

2014

# GEOMETRIC OPTIMIZATION IN SOME PROXIMITY AND BIOINFORMATICS PROBLEMS

Satish Chandra Panigrahi  
*University of Windsor*

Follow this and additional works at: <http://scholar.uwindsor.ca/etd>

---

## Recommended Citation

Panigrahi, Satish Chandra, "GEOMETRIC OPTIMIZATION IN SOME PROXIMITY AND BIOINFORMATICS PROBLEMS" (2014). *Electronic Theses and Dissertations*. Paper 5198.

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email ([scholarship@uwindsor.ca](mailto:scholarship@uwindsor.ca)) or by telephone at 519-253-3000ext. 3208.

GEOMETRIC OPTIMIZATION IN SOME PROXIMITY AND  
BIOINFORMATICS PROBLEMS

by

Satish Chandra Panigrahi

A Dissertation  
Submitted to the Faculty of Graduate Studies  
through School of Computer Science  
in Partial Fulfillment of the Requirements for  
the Degree of Doctor of Philosophy  
at the University of Windsor

Windsor, Ontario, Canada

2014

© 2014 Satish Chandra Panigrahi

Geometric optimization in some proximity and bioinformatics problems

by

Satish Chandra Panigrahi

APPROVED BY:

---

A. Maheshwari, External Examiner  
Carleton University

---

M. Hlynka  
Department of Mathematics and Statistics

---

D. Wu  
School of Computer Science

---

A. Jaekel  
School of Computer Science

---

A. Mukhopadhyay, Advisor  
School of Computer Science

5 September, 2014

# Declaration of Co-Authorship / Previous Publication

## I. Co-Authorship Declaration

I hereby declare that this dissertation incorporates material that is result of joint research, as follows:

This dissertation also incorporates the outcome of a joint research undertaken in collaboration with Dr. Md. Shfiul Alam under the supervision of my supervisor Dr. A. Mukhopadhyay. The collaboration is covered in Chapter 3 of the dissertation. In all cases, the key ideas, primary contributions, experimental designs, data analysis and interpretation, were performed by the author, and the contribution of co-author Dr. Md. Shafiul Alam was primarily through the discussion of technical content and time complexity during the implementation.

I am aware of the University of Windsor Senate Policy on Authorship and I certify that I have properly acknowledged the contribution of other researchers to my dissertation, and have obtained written permission from each of the co-authors to include the above materials in my dissertation.

I certify that, with the above qualification, this dissertation, and the research to which it refers, is the product of my own work.

## II. Declaration of Previous Publication

This dissertation includes 3 original papers that have been previously published in peer reviewed journal and conferences, as follows:

| Dissertation Chapter | Publication title/full citation  | Publication status |
|----------------------|--|--------------------|
| Chapter 2            | Asish Mukhopadhyay, Satish Panigrahi. “All-maximum and all-minimum problems under some measures”; Journal of Discrete Algorithms, Volume 21, Pages 18 - 31, 2013   | published          |
| Chapter 3            | Satish Panigrahi, Md. Shafiqul Alam, and Asish Mukhopadhyay. “An incremental linear programming based tool for analyzing gene expression data”; In Beniamino Murgante, Sanjay Misra, Maurizio Carlini, CarmeloM. Torre, Hong-Quang Nguyen, David Taniar, BernadyO. Apduhan, and Osvaldo Gervasi (Eds.), Computational Science and Its Applications ICCSA 2013, volume 7975 of Lecture Notes in Computer Science, pages 48 - 64, 2013 | published          |
| Chapter 5            | Satish Panigrahi and Asish Mukhopadhyay. “An eigendecomposition method for protein structure alignment”, In M. Basu, Y. Pan, and J. Wang (Eds.), Bioinformatics Research and Applications, 10th International Symposium, ISBRA 2014, volume 8492 of Lecture Notes in Bioinformatics, pages 24 - 37, 2014   | published          |

I certify that I have obtained a written permission from the copyright owner(s) to include the above published material(s) in my dissertation. I certify that the above material describes work completed during my registration as graduate student at the University of Windsor.

I declare that, to the best of my knowledge, my dissertation does not infringe upon anyones copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in

my dissertation, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my dissertation.

I declare that this is a true copy of my dissertation, including any final revisions, as approved by my dissertation committee and the Graduate Studies office, and that this dissertation has not been submitted for a higher degree to any other University or Institution.

# Abstract

The theme of this dissertation is geometric optimization and its applications. We study geometric proximity problems and several bioinformatics problems with a geometric content, requiring the use of geometric optimization tools.

We have investigated the following type of proximity problems. Given a point-set in a plane with  $n$  distinct points, for each point in the set find a pair of points from the remaining points in the set such that the three points either maximize or minimize some geometric measure defined on these. The measures include (a) sum and product; (b) difference; (c) line-distance; (d) triangle area; (e) triangle perimeter; (f) circumcircle-radius; and (g) triangle-distance in three dimensions.

We have also studied the application of a linear time incremental geometric algorithm to test the linear separability of a set of blue points from a set of red points, in two and three-dimensional euclidean spaces. We have used this geometric separability tool on 4 different gene expression data-sets, enumerating gene-pairs and gene-triplets that are linearly separable. Pushing on further, we have exploited this novel tool to identify some bio-marker genes for a classifier. The gene selection method proposed in the dissertation exhibits good classification accuracy as compared to other known feature (or gene) selection methods such as  $t$ -values, FCS (Fisher Criterion Score) and SAM (Significance Analysis of Microarrays). Continuing this line of investigation further, we have also designed an efficient algorithm to find the minimum number

of outliers when the red and blue point sets are not fully linearly separable.

We have also explored the applicability of geometric optimization techniques to the problem of protein structure similarity. We have come up with two new algorithms, *EDAlign<sub>res</sub>* and *EDAlign<sub>sse</sub>*, for pairwise protein structure alignment. *EDAlign<sub>res</sub>* identifies the best structural alignment of two equal length proteins by refining the correspondence obtained from eigendecomposition and to maximize the similarity measure for the refined correspondence. *EDAlign<sub>sse</sub>*, on the other hand, does not require the input proteins to be of equal length. These have been fully implemented and tested against well-established protein alignment programs.



*Dedicated at the feet of my Parents*

Mother: Jayanti Panigrahi

Father: Trilochan Panigrahi

Also special thanks to my

Wife: Sasmita Sahu

Son: Soham Panigrahi

# Acknowledgements

I consider myself very fortunate to have Dr. Asish Mukhopadhyay as my supervisor. I would like to express my sincere gratitude to Dr. Mukhopadhyay, for his constant guidance and extensive support throughout my doctoral research. This work could not have been achieved without his continuous encouragement, astute advices, suggestions and cooperation. Words cannot thank him enough.

I would like to thank Dr. Anil Maheshwari for his kind acceptance to be the examiner of my dissertation defense.

I would like to thank members of my Ph.D. thesis committee, Dr. Myron Hlynka, Dr. Arunita Jaekel and Dr. Dan Wu for their instructive advices, suggestions and comments.

I would like to thank my wife, Sasmita Sahu, for her endless love and care and for always being there for me.

Last but not the least, I would like to thank my parents, brother and sister for their love, understanding and cooperation.

# Table of Contents

|   | Page |
|---|------|
| <b>Declaration of Co-Authorship / Previous Publication</b> . . . . .        | iii  |
| <b>Abstract</b> . . . . .   | vi   |
| <b>Acknowledgements</b> . . . . .   | ix   |
| <b>List of Tables</b> . . . . .   | xv   |
| <b>List of Figures</b> . . . . .  | xvi  |
| <b>Chapters</b>   |      |
| <b>1 Introduction</b> . . . . .   | 1    |
| 1.1 Outline of Dissertation . . . . .                                       | 2    |
| 1.2 Background and Motivation . . . . .                                     | 4    |
| 1.2.1 Geometric proximity . . . . .   | 4    |
| 1.2.2 Geometric separability and gene expression data sets . . . . .        | 6    |
| 1.2.3 Geometric separability with few violated constraints . . . . .        | 9    |
| 1.2.4 Point pattern matching and protein structure similarity . . . . .     | 10   |
| 1.3 Contributions . . . . .   | 15   |
| <b>2 All-maximum and all-minimum problems under some measures</b> . . . . . | 19   |
| 2.1 Overview . . . . .  | 19   |
| 2.2 Introduction . . . . .  | 19   |
| 2.2.1 Motivation . . . . .  | 20   |

|          |  |           |
|----------|--|-----------|
| 2.2.2    | Prior Work . . . . .   | 21        |
| 2.2.3    | Our results . . . . .  | 22        |
| 2.3      | Sum and Product Measures . . . . .   | 23        |
| 2.4      | Difference Measure . . . . .   | 24        |
| 2.5      | Line Distance Measure . . . . .  | 27        |
| 2.6      | Triangle Area Measure . . . . .  | 30        |
| 2.6.1    | Maximum Area Triangle . . . . .  | 31        |
| 2.6.2    | Minimum Area Triangle . . . . .  | 35        |
| 2.7      | Triangle Perimeter Measure . . . . .   | 37        |
| 2.7.1    | Maximum perimeter . . . . .  | 38        |
| 2.7.2    | Minimum perimeter . . . . .  | 41        |
| 2.8      | Circumcircle radius measure . . . . .  | 43        |
| 2.9      | Triangle Distance Measure . . . . .  | 45        |
| 2.9.1    | Characterizing farthest triangles . . . . .  | 46        |
| 2.9.2    | Algorithm . . . . .  | 55        |
| 2.10     | Summary . . . . .  | 58        |
| <b>3</b> | <b>An Incremental Linear Programming Based Tool for Analyzing<br/>Gene Expression Data . . . . .</b> | <b>60</b> |
| 3.1      | Overview . . . . .   | 60        |
| 3.2      | Introduction . . . . .   | 61        |
| 3.3      | LP Formulation of Separability . . . . .   | 65        |

|          |  |            |
|----------|--|------------|
| 3.4      | Offline Approach . . . . .   | 70         |
| 3.5      | Incremental Approach . . . . .   | 73         |
| 3.5.1    | Incremental approach-2d . . . . .  | 73         |
| 3.5.2    | Incremental approach-3d . . . . .  | 76         |
| 3.5.3    | Linear programming formulation of Unger and Chor’s incremental algorithm . . . . .               | 78         |
| 3.6      | Gene Selection . . . . .   | 81         |
| 3.6.1    | Background . . . . .   | 81         |
| 3.7      | A new methodology for gene selection . . . . .   | 83         |
| 3.7.1    | Coarse Filtration . . . . .  | 84         |
| 3.7.2    | Fine Filtration . . . . .  | 84         |
| 3.8      | Results and Discussions . . . . .  | 86         |
| 3.9      | Summary . . . . .  | 102        |
| <b>4</b> | <b>On the linear separability of a bichromatic point set with violated constraints . . . . .</b> | <b>104</b> |
| 4.1      | Overview . . . . .   | 104        |
| 4.2      | Introduction . . . . .   | 104        |
| 4.2.1    | Problem statement . . . . .  | 104        |
| 4.2.2    | Motivation . . . . .   | 105        |
| 4.2.3    | Prior work . . . . .   | 106        |
| 4.2.4    | Our contributions . . . . .  | 107        |

|          |  |            |
|----------|--|------------|
| 4.3      | Preliminaries . . . . .  | 108        |
| 4.3.1    | Megiddo’s algorithm . . . . .  | 108        |
| 4.4      | Proposed Algorithm . . . . .   | 110        |
| 4.4.1    | Construct the <i>tentative-set</i> $T$ . . . . .   | 112        |
| 4.4.2    | Explore the arrangement of the lines in the <i>tentative-set</i> , $T$ , for<br>outlier sets . . . . . | 113        |
| 4.4.3    | Validate the outlier set . . . . .   | 116        |
| 4.4.4    | Time Complexity . . . . .  | 118        |
| 4.5      | Experiment on gene expression datasets . . . . .   | 119        |
| 4.6      | Summary . . . . .  | 123        |
| <b>5</b> | <b>An eigendecomposition method for protein structure alignment .</b>                                  | <b>124</b> |
| 5.1      | Overview . . . . .   | 124        |
| 5.2      | Introduction . . . . .   | 125        |
| 5.3      | Preliminaries . . . . .  | 129        |
| 5.3.1    | Notations and Definitions . . . . .  | 129        |
| 5.3.2    | Similarity measures . . . . .  | 130        |
| 5.3.3    | Umeyama’s matrix eigendecomposition method . . . . .   | 132        |
| 5.4      | Methods . . . . .  | 136        |
| 5.5      | Results and Discussions . . . . .  | 142        |
| 5.6      | Summary . . . . .  | 148        |
| <b>6</b> | <b>Conclusions and Future works . . . . .</b>  | <b>150</b> |

|  |     |
|--|-----|
| <b>References</b> . . . . .                              | 154 |
| <b>Appendices</b>  |     |
| <b>A Offline implementation</b> . . . . .                | 177 |
| A.1 Code for linearly separating gene pairs . . . . .    | 177 |
| A.2 Code for linearly separating gene triplets . . . . . | 178 |
| A.3 Data Sets . . . . .                                  | 179 |
| <b>B A sample PDB file</b> . . . . .                     | 182 |
| <b>C Co-authors' approval letters</b> . . . . .          | 197 |
| <b>VITA AUCTORIS</b> . . . . .                           | 199 |

# List of Tables

|     |   |     |
|-----|---|-----|
| 1.1 | Selected Protein Data Bank Record Types . . . . .                               | 16  |
| 2.1 | Our Results . . . . .   | 22  |
| 3.1 | Five Gene Expression Datasets . . . . .   | 86  |
| 3.2 | 2-D and 3-D Separability Test with Runtime . . . . .                            | 88  |
| 3.3 | Top Ten Significant Genes based upon P-values . . . . .                         | 103 |
| 4.1 | Five gene expression datasets before and after pruning . . . . .                | 119 |
| 4.2 | Almost linear separability for $k = 0, 1, \dots, 10$ . . . . .                  | 121 |
| 5.1 | Pairwise structural alignment of equal length proteins . . . . .                | 143 |
| 5.2 | NMR models 1m2f_A_1-1mef_A_25, compared to an average model<br>1m2e_A . . . . . | 145 |
| 5.3 | Pairwise structural alignment of unequal length proteins . . . . .              | 146 |



# List of Figures

|     |   |    |
|-----|---|----|
| 1.1 | The United States partitioned by closest national park [Figure from [1]]  | 6  |
| 1.2 | Steps followed in microarray experiment [Figure from [2]] . . . . .   | 7  |
| 1.3 | Geometric separability on expression profiles of lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML) samples [Figure taken from [3]]  | 8  |
| 1.4 | Geometric separability on expression profiles of lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML) samples with a violated constraint (e.g. red dot within a blue circle) [Figure taken from [3]] .   | 9  |
| 1.5 | Geometric reduction of protein structure similarity as a point pattern matching . . . . .   | 10 |
| 1.6 | Structure superposition of LTP1 from maize (PDB code 1MZL [4]) and LTP1 from rice (PDB code 1RZL [5]). (a) cartoon representation of 1MZL (b) ribbon representation of 1MZL (c) cartoon representation of 1RZL (d) ribbon representation of 1RZL (e) superposition of 1MZL and 1RZL (cartoon) (f) superposition of 1MZL and 1RZL (ribbon) . | 12 |
| 1.7 | The 20 amino acids [Figure taken from [6]] . . . . .  | 14 |
| 1.8 | The 20 amino acids as sidechain [Figure taken from [7]] . . . . .   | 14 |
| 2.1 | <i>A farthest line from <math>p_j</math>, incident on <math>p_i</math></i> . . . . .  | 28 |
| 2.2 | <i>A closest line from <math>p_j</math>, incident on <math>p_i</math></i> . . . . .   | 29 |

|      |  |    |
|------|--|----|
| 2.3  | When $p_k$ is an internal vertex of the convex hull, $CH(P)$ . . . . .   | 32 |
| 2.4  | Convex hull and its corresponding ray diagram . . . . .  | 32 |
| 2.5  | Similar triangles $\triangle a'b'p'_i$ and $\triangle abp_i$ . . . . .   | 36 |
| 2.6  | Walking the lower part of $\text{zone}(\ell)$ . . . . .  | 38 |
| 2.7  | A maximum perimeter triangle rooted at $p_i$ has the other two points<br>on $CH(P)$ . . . . .                                | 39 |
| 2.8  | Diameter of the convex figure is equal to the maximum perimeter tri-<br>angle rooted at $p_i$ . . . . .                      | 40 |
| 2.9  | The perimeter of $\triangle(p_i, p_j, p_k) > \mathcal{P}_i$ . . . . .  | 41 |
| 2.10 | (a) Inversion transformation (b) Transforming a circle $r = d \cos \theta$ to a<br>line $1/d = r' \cos \theta$ . . . . .     | 45 |
| 2.11 | <i>Type A distance</i> . . . . .   | 46 |
| 2.12 | <i>Type B distance</i> . . . . .   | 47 |
| 2.13 | <i>Type C distance</i> . . . . .   | 47 |
| 2.14 | <i>If a type A triangle is not a convex hull facet</i> . . . . .   | 51 |
| 3.1  | <i>A separating line <math>H</math> in primal space is a feasible solution <math>H^*</math> in dual<br/>space.</i> . . . . . | 68 |
| 3.2  | <i>A counterexample: black circles represent red points, white ones blue.</i> .  | 69 |
| 3.3  | (a) Pruning Constraints (b) Testing Feasibility . . . . .  | 71 |

|      |  |     |
|------|--|-----|
| 3.4  | <i>Updation of min-curve (a) addition of line <math>l_1</math> (b) test line continues to pass through feasible region on updating of min-curve (c) addition of line <math>l_2</math> (d) test line goes out of feasible region on updating of min-curve</i> | 74  |
| 3.5  | Linear programming formulation of ‘180 strict containment condition’<br>(a) construction of vectors (b) projection of the vectors onto unit circle<br>(c) mapping of points on the unit circle to dual plane . . . . .                                       | 80  |
| 3.6  | <i>10 Fold cross validation on gene expression data-set . . . . .</i>  | 92  |
| 3.7  | Accuracy vs Feature Space (Leukemia) . . . . .   | 94  |
| 3.8  | Accuracy vs Feature Space (SRBCT) . . . . .  | 95  |
| 3.9  | Accuracy vs Feature Space (Lung Cancer) . . . . .  | 96  |
| 3.10 | Accuracy vs Feature Space (Breast Cancer) . . . . .  | 97  |
| 3.11 | Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - Leukemia . . . . .   | 98  |
| 3.12 | Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - SRBCT . . . . .  | 99  |
| 3.13 | Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - Lung Cancer . . . . .  | 100 |
| 3.14 | Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - Breast Cancer . . . . .  | 101 |

|     |   |     |
|-----|---|-----|
| 4.1 | Classification of red set (dots (primal) or lines (dual)) from blue set (solid dots (primal) or dark lines (dual)). A separating line $l$ in the primal space has a feasible point $l^*$ in the dual space. Two points in primal (lines in dual) are misclassified. . . . .   | 109 |
| 4.2 | Eccentric-lines from max and min curve. In all cases $slope(l_{min}^l) \geq slope(l_{max}^l)$ and $slope(l_{max}^r) \geq slope(l_{min}^r)$ . . . . .  | 111 |
| 4.3 | The levels to the edges on the line $l_1^{R*}$ of arrangement. . . . .  | 115 |
| 4.4 | Percentage of linear separable pairs vs number of outliers ( $k$ ) . . . . .  | 121 |
| 4.5 | Rate of growth of linear separable pairs with increase in $k$ . . . . .   | 122 |
| 5.1 | Structural Alignment of 1ffx with 1aep . . . . .  | 147 |
| 5.2 | $EDAlign_{sse}$ on difficult alignment (a) $EDAlign_{sse}$ : cRMSD 1.94 - 20 and TM-score* 0.81 TM-align: cRMSD 4.2 - 68 and TM-score 0.42 SuperPose: cRMSD 13.21 - 72 and TM-score 0.07 (b) $EDAlign_{sse}$ : cRMSD 2.42 - 16 and TM-score* 0.77 TM-align: cRMSD 4.23 - 63 and TM-score 0.31 SuperPose: cRMSD 13.77 - 111 and TM-score 0.2 . . . | 148 |

# Chapter 1

## Introduction

Computational geometry is now a well-established discipline, dealing with problems emerging from the applications of geometric principles to objects such as points, lines, polygons, to name a few. Its origin can be traced to the publication of a thesis by Shamos [8] that established a connection between computing and geometry. In retrospect, its roots lie in the branch of mathematics that deals with the measurement of the shape, size and relative position of geometric objects and properties of the space in which these are embedded.

Computational geometry has great practical importance with a large number of applications to different problems related to geometric optimization, including facility location, proximity problems, statistical estimators and metrology, placement and intersection of polygons and polyhedra, ray shooting and other query-type problems [9]. Geometric optimization problems involve a constant number of variables with large number of constraints induced by the collection of geometric objects. One approach to such problems is to exploit the geometric nature of the problem. Much work has been done on geometric optimization problems and its applications, tools and techniques

to other areas, offering scope for exploring exciting problems. Thus my dissertation includes (a) problems on geometric proximity, which has applications that include object classification in pattern recognition, computer graphics, geographic information systems, and robotics; (b) application of linear separability to gene expression analysis which has an important application to the discovery of bio-markers, leading to effective diagnosis and treatment of various diseases; (c) protein structure similarity, which is useful for understanding biological functions of proteins and their evolutionary relationships.

## 1.1 Outline of Dissertation

The primary motivation of this dissertation is the application of geometric optimization tools to some proximity and bioinformatics problems. We address problems and their applications that arise in the context of proximity for a given point set or a pair of point sets. The objective is to identify subset(s) of a point set or a pair of point sets having desired properties. As an application to bioinformatics we use linear geometric separability to extract suitable genes from gene expression data which can be used for classification purposes. Another interesting application of geometric optimization is the study of protein structure similarity, which is an active and promising area of research in bioinformatics.

The Introductory chapter includes background study to establish the importance of

each problem addressed in the dissertation.

In Chapter 2, we investigate the following type of proximity problems: given a set of  $n$  points in the plane  $P = \{p_1, p_2, p_3, \dots, p_n\}$ , for each point  $p_i$  find a pair  $\{p_j, p_k\}$ , where  $i \neq j, i \neq k, j \neq k$ , such that a measure  $\mathcal{M}$  defined on the triplet of points  $\{p_i, p_j, p_k\}$  is maximized or minimized. We also discuss the all-farthest triangle problem in the triangle-distance measure when  $P$  is a set of points in 3 dimensions.

In Chapter 3, we discuss a new profiling tool based on linear programming. Given gene expression data from two subclasses of the same disease (e.g. leukemia), we are able to determine efficiently if the samples are linearly separable with respect to triplets of genes. We have used this geometric tool to propose an effective gene selection strategy.

In Chapter 4, we present an efficient algorithm to determine when two point sets are not linearly separable. In the presence of a few outliers, say  $k$  (or violated constraints), we present an output sensitive  $O(nk^2)$  time algorithm, where  $n$  is the total number of data points or samples. It works better than known algorithms by Everett et al. [10], Matausek [11] and Chan [12] when  $k = o(\log^2 n)$ .

