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct., 2008.

[42] A. Bounceur, S. Mir, and H. G. Stratigopoulos, *Estimation of analog parametric test metrics using copulas*, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems* **30** (Sept., 2011) 1400–1410.

[43] N. Sumikawa, L. C. Wang, and M. S. Abadir, *A pattern mining framework for inter-wafer abnormality analysis*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Sept., 2013.

[44] I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh (Eds.), *Feature Extraction Foundations and Applications*. Springer, 2006.

[45] S. Krishnan and H. G. Kerkhoff, *Exploiting multiple mahalanobis distance metrics to screen outliers from analog product manufacturing test responses*, *IEEE Design & Test* **30** (June, 2013) 18–24.

[46] S. Biswas and R. D. S. Blanton, *Statistical test compaction using binary decision trees*, *IEEE Design & Test of Computers* **23** (June, 2006) 452–462.

[47] E. Yilmaz, S. Ozev, and K. M. Butler, *Adaptive test flow for mixed-signal/RF circuits using learned information from device under test*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov., 2010.

[48] E. Yilmaz, S. Ozev, and K. M. Butler, *Per-device adaptive test for analog/RF circuits using entropy-based process monitoring*, *IEEE Trans. on Very Large Scale Integration (VLSI) Systems* **21** (June, 2013) 1116–1128.

[49] F.-L. Chen and S.-F. Liu, *A neural-network approach to recognize defect spatial pattern in semiconductor fabrication*, *IEEE Trans. on Semiconductor Manufacturing* **13** (Aug., 2000) 366–373.

[50] T. Yuan, W. Kuo, and S. J. Bae, *Detection of spatial defect patterns generated in semiconductor fabrication processes*, *IEEE Trans. on Semiconductor Manufacturing* **24** (Aug., 2011) 392–403.

[51] M. P. L. Ooi, S. H. Kuan, Y. C. Kuang, H. Cheng, E. K. J. Sim, S. N. Demidenko, and C. W. K. Chan, *Identifying systematic failures on semiconductor wafers using ADCAS*, *IEEE Design & Test* **30** (Oct., 2013) 44–53.

[52] W. Zhang, X. Li, S. Saxena, A. Strojwas, and R. Rutenbar, *Automatic clustering of wafer spatial signatures*, in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, May, 2013.

[53] M.-J. Wu, J.-S. R. Jang, and J.-L. Chen, *Wafer map failure pattern recognition and similarity ranking for large-scale data sets*, *IEEE Trans. on Semiconductor Manufacturing* **28** (Feb., 2015) 1–12.

[54] F. Wang, W. Zhang, S. Sun, X. Li, and C. Gu, *Bayesian model fusion: Large-scale performance modeling of analog and mixed-signal circuits by reusing early-stage data*, in *Proc. IEEE/ACM Design Automation Conf. (DAC)*, May, 2013.

[55] S. Kang, S. Cho, D. An, and J. Rim, *Using wafer map features to better predict die-level failures in final test*, *IEEE Trans. on Semiconductor Manufacturing* **28** (Aug., 2015) 431–437.

[56] K. M. Butler, S. Subramaniam, A. Nahar, J. M. Carulli Jr., T. J. Anderson, and W. R. Daasch, *Successful development and implementation of statistical outlier techniques on 90nm and 65nm process driver devices*, in *Proc. IEEE Int'l Reliability Physics Symp.*, pp. 552–559, Mar., 2006.

[57] L. Fang, M. Lemnawar, and Y. Xing, *Cost effective outliers screening with moving limits and correlation testing for analogue ICs*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct., 2006.

[58] P. M. O'Neill, *Statistical test: A new paradigm to improve test effectiveness & efficiency*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Oct., 2007.

[59] A. Nahar, K. M. Butler, J. M. Carulli Jr., and C. Weinberger, *Quality improvement and cost reduction using statistical outlier methods*, in *Proc. IEEE Int'l Conf. on Computer Design (ICCD)*, Sept., 2009.