In Chapter 5, we examine the application of geometric optimization to the problem of

protein structure similarity. The alignment of two protein structures is a fundamental problem in structural bioinformatics. Their structural similarity carries with it the connotation of similar functional behavior that could be exploited in various applications. Thus the structural similarity of a new protein with unknown functionalities and a protein with known functionalities could reveal some common behavior.

In Chapter 6, we summarize our results, provide a list of open problems and suggest avenues for future work.

## **1.2 Background and Motivation**

### **1.2.1 Geometric proximity**

Proximity problems in computational geometry involve computation of distances between geometric objects. Typically, such problems require the construction of geometric structures like Voronoi diagram, Delaunay triangulation and related graph structures such as the relative neighborhood graph, using suitable distance metrics. Algorithms for geometric proximity problems also has applications to nearest neighbor searching as well as range searching [13].

The motivation of such problems lies in various application areas that include pattern recognition [14], computer graphics [15,16], image processing [17], operations research,



statistics [18, 19], computer-aided design and robotics [20]. Geometric proximity also has applications to some geometric optimization problems in manufacturing, such as wire layout [21], cutting stock [22] and facility location [23].

In a typical proximity problem a finite point set is provided as input and the objective is to find a subset of this point set having some desired properties. For example given a set of  $n$  points in the plane  $P = \{p_1, p_2, p_3, \dots, p_n\}$ , for each point  $p_i$  find a pair  $\{p_j, p_k\}$ , where  $i \neq j \neq k$ , such that a measure  $\mathcal{M}$  defined on the triplet of points  $\{p_i, p_j, p_k\}$  is maximized or minimized. A more natural version of this type of problem, studied by Barquet et al. [24], is the 2-point site Voronoi diagram under different measures.

In view of the importance of the problem, a number of variations have been studied by different researchers, giving rise to an extensive literature. [24–29].

In Figure 1.1 an interesting application of the Voronoi diagram data structure is shown. The United States is home to 59 national parks and the figure shows the area closest to each national park.

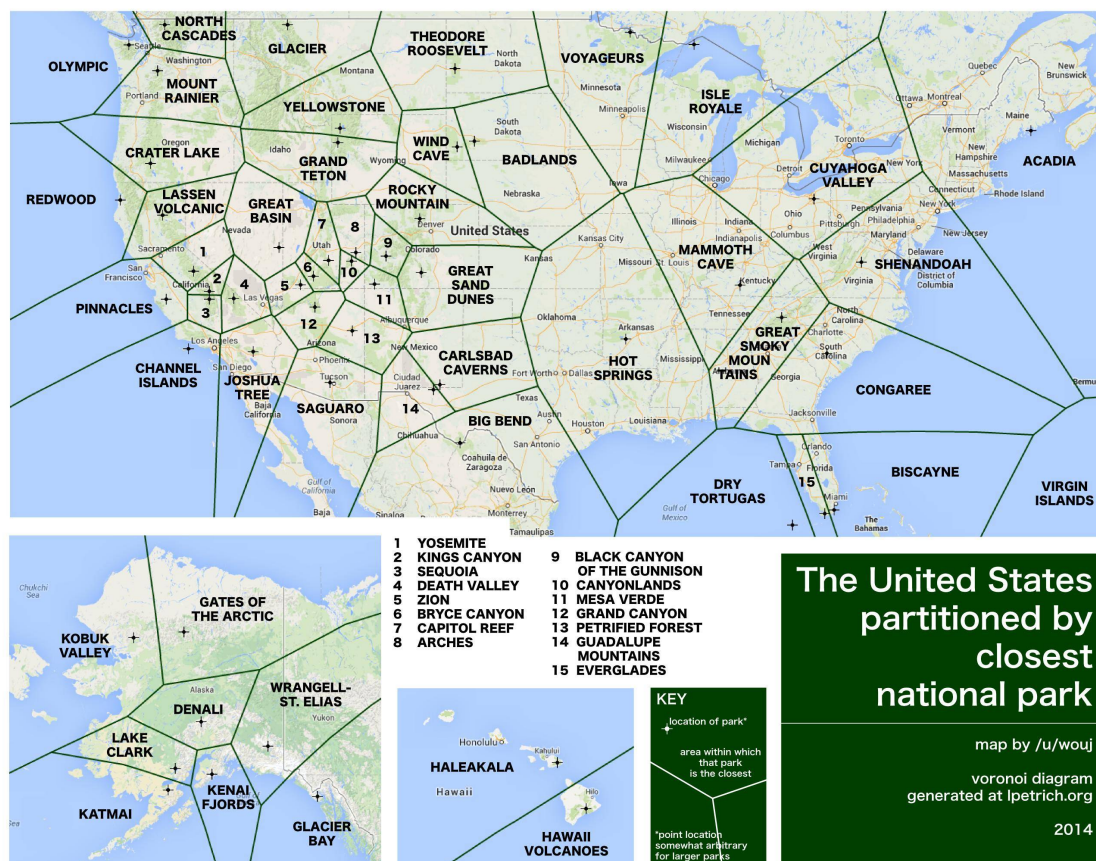


Figure 1.1: The United States partitioned by closest national park [Figure from [1]]

## 1.2.2 Geometric separability and gene expression data sets

Classification is an important and significant tool for biologists to extract information from gene expression data sets. One of the conventional approaches to learning a new object or phenomenon is to look into the features that describe it. Taking note of this approach, formulating the analysis of gene expression data sets in geometric setting is an important step for identifying bio-marker genes, leading to effective diagnosis and treatment of various diseases [30–34].

The transcriptome, or mRNA expressed by the genome, reflects the activity of all genes within a cell. The quantitative measure of mRNA concentration, known as expression level, in a cell can be obtained by microarray technologies. In Figure 1.2 we explain how microarrays are used to measure expression levels of mRNA in cells

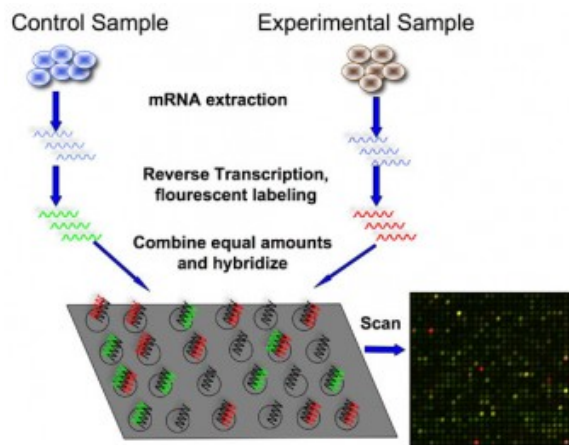


Figure 1.2: Steps followed in microarray experiment [Figure from [2]]

1. extract mRNA from samples (e.g. sample from cancer cell and normal cell)
2. make labelled cDNA through reverse transcription
3. mix samples and hybridize to cDNA microarray
4. wash to remove non-specific bindings
5. scan and calculate expression levels of mRNA

For more details on microarrays refer to Riva et al. [35].

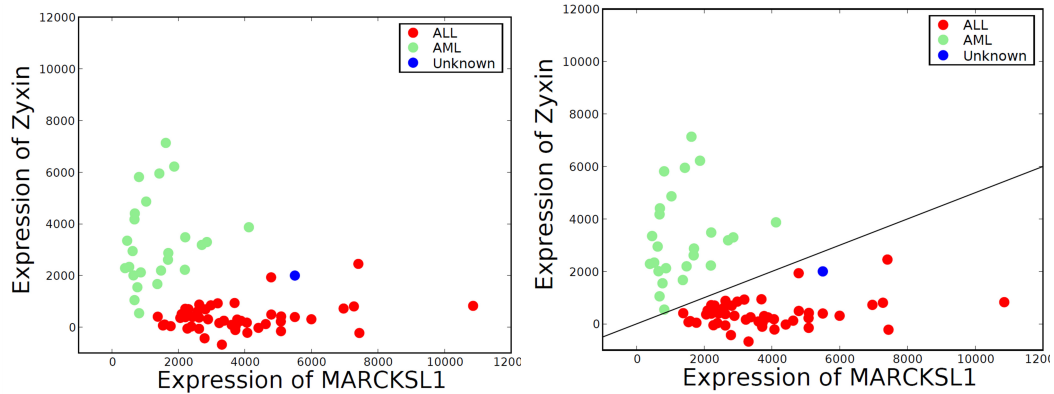


Figure 1.3: Geometric separability on expression profiles of lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML) samples [Figure taken from [3]]

Unger and Chor [31] studied linear separability on 10 different publicly available gene expression data sets and observed that 7 out of 10 are highly separable. The term linear separability means two classes are completely separable by a line in a two-dimensional Euclidean space. Each sample is a point in Euclidean space whose coordinates are expression values of pair of genes. The concept can be extended to any dimensions. In this dissertation we propose a geometric tool for linear separability which is used to achieve a larger objective of identifying bio-marker genes for an efficient classifier. We also study the linear separability of gene triplets, which was left as an open problem by Unger and Chor [31]

Figure 1.3 shows the linear separability of the gene pairs MARKSL1 and Zyxin. The

figure shows two-dimensional expression profiles of lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML) samples. Each dimension corresponds to the measured mRNA expression level of a given gene. The separating line can be used as a classifier to determine whether an unknown sample belongs to ALL or AML.

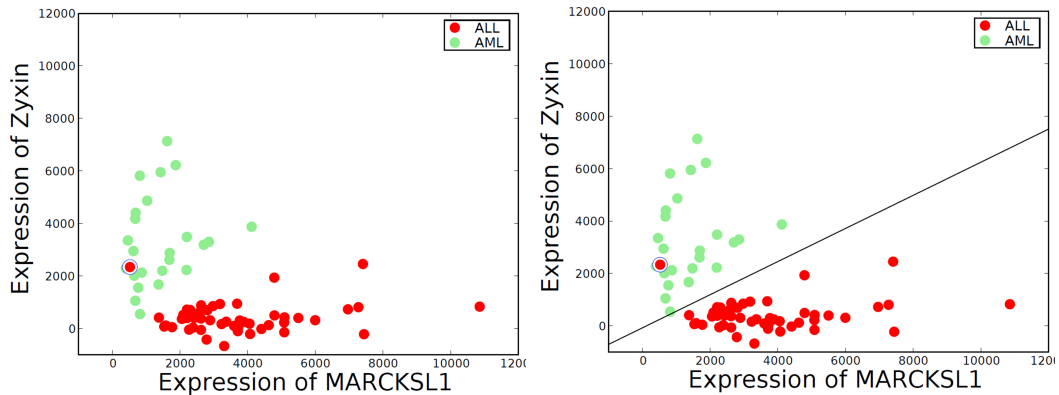


Figure 1.4: Geometric separability on expression profiles of lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML) samples with a violated constraint (e.g. red dot within a blue circle) [Figure taken from [3]]

### 1.2.3 Geometric separability with few violated constraints

More often than not data is noisy. This motivates us to study almost linear separability. The term “almost linear separability” means a two class point set (e.g. in the Figure 1.4, ALL belongs to one class where as AML belongs to other) is linearly separable for all but  $k$  of given points. The parameter  $k$  is the measure of number of outliers present in the point set.

Figure 1.4 illustrates an example of almost linear separability for a given gene pair where as each co-ordinate represents expression levels of the pair of gene taken from a data set. The outlier is represented as red dot within a blue circle. Identifying and pruning of outliers is an oft-recurring problem and has attracted much attention from various researches [10–12], and is also studied in this dissertation.

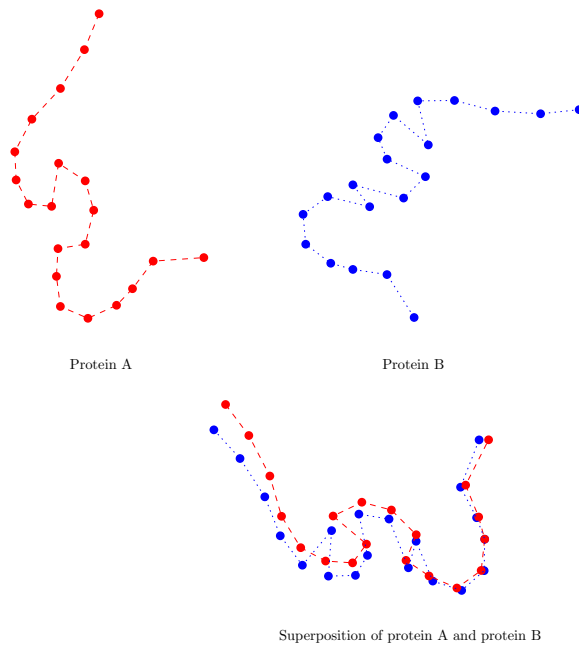


Figure 1.5: Geometric reduction of protein structure similarity as a point pattern matching

### 1.2.4 Point pattern matching and protein structure similarity

The study of protein structure similarity is a very promising area of research. It is important to our understanding of the biological activities of proteins. This includes

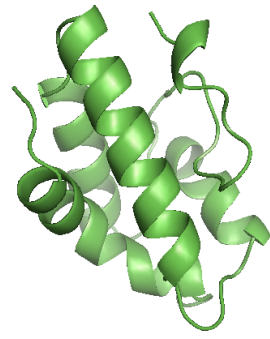
(a) finding homology between distantly related proteins; (b) inferring functional properties of an unannotated protein; (c) evaluating accuracy of protein folding algorithms.

From a geometric optimization perspective, the protein structure alignment problem can be viewed as a point pattern matching problem in three-dimensional space (see Figure 1.5). Each protein is considered as a collection of points in three dimensional space, where the points represent the co-ordinate of  $\alpha$ -carbon atoms along the backbone of the protein chain. Figure 1.6 shows structural superposition lipid transfer proteins (LPT1) of rice and maize. Lipid transfer proteins are believed to participate in membrane biogenesis and regulation of the intracellular fatty acid pools [36].

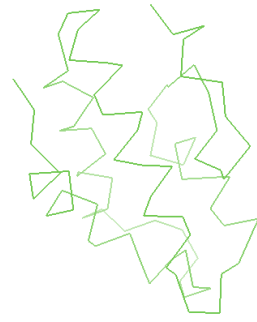
To make the dissertation self-contained we include some basics facts about protein structures. We also describe how a structure of a protein store in the Protein Data Bank [37] as it is helpful when using PDB file as an input to a structure alignment program.

## **Preliminaries**

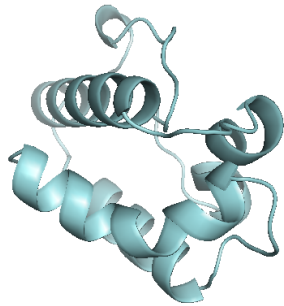
Proteins are large linear macromolecules built from an “alphabet” of 20 different amino acids (see Figure 1.7). These are the molecules along a protein and are called “residues”. The amino acids are organic compounds composed of a backbone and a side chain (see Figure 1.8 and Figure 1.7). Depending on the side chain, an amino



(a)



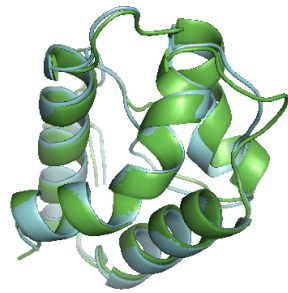
(b)



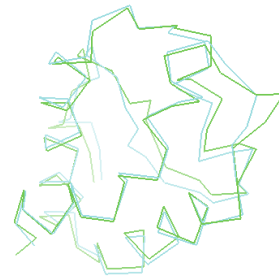
(c)



(d)



(e)



(f)

Figure 1.6: Structure superposition of LTP1 from maize (PDB code 1MZL [4]) and LTP1 from rice (PDB code 1RZL [5]). (a) cartoon representation of 1MZL (b) ribbon representation of 1MZL (c) cartoon representation of 1RZL (d) ribbon representation of 1RZL (e) superposition of 1MZL and 1RZL (cartoon) (f) superposition of 1MZL and 1RZL (ribbon)



acid can be classified into one of 5 different groups. In Figure 1.7 the name and background shading indicates (a) hydrophobic amino acids as yellow; (b) hydrophilic non-charged amino acids as white; (c) positive charged amino acids as blue; (d) negative charged amino acids as red; (e) cysteine as green. In Figure 1.7 atom types are color-coded: carbon with gray, oxygen with red, nitrogen with blue, and sulfur with yellow.

The backbone of an amino acid has two functional groups: (1) amino group ( $-NH_2$ ) and (2) carboxyl group ( $-COOH$ ) which are responsible for formation of linear polymers by linking each other with a peptide bond. A peptide bond is formed when two amino acids chemically bond, releasing a water molecule. The amino acid residue sequence along a protein is known as a primary structure. Segments of polypeptides often fold locally into secondary structures, for example  $\alpha$ -helices and  $\beta$ -sheets.

The three dimensional tertiary structure is built up from secondary structure elements and determines the biological functions of the protein. Thus the tertiary structure of a protein is a subject of interest in the study of protein structure similarity (for more details on protein structures refer to [38]).

| Name          | Amino Acid | Sidechain | Name          | Amino Acid | Sidechain | Name          | Amino Acid | Sidechain | Name       | Amino Acid | Sidechain |
|---------------|------------|-----------|---------------|------------|-----------|---------------|------------|-----------|------------|------------|-----------|
| Alanine       | Ala        | A         | Glutamine     | Gln        | Q         | Leucine       | Leu        | L         | Serine     | Ser        | S         |
| Arginine      | Arg        | R         | Glutamic Acid | Glu        | E         | Lysine        | Lys        | K         | Threonine  | Thr        | T         |
| Asparagine    | Asn        | N         | Glycine       | Gly        | G         | Methionine    | Met        | M         | Tryptophan | Trp        | W         |
| Aspartic Acid | Asp        | D         | Histidine     | His        | H         | Phenylalanine | Phe        | F         | Tyrosine   | Tyr        | Y         |
| Cysteine      | Cys        | C         | Isothreonine  | Ile        | I         | Proline       | Pro        | P         | Valine     | Val        | V         |

Figure 1.7: The 20 amino acids [Figure taken from [6]]

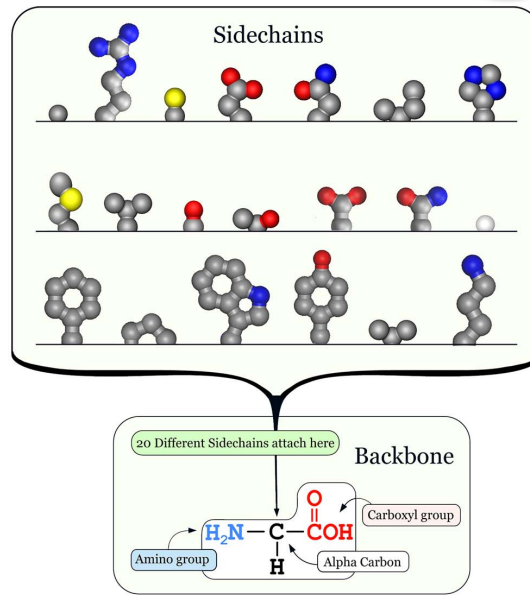


Figure 1.8: The 20 amino acids as sidechain [Figure taken from [7]]

The tertiary structure of a protein is represented in a standard format known as the PDB (Protein Data bank) file format [37]. The data contained in such a file is derived from X-ray diffraction and NMR (Nuclear magnetic resonance) studies. The PDB file format contains the three dimensional coordinates of every atom in a protein. Proteins are tightly packed globs of atomic spheres where each atom is assumed to be a sphere of radius  $a_i(x_i, y_i, z_i)$ . A typical value of  $a_i$  lies between  $1A^\circ$  and  $2A^\circ$ .

Each protein structure archived in the Protein Data Bank [37, 39] assigned a 4-character unique identifier known as its PDB-id. To describe the structure of a protein molecule the PDB file contains different types of records. The Table 1.1 presents some selected record types while a sample PDB file is given in Appendix B. In the dissertation we have used molecular graphics software Pymol [40] to generate protein figures.

### 1.3 Contributions

The main contributions of this dissertation are as follows.

1. **Geometric proximity:** We studied the following optimization problem from a geometric proximity perspective. Given a point set  $P = \{p_1, p_2, p_3, \dots, p_n\}$ , for each point  $p_i$  find a pair  $\{p_j, p_k\}$ , where  $i \neq j, i \neq k, j \neq k$ , such that a measure  $\mathcal{M}$  defined on the triplet of points  $\{p_i, p_j, p_k\}$  is maximized or minimized. We

Table 1.1: Selected Protein Data Bank Record Types

| Record Type | Data Provided by Record   |
|-------------|---|
| HEADER      | provides the information like PDB-id, classification for entry and the date when deposited to PDB archive                       |
| SEQRES      | provides the information about the residues covalently linked in a linear fashion to form a polymer                             |
| HELIX       | provides the information like position of helix in the molecule and type of the helix   |
| SHEET       | provides the information to identify position of sheet in the molecule and number of strand in the sheet                        |
| ATOM        | provides the information like atomic co-ordinates, occupancy and temperature factor of each atom of the residues in the polymer |
| HETATM      | provides the information about atomic co-ordinates of non-polymer or other non-standard” chemical compound such water molecule  |
| TER         | indicates the end of chain  |

propose efficient algorithms for each of the following distance measures

- (a) Sum and product measures
- (b) Difference measure
- (c) Line-distance measure
- (d) Triangle area measure
- (e) Triangle perimeter measure
- (f) Circumcircle-radius measure
- (g) Triangle-distance measure in three dimension

## 2. Geometric separability and gene expression data set:

We also study the geometric separation of gene expression data sets. Our contributions are as follows

- (a) An offline adaption of Megiddo’s algorithm to test linear separability by gene pairs/triplets, fully implemented and tested.
- (b) An incremental version of Megiddo’s algorithm that is particularly useful for gene expression datasets, fully implemented and tested.
- (c) Demonstration of the usefulness of linear separability as a tool to build a good classifier with application to concrete examples.
- (d) Reformulation of Unger and Chor’s [31] method in the linear programming framework.

The dissertation highlights the advantage of using a “linear time” incremental algorithm as compared to a “quadratic time” algorithm of Unger and Chor [31].

**3. Geometric separability with few violated constraints:** We also extend the idea of linear separability to almost linear separability. We propose an efficient algorithm to find a minimum set of outliers (or violated constraints), say  $k$ , when two point sets (colored respectively red and blue) are not completely separable.

We propose an  $O(nk^2)$  time algorithm for this problem. When  $k = o(\log n)$ , this is better than the so-far-best  $O((n + k^2) \log n)$  time algorithm known for

this problem. For  $k = O(1)$ , which holds for the application to gene expression analysis that we have in mind, we have the first linear time algorithm known for this problem.

4. **Point pattern matching and protein structure similarity:** We also study the problem of protein structure similarity from a point pattern matching perspective. We propose two new algorithms,  $EDAlign_{res}$  and  $EDAlign_{sse}$ , for pairwise protein structure alignment.

- (a)  $EDAlign_{res}$  identifies the best structural alignment of two equal length proteins by refining the correspondence obtained from eigendecomposition and to maximize similarity measure, TM-score [41], for the refined correspondence.
- (b)  $EDAlign_{sse}$ , on the other hand, does not require the input proteins to be of equal length.

We report the TM-score and cRMSD as measures of structural similarity. These new methods are able to report sequence and topology independent alignments, with similarity scores that are comparable to those of the state-of-the-art algorithms such as, TM align [42] and SuperPose [43].

# Chapter 2

## All-maximum and all-minimum problems under some measures

### 2.1 Overview

In this chapter [44] we investigate the following type of proximity problems: given a set of  $n$  points in the plane  $P = \{p_1, p_2, p_3, \dots, p_n\}$ , for each point  $p_i$  find a pair  $\{p_j, p_k\}$ , where  $i \neq j, i \neq k, j \neq k$ , such that a measure  $\mathcal{M}$  defined on the triplet of points  $\{p_i, p_j, p_k\}$  is maximized or minimized. The cases where  $\mathcal{M}$  is the distance from  $p_i$  to the segment or line defined by  $\{p_j, p_k\}$  have been extensively studied. We study the cases where  $\mathcal{M}$  is the sum, product or the difference of the distances from  $p_i$  to the points  $p_j$  and  $p_k$ ; distance from  $p_i$  to the line defined by  $p_j$  and  $p_k$ ; the area, perimeter of the triangle defined by  $p_i, p_j$  and  $p_k$ , as well as the radius of the circumcircle defined by them. We also discuss the all-farthest triangle problem in the triangle-distance measure when  $P$  is a set of points in 3 dimensions.

### 2.2 Introduction

In computational geometry, an oft-recurring problem is to identify subsets of a point set having desirable properties. The following type of proximity problems belong to

this genre: for each point  $p_i$  in a planar point-set  $P = \{p_1, p_2, p_3, \dots, p_n\}$ , find a pair  $\{p_j, p_k\}$ ,  $i \neq j, i \neq k, j \neq k$ , such that a measure  $\mathcal{M}$  defined on the points  $\{p_i, p_j, p_k\}$  is maximized or minimized. The cases where  $\mathcal{M}$  is the distance from  $p_i$  to the segment or line defined by  $\{p_j, p_k\}$  have been extensively studied (see [28], [25], [26]). In this chapter, we study a number of other measures in two and higher dimensions.

A more general and natural version of the problem is to allow the query to be any point in the plane, which would make all the problems in this genre solvable, in principle, in the 2-point site Voronoi diagram scheme of Barequet et al. [24]. However, restricting the queries to the points in  $P$  often allows us to solve the problems more efficiently by defining suitable data structures on the point set  $P$ . An example that readily comes to mind is the construction of the Voronoi diagram data structure on  $P$  to find the nearest neighbor of each point in  $P$ .

### 2.2.1 Motivation

Our motivation is primarily theoretical, originally triggered by an interest in knowing whether an  $O(n^2)$  algorithm for the segment distance problem for the all-nearest measure problem in the plane [25] is also 3sum-hard for the all-farthest measure. In [26] it was shown that it is not so, and this work also initiated investigation into the surprisingly unexplored problem of computing farthest-segment Voronoi diagrams [27]. In this vein, we extend our investigation to other measures. From a theoretical per-



spective, what also interests us is the intimate connection of this problem to the combinatorial complexities of 2-point site Voronoi diagrams studied in [24].

In [28], for the segment distance measure an interesting and more practical motivation in the form of an application to graph drawing is discussed, while in [29] for the line distance measure a military application in the form of communication disruption is described. There may possibly be other interesting practical applications that we are not aware of or even some that are awaiting discovery.

### 2.2.2 Prior Work

Ovidiu et al. [28] introduced a type of proximity problem which can be stated this way: for each of a given set of  $n$  points,  $P = \{p_1, p_2, p_3, \dots, p_n\}$ , find a segment defined by two other points that is nearest with respect to some measure  $\mathcal{M}$ .

Following up on [45], Duffy et al. [25] proposed an algorithm for solving the all-nearest version of the problem in  $O(n^2)$  time, while Mukhopadhyay et al. [26] solved the all-farthest version of the problem in  $O(n \log n)$  time.

In this chapter, we explore this line of work further by systematically looking at the restricted version of the query problem with respect to several other measures, viz. when  $\mathcal{M}$  is the sum, product, or the difference of the distances from  $p_i$  to the points

$p_j$  and  $p_k$ ; the closest and farthest distance from  $p_i$  to the line spanned by  $p_j$  and  $p_k$ ; the area, perimeter of the triangle defined by  $p_i, p_j$  and  $p_k$ , as well as the radius of the circumcircle defined by them. Barequet et al. [24] studied the combinatorial complexities of the farthest and nearest point Voronoi diagrams in all these measures, except for the circumcircle-radius measure that they left open. We propose a novel solution to this last problem in this restricted setting. In addition, we have also studied the all-farthest triangle-distance problem as a generalization to 3 dimensions of the all-farthest segment distance problem [26].

### 2.2.3 Our results

We summarize our results in Table 2.1 for a planar set of points, where in the minimum column for the Triangle Perimeter measure,  $\phi_i^j$  is a parameter related to the  $i$ -th point  $p_i$ . In some of these cases we have also discussed the extension of the results to higher dimensions.

Table 2.1: Our Results

| Measure            | all-maximum     | all-minimum                                  |
|--------------------|-----------------|--|
| Sum                | $O(n \log n)$   | $O(n \log n)$                                |
| Product            | $O(n \log n)$   | $O(n \log n)$                                |
| Difference         | $O(n \log n)$   | $O(n^2 \log n)$                              |
| Line-distance      | $O(n^2)$        | $O(n^2)$                                     |
| Triangle Area      | $O(nh)$         | $O(n^2)$                                     |
| Triangle Perimeter | $O(nh)$         | $O(n^2 \log n + \sum_i \sum_j (\phi_i^j)^2)$ |
| Circumradius       | $O(n^2 \log n)$ | $O(n^2 \log n)$                              |
| Triangle-distance  | $O(nhh')$       | $O(n^4)$                                     |

## 2.3 Sum and Product Measures

The *sum*  $\mathcal{S}(p_i, p_j, p_k)$  and *product*  $\mathcal{P}(p_i, p_j, p_k)$  measures are respectively  $|\overline{p_i p_j}| + |\overline{p_i p_k}|$  and  $|\overline{p_i p_j}| * |\overline{p_i p_k}|$ , where  $|s|$  is the length of a segment  $s$ .

The computational problem is to find for each  $p_i$  in  $P$  a pair  $\{p_j, p_k\}$  in  $P - \{p_i\}$ ,  $j \neq k$ , such that the *sum* measure  $\mathcal{S}(p_i, p_j, p_k)$  and the *product* measure  $\mathcal{P}(p_i, p_j, p_k)$  is maximum (minimum).

We have the following obvious characterization.

**Claim 1** *For a point  $p_i \in P$ ,  $\mathcal{S}(p_i, p_j, p_k)$  and  $\mathcal{P}(p_i, p_j, p_k)$  is maximum (minimum) when  $p_j, p_k \in P - \{p_i\}$ , realize the farthest (nearest) and second farthest (second nearest) distance from the point  $p_i$ .*

An  $O(n^2)$  algorithm for the all-maximum as well as the all-minimum in both the measures is immediate from the above claim.

For a more efficient  $O(n \log n)$  algorithm, we construct the *third order nearest* and *second order farthest* Voronoi diagrams on  $P$  and construct point location structures on both that allow (point) location of  $p_i$  in  $O(\log n)$  time.

The all-minimum problem solves the closest pair problem and thus has a lower bound

of  $\Omega(n \log n)$  in the algebraic decision tree model. The all-maximum problem solves the all-farthest pairs problem and has a lower bound of  $\Omega(n \log n)$  in the same model. Thus the  $O(n \log n)$  algorithms are optimal in the algebraic decision tree model.

## 2.4 Difference Measure

The *difference* measure is  $\mathcal{D}(p_i, p_j, p_k) = ||\overline{p_i p_j}| - |\overline{p_i p_k}||$ . The computational problem is to find for each  $p_i$  in  $P$  a pair  $\{p_j, p_k\}$  in  $P - \{p_i\}$ ,  $j \neq k$  such that  $\mathcal{D}(p_i, p_j, p_k)$  is maximum (minimum).

The following characterization is again obvious.

**Claim 2** *For a point  $p_i \in P$ , the pair  $\{p_j, p_k\} \in P - \{p_i\}$  realize the maximum  $\mathcal{D}(p_i, p_j, p_k)$  iff  $p_j$  and  $p_k$  are respectively the nearest and farthest point from  $p_i$  or vice versa.*

For the maximum  $\mathcal{D}\{p_i, p_j, p_k\}$  for all  $p_i \in P$ , a brute-force  $O(n^2)$  algorithm is immediate. We can improve on this by constructing the nearest-point Voronoi diagram of  $P$  [46] to obtain for each  $p_i$  its nearest point in  $P - \{p_i\}$ ; for the farthest point of each  $p_i$ , we have to construct the farthest-point Voronoi diagram as well as a point location structure [46] on top of this. This is because we need to locate  $p_i$  in the farthest-point Voronoi region of a point  $p_j$  it is farthest from.

This gives us an  $O(n \log n)$  time algorithm for the maximum problem. The brute-force version is easily extended to  $d$ -dimensions to run in  $O(dn^2)$  time, the additional factor  $d$  accounting for the distance calculations.

The maximum problem has a lower bound of  $\Omega(n \log n)$  in the algebraic decision tree model since it implicitly solves the closest pair problem.

For each  $p_i \in P$ , we can determine the minimum  $\mathcal{D}\{p_i, p_j, p_k\}$  by finding a pair of points  $\{p_j, p_k\}$  that have the smallest difference in their Euclidean distances from  $p_i$ . This amounts to solving the closest-pair problem on the line by mapping the distances from  $p_i$  onto it. This takes  $O(n \log n)$  time, which is optimal in the algebraic decision-tree model. Thus the time complexity for the minimum of  $\mathcal{D}(p_i, p_j, p_k)$ , for all  $p_i \in P$ , is in  $O(n^2 \log n)$ .

The time complexity of the minimum version of the problem can be reduced to expected time  $O(n^2)$ , since finding a pair  $p_j, p_k$  is equivalent to finding a closest pair on the line when the distances from  $p_i$  are measured from an origin point,  $O$ , on the line. The latter problem can be solved in expected  $O(n)$  time by a randomized algorithm, assuming that floor and square-root operations can be done in constant time [47]. However, coming up with an  $O(n^2)$  deterministic algorithm remains an interesting open question.

It is worth investigating the special case when the points lie on a line. Assume that the points are in sorted order. For the two extreme points the solution consists of a closest pair of points. For an intermediate point  $p_i$ , we reduce the problem to the extreme points case by embedding into the points, say, on the right of  $p_i$  the reflections in  $p_i$  of the points to its left and then find a closest pair of the combined set. Thus the all-minimum problem can be solved in this special case in  $O(n^2)$  time.

Since we implicitly solve the closest pair problem for the point set, we have a lower bound of  $\Omega(n \log n)$  in the for the all-minimum problem in the algebraic decision tree model.

The difference in the time-complexities of these two problems is noteworthy. Clearly, in the second case relative to a  $p_i$  any pair of points  $p_j, p_k$  can realize the minimum distance, whereas the maximum distance is realized by a definite pair of points.

The minimum version is also easily extended to  $d$ -dimensions to run in  $O(dn^2 \log n)$  time, the additional factor  $d$  accounting for the distance calculations.

## 2.5 Line Distance Measure

The line-distance measure is  $\mathcal{LD}(p_i, p_j, p_k) = d(p_i, \overleftrightarrow{p_j p_k})$ , where  $d(p, l)$  denotes the minimum distance from a point  $p$  to a line  $l$ .

For the *all-farthest* line distance problem we borrow an idea that Duffy et al. [25] used for the all-nearest segment distance problem. Assume we know the angular order of the points in  $P - \{p_i\}$  about  $p_i$ . Call the polygon obtained by joining the points in this angular order the *surrounding polygon* of  $p_i$ . Fix a  $p_j$  in  $P - \{p_i\}$ . As we traverse the boundary of this polygon, we find a point  $p_k$ ,  $k \neq j$ , such that line incident on  $p_i$  and  $p_k$  is farthest from  $p_j$ . This is updated vis-a-vis  $p_j$  as we consider the angular orders about the remaining  $n - 2$  points in  $P$ . We search for  $p_k$ , following the scheme below.

Consider a circle of radius  $|p_i p_j|$ , centered at  $p_j$ . The line through  $p_i$  tangent to this circle is a farthest line if there is another point  $p_k$  incident on it; otherwise, we determine at most four candidate points on the surrounding polygon, at most one in each of the 4 quadrants determined by the supporting line of  $\overline{p_i p_j}$  and the tangent to the circle at  $p_i$ , and lying immediately above and below the tangent line. The line farthest from  $p_j$  is the one incident on one of these candidate points and  $p_i$ , making the *largest* acute angle with  $\overline{p_i p_j}$ .

As we move counterclockwise to the next point  $p_{j+1}$  on the polygon, we either find a point incident on the new tangent line, else find a new set of at most four candidate points in constant time. As no backtracking is involved in this latter update, for each  $p_j$  in  $P - \{p_i\}$  we can find the farthest line incident on  $p_i$  in  $O(n)$  time, including the time that it takes to find the intersection of the initial tangent line with the surrounding polygon of  $p_i$ , all by a single tour of the boundary.

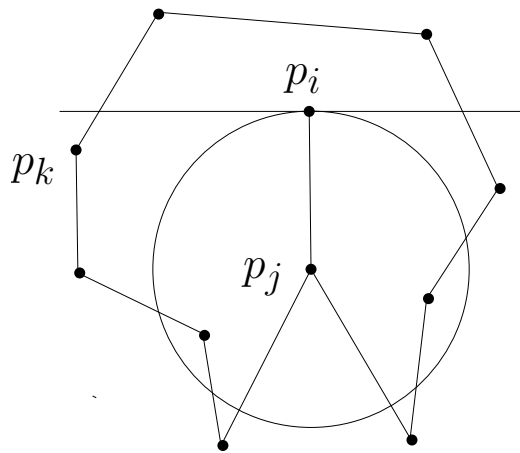


Figure 2.1: A farthest line from  $p_j$ , incident on  $p_i$

We repeat the above steps for each of the angular orders about the remaining  $n - 1$  points in  $P$ . Since we can determine the angular orders about all the  $p_i$ 's in  $O(n^2)$  time [48], [49], the time-complexity of the all-farthest problem in the line-distance measure is in  $O(n^2)$ .

An interesting open problem is to establish if the above algorithm is optimal in the algebraic decision tree model. It would also be interesting to generalize the above



algorithm to  $d$ -dimensions with time complexity in  $O(n^d)$ . When  $d = 3$ , we can use the algorithm of Bespamyatnikh and Segal [29] to obtain an algorithm whose time-complexity is in  $O(n^3 \log^2 n)$ . Paring away the  $\log^2 n$  factor is another interesting problem.

We next consider the *all-closest* problem in the line-distance measure.

For a fixed  $p_i$ , Mount et al. [28] gave an optimal  $O(n \log n)$  time and  $O(n)$  space algorithm to find the line closest to it spanned by a pair of points  $p_j, p_k \in P - \{p_i\}$ . Here we show that the all-closest problem can be solved in  $O(n^2)$  time.

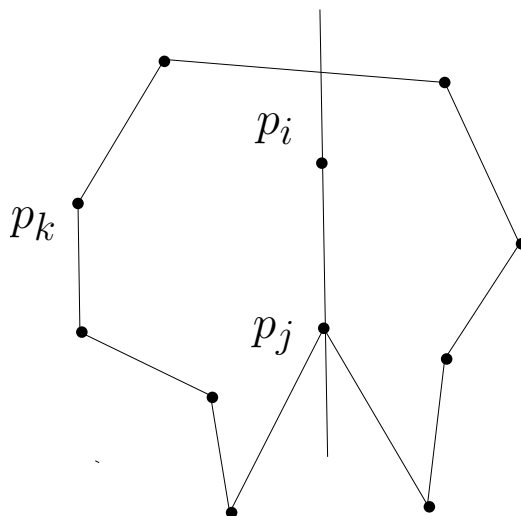


Figure 2.2: A closest line from  $p_j$ , incident on  $p_i$

Indeed, exactly the same approach as used for the *all-farthest* problem also works, except that the closest line to  $p_j$ , incident on a candidate point in a quadrant and

$p_i$ , makes the *smallest* acute angle with the supporting line of  $\overline{p_i p_j}$ . Thus we have an  $O(n^2)$  algorithm again.

Since the above algorithm allows us to determine a point-and-closest-line pair such that the distance from the point to the line is minimum among all such pairs, as in Duffy et al. [25], we can argue that this problem is 3sum-hard by 1-reduction from the problem of determining if 3 of  $n$  points  $p_1, p_2 \dots, p_n$  in the plane are collinear (see also [50]).

An interesting open problem is to design an algorithm for the all-closest problem to  $d$  dimensions. This would probably require a different approach than what we have used for the  $2-d$  case.

## 2.6 Triangle Area Measure

Let  $\mathcal{A}(p_i, p_j, p_k)$  denote the area of the triangle formed by the points  $p_i, p_j$  and  $p_k$ . In this measure, for each  $p_i$  in  $P$ , we have to find out a pair of points  $\{p_j, p_k\}$  in  $P - \{p_i\}$  such that  $\mathcal{A}(p_i, p_j, p_k)$  is maximum (minimum).

For a fixed  $p_i$ , this problem has been solved in [28] for both the maximum and minimum measures in  $O(n \log n)$  time and  $O(n)$  space that are optimal in the algebraic decision tree model. This gives  $O(n^2 \log n)$  algorithms for the all-maximum as well

as the all-minimum version. The proposed algorithms employ the dynamic convex hull maintenance algorithm of Brodal and Jakob [51]. Below, we propose a simple  $O(nh)$  algorithm for the all-maximum area problem and then, by dualization, an  $O(n^2)$  algorithm for the all-minimum version.

### 2.6.1 Maximum Area Triangle

The following claim gives a structural characterization of a maximum area triangle, anchored at a point  $p_i$ .

**Claim 3** *For an anchor point  $p_i$ ,  $\mathcal{A}(p_i, p_j, p_k)$  is maximum when the points  $p_j$  and  $p_k$  lie on the boundary of the convex hull,  $CH(P)$ .*

**Proof:** Assume that at least one of the points  $p_j$  and  $p_k$  does not lie on  $CH(P)$ . Without loss of generality let it be  $p_k$ . If we draw a line  $l$  through  $p_k$ , parallel to  $\overline{p_i p_j}$ , then there exists a point  $p'_k$  on the convex hull boundary in the open half-space defined by  $l$  that does not contain  $p_i$  (see Figure 2.3) such that  $\mathcal{A}(p_i, p_j, p'_k)$  is greater than  $\mathcal{A}(p_i, p_j, p_k)$ . This contradicts our assumption that  $\mathcal{A}(p_i, p_j, p_k)$  is of maximum area. Thus  $p_j$  and  $p_k$  both lie on the convex hull boundary.  $\square$

**Claim 4** *For a point  $p_i$ , if  $\mathcal{A}(p_i, p_j, p_k)$  is maximum for a pair  $\{p_j, p_k\}$ , then  $p_j$  is a farthest point from the supporting line of  $\overline{p_i p_k}$  and  $p_k$  is a farthest point from the supporting line of  $\overline{p_i p_j}$ .*

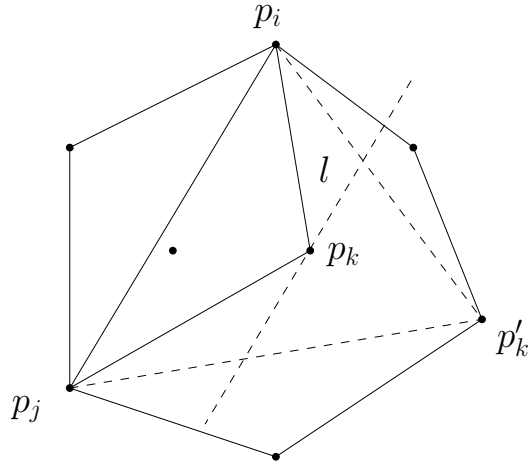


Figure 2.3: When  $p_k$  is an internal vertex of the convex hull,  $CH(P)$

**Proof:** Suppose  $p_j$  is not farthest point from the line  $\overline{p_i p_k}$ . This implies that there exists another point  $p'_j$ , which is farthest from  $\overline{p_i p_k}$ . Thus  $\mathcal{A}(p_i, p'_j, p_k)$  is greater than  $\mathcal{A}(p_i, p_j, p_k)$ . This contradicts our assumption. By a similar argument,  $p_k$  is farthest from  $\overline{p_i p_j}$ . □

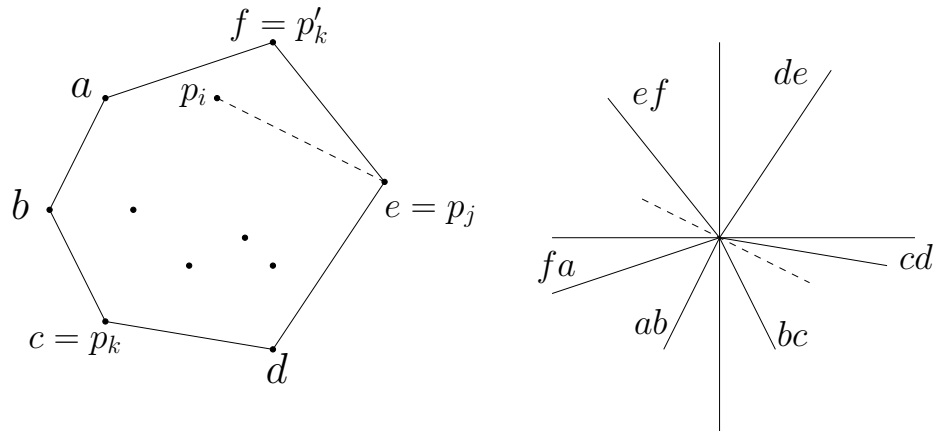


Figure 2.4: Convex hull and its corresponding ray diagram

For an efficient algorithm for computing a maximum area triangle for a  $p_i \in P$  we borrow the ‘rotating calipers’ idea that Shamos [8] used to find diameter of a set. In a preprocessing step, we first construct the convex hull of  $P$  and then the ray diagram of the convex hull. Next, we scan the boundary of the convex hull,  $CH(P)$ , in counter-clockwise order, starting with some vertex  $p_j$ , say. If  $p_k$  is the farthest antipodal vertex corresponding to  $\overline{p_i p_j}$ , we tag  $p_k$  with the index  $j$ ; when the scan reaches  $p_k$ , we determine its farthest antipodal vertex and use the tag attached to  $p_k$  to check if it is the same as  $p_j$ . If so, this is another candidate pair. We determine the area of  $\triangle(p_i, p_j, p_k)$  and update the current maximum area triangle if necessary. Doing this for each  $p_i$ , gives an  $O(n(h + \log h))$  algorithm for the all-maximum problem.