[60] N. Sumikawa, J. Tikkanen, L.-C. Wang, L. Winemberg, and M. S. Abadir, *Screening customer returns with multivariate test analysis*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov., 2012.

[61] H. H. Chen, R. Hsu, P. Yang, and J. J. Shyr, *Predicting system-level test and in-field customer failures using data mining*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Sept., 2013.

[62] R. Tibshirani, *Regression shrinkage and selection via the lasso*, *Jour. of Royal Statistical Society* **58** (Jan., 1996) 267–288.

[63] M. Yuan and Y. Lin, *Model selection and estimation in regression with grouped variables*, *Jour. of Royal Statistical Society* **68** (Feb., 2006) 49–67.

[64] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, *Sparse solutions to linear inverse problems with multiple measurement vectors*, *IEEE Trans. on Signal Processing* **53** (July, 2005) 2477–2488.

[65] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, *Applications of second-order cone programming*, *Linear Algebra and Its Applications* **284** (Nov., 1998) 193–228.

[66] D. M. Malioutov, M. Cetin, and A. S. Willsky, *Source localization by enforcing sparsity through a Laplacian prior: an SVD-based approach*, in *Proc. IEEE Workshop on Statistical Signal Processing*, pp. 573–576, IEEE, 2003.

[67] J. Chen and X. Huo, *Theoretical results on sparse representations of multiple-measurement vectors*, *IEEE Trans. on Signal Processing* **54** (Dec., 2006) 4634–4643.

[68] J. A. Tropp, *Algorithms for simultaneous sparse approximation. Part II: Convex relaxation*, *Signal Processing* **86** (Mar., 2006) 589–602.

[69] D. Baron, M. F. Duarte, M. B. Wakin, S. Sarvotham, and R. G. Baraniuk, *Distributed compressive sensing*, *arXiv preprint arXiv:0901.3403* (Jan., 2009).

[70] E. J. Marinissen, A. Singh, D. Glotter, M. Esposito, J. M. Carulli Jr., A. Nahar, K. M. Butler, D. Appello, and C. Portelli, *Adapting to adaptive testing*, in *Proc. Design, Automation, and Test in Europe (DATE)*, Mar., 2010.

[71] W. R. Daasch and R. Madge, *Variance reduction and outliers: Statistical analysis of semiconductor test data*, in *Proc. IEEE Int'l Test Conf. (ITC)*, Nov., 2005.

[72] S. Hochreiter, U. Bodenhofer, M. Heusel, A. Mayr, A. Mitterecker, A. Kasim, T. Khamiakova, S. Van Sanden, D. Lin, W. Talloen, *et. al.*, *FABIA: factor analysis for bicluster acquisition*, *Bioinformatics* **26** (Apr., 2010) 1520–1527.

[73] S. C. Madeira and A. L. Oliveira, *Biclustering algorithms for biological data analysis: A survey*, *IEEE/ACM Trans. on Computational Biology and Bioinformatics* **1** (Jan., 2004) 24–45.

[74] J. Lewis, *Fast normalized cross-correlation*, in *Vision interface*, vol. 10, pp. 120–123, 1995.

[75] K. Briechle and U. D. Hanebeck, *Template matching using fast normalized cross correlation*, in *Proc. SPIE Optical Pattern Recognition XII*, vol. 4387, pp. 95–102, Mar., 2001.

[76] S.-D. Wei and S.-H. Lai, *Fast template matching based on normalized cross correlation with adaptive multilevel winner update*, *IEEE Trans. on VLSI Image Processing* **17** (Nov., 2008) 2227–2235.

[77] J. K. Gupta, S. Singh, and N. K. Verma, *MTBA: Matlab toolbox for biclustering analysis*, in *IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions*, pp. 94–97, July, 2013.

[78] C.-C. Chang and C.-J. Lin, *LIBSVM: A library for support vector machines*, *ACM Trans. on Intelligent Systems and Technology* **2** (2011) 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[79] M. ApS, *The MOSEK optimization toolbox for MATLAB manual. Version 7.1 (Revision 28).*, 2015.