It is also possible to devise an  $O(n^2)$  time algorithm (matching the worst case of the previous algorithm) by dualization which can be generalized to 3-dimensions. We first note that for a fixed anchor point  $p_i$  if  $\triangle p_i p_j p_k$  has maximum area then  $p_k$  is farthest from the supporting line of  $p_i p_j$ . Thus we have to find the farthest  $p_k$  for each of  $n - 1$  different line segments  $\overline{p_i p_j}$ . The dual version helps us solve our problem if for each intersection point, which corresponds to a point pair  $\{p_i, p_j\}$  in the dual plane, we determine the line that is vertically farthest from this intersection point. This line will be part of the lower or upper envelope in the arrangement, each of which is of size  $O(h)$ . The upper and lower envelopes can be obtained in  $O(n \log n)$  from the

convex hull of the points in the primal plane.

Now, we do a topological sweep of the arrangement in the dual plane. This sweep discovers the intersection points on each line in a left to right order. This means that we can determine the intersection of a vertical line through each intersection point on this line with the upper and lower envelopes in a left to right order. We maintain an intersection history for each line. This requires  $O(h)$  time for each line and hence  $O(nh)$  time for all the lines. Now that we have the farthest point for each line, we can determine the maximum area triangle for an anchored point by selecting from among the intersection points on the dual of  $p_i$  the one whose farthest line is the farthest of all.

Since topological sweep can be done in  $O(n^2)$  time, the time complexity of this alternate algorithm is in  $O(n^2)$ . This algorithm can be generalized to 3-dimensions, using a topological sweep algorithm to compute an arrangement of planes in 3-dimensions [52] to find a maximum volume tetrahedron for each  $p_i$  in  $O(n^3)$  time.

In [53] an  $\Omega(n \log n)$  lower bound has been established in the algebraic decision tree model for the complexity determining a maximum area  $k$ -gon, such that the  $k$  vertices of this polygon are a subset of a given set of  $n$  points. Since we solve this problem for  $k = 3$  by solving the all-maximum triangle area problem, we have the same lower bound for this problem. Closing the complexity gap is an interesting open

problem.

## 2.6.2 Minimum Area Triangle

A minimum area triangle  $\triangle(p_i, p_j, p_k)$  anchored at  $p_i$  must be empty, for if it contained a point  $p_l$ , then  $\triangle(p_i, p_l, p_k)$ , for example, has a smaller area. This observation, however, does not yield an efficient algorithm for if the points in  $P$  are the vertices of a convex polygon, we have to consider  $O(n^2)$  empty simplexes for each anchored point.

The main difficulty with the minimum area triangle computation lies in the absence of locality. A triangle of small area can have very long edges. Chazelle et al. [54] and Edelsbrunner et al. [52] used geometric duality to find a triangle  $\triangle(p_i, p_j, p_k)$  whose area is globally minimum. We explore this approach for finding anchored minimum area triangles.

We can view the computation of a minimum area triangle anchored at a point  $p_i$  this way: choose a  $p_j$  from among the remaining  $n - 1$  points and for this pair choose a third point  $p_k$  such that the area of the triangle  $\triangle(p_i, p_j, p_k)$  is a minimum. This gives us the following alternate characterization of an anchored minimum area triangle.

**Claim 5** *If  $\mathcal{A}(p_i, p_j, p_k)$  is the minimum area of a triangle anchored at the point  $p_i$ , then  $p_k$  is vertically closest to the supporting line,  $\ell$ , of  $p_i$  and  $p_j$ .*

**Proof:** If there is a point  $p'_k \in P$  which is vertically closer to  $\ell$  than  $p_k$ , it would have

to lie in the strip defined by  $\ell$  and a line through  $p_k$  parallel to  $\ell$ . In that case the area of  $\triangle(p_i, p_j, p'_k)$  would be smaller than that of  $\triangle(p_i, p_j, p_k)$ .  $\square$

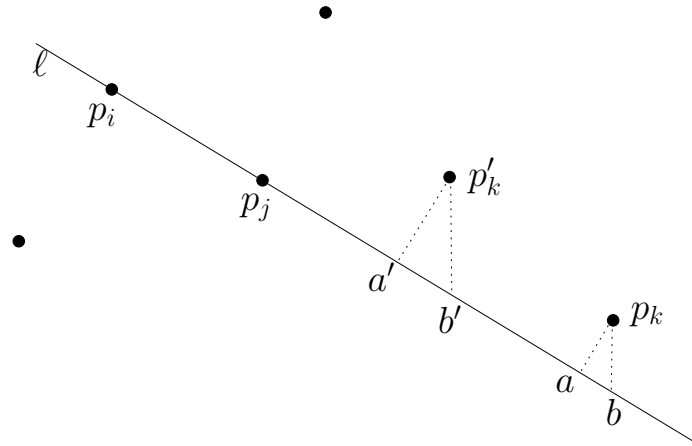


Figure 2.5: Similar triangles  $\triangle a'b'p'_i$  and  $\triangle abp_i$

The above characterization is helpful because dualization preserves vertical distances; in the dual plane a fixed  $p_i$  corresponds to a fixed line  $p_i^*$ , and for each  $p_j$ , the pair  $(p_i, p_j)$  corresponds to an intersection on the line  $p_i^*$ . For each intersection point we have to find a line  $p_k^*$  that is vertically closest to it.

We can solve our problem by simulating Chazelle et al.'s [54] construction of an arrangement of lines (in the dual plane for the given point set in the primal plane) with some additional book-keeping.

We use any reasonable data structure (e.g doubly connected edge list) to represent the planar graph corresponding to the arrangement. To ensure that the addition of each new line to the current arrangement takes linear time, we assume that the



arrangement lies inside a large bounding box. Since a line intersects the boundary of this bounding box twice, we can easily determine the entry face of the  $i$ -th line  $\ell_i$  of the arrangement. The arrangement can be updated by walking along the lower parts of  $\text{zone}(\ell_i)$  (i.e. zone of  $\ell_i$ ) as shown in Figure 2.6. As the combinatorial complexity of the faces of the arrangement that intersect  $\ell_i$  is at most  $8i$  [54], the walk along the  $\text{zone}(\ell_i)$  takes  $O(i)$  time. While updating the data structure, we maintain the vertically closest line from each vertex.

Since computing the arrangement takes  $O(n^2)$  time and  $O(n^2)$  space, the same complexities hold for the all-minimum area triangle problem. This problem is 3sum-hard since we can use this to determine if 3 of  $n$  points in the plane are collinear.

We can also generalize the algorithm to  $d$ -dimensions to run in  $O(n^d)$  time since the size of a zone is  $O(n^{d-1})$  [55].

## 2.7 Triangle Perimeter Measure

In this measure, for each  $p_i$ , we want to find a pair of points  $\{p_j, p_k\} \in P - \{p_i\}$ , such that  $\mathcal{P}(p_i, p_j, p_k) = |\overline{p_i p_j}| + |\overline{p_j p_k}| + |\overline{p_i p_k}|$  is maximum or minimum.

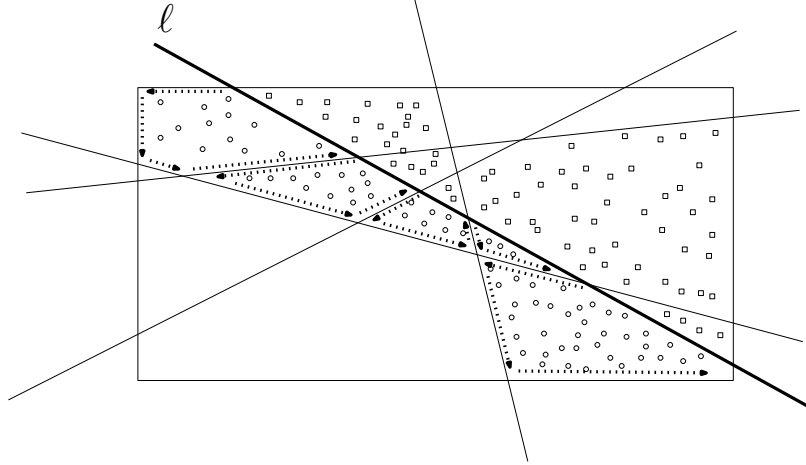


Figure 2.6: Walking the lower part of zone( $\ell$ )

### 2.7.1 Maximum perimeter

We have the following characterization of a pair  $\{p_j, p_k\}$  in  $P$  that gives the maximum perimeter for a given  $p_i \in P$ .

**Claim 6** *If for a point  $p_i \in P$ , the pair  $\{p_j, p_k\} \in P - \{p_i\}$  realizes the maximum perimeter  $\mathcal{P}(p_i, p_j, p_k)$  then  $p_j$  and  $p_k$  are vertices on the convex hull of  $P$ ,  $CH(P)$ .*

**Proof.** Let the maximum perimeter be realized by a pair  $\{p_j, p_k\}$ , both of which are internal to the hull boundary. Extend  $\overline{p_i p_j}$  and  $\overline{p_i p_k}$  to meet the boundary of  $CH(P)$  at  $x$  and  $y$  respectively. If both of the latter points are vertices of  $CH(P)$ , we are done. Otherwise assume that at least  $x$  or  $y$  is internal point of convex hull edge. Without loss of generality assume that  $x$  is internal point of convex hull edge. Consider an ellipse, one of whose axis lies along  $\overline{p_i y}$ , the other orthogonal to it, foci at the points  $p_i$  and  $y$  and focal radius  $|\overline{p_i x}| + |\overline{y x}|$  (see Figure 2.7). This crosses the convex hull boundary at  $x$  and from convexity of the ellipse and  $CH(P)$ , there will

be a point  $p_i$  on  $CH(P)$  that lies outside this ellipse. This implies that the triangle  $\triangle(p_i, y, p_i)$  has larger perimeter than  $\triangle(p_i, x, y)$  and hence  $\triangle(p_i, p_j, p_k)$ .  $\square$

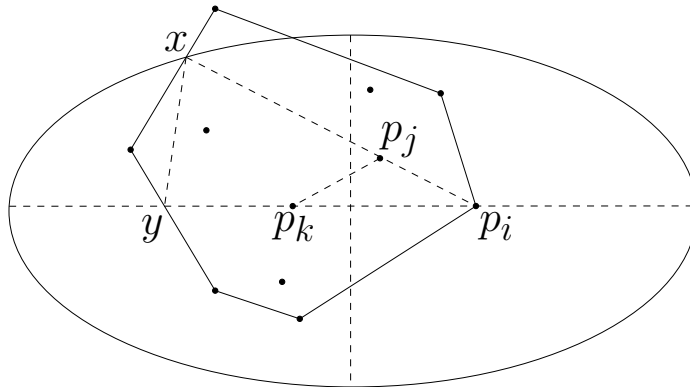


Figure 2.7: A maximum perimeter triangle rooted at  $p_i$  has the other two points on  $CH(P)$

For all the  $p_i$ 's on the convex hull boundary, we can find the maximum perimeter distance by using the monotone matrix method of [56] in  $O(h \log h)$  time. If  $p_i$  is internal to the hull boundary we have the problem of finding a maximal perimeter triangle, rooted at  $p_i$ , with the other two points on the hull boundary. Boyce et al. [57] gave an ingenious reduction of this problem to the problem of computing the diameter of a convex polygon bounded by circular arcs and segments that are common external tangents to two circles (see Figure 2.8 below). The circles in question are obtained by centering them at the points in  $P - \{p_i\}$  and letting them pass through the anchor point  $p_i$ . The diameter is the distance between parallel lines of support that are tangents to a pair of antipodal circular arcs. Moreover, the segment joining the points

of contact passes through the centers of the two circles and is easily seen to be the perimeter of the triangle on  $p_i$  and the centers of the circles in question. This takes  $O(h)$  time for a fixed  $p_i$  and thus  $O((n - h) * h)$  time for all the  $n - h$  points inside the convex hull.

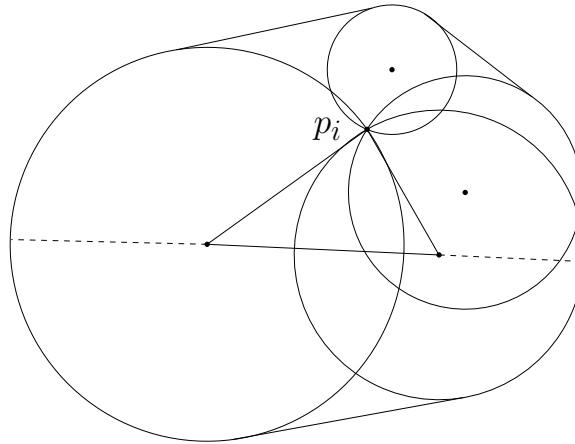


Figure 2.8: Diameter of the convex figure is equal to the maximum perimeter triangle rooted at  $p_i$

Thus the complexity of the all-farthest problem in this distance measure is in  $O(h + (n - h) * h)$ , that is in  $O(nh)$ .

In [53] an  $\Omega(n \log n)$  lower bound has been established in the algebraic decision tree model for the complexity determining a maximum perimeter  $k$ -gon, such that the  $k$  vertices of this polygon are a subset of a given set of  $n$  points. Since we solve this problem for  $k = 3$  by solving the all-maximum triangle perimeter problem, we have the same lower bound for this problem. Closing the complexity gap is an interesting

open problem.

## 2.7.2 Minimum perimeter

The following interesting observation helps in localizing the search, relative to a given  $p_i$ , for a pair of points  $p_j$  and  $p_k$  that minimizes the perimeter of  $\Delta(p_i, p_j, p_k)$ .

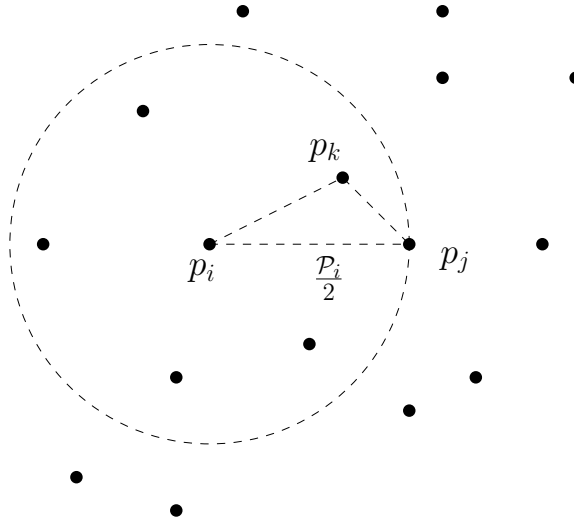


Figure 2.9: The perimeter of  $\Delta(p_i, p_j, p_k) > \mathcal{P}_i$

**Claim 7** *If  $\mathcal{P}_i$  is the perimeter of any triangle anchored at  $p_i$ , then neither  $p_j$  nor  $p_k$  of a minimum perimeter triangle  $\Delta p_i p_j p_k$  can be at a distance greater than or equal to  $\frac{\mathcal{P}_i}{2}$  from  $p_i$ .*

**Proof:** If  $|p_i p_j| \geq \frac{\mathcal{P}_i}{2}$ , then the perimeter of  $\Delta p_i p_j p_k$  is greater than  $\mathcal{P}_i$  no matter what the value of  $|p_i p_k|$  is (see Figure 2.9, where  $|p_i p_j| = \frac{\mathcal{P}_i}{2}$ ). Therefore,  $\Delta p_i p_j p_k$  cannot be of minimum perimeter. Thus  $p_j$ , and by an identical argument  $p_k$ , lies

inside a circle of radius  $\mathcal{P}_i/2$  centered at  $p_i$ . □

Let  $S(p_i, \mathcal{P}_i/2)$  denote the set of points that lie strictly inside a circle of radius  $\frac{\mathcal{P}_i}{2}$ , centered at  $p_i$ . An algorithm for the all-minimum perimeter problem, based on the above observation, is given below.

---

**Algorithm** *MinimumPerimeterTriangle*

**Input:** The set  $P$ .

**Output:** For each  $p_i \in P$ , a pair  $\{p_j, p_k\}$  such that  $\mathcal{P}(p_i, p_j, p_k)$  is minimum.

**begin**

**for** each point  $p_i \in S$  **do**

**Step 1:** Sort the points in  $P - \{p_i\}$  by their distances from  $p_i$ .

**Step 2:** Compute the perimeter  $\mathcal{P}_i$  of the triangle formed by  $p_i$  and two points closest to it.

**Step 3:** Compute  $S(p_i, \mathcal{P}_i/2)$ .

**Step 4:** while {there exists another pair of points  $p_j, p_k \in S(p_i, \mathcal{P}_i/2)$  }

**Step 4.1:** Compute perimeter  $\mathcal{P}_i'$  of  $\Delta(p_i, p_j, p_k)$ .

**Step 4.2:** If  $\mathcal{P}_i' < \mathcal{P}_i$ , set  $\mathcal{P}_i \leftarrow \mathcal{P}_i'$  and go to Step 3, else continue.

**Step 5:** Return  $\mathcal{P}_i$ .

**endfor**

**end**

---

Let  $\Delta_i$  be the smallest difference of distances of a pair of points in  $P - \{p_i\}$  relative to  $p_i$ . We determine this by sorting the distances relative to  $p_i$ . Then,  $\mathcal{P}_i/2\Delta_i$  is an upper bound on the number of points inside a circle of radius  $\mathcal{P}_i/2$ . Thus the running time of the above algorithm is  $O(n^2 \log n + \sum_i \sum_j (\mathcal{P}_i^j / 2\Delta_i)^2)$ , where, in the second term, the outer sum is over all the  $p_i$ 's, while the inner sum accounts for the number of times we reset  $\mathcal{P}_i$  for a fixed  $p_i$ .

Note that inasmuch as the time complexity for this problem depends on the  $\Delta_i$ 's as well as the input size, it is anomalous vis-a-vis the time complexities of the remaining problems studied in this chapter that depend only on the size of the input. It is an interesting open problem to obtain a solution that depends entirely on the input size.

## 2.8 Circumcircle radius measure

The circumcircle radius measure  $\mathcal{R}(p_i, p_j, p_k)$  of  $\{p_i, p_j, p_k\}$  is defined to be the radius of a circle that circumscribes the  $\triangle(p_i, p_j, p_k)$ . The computational problem is to determine for each  $p_i$  a pair  $\{p_j, p_k\}$  such that the circumradius,  $\mathcal{R}(p_i, p_j, p_k)$  of  $\{p_i, p_j, p_k\}$  is maximum (minimum).

The combinatorial complexities of the farthest and nearest 2-point site Voronoi diagrams in this distance measure were left open by Barequet et al. [24]. For the restricted query case, we can use the inversion transformation to reduce the maxi-

imum and minimum problem to finding a nearest and farthest line respectively from  $p_i$ , spanned by pairs of points in the inverted set.

An *inversion transformation* is defined as follows. Let  $C$  be a circle of unit radius centered at the origin of a rectangular coordinate system. A point  $p'$  is said to be the inverse of another point  $p$  with respect to the unit circle if  $|op| * |op'| = 1$  (Figure 10(a)). Thus inversion maps circles passing through the center of inversion into lines not passing through the center of inversion and conversely (Figure 10(b)). More details on this transformation are available in [58].

Daescu et al. [28] has given an  $O(n \log n)$  solution for the nearest and farthest line problem for a fixed  $p_i$ . Using these algorithms, we can solve the all-maximum and all-minimum radius circle problem in  $O(n^2 \log n)$  time.

In [29] Bespamyatnikh and Segal considered the problem of selecting a hyperplane spanned by  $d$  of  $n$  points in  $d$ -dimensional space the rank of whose distance from the origin is  $k$ . They showed that the 3-dimensional version of this problem is almost 3-SUM hard and proposed an  $O(n^2 \log^2 n)$  algorithm. Thus using each  $p_i$  as the origin of coordinates we can determine the farthest and nearest planes spanned by 3 points in  $P - \{p_i\}$  in the same time and hence solve the all-farthest and all-nearest problems in this measure in  $O(n^3 \log^2 n)$  time. Thus by using inversion we can solve



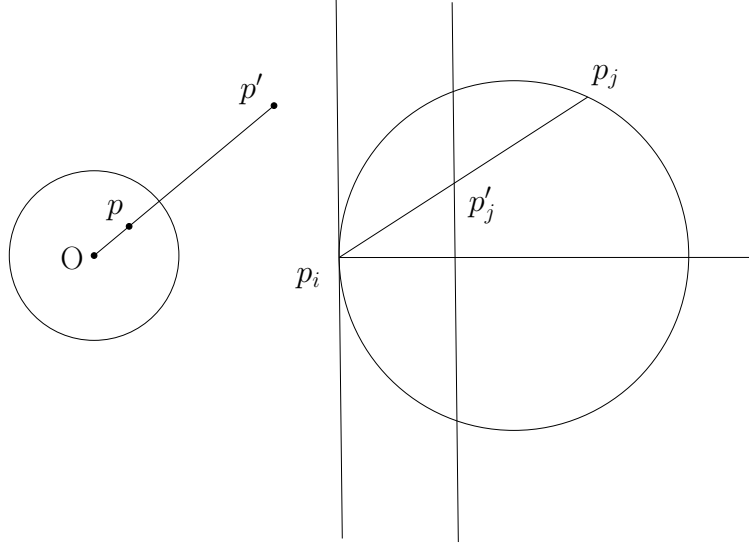


Figure 2.10: (a) Inversion transformation (b) Transforming a circle  $r = d \cos \theta$  to a line  $1/d = r' \cos \theta$

the all-nearest and all-farthest measures in the same time.

## 2.9 Triangle Distance Measure

Let  $P = \{p_1, p_2, p_3, \dots, p_n\}$  be a point set in  $E^3$ . For the purpose of our discussion below, a triangle on a set of three points  $\{p_i, p_j, p_k\}$  is the area bounded by the segments obtained by joining the points in pairs. The triangle distance measure is  $\mathcal{TD}(p_i, p_j, p_k, p_l) = d(p_i, \Delta(p_j, p_k, p_l))$ ,  $i \neq j \neq k \neq l$ , where  $d(p, \Delta)$  denotes the distance from a point  $p$  to a triangle  $\Delta$ . It is thus a generalization of the segment distance measure discussed in [26], [59], where an  $O(n \log n)$  algorithm was proposed for the all-farthest version.

The computational problem is to find for each  $p_i$  a triangle formed by 3 distinct points  $p_j, p_k, p_l \in P - \{p_i\}$  such that the distance from  $p_i$  to this triangle is maximum (minimum). Below, we discuss the all-maximum version of the problem.

### 2.9.1 Characterizing farthest triangles

Let  $\triangle(p_j, p_k, p_l)$  be a triangle that is farthest from  $p_i$ ,  $i \neq j \neq k \neq l$ . The farthest distance is realized in one of the following 3 ways.

- Type **A** distance: The perpendicular distance from  $p_i$  to an interior point of  $\triangle(p_j, p_k, p_l)$  (Figure 2.11)

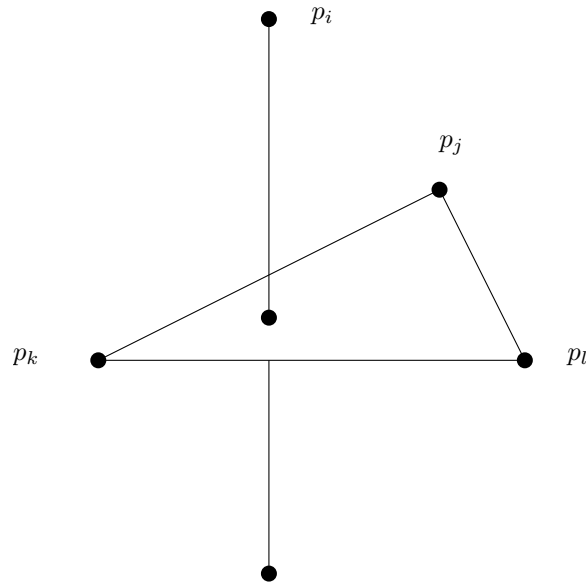


Figure 2.11: *Type A distance*

- Type **B** distance: The distance from  $p_i$  to the nearest of the vertices  $p_j$ ,  $p_k$ , or  $p_l$  of  $\triangle(p_j, p_k, p_l)$  (Figure 2.12)

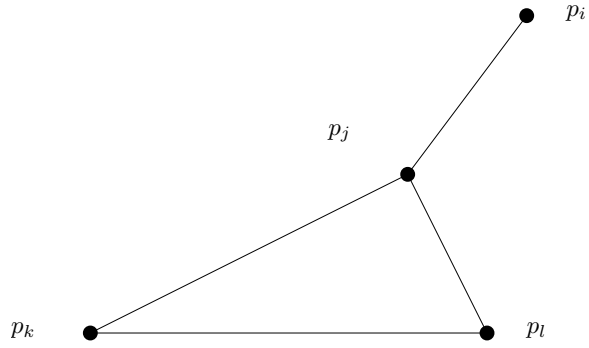


Figure 2.12: *Type B distance*

- Type C distance: The perpendicular distance from  $p_i$  to the nearest (open) edge of  $\triangle(p_j, p_k, p_l)$  (Figure 2.13).

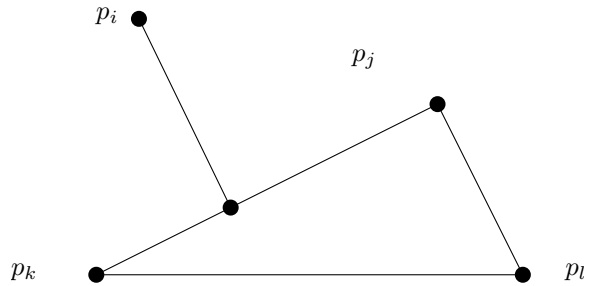


Figure 2.13: *Type C distance*

Let us call the vertices internal to the convex hull of  $P$ ,  $CH(P)$ , interior vertices. Intuitively, it seems plausible that the vertices of the triangle farthest from a given  $p_i$  should lie as far as possible on the “boundary” of the point set  $P$ . The following theorem confirms this intuition.

**Theorem 1** *If  $T = \triangle(p_j, p_k, p_l)$  is a triangle that is farthest from  $p_i$ , then  $p_j$ ,  $p_k$ , and  $p_l$  cannot all be interior vertices.*

**Proof:** Assume otherwise.

- Type **A** distance:

Let the distance from  $p_i$  to  $T$  be realized by a point  $q$  interior to  $T$ . Clearly,  $\overline{p_i q}$  is orthogonal to  $T$ . Let  $P(T)$  be the plane containing  $T$ . Consider the part of  $CH(P)$  that lies on the side of  $P(T)$  that does not contain  $p_i$ . Because of convexity, there must be a point  $p_m$  in  $P$  on this side of  $P(T)$  that is a convex-hull vertex. We claim that the triangle  $T' = \triangle(p_k, p_l, p_m)$  is farther from  $p_i$  than  $T$ . Let  $q'$  be the point on  $T'$  that is closest to  $p_i$ . Consider the sphere  $B(p_i, |\overline{p_i q}|)$ , centered at  $p_i$ , of radius  $|\overline{p_i q}|$ . The plane  $P(T)$  separates the points  $\{p_k, p_l, p_m\}$  from  $B$ , putting  $T'$  entirely outside  $B$ . Thus  $|\overline{p_i q}| < |\overline{p_i q'}|$  and the claim is established in this case.

- Type **B** distance:

Let the distance from  $p_i$  to  $T$  be realized by the segment  $\overline{p_i p_j}$ . Let  $P(p_j)$  be a plane through  $p_j$  orthogonal to  $\overline{p_i p_j}$ . From the convexity of  $CH(P)$  there exists a vertex  $p_m$  of  $CH(P)$  on the side of  $P(p_j)$  not containing  $p_i$ . Consider the sphere  $B(p_i, |\overline{p_i p_j}|)$ , centered at  $p_i$ , with radius  $|\overline{p_i p_j}|$ . Now  $\overline{p_k p_l}$  and  $B$  are on opposite sides of  $P(p_j)$ . So also are  $p_m$  and  $B$ . Thus all points of the triangle  $T' = \triangle(p_k, p_l, p_m)$  are separated from  $B$  by  $P(p_j)$ . Hence  $T'$  must be farther from  $p_i$  than  $T$ .

- Type **C** distance:

Let the distance from  $p_i$  to  $T$  be realized by the segment  $\overline{p_i q}$  orthogonal to the triangle edge  $\overline{p_j p_k}$ ,  $q$  being an internal point of this edge. Let  $P(\overline{p_j p_k})$  be a plane through  $\overline{p_j p_k}$  orthogonal to  $\overline{p_i q}$ . From the convexity of  $CH(P)$  there exists a vertex  $p_m$  of  $CH(P)$  on the side of  $P(\overline{p_j p_k})$  not containing  $p_i$ . Thus the triangle  $T' = \triangle(p_k, p_l, p_m)$  lies outside the sphere  $B(p_i, |\overline{p_i q}|)$ , centered at  $p_i$  and radius  $|\overline{p_i q}|$ . Therefore  $T'$  must be farther from  $p_i$  than  $T$ .

Thus in all cases we can find a triangle, with a vertex on  $CH(P)$ , that is farther from  $p_i$  than  $T$ . □

Therefore, the vertex configuration of a farthest triangle from a point  $p_i$  can be categorized into the following cases:

- **Case I:** One vertex on  $CH(P)$ , while the other two vertices are points internal to  $CH(P)$ ;
- **Case II:** Two of its vertices are on  $CH(P)$ , while the third vertex is a point internal to  $CH(P)$ ;
- **Case III:** All three vertices are on  $CH(P)$ .

Over the next few lemmas, we sharpen the above characterizations further to help us design efficient algorithms for computing a farthest triangle. Indeed, the first of these

shows that we can be more precise about the location of the farthest triangle when the farthest distance is of Type **A**.

**Lemma 1** *If the distance from  $p_i$  to a farthest triangle  $\Delta(p_j, p_k, p_l)$  is of type **A**, then  $\Delta(p_j, p_k, p_l)$  is a facet of  $CH(P)$ .*

**Proof:** If the triangle  $\Delta(p_j, p_k, p_l)$  is not a convex hull facet, then there exists a point  $p_m$  of  $P$  in the open half-space defined by the supporting plane through  $p_j$ ,  $p_k$ , and  $p_l$  that does not contain  $p_i$  (Figure 2.14). This gives a triangle  $\Delta(p_j, p_k, p_m)$  that is farther from  $p_i$  than  $\Delta(p_j, p_k, p_l)$  since every point on the triangle  $\Delta(p_j, p_k, p_m)$  is farther from  $p_i$  than the distance from  $p_i$  to triangle  $\Delta(p_j, p_k, p_l)$ .  $\square$

This implies that for Case **I** or Case **II**, the farthest distance cannot be of Type **A**.

**Lemma 2** *For the vertex configuration of Case **I**, let the vertices  $p_j$  and  $p_k$  of the farthest triangle  $T$  be interior vertices. Then  $\overline{p_j p_k}$  is the farthest from  $p_i$  among all edges that can be formed by taking pairs of interior vertices.*

**Proof:** Let the point  $q$  of  $T$  be closest to  $p_i$ . If  $r$  is the point on the internal edge  $\overline{p_j p_k}$  closest to  $p_i$ , then  $|\overline{p_i q}| \leq |\overline{p_i r}|$ . Assume that there is another completely internal edge,  $\overline{p_a p_b}$ , that is farthest from  $p_i$ . If  $q'$  is the point on  $\overline{p_a p_b}$  that is closest to  $p_i$ , then  $|\overline{p_i r}| < |\overline{p_i q'}|$  and hence  $|\overline{p_i q}| < |\overline{p_i q'}|$ . Consider a plane  $P(q')$  through  $q'$  orthogonal to  $\overline{p_i q'}$ . There exists a vertex  $p_c$  of  $CH(P)$  on the side of  $P(q')$  not containing  $p_i$ . Now all points of  $T' = \Delta(p_a, p_b, p_c)$  are at least as far from  $p_i$  as  $q'$  is. Thus triangle  $T'$  is

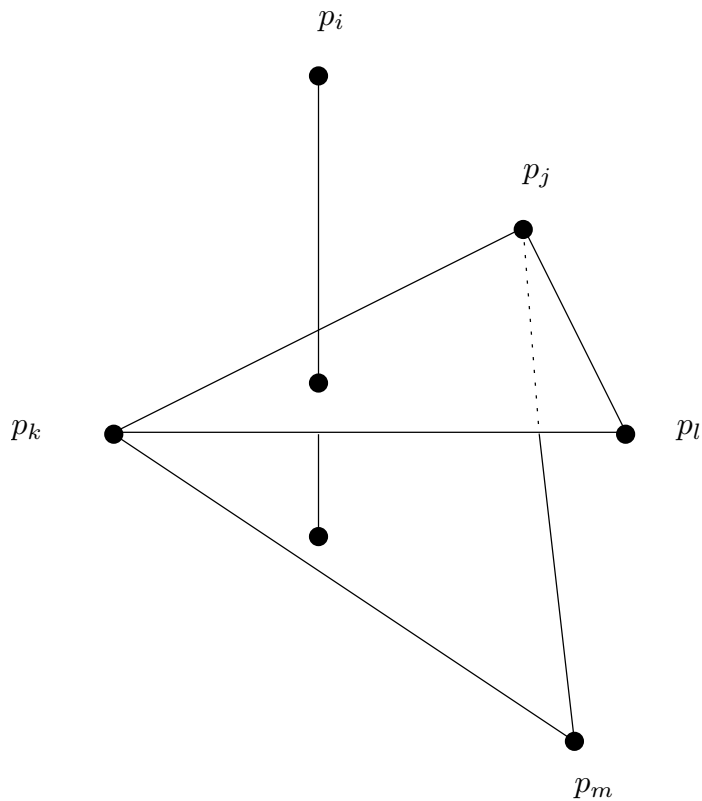


Figure 2.14: *If a type **A** triangle is not a convex hull facet*

farther from  $p_i$  than  $T$ , giving us a contradiction. □

Lemma 2 reaffirms the intuition that the vertices of the farthest triangle  $T$  should be as near the “boundary” as possible even when some of its vertices are interior vertices.

Let us call the convex hull of the points internal to  $CH(P)$ , namely  $CH(P - CH(P))$ , the iterated convex hull. With the help of this we can be more precise about the location of the internal edge in Case **I**.

**Lemma 3** *The farthest internal edge  $\overline{p_j p_k}$  is: (a) either the farthest edge of  $CH(P - CH(P))$  from  $p_i$ ; (b) or has one endpoint as a vertex of  $CH(P - CH(P))$  and the other endpoint as the farthest point of  $P - CH(P) - CH(P - CH(P))$  from  $p_i$ .*

**Proof:** Let  $p_f$  be the farthest point of  $P - CH(P) - CH(P - CH(P))$  from  $p_i$ . Let  $P(p_f)$  be the plane containing  $p_f$  that is perpendicular to  $\overline{p_i p_f}$ . There must be a vertex  $p_g$  of  $CH(P - CH(P))$  such that  $p_g$  and  $p_i$  are on opposite sides of  $P(p_f)$ . If there is another such vertex,  $p_h$ , then it must be that  $\overline{p_j p_k}$  is an edge of  $CH(P - CH(P))$ , since  $\overline{p_g p_h}$  is farther from  $p_i$  than  $p_f p_g$  is. But if  $p_g$  is the only such vertex, then  $\overline{p_j p_k}$  must be either  $\overline{p_f p_g}$  or some edge of  $CH(P - CH(P))$ . Thus in all cases the conclusion of the lemma holds.  $\square$

Note that for the above vertex-configuration, we know  $p_l$  up to being a vertex of  $CH(P)$  and the distance from  $p_i$  to  $\triangle(p_j, p_k, p_l)$  up to being of Type **B** or Type **C**.

**Lemma 4** *For the vertex configuration of Case **II**, if the vertices  $p_k$  and  $p_l$  of  $T$  lie on  $CH(P)$ , then  $\overline{p_k p_l}$  is an edge of  $CH(P)$ .*

**Proof:** Assume that  $\overline{p_k p_l}$  is not an edge of the convex hull boundary. We can once again imitate the proof of Theorem 1 to find a convex hull vertex that lies below the plane of the triangle  $\triangle(p_j, p_k, p_l)$  which, along with  $p_k$  and  $p_l$ , forms a triangle that is farther from  $p_i$  than  $T$  is, giving us a contradiction.  $\square$



The remaining (interior) vertex  $p_j$  of  $T$  has the following characterization.

**Lemma 5** For Case **II**, the vertex internal to  $CH$  lies on the iterated convex hull,  $CH(P - CH(P))$ .

**Proof:** Assume otherwise. Let  $q$  be the point of  $T$  closest to  $p_i$ . By Lemma 1,  $q$  is either an internal point of an edge or a vertex of  $T$ . Several cases arise, depending on the location of  $q$ . In each case, we find a point  $p \in P$  such that  $p$  along with 2 of the points in  $\{p_j, p_k, p_l\}$  form a triangle that is farther from  $p_i$  than  $T$ , thus obtaining a contradiction.

Let  $q = p_l$  be the point of  $T$  closest to  $p_i$ . Consider the sphere  $B(p_i, |\overline{p_i q}|)$ , center  $p_i$  and radius  $|\overline{p_i q}|$ , and a plane  $\pi$  tangent to it at  $p_l$ . Let  $\pi'$  be a plane through  $p_j$  parallel to  $\pi$ . Two cases arise:

(a) Vertex  $p_k$  of triangle  $T$  is below  $\pi'$

Let  $p_a$  be a vertex of the iterated hull that also lies below  $\pi'$ . Such a vertex exists since  $p_j$  is assumed to be internal to the iterated hull  $CH(P - CH(P))$ . Thus the triangle  $T' = \Delta(p_j p_k p_a)$  is farther from  $p_i$  than  $T$  since its vertices are separated from  $B$  by the plane  $\pi$ .

(b) Vertex  $p_k$  of triangle  $T$  is above  $\pi'$

In this case we consider a plane  $\pi''$  through  $p_k$  parallel to  $\pi$ . Let the point  $p_a$  be as in the previous case. Again the triangle  $T' = \triangle(p_j p_k p_a)$  is farther from  $p_i$  than  $T$  since its vertices are separated from  $B$  by the plane  $\pi$ .

If  $q$  is interior to  $\overline{p_k p_l}$ , consider a sphere  $B(p_i, |\overline{p_i q}|)$ , center  $p_i$  and radius  $|\overline{p_i q}|$ , and a plane  $\pi$  tangent to it at  $q$ . Let  $\pi'$  be a plane through  $p_j$  parallel to  $\pi$ . Since, by assumption,  $p_j$  is an interior point of  $CH(P - CH(P))$ , there exists a point  $p_a$  on the iterated hull that lies below  $\pi'$ . The triangle  $T' = \triangle(p_j, p_k, p_a)$  is farther from  $q$  than the triangle  $T$  since the segment joining any point on  $T'$  to  $q$  intersects the plane  $\pi$ .

The above argument holds even in the limiting case when any of the points  $p_k$  or  $p_l$  lies on  $\pi$ .

Let  $q$  be on  $\overline{p_j p_k}$  or  $\overline{p_j p_l}$ . Consider the plane  $P(q)$  through  $q$  that is orthogonal to  $\overline{p_i q}$ . If  $q$  is  $p_j$ , then  $P(q)$  contains  $p_j$ . If  $q$  is an internal point of  $\overline{p_j p_k}$  ( $\overline{p_j p_l}$ ), then  $\overline{p_j p_k}$  ( $\overline{p_j p_l}$ ) is perpendicular to  $\overline{p_i q}$  and so  $P(q)$  contains  $p_j$ . and so there must be some point  $p_b$  on  $CH(P - CH(P))$  such that  $p_b$  and  $p_i$  are on opposite sides of  $P(q)$ . Then  $T'' = \triangle(p_b, p_k, p_l)$  is farther from  $p_i$  than  $T$  is.  $\square$

Thus for the vertex configuration of Case **II**, we know the edge  $\overline{p_k p_l}$  up to being an edge of  $CH(P)$  and the vertex  $p_j$  up to being a vertex of the iterated convex hull  $CH(P - CH(P))$ .

**Lemma 6** *For the vertex configuration of Case **III**,  $T$  is a facet of  $CH(P)$ .*

**Proof:** If we assume that at least one pair of the convex hull vertices  $p_j$ ,  $p_k$ , and  $p_l$  are not adjacent in the face-graph structure of  $CH(P)$  so that  $\Delta(p_j, p_k, p_l)$  is not a facet of  $CH(P)$ , then the arguments adduced in the proof of Theorem 1 shows that  $T$  cannot be the farthest triangle. Of course, the farthest triangle will be the farthest facet. □

### 2.9.2 Algorithm

For each  $p_i$  in  $P$ , the farthest triangle is found by finding a farthest triangle for each of the 3 types of vertex configuration and then returning the farthest of the 3 as the answer.

We start with the simplest vertex configuration, viz., Case **III**. In this case, we find a convex hull facet that is farthest from  $p_i$ . Assuming that  $CH(P)$  is fully triangulated, the number of facets are in  $O(h)$ . So we can find the farthest facet by a simple brute-force search whose complexity is in  $O(h)$ . Here, it would help to have a farthest-triangle Voronoi diagram in the special case that the triangles are the facets

of a convex polyhedron.

For the vertex configuration of Case **II**, we proceed in somewhat a brute-force manner. We consider all the triangles that can be formed by choosing an edge on the convex hull of  $P$ ,  $CH(P)$ , and a vertex on the iterated convex hull  $CH(P - CH(P))$  and compute the minimum distance from the query point to these triangles, returning the corresponding triangle. The complexity of this is in  $O(hh')$  as the number of edges of  $CH(P)$  are in  $O(h)$  and the number of vertices of the iterated convex hull of  $P$  is in  $O(h')$ .

Finally, we consider the vertex configuration of Case **I**. The search for the edge  $\overline{p_j p_k}$  of the farthest  $\triangle(p_j, p_k, p_l)$  is guided by Lemmas 2 and 3. Since the farthest edge on  $CH(P - CH(P))$  from  $p_i$  is a candidate, we first find this edge  $\overline{p_a p_b}$ . Here once again we remark that it would be helpful to have a farthest-segment Voronoi diagram when the segments are edges of a convex polytope.

For the other candidate edge, we find the farthest point  $p_f$  in  $P - CH(P) - CH(P - CH(P))$  from  $p_i$ . Here we take advantage of a farthest-point Voronoi diagram for the point set  $P - CH(P) - CH(P - CH(P))$ . To find the point on  $CH(P - CH(P))$ , we shoot a ray  $\vec{r}$  from  $p_i$ , through  $p_f$ , to intersect  $CH(P - CH(P))$ , using an algorithm due to Matousek and Schwarzkopf [60]. We set the other end point  $p_g$  to one of the

vertices of the facet that is hit by the ray.

Of the segments  $\overline{p_a p_b}$  and  $\overline{p_f p_g}$ , we set  $\overline{p_j p_k}$  to be the one that is farther from  $p_i$ .

Next, we locate a vertex  $p_l$  on  $CH(P)$ , and join it to  $\overline{p_j p_k}$  to complete the construction of the farthest triangle. Let  $q$  be a point on the support line of  $\overline{p_j p_k}$  such  $\overline{p_i q}$  is orthogonal to it. We now shoot a ray  $\vec{r}$  from  $p_i$  in the direction of  $q$  to hit  $CH(P)$ . Let  $P(\overline{p_j p_k})$  be the plane containing  $\overline{p_j p_k}$ , orthogonal to  $\vec{r}$ . We look at the vertices adjacent to the face of  $CH(P)$  that was hit by  $\vec{r}$ . Assuming  $CH(P)$  has been triangulated, we have to examine at most three vertices before we find one that is on the opposite side of  $P(\overline{p_j p_k})$  (we are guaranteed to find at least one because of convexity). We only need to find one vertex on that side of  $P(\overline{p_j p_k})$  because if there is more than one, then Case **I** does not result in a farthest triangle. We set this vertex to  $p_l$ .

Let  $h$  be the number of vertices of  $CH(P)$  and  $h'$  the number of vertices of  $CH(P - CH(P))$ . The complexity of finding a farthest internal edge is in  $O(h' + (n - h - h' + \log h'))$ , where the first term accounts for the case when the farthest edge lies on the iterated hull boundary. The second term accounts for the case when the farthest internal edge has one point internal to the iterated hull boundary that is farthest from  $p_i$ , and the other end point is obtained by ray-shooting, following [60]. The complexity of finding the third vertex on the outer hull boundary is in  $O(\log h)$ , found again

by ray-shooting, following [60]. Thus the complexity of finding a farthest triangle of this type is in  $O(h' + (n - h - h') + \log h' + \log h)$ .

The farthest of the three triangles found from the three cases above gives us the triangle  $\Delta(p_j, p_k, p_l)$  that is farthest from  $p_i$ . Summarizing the above discussions, we have the following theorem.

**Theorem 2** *The complexity of the all-farthest triangles problem is in  $O(nhh')$ , where  $h$  is the number of vertices of  $CH(P)$  and  $h'$  the number of vertices of  $CH(P - CH(P))$ .*

**Proof.** This follows from the fact that the complexity of computing the two iterated convex hulls is in  $O(n \log n + (n - h) \log(n - h))$  [46], while the complexity of finding a farthest triangle for a point  $p_i$  is in  $O(hh')$ . □

## 2.10 Summary

We have made an exhaustive study of a restricted kind of proximity problem under various measures. A number of problems remain open. These are: (a) an  $O(n^2)$  deterministic algorithm for the all-minimum problem in the line-difference measure; (b) closing the complexity gaps for the all-maximum problems for the area and perimeter measures; (c) improving the complexity of the all-minimum problem in the perime-

ter measure and establishing a corresponding lower bound; (d) whittling away the  $\log n$  factor from the complexities of the all-minimum and all-maximum problems in the circumcircle measure; (e) the design of an  $O(n^3)$  algorithm for the all-minimum problem in the triangle distance measure to improve on the trivial  $O(n^4)$  algorithm; for this last problem, an effective characterization will have to be found as a first step.

In [28] a randomized algorithm was suggested for finding the  $k$ -th closest distance from a given point  $q$  to a line determined by a pair of  $n$  given points whose time complexity is in  $O(n \log n)$ . It would be interesting to design algorithms for the all- $k$ -closest problems in all of the above the measures we have discussed.

In the line of the study in [27], the problems of constructing the farthest-segment Voronoi diagram of a set of segments that are edges of a convex polytope, or the farthest-triangle Voronoi diagram of the facets of a triangulated polytope are also worthy of further investigation.

## Chapter 3

# An Incremental Linear Programming Based Tool for Analyzing Gene Expression Data

### 3.1 Overview

The availability of large volumes of gene expression data from microarray analysis (cDNA and oligonucleotide) has opened a new door to the diagnoses and treatments of various diseases based on gene expression profiling. In this chapter [61], we discuss a new profiling tool based on linear programming. Given gene expression data from two subclasses of the same disease (e.g. leukemia), we are able to determine efficiently if the samples are linearly separable with respect to triplets of genes. This was left as an open problem in an earlier study that considered only pairs of genes as linear separators. Our tool comes in two versions - offline and incremental. Tests show that the incremental version is markedly more efficient than the offline one. This chapter also introduces a gene selection strategy that exploits the class distinction property of a gene by separability test by pairs and triplets. We applied our gene selection strategy to 4 publicly available gene-expression data sets. Our experiments show that gene spaces generated by our method achieves similar or even better classification accuracy than the gene spaces generated by  $t$ -values, FCS(Fisher Criterion Score)



and SAM(Significance Analysis of Microarrays).

## 3.2 Introduction

The availability of large volumes of gene expression data from microarray analysis (cDNA and oligonucleotide) has opened a new door to the diagnoses and treatments of various diseases based on gene expression profiling.

In a pioneering study, Golub et al [62] identified a set of 50 genes that can distinguish an unknown sample with respect to 2 kinds of leukemia with a low classification error rate. Following this work, other researchers attempted to replicate this effort in the diagnoses of other diseases. There were several notable successes. van't Veer *et al.* [63] found that 231 genes are significantly related to breast cancer. Their FDA approved MammaPrint uses 70 genes as biomarkers to predict the relapse of breast-cancer in patients whose condition has been detected early [63]. Khan *et al.* [64] found 96 genes to classify small, round, blue-cell cancers. Ben-Dor *et al.* [65] used 173-4,375 genes to classify various cancers. Alon *et al.* [66] used 2,000 genes to classify colon cancers.

A major bottleneck with any classification scheme based on gene expression data is that while the sample size is small, numbering in hundreds, the feature space is much larger, running into tens of thousands of genes. Using too many genes as classifiers results in over-fitting, while using too few leads to under-fitting. Thus the main dif-

faculty of this effort is one of scale: the number of genes is much larger than the number of samples. The consensus is that genes numbering between 10 and 50 may be sufficient for good classification [62, 67].

In [31], Computational Geometry tools were used for testing the linear separability of gene expression data by pairs of genes. Applying their tool to 10 different publicly available gene-expression data-sets, they determined that 7 of these are highly separable. From this they inferred that there might be a functional relationship “between separating genes and the underlying phenotypic classes”. Their method of linear separability, applicable to pairs of genes only, checks for separability incrementally. For separable datasets, the running time is quadratic in the sample size  $m$ .

Alam et al. [32] in a short abstract, proposed a different geometric tool for testing the separability of gene expression data sets. This is based on a linear programming algorithm of Megiddo [68–70] that can test linear separability with respect to a fixed set of genes in time proportional to the size of the sample set.

In this chapter, we extend this work to testing separability with respect to triplets of genes. Since most gene sets do not separate the sample expression data, we have proposed and implemented an incremental version of Megiddo’s scheme that terminates as soon as linear inseparability is detected. The usefulness of such incremental

algorithm to detect inseparability in gene expression dataset is also observed by [31]. The performance of the incremental version turned out to be better than the offline version when we tested the separability of 5 different data sets by pairs/triplets of genes. In the chapter we have also conclusively demonstrated that linear separability can be put to good use as feature for classification. The chapter also reformulates Unger and Chor's method as a linear programming framework. A conference version of the chapter is appeared in the proceedings of ICCSA 2013 [61].

In a study, Anastassiou [71] reveals that diseases (e.g. cancer) are due to the collaborative effect of multiple genes within complex pathways, or to combinations of multiple SNPs. Motivated by this, we illustrate the effect of the separability property of a gene to build a good classifier. In order to do so this chapter introduces a gene selection strategy, based upon the individual ranking of a gene. The ranking scheme uses the above geometric tools and exploits class distinction of a gene by testing separability with respect to pairs and triplets of genes.

An important biological consequence of perfect linear separability in low dimensions is that the participating genes can be used as biomarkers. These genes can be used in clinical studies to identify samples from the input classes. This objective of linear separability in low dimensions can be achieved in an efficient way by an adaptation of Megiddo's algorithm. Since in gene expression dataset(s) the total number of possible

combination of genes which may be considered for good classification is too high, we are justified in confining ourselves to separability in low dimensions and thus limiting the group size to pairs and triplets. Furthermore, taking a cue from the observation in [31] that most of gene pairs are not separating, we have laid particular emphasis on an incremental version of Megiddo's algorithm that is more efficient in this situation than an offline one. For any classification purpose as groups of two or three genes may lead to under-fitting we have also discussed a feature selection method by using the above geometric tool of linear separability.

The major contributions of this chapter can be summarized as follows:

1. An offline adaptation of Megiddo's algorithm to test separability by gene pairs/triplets, fully implemented and tested.
2. An incremental version of Megiddo's algorithm that is particularly useful for gene expression datasets, fully implemented and tested.
3. Demonstration of the usefulness of linear separability as a tool to build a good classifier with application to concrete examples.
4. Reformulation of Unger and Chor's method [31] in a linear programming framework.

For the completeness of the chapter, in the following section we briefly discuss about LP formulation of separability [32], [61].

### 3.3 LP Formulation of Separability

We have  $m$  samples,  $m_1$  from a cancer type  $C_1$  and  $m_2 (= m - m_1)$  from a cancer type  $C_2$  (for example,  $m_1$  from ALL and  $m_2$  from AML [62]). Each sample is a point in a  $d$ -dimensional Euclidean space, whose coordinates are the expression values of the samples with respect to the  $d$  selected genes. This  $d$ -dimensional space is called the *primal space*. If a hyperplane in this primal space separates the sample-points of  $C_1$  from those of  $C_2$ , then the test group of genes is a linear separator and the resulting linear program in dual space has a feasible solution. Suppose there is a separating hyperplane in primal space and, say, the sample points of  $C_1$  are above this plane, while the sample points of  $C_2$  are below (Figure 3.1(a) is a 2-dimensional illustration of this). Figure 3.1(b) shows that the separating line maps to a point inside a convex region. The set of all points inside this convex region make up the feasible region of a linear program in dual space and correspond to all possible separating lines in primal space.

Thus there is a separating hyperplane in primal space if the resulting linear program in dual space has a feasible solution. Note, however, that we will have to solve 2 linear programs since it is not known a priori if the  $m_1$  samples of  $C_1$  lie above or below the separating hyperplane  $H$ .

Thus if  $d = 2$  and the selected genes are  $g_1$  and  $g_2$ , then the gene pair  $\{g_1, g_2\}$  is a

linear separator of the  $m_1$  samples from  $C_1$  and the  $m_2$  samples from  $C_2$ .

We reformulate the above problem as a linear program in *dual space*. This is another  $d$ -dimensional Euclidean space such that points(planes) in the primal space are mapped into planes(points) in this space such that if a point is above(below) a plane in the primal space, the mapping preserves this point-plane relationship in the dual space. For  $d = 2$ , read the text of this paragraph by substituting all occurrences of the word “plane” with the word “line”.

For  $d = 2$  (see [72]), one such mapping of a point  $p$  and a line  $l$  in the primal space  $(x, y)$  to the line  $p^*$  and the point  $l^*$  respectively in the dual space  $(u, v)$  is:

$$\begin{aligned} p = (p_x, p_y) &\rightarrow p^* : v = p_x u - p_y \\ l : y = l_u x - l_v &\rightarrow l^* = (l_u, l_v) \end{aligned} \tag{3.1}$$

It is straightforward to extend this definition to any dimension greater than 2.

Suppose there is a separating hyperplane in primal space and, say, the sample points of  $C_1$  are above this plane, while the sample points of  $C_2$  are below (Figure 3.1(a) is a 2-dimensional illustration of this). Figure 3.1(b) shows that the separating line maps to a point inside a convex region. The set of all points inside this convex region

make up the feasible region of a linear program in dual space and correspond to all possible separating lines in primal space.

Thus there is a separating hyperplane in primal space if the resulting linear program in dual space has a feasible solution. Note, however, that we will have to solve 2 linear programs since it is not known a priori if the  $m_1$  samples of  $C_1$  lie above or below the separating hyperplane  $H$ .

Formally, one of these linear programs in  $d$ -dimensional dual space

$(u_1, u_2, \dots, u_d)$  is shown below:

minimize  $u_d$

$$p_1^i u_1 + \dots + p_{d-1}^i u_{d-1} - u_d - p_d^i < 0, \quad i = 1, \dots, m_1 \tag{3.2}$$

$$p_1'^i u_1 + \dots + p_{d-1}'^i u_{d-1} - u_d - p_d'^i > 0, \quad i = 1, \dots, m_2$$

where  $(p_1^i, p_2^i, \dots, p_d^i)$  is the  $i$ -th sample point from  $C_1$ , and the first set of  $m_1$  linear inequalities express the conditions that these sample points are above the separating plane, while the second set of  $m_2$  linear inequalities express the conditions that the sample points  $(p_1'^i, p_2'^i, \dots, p_d'^i)$  from  $C_2$  are below this plane. The linear inequalities above that describe the linear program are called constraints, a term that we shall also use from now on.

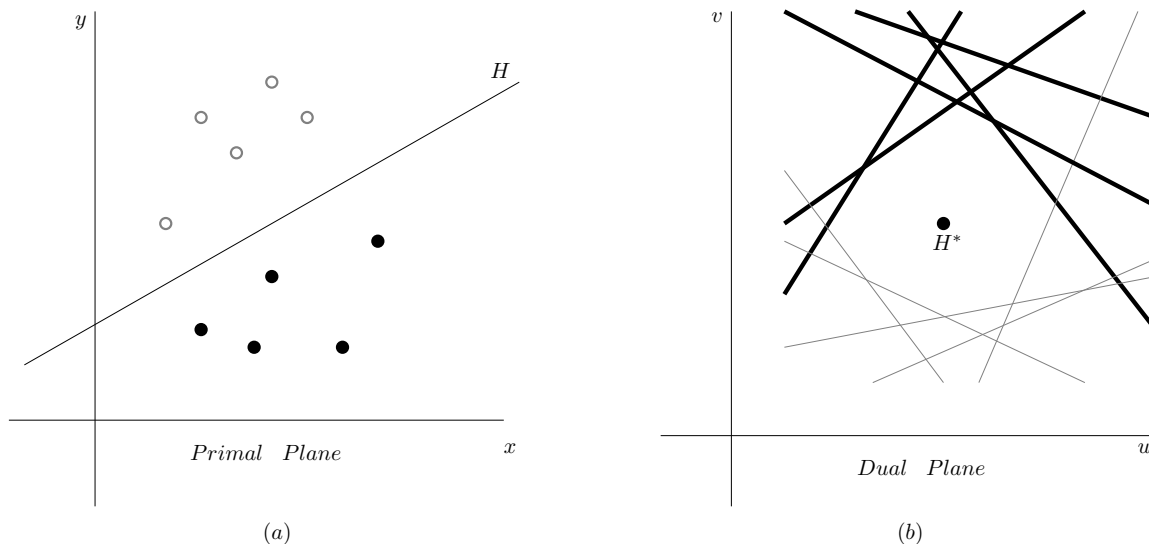


Figure 3.1: A separating line  $H$  in primal space is a feasible solution  $H^*$  in dual space.

Megiddo in [68, 69] and Dyer in [70] both proposed an ingenious prune-and-search technique for solving the above linear program that, for fixed  $d$  (dimension of the linear program), takes time linear in (proportional to) the number of constraints. Over the next two sections, we discuss how the above LP-framework achieve the more limited goal of testing separability of the samples from the two input classes by an offline algorithm and an incremental one, both of which are based on an adaptation Megiddo's (and Dyer's) technique.

This approach is of interest for two reasons: (a) in contrast to the algorithm of Unger and Chor [31], the *worst case* running time of this algorithm is linear in the sample size; and (b) in principle it can be extended to study the separability of the sample



classes with respect to *any number of genes*.

In what follows, we adopt a coloring scheme to refer to the points that represent the samples: those in the class  $C_1$  are colored blue and make up the set  $S_B$ , while those in  $C_2$  are colored red and make up the set  $S_R$ .

If a pair of genes separate the sample classes, then a (blue) segment that joins a pair of blue points is disjoint from a (red) segment that joins a pair of red points. Unger and Chor (p. 375, para. 6) [31] suggests an algorithm to test separability by testing if each blue segment is disjoint from a red segment. Figure 3.2 shows that the above test succeeds even when the point sets are not separable. Unger and Chor's [31] conclusions on separability by pairs of genes is, however, based on an incremental algorithm that works correctly. Its extension to testing separability by 3 or more genes is not obvious.

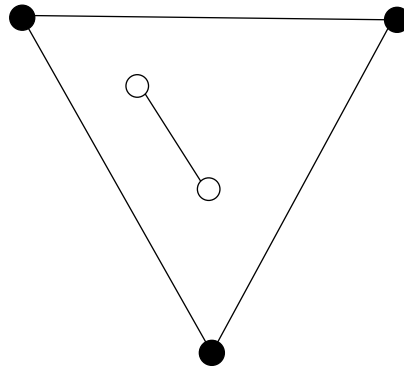


Figure 3.2: *A counterexample: black circles represent red points, white ones blue*

### 3.4 Offline Approach

As Megiddo's algorithm is central to our discussion, we briefly review this algorithm for  $d = 2$  and refer the reader to [69] for the cases  $d \geq 3$ . A bird's eye view is this: in each of  $\log m$  iterations it prunes away at least a quarter of the constraints that do not determine the optimum (minimum in our formulation), at the same time reducing the search space (an interval on the  $u$ -axis) in which the optimal solution lies.

**Definition:** If  $f_1, f_2, \dots, f_n$  is a set of real single-valued functions defined in an interval  $[a, b]$  on the real line, their point wise minimum (maximum) is another function  $f$  such that for every  $x \in [a, b]$

$$f(x) = \min(f_1(x), f_2(x), \dots, f_n(x)) \quad (f(x) = \max(f_1(x), f_2(x), \dots, f_n(x)))$$

Let us call the point wise minimum (maximum) of the heavy (light) lines in Figure 3.1(b) the *min-curve* (*max-curve*). These are also called the upper and lower envelope respectively.

Assume that after  $i$  iterations we have determined that the minimum lies in the interval  $[u_1, u_2]$ . Let us see how to prune redundant constraints from the set of constraints that determine the min-curve. We make an arbitrary pairing of the bounding lines of these constraints. With respect to the interval  $[u_1, u_2]$ , the intersection of such a constraint pair can lie as shown in Figure 3.3(a). For the ones that lie to the left

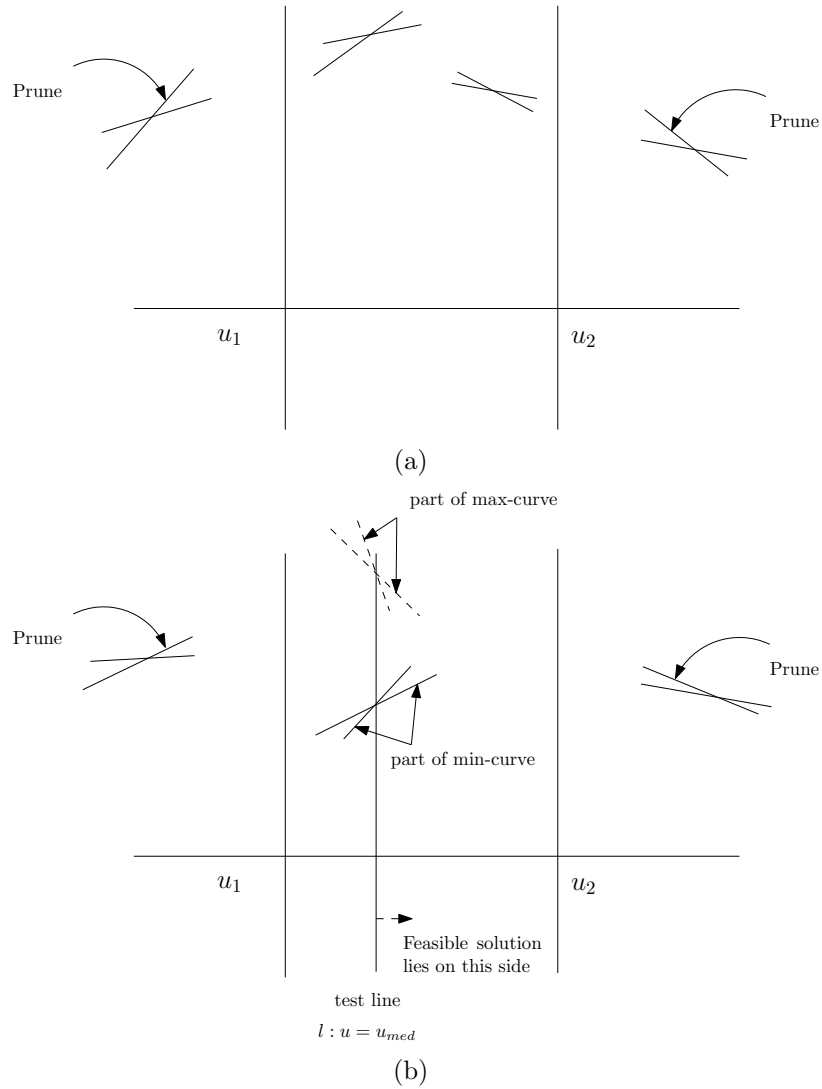


Figure 3.3: (a) Pruning Constraints (b) Testing Feasibility

(right) of the line  $u = u_1$  ( $u = u_2$ ), we prune the constraint whose bounding line has larger (smaller) slope. We can likewise prune redundant constraints from the set of constraints that determine the max-curve.

In order to further narrow down the interval on the  $u$ -axis where the minimum lies,

for all other pairs of constraints whose intersections lie within the interval  $[u_1, u_2]$ , we find the median  $u_{med}$  of the  $u$ -coordinates of the intersections and let  $l : u = u_{med}$  be the line with respect to which we test for the location of the minimum. We do this test by examining the intersections of the min-curve and max-curve with  $l$ . This is accomplished by using the residual constraint sets that implicitly define the min-curve and max-curve. From the relative positions of these intersections and the slopes of the bounding lines of the constraints that determine these intersections, we can determine on which side of  $l$ , the minimum lies (see Figure 3.3(b)). Next we prune a constraint from each pair whose intersections lie within  $[u_1, u_2]$  but on the side opposite to which the minimum lies. Because of our choice of the test-line, we are guaranteed to throw a quarter of the constraints from those that determine these intersections. We now reset the interval that contains the minimum to  $[u_{med}, u_2]$  or  $[u_1, u_{med}]$ .

The above algorithm allows us to determine feasibility as soon as we have found a test line such that the intersection of the min-curve with  $l$  lies above its intersection with the max-curve.

We have implemented the above offline algorithm both in 2 and 3 dimensions from scratch. Ours is probably the first such implementation in 3 dimensions. In Appendix A we provide the pseudo code for offline implementation for both 2 and 3 dimensions.

Offline algorithms are effective for determining linear separability. However, as most of the gene-pairs and gene-triplets are not linearly separating, incremental algorithms would be more efficient than offline ones. This was also observed by Unger and Chor [31]. In view of this, in the next section we discuss in details an incremental version of the above algorithm.

## 3.5 Incremental Approach

The following obvious but useful theorem (true in any dimension  $d \geq 1$ ) underlies our algorithm in dual space.

**Theorem 3** *Let  $S'_B$  and  $S'_R$  be arbitrary subsets of  $S_B$  and  $S_R$  respectively. If  $S'_B$  and  $S'_R$  are linearly inseparable, then so are  $S_B$  and  $S_R$ .*

**Proof:** Straightforward, since if  $S_B$  and  $S_R$  are linearly separable, then so are  $S'_B$  and  $S'_R$ . □

### 3.5.1 Incremental approach-2d

First, we choose a small constant number of lines from each of the duals of  $S_R$  and  $S_B$ , and use the offline approach of the previous section to determine if there is a feasible solution to this constant-size problem. If not, we declare infeasibility (Theorem 3)

and terminate. Otherwise, we have an initial feasible region and a test-line  $l : u = \bar{u}$ .

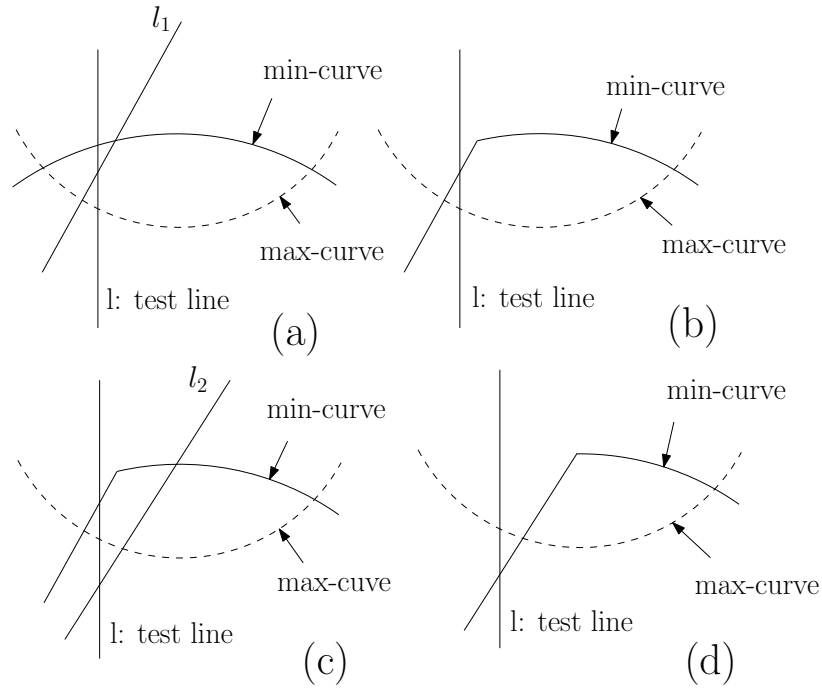


Figure 3.4: *Updation of min-curve (a) addition of line  $l_1$  (b) test line continues to pass through feasible region on updation of min-curve (c) addition of line  $l_2$  (d) test line goes out of feasible region on updation of min-curve*

We continue, adding a line from one of the residual sets  $S_A^*$  or  $S_B^*$ , also chosen randomly. Several cases arise. This line (a) either becomes a part of the boundary of the feasible region, or (b) leaves it unchanged or (c) establishes infeasibility, in which case the algorithm terminates. Case(a) spawns two sub-cases as shown in Figure 3.4. (a.1) the test line  $l$  still intersects the feasible region. (a.2) the test line  $l$  goes outside the feasible region. The lines belonging to the case (b) always leads to a condition mentioned in (a.1).

If the test line  $l$  goes outside the feasible region, constraint-pruning is triggered. This consists of examining pairs of constraints whose intersections lie on the side of  $l$  that does not include the feasible region. One of the constraints of each such pair does not intersect the feasible region and is therefore eliminated from further consideration. However, if  $l$  lies inside the new feasible interval, we continue to add new lines.

If we are able to add all lines without hitting case (c), then we have a feasible region and hence a separating line in the primal space.

A formal description of the iterative algorithm is as below.

---

**Algorithm** IncrementallySeparatingGenepairs

**Input:** Line duals  $S_R^*$  and  $S_B^*$  of the point sets  $S_R$  and  $S_B$ .

**Output:** LP feasible or infeasible.

**1:** Choose  $S_R^{*'} \subset S_R^*$  and  $S_B^{*'} \subset S_B^*$  so that  $|S_R^{*'}| = |S_B^{*'}| = 2$ .

**2:** Apply the offline approach to  $S_R^{*'}$  and  $S_B^{*'}$ , distinguishing between the following cases:

Case 1: If infeasible then report this and halt.

Case 2: If feasible then return the vertical test line  $l$  and continue with Step 3.

**3:** Repeatedly add a line from  $|S_R^* - S_R^{*' }|$  or  $|S_B^* - S_B^{*' }|$  until no more lines remain to be added or there exists no feasible point on the test line  $l$ .

**4:** If there is a feasible point on the test line  $l$  we report separability and halt.

**5:** If there is no feasible point on the test line  $l$ , determine on which side of  $l$  the feasible solution lies.

**6:** Update  $S_R^{*'}$  and  $S_B^{*'}$  by eliminating a line from each pair whose intersection does not lie in the feasible region and was earlier used to determine  $l$ .

**7:** Update  $S_R^{*'}$  and  $S_B^{*'}$  by including all those lines added in step 3 and go to Step 2.

---

When  $S_R$  and  $S_B$  are linearly separable the running time of the incremental algorithm is linear in the total number of inputs. Otherwise, as the algorithm terminates when a line added that reveals inseparability, the time complexity for this case is linear in the number of lines added so far.

In an Appendix A we provide the pseudo code for the extension of this incremental algorithm to 3 dimensions.

**Theorem 4** *If  $m$  is the total number of samples then time complexity of the incremental algorithm is  $O(m)$ .*

**Proof:** In each iteration, the algorithm prunes one quarter of the constraints (i.e. samples) from the current set  $S_R^* \cup S_B^*$ . The time complexity of each iteration is  $O(m)$ . The run-time  $T(m)$  satisfies the recurrence  $T(m) = O(m) + T(\frac{3m}{4})$ , whose solution is  $T(m) = O(m)$ . □

### 3.5.2 Incremental approach-3d

Suppose constraints (i.e. planes) belong to three-dimensional Cartesian coordinate system with axes labelled as U, V and Z and the position of any point in three-dimensional space is given by an ordered triple of real numbers  $(u_1, u_2, u_3)$ . These



numbers giving the distance of that point from the origin measured along the axes.

First, we apply 3 dimensional offline approach on a small constant numbers constraints (i.e. planes) from each duals of  $S_R$  and  $S_B$ . If this constant size problem is infeasible then we report infeasibility and terminate (see Theorem 3). Otherwise, we have a vertical test plane that pass through the feasible region. This vertical test plane is parallel to either VZ-plane (say  $\bar{U}$ ) or UZ-plane (say  $\bar{V}$ ) and chosen suitably (see pseudo-code in the appendix) as suggested by Megiddo.

A formal description of the iterative algorithm is as below.

---

**Algorithm** IncrementallySeparatingGeneTriplets

**Input:** Plane duals  $S_R^*$  and  $S_B^*$  of the point sets  $S_R$  and  $S_B$ .

**Output:** LP feasible or infeasible.

**1:** Initialize  $S_R^{*'} \subset S_R^*$  and  $S_B^{*'} \subset S_B^*$ . We can choose  $|S_R^{*'}| = |S_B^{*'}| = 4$ .

**2:** Apply Megiddo's approach to  $S_R^{*'}$  and  $S_B^{*'}$ . We distinguish with following cases

*Case 1:* If infeasible then report the inseparability and halt.

*Case 2:* If feasible then return a vertical test plane  $\bar{U}$  (or  $\bar{V}$ ) and continue with step 3.

**3:** Repeatedly add a constraint from  $|S_R^* - S_R^{*'}$  or  $|S_B^* - S_B^{*'}$  and initiate an incremental 2-D approach on vertical test plane  $\bar{U}$  (or  $\bar{V}$ ). We distinguish with following cases

*Case 1:* If the test plane  $\bar{U}$  (or  $\bar{V}$ ) is feasible after all the constraints being considered then report separability and halt.

*Case 2:* If the test plane  $\bar{U}$  (or  $\bar{V}$ ) is infeasible then solve two 2D linear program to determine which side of test plane the feasible region lies.

*Case 2.1:* If any one of 2D linear program is feasible then identify the side of feasible solution and continue with step 4.

*Case 2.2:* If both 2D linear program are not feasible or both are feasible then report the inseparability and halt.

4: Identify a second vertical test plane  $\bar{V}$  (or  $\bar{U}$ ) and initiate an incremental 2-D approach by resuming the addition of constraints from  $|S_R^* - S_R^{*'}|$  or  $|S_B^* - S_B^{*'}|$  excluding those which are already being considered with  $\bar{U}$  (or  $\bar{V}$ ). In case if we do not have any second vertical test plane then continue with step 5. We distinguish with following cases

*Case 1:* If the test plane  $\bar{V}$  (or  $\bar{U}$ ) is feasible after all the constraints being considered then report separability and halt.

*Case 2:* If the test plane  $\bar{V}$  (or  $\bar{U}$ ) is infeasible then solve two 2D linear program to determine which side of test plane the feasible region lies.

*Case 2.1:* If any one of 2D linear program is feasible then identify the side of feasible solution and continue with step 5.

*Case 2.2:* If both 2D linear program are not feasible or both are feasible then report the inseparability and halt.

5: Update  $S_R^{*'}$  and  $S_B^{*'}$  by eliminating a constraint from each coupled line which does not pass through the feasible quadrant  $\bar{U}$  and  $\bar{V}$ .

6: Update  $S_R^{*'}$  and  $S_B^{*'}$  by including all those constraints considered in step 3 and step 4.

7: Repeat the algorithm for updated set of  $S_R^{*'}$  and  $S_B^{*'}$ .

---

### 3.5.3 Linear programming formulation of Unger and Chor's incremental algorithm

In [31], Unger and Chor proposed an incremental algorithm for testing separability with respect to gene pairs. They consider  $m_1.m_2$  vectors, obtained by joining every point in the class  $S_R$  (or  $S_B$ ) to every point of the class  $S_B$  (or  $S_R$ ) (see Figure 3.5).

The directions corresponding to these vectors map to  $m_1.m_2$  points on a unit circle, with center at  $O$ . In this formulation, the sample classes  $S_R$  and  $S_B$  are linearly separable if the points on the unit circle span an arc less than  $\pi$ .

We can reformulate this in our linear programming framework. Let  $p_i(x_i, y_i), 1 \leq i \leq m_1 * m_2$  be the coordinates of the points corresponding to all the directions on the perimeter of the unit circle. We have the following observation.

**Observation 1** *The points  $p_i(x_i, y_i), 1 \leq i \leq m_1 * m_2$ , span an angle less than  $\pi$  iff there exists a line,  $l$ , through  $O$  such that all the points lie on one side of it.*

$$\begin{aligned}
 & \text{maximize/minimize } u \\
 & u > \frac{y_i}{x_i}, x_i > 0 \\
 & u < \frac{y_i}{x_i}, x_i < 0
 \end{aligned} \tag{3.3}$$

Or

$$\begin{aligned}
 & \text{maximize/minimize } u \\
 & u < \frac{y_i}{x_i}, x_i > 0 \\
 & u > \frac{y_i}{x_i}, x_i < 0
 \end{aligned} \tag{3.4}$$

We summarize the above discussion in the following claim:

**Claim 8** *If there  $\exists u$ , then  $l^*(u, 0)$  is a point in dual plane such that it is either above or below of all the lines  $p_i^*(v = u.x_i - y_i), 1 \leq i \leq m_1.m_2$ .*

Equivalently,

**Claim 9** If there  $\exists u$ , then  $l(y = ux)$  is a line in primal plane such that all the points  $p_i(x_i, y_i), 1 \leq i \leq m_1 \cdot m_2$  lie on one side of this line.

The incremental implementation based on the above LP formulation is as simple as

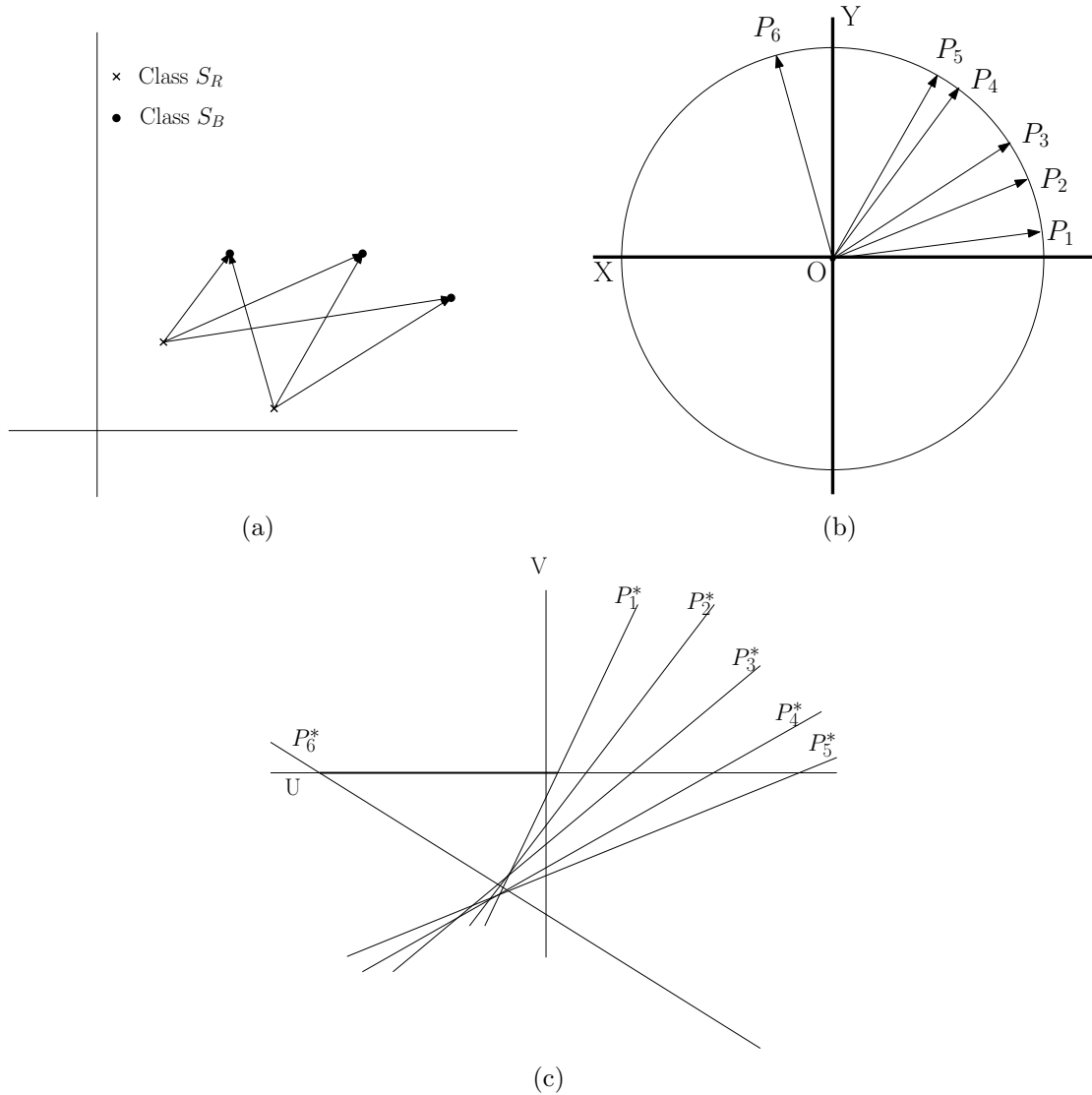


Figure 3.5: Linear programming formulation of '180 strict containment condition'  
 (a) construction of vectors (b) projection of the vectors onto unit circle  
 (c) mapping of points on the unit circle to dual plane

the one [31], and also provide for early termination when inseparability is detected. In the worst case, the running time of both formulations is quadratic in the sample size  $m$ .

## 3.6 Gene Selection

Gene selection is an important preprocessing step for the classification of gene expression dataset. This helps (a) to reduce the size of the gene expression dataset and improve classification accuracy; (b) to cut down the presence of noise in the gene expression dataset by identifying informative genes; and (c) to improve the computation by removing irrelevant genes that not only add to the computation time but also make classification harder.

### 3.6.1 Background

In this subsection we briefly discuss some popular score functions used for gene selection. We compare these with our gene selection method, proposed in the next section. A simple approach to feature selection is to use the correlation between gene expression values and class labels. This method was first proposed by Golub et al [62]. The correlation metric defined by Nguyen and Rocke [73] and by Golub et al [62] reflects the difference between the class mean relative to standard deviation within the class. High absolute value of this correlation metric favors those genes

that are highly expressed in one class as compared to the other class, while their sign indicates the class in which the gene is highly expressed. We have chosen to select genes based on a  $t$ -statistic defined by Nguyen and Rocke [73].

For  $i$ th gene, a  $t$ -value is computed using the formula

$$t_i = \frac{\mu_1^i - \mu_2^i}{\sqrt{\frac{\sigma_1^{i2}}{n_1} + \frac{\sigma_2^{i2}}{n_2}}} \quad (3.5)$$

where  $n_k$ ,  $\mu_k^i$  and  $\sigma_k^{i2}$  are the sample size, mean and variance of  $i$ th gene respectively of class  $k = 1, 2$ .

Another important feature selection method is based on the Fisher Score [74] [75].

The Fisher Score Criterion(FCS) for  $i$ th gene can be defined as

$$F_i = \frac{n_1(\mu_1^i - \mu^i)^2 + n_2(\mu_2^i - \mu^i)^2}{n_1(\sigma_1^i)^2 + n_2(\sigma_2^i)^2} \quad (3.6)$$

where  $n_k$ ,  $\mu_k^i$  and  $\sigma_k^{i2}$  are the sample size, mean and variance of  $i$ th gene respectively of class  $k = 1, 2$ .  $\mu^i$  represents mean of the  $i$ th gene.

Significance Analysis of Microarrays (SAM) proposed by Tusher et al [76] is another important gene filter technique for finding significant genes in a set of microarray

experiments. The SAM score for each gene can be defined as

$$M_i = \frac{\mu_2^i - \mu_1^i}{s_i + s_0} \quad (3.7)$$

For simplicity the correcting constant  $s_0$  is set to 1 and  $s_i$  is computed as follows

$$s_i = \left[ \left( \frac{1}{n_1} + \frac{1}{n_2} \right) \frac{\left\{ \sum_{j \in C_1} (x_j^i - \mu_1^i)^2 + \sum_{j \in C_2} (x_j^i - \mu_2^i)^2 \sum_{j \in C_2} \right\}}{(n_1 + n_2 - 2)} \right]^{\frac{1}{2}} \quad (3.8)$$

where  $x_j^i$  is  $j$ th sample of  $i$ th gene. The classes 1 and 2 are represented by  $C_1$  and  $C_2$ . Similarly  $n_k, \mu_k^i$  and  $\sigma_k^{i,2}$  are the sample size, mean and variance of  $i$ th gene respectively of class  $k = 1, 2$ . For the purpose of generating significant genes by SAM we have used the software written by Chu et al [77] which is publicly available at <http://www-stat.stanford.edu/tibs/clickwrap/sam/academic>.

### 3.7 A new methodology for gene selection

To find a set of genes of suitable size that is large enough to be robust against noise and small enough to be applied to the clinical setting, we propose a simple gene selection strategy based on an individual gene ranking approach. This consists of two steps: *coarse filtration*, followed by *fine filtration*.

### 3.7.1 Coarse Filtration

The purpose of coarse filtration is to remove most of the attributes that contribute to noise in the gene expression dataset. This noise can be categorized into (i) *biological noise* and (ii) *technical noise* [78]. Biological noise refers to the genes in gene expression dataset that are irrelevant for classification. Technical noise refers to errors incurred at various stages during data preparation.

For coarse filtration we follow an established approach based upon t-metric discussed in the previous section. Following a general consensus [62, 67], we chose to select a sufficient number genes that can be further considered for fine filtration. This is a set of 100 genes obtained by taking 50 genes with the largest positive  $t$ -values and another 50 genes with the smallest negative  $t$ -values.

### 3.7.2 Fine Filtration

One of the problems with the above correlation metric is that the  $t$ -value is calculated from the expression values of a single gene, ignoring the information available from the other genes. To rectify this, we propose the following scheme.

Let the set of genes  $\Delta = \{g_1, g_2, \dots, g_n\}$  be the output of the coarse filtration step where  $n = 100$ . For a gene  $g_i \in \Delta$ , let  $S_i = \{g_j | (g_i, g_j) \text{ is an LS(Linearly Separable) pair, } g_j \in \Delta \text{ and } i \neq j\}$ . In words,  $S_i$  consists of all genes that form linearly separable



pairs with  $g_i$ . For each gene  $g_i \in \Delta$ , its  $P_i$ -value is set to be  $P_i = |S_i|$ .

The intuition underlying the above definition is that the informative genes have quite different expression values in the two classes. If such genes exist in the gene expression data set then the above ranking strategy will assign the highest rank to those genes.

A drawback of this gene selection method is that it is applicable only to those gene expression datasets that have linearly separable pairs. For those datasets that have few linearly separable pairs, such as Lung Cancer [79] and Breast Cancer [63], we can extend the definition, using linearly separable gene triplets.

For a gene  $g_i \in \Delta$ , set

$$Q_i = \{(g_j, g_k) | (g_i, g_j, g_k) \text{ is an LS(Linearly Separable) triplet, } g_j, g_k \in \Delta, \text{ and } i \neq j, i \neq k, j \neq k\},$$

In words,  $Q_i$  consists of all gene-pairs  $(g_j, g_k)$  that make up a linearly separable triplet with the gene  $g_i$ . For each gene  $g_i \in \Delta$ , define  $T_i = |Q_i|$ . Clearly,  $T_i$  lies between 0 and  ${}^{n-1}C_2$ .

Table 3.1: Five Gene Expression Datasets

|    | Dataset            | No. of Genes | Total Samples |
|----|--------------------|--------------|---------------|
| 1. | Lung Cancer [79]   | 12533        | 181(31+150)   |
| 2. | Leukemia [80]      | 12582        | 52(24+28)     |
| 3. | SRBCT [64]         | 2308         | 43(23+20)     |
| 4. | Colon [66]         | 2000         | 62(40+22)     |
| 5. | Breast Cancer [63] | 21682        | 77(44+33)     |

### 3.8 Results and Discussions

In this chapter we have developed an offline as well as an incremental version of a geometric tool to test linear separability of pairs and triplets of genes, followed by a simple gene selection strategy that uses this tool to rank the genes. Based upon this ranking, we choose a suitable number of top-scoring genes for a good classifier.

We demonstrate the usefulness of the proposed methodology by testing with five publicly available gene expression datasets: (a) Lung Cancer [79] (b) Leukemia Data [80] (c) SRBCT [64] (d) Colon Data [66] (e) Breast Cancer [63] (see Appendix A for detail about the datasets). Table 3.1 shows number of samples belongs to different datasets and the number of samples from each class appear in the parenthesis.

The 100 genes that we select from each of these datasets in the *Coarse Filtration* step effectively prunes away most of the attributes(genes) that are irrelevant for classification. On the other hand, this number is large enough to provide us with a number

attributes(genes) that may be over fitting for classifier construction.

To get the best subset of genes for good classification we chose to populate the attribute space with 5, 10, 15, 25 and 30 genes from each dataset by applying *Fine Filtration*. The choices of these attribute/feature-space sizes are somewhat arbitrary but the chosen attribute/feature-spaces are sufficiently large in comparison to the size of the sample spaces as Table 3.1 shows.

The computational time of the *Fine Filtration* step depends upon the geometric tool that we use to check the separability of gene expression data. In this chapter, we have presented linear time incremental algorithms for both gene pairs and gene triplets. In order to illustrate the effectiveness of this approach we ran both versions (offline and incremental) on each of the five datasets obtained by *Coarse Filtration*. The computing platform was a Dell inspiron 1545 model-Intel Core2 Duo CPU, 2.00 GHz and 2 GB RAM, running under Windows Vista. The run-time efficiency of the incremental version over the offline one is evident from Table 3.2.

A group of genes that is being tested for linear separability may include a gene that is a perfect 1-D separator with TNoM score zero, using the terminology of [65]. In this case, such a group will provide a positive separability test. In order to exclude such groups, we checked for the existence of such 1-D separators, and found that no such

Table 3.2: 2-D and 3-D Separability Test with Runtime

| Dataset       | 2-D Separability Test with RT |                                |                              |                                      | 3-D Separability Test with RT |                             |                           |                                      |
|---------------|-------------------------------|--------------------------------|------------------------------|--------------------------------------|-------------------------------|-----------------------------|---------------------------|--------------------------------------|
|               | % LSP                         | RT of<br>offline<br>in<br>msec | RT of<br>incr.<br>in<br>msec | Impr.<br>of incr.<br>over<br>offline | %<br>PLST                     | RT of<br>offline<br>in msec | RT of<br>incr. in<br>msec | Impr.<br>of incr.<br>over<br>offline |
| Lung Cancer   | 0.72%                         | 4617                           | 1537                         | 66.71%                               | 0.946%                        | 1114467                     | 1666662                   | 85.04%                               |
| Leukemia      | 11.92%                        | 1138                           | 418                          | 63.27%                               | 3.72%                         | 115791                      | 49263                     | 57.45%                               |
| SRBCT         | 8.86%                         | 987                            | 356                          | 63.93%                               | 4.11%                         | 92825                       | 52080                     | 43.89%                               |
| Colon         | 0%                            | 1328                           | 275                          | 79.29%                               | 0%                            | 170143                      | 39955                     | 76.516                               |
| Breast Cancer | 0.93%                         | 1606                           | 440                          | 72.6%                                | 0.137%                        | 274141                      | 83913                     | 69.39%                               |

*Abbreviations RT: Run Time, Incr.: Incremental, Impr.: Improvement*

genes exists in the above datasets. Likewise, if a gene pair shows linear separability then all gene triplets that include these gene pairs will also be linearly separating. In order to count gene triplets that exhibit pure 3-D linear separability, we avoid testing gene triplets that include a linearly separable pair. Thus our 3-D test results shown here include only such gene triplets. We call such gene triplets as Perfect Linearly Separable Triplet(*PLST*). The percentage of Linearly Separable Pairs(*LSP*), Linearly Separable Triplets(*LST*) and Perfect Linearly Separable Triplets(*PLST*) are calculated using the formulas below.

$$\% \text{ of LSP} = \frac{\# \text{ of } LSP}{\text{Total possible } LSP} \times 100 = \frac{\# \text{ of } LSP}{{}^2C_n} \times 100$$

$$\% \text{ of LST} = \frac{\# \text{ of } LST}{\text{Total possible } LST} \times 100 = \frac{\# \text{ of } LST}{{}^3C_n} \times 100$$

$$\% \text{ of PLST} = \frac{\# \text{ of } PLST}{\text{Total possible } LST - ((\# \text{ of } LSP) \times (n-2))} \times 100$$

$$= \frac{\# \text{ of } PLST}{{}^3C_n - ((\# \text{ of } LSP) \times (n-2))} \times 100$$

where  $n$  is total number genes in the gene expression data set.

The above formulas show that the total number of triplets relative to the *PLST*s is much higher than the total number of pairs relative to the *LSP*s. Thus the increase

in the actual number of *PLST* over the number of *LSP* is suppressed by the high value of the denominator in the former case. The separability test shows that Colon Data [66] has neither any *LSP* nor any *PLST*. The Lung Cancer [79] and Breast Cancer [63] datasets have a few *LSP*, whereas the number of *PLST* is respectively 41 and 5 times (approximately) the number of *LSP*. The Leukemia Data [80] and SRBCT [64] show a good number of *LSP*, while the number of *PLST* is respectively 6 and 11 times (approximately) the number of *LSP*.

The motive underlying our gene selection strategy is to identify if a gene, jointly with some other genes, has the class distinction property or not. In the current study, we identify the class distinction property by separability tests where we restricted the group size to pairs and triplets. As the Colon Data [66] did not show any positive separability result we continued our study with the remaining four gene expression datasets. This result in Colon Data [66] is not surprising at all since according to Alon et al. [66] some samples such as T2, T30, T33, T36, T37, N8, N12, N34 in Colon Data have been identified as outliers and presented with anomalous muscle-index. This confirms the uncertainty of these samples.

To continue, in the *Fine Filtration* stage we use the incremental version of our algorithm to test separability by gene pairs and assign a  $P_i$  value to a gene  $g_i \in \Delta$ . Based on the ranking, we choose a set of top-scoring genes to populate five different

feature spaces of size 5, 10, 15, 20, 25 and 30. If more than one gene have same rank then we choose an arbitrary gene from that peer group. To compare our method with other selection methods such as  $t$ -metric, FCS and SAM, we populate similar feature spaces respectively.

For classification we used machine learning tools supported by WEKA version 3.6.3 [81]. We used the following two classifiers :(a) *Support Vector Classifier*: WEKA SMO class implements John C. Platt's [82] sequential minimal optimization algorithm for training a support vector classifier. We used a linear kernel. (b) *Bayes Network Classifier*: Weka BayesNet class implements Bayes Network learning using various search algorithms and quality measures [83]. We have chosen Bayes Network classifier based on  $K2$  for learning structure [84]. Both of the above classifiers normalized the attributes by default to provide a better classification result. We used a 10-fold cross-validation [85] for prediction. as shown in Figure 3.6. As suggested by Kohavi [85] we have used ten-fold stratified cross-validation. In stratified cross-validation the folds are stratified so that they contain approximately same proportions of labels as original datasets.

A comparative classification accuracy of the feature spaces generated from  $P$ -values,  $t$ -values, FCS and SAM is shown in Figure 3.7 - 3.10. The results clearly show that the gene spaces generated by  $P$ -values yields a good classifier. Specifically, the feature

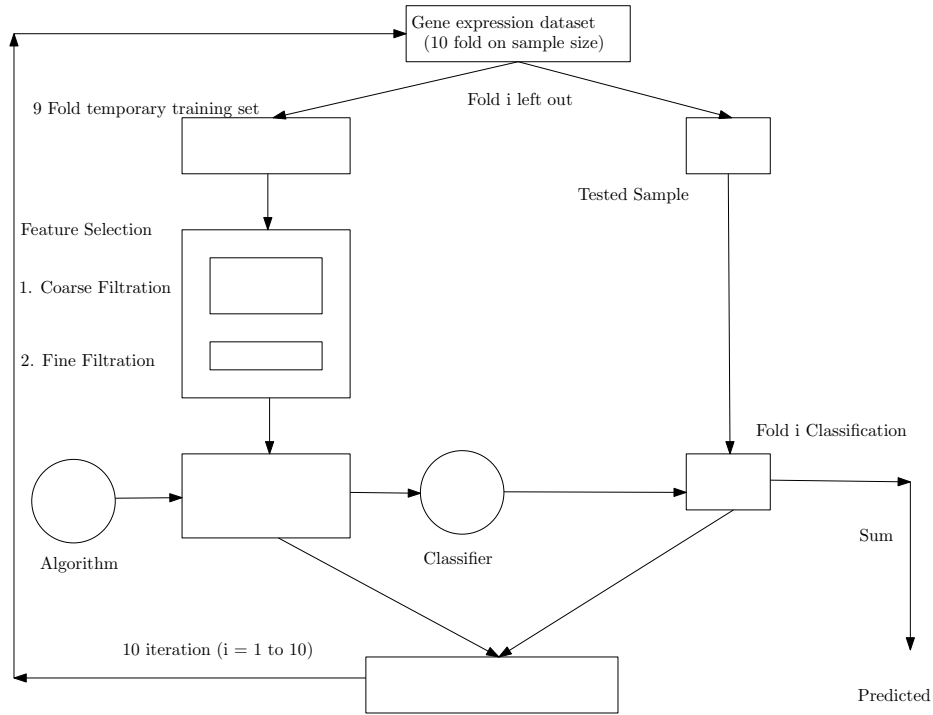


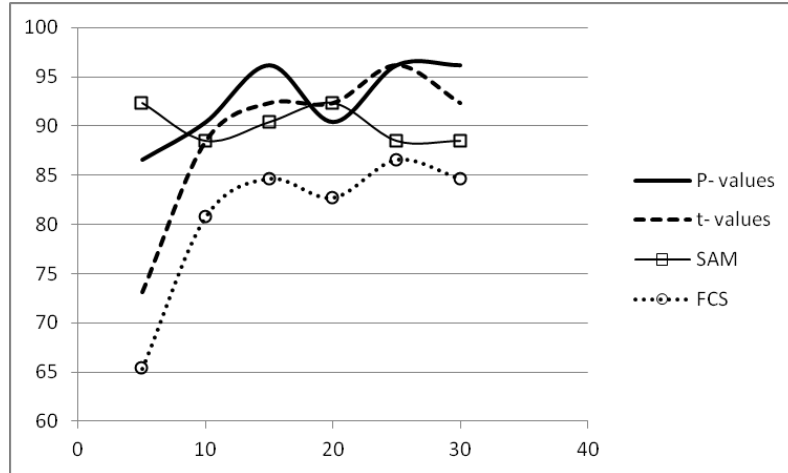
Figure 3.6: 10 Fold cross validation on gene expression data-set

spaces of sizes 10, 15, 20, 25 and 30 generated by the  $P$ -values perform mostly better than or as good compared to the feature spaces generated by the  $t$ -values, FCS and SAM.

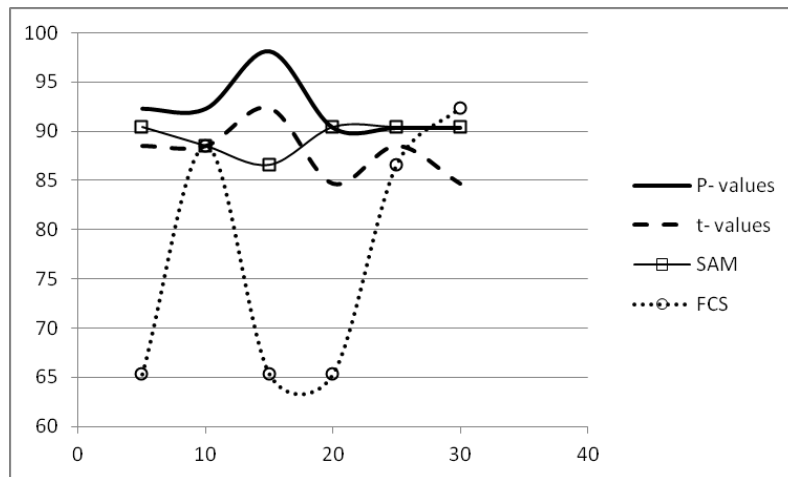
To illustrate the performance of the classifiers with respect to the feature spaces generated by the  $T$ -values we considered two datasets with few  $LSP$ , such as Lung Cancer [79] and Breast Cancer [63]. To make sure that the dataset has no  $LSP$  we removed all genes that are responsible for pair separability in feature the selection process. Then feature spaces of size 5, 10, 15, 20, 25 and 30 are populated based upon the  $T$ -values. The classification results are shown in Figure 3.7 - 3.10. It is interesting



to note that the feature space generate from lung Cancer [79] dataset by  $T$ -values achieves similar or even better classification accuracy as compared to  $t$ -values, FCS and SAM. In Figure 3.11 - 3.14. we have shown the classification accuracy of feature space confined to 25 and 30.

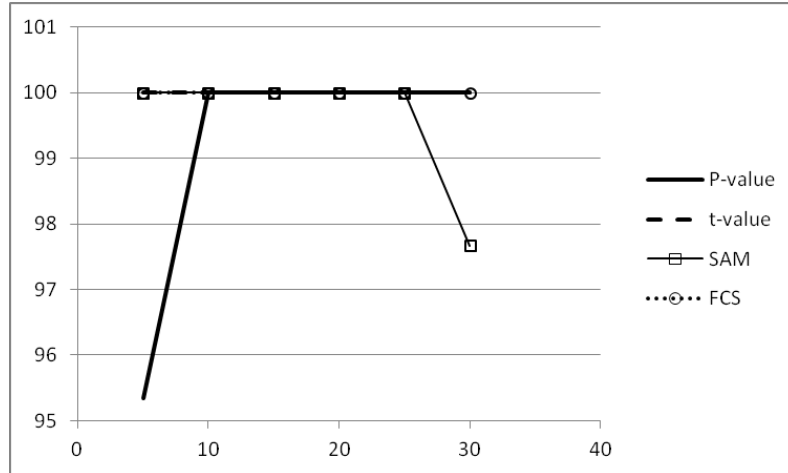


(a) Leukemia SVM

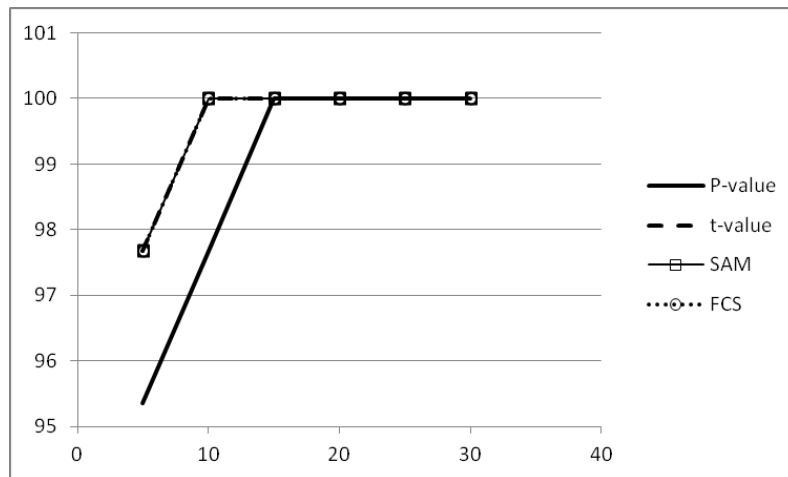


(b) Leukemia BayesNet

Figure 3.7: Accuracy vs Feature Space (Leukemia)

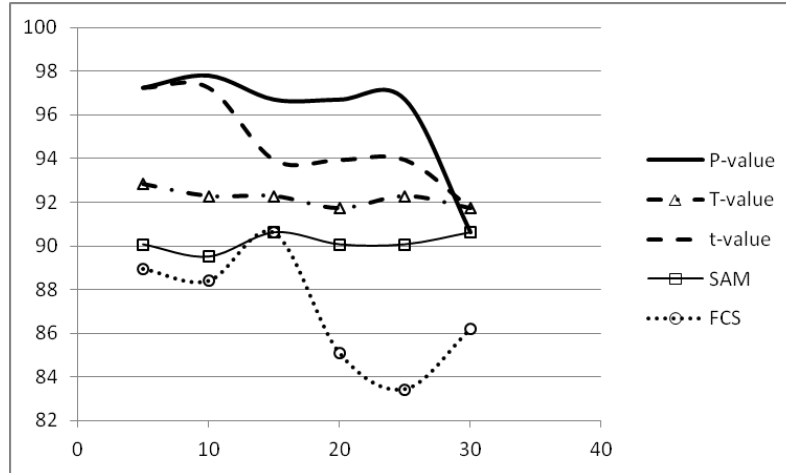


(a) SRBCT SVM

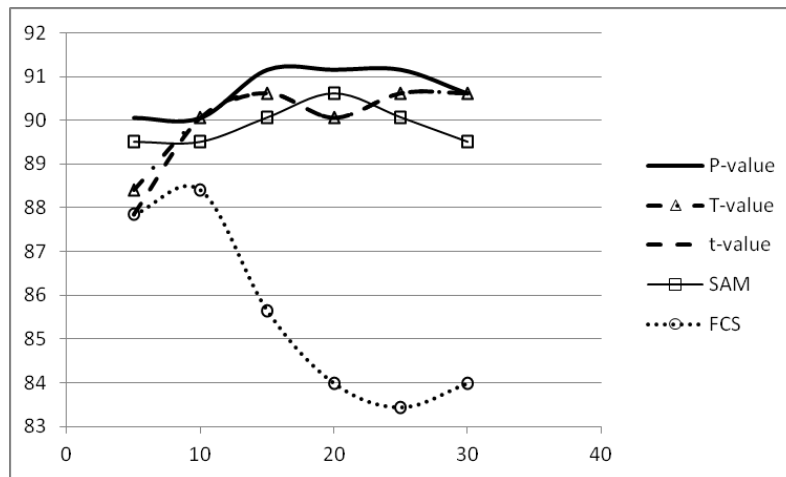


(b) SRBCT BayesNet

Figure 3.8: Accuracy vs Feature Space (SRBCT)

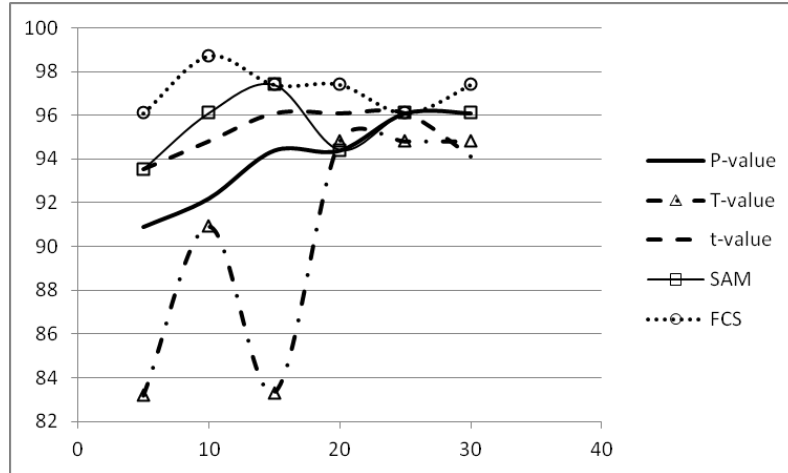


(a) Lung Cancer SVM

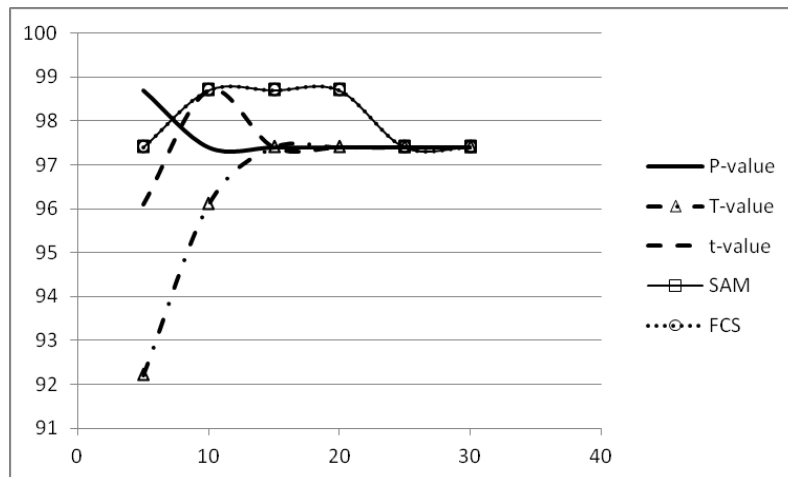


(b) Lung Cancer BayesNet

Figure 3.9: Accuracy vs Feature Space (Lung Cancer)

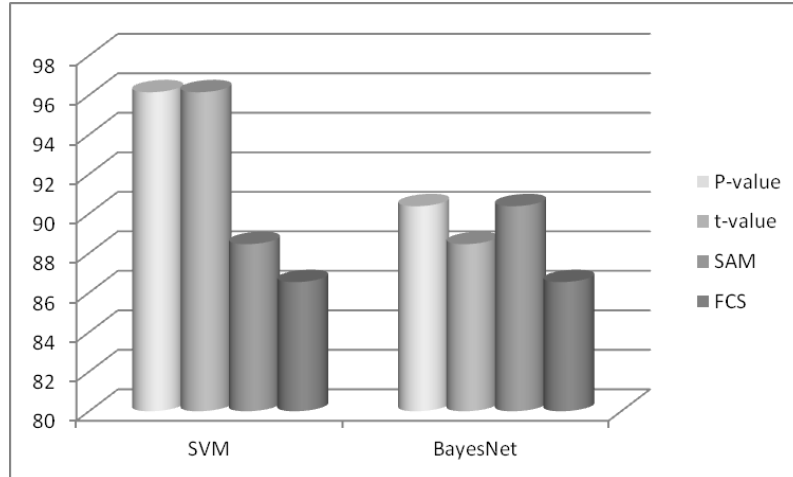


(a) Breast Cancer SVM

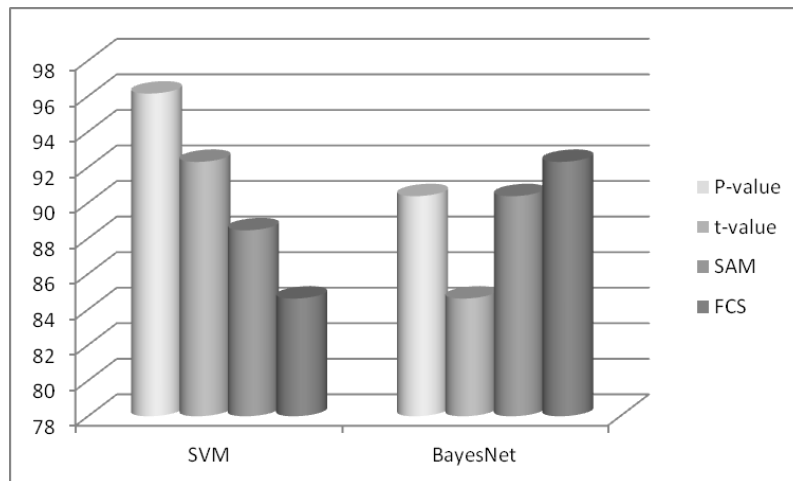


(b) Breast Cancer BayesNet

Figure 3.10: Accuracy vs Feature Space (Breast Cancer)

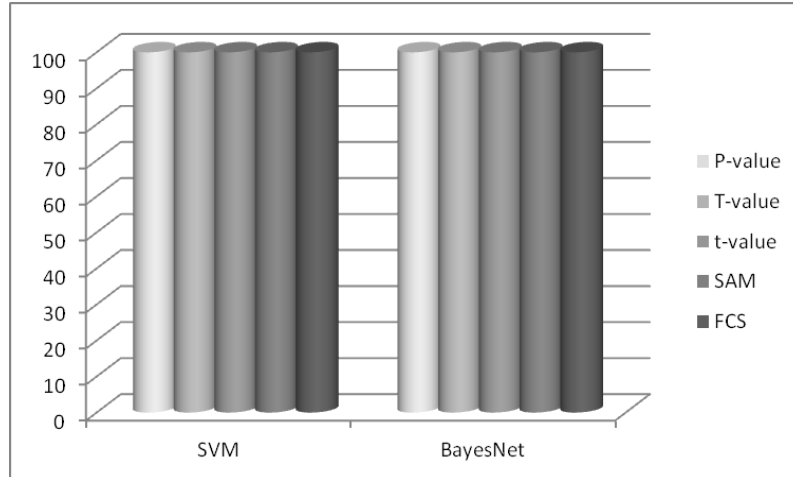


(a) 25 FS of Leukemia

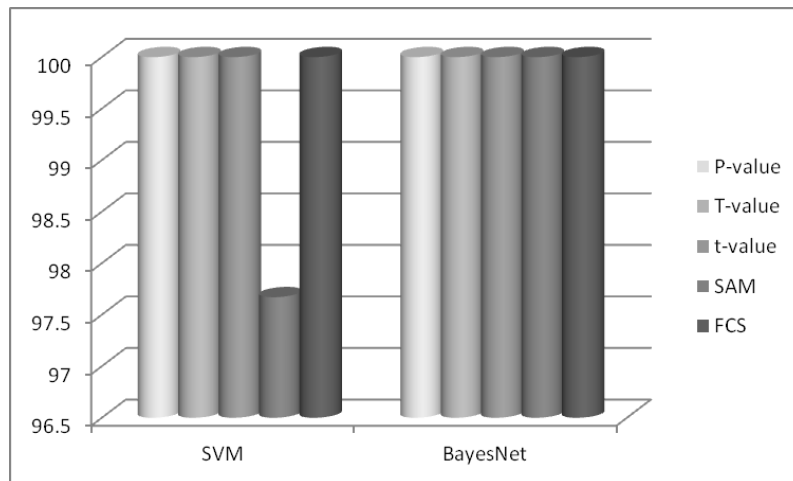


(b) 30 FS of Leukemia

Figure 3.11: Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - Leukemia

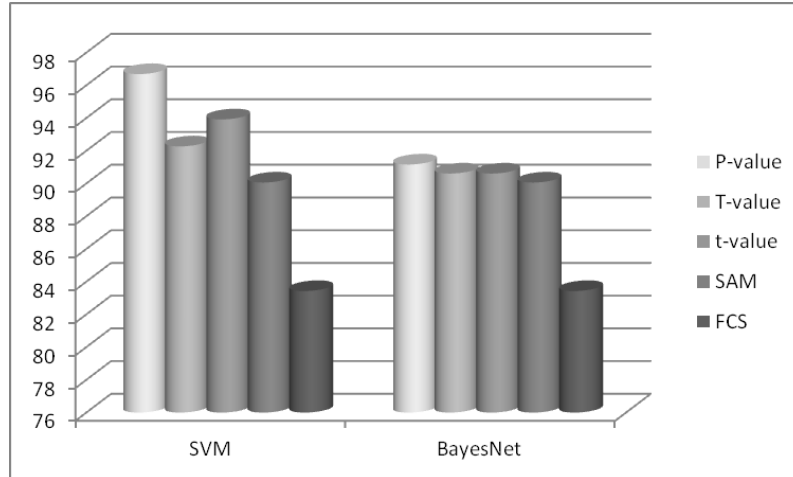


(a) 25 FS of SRBCT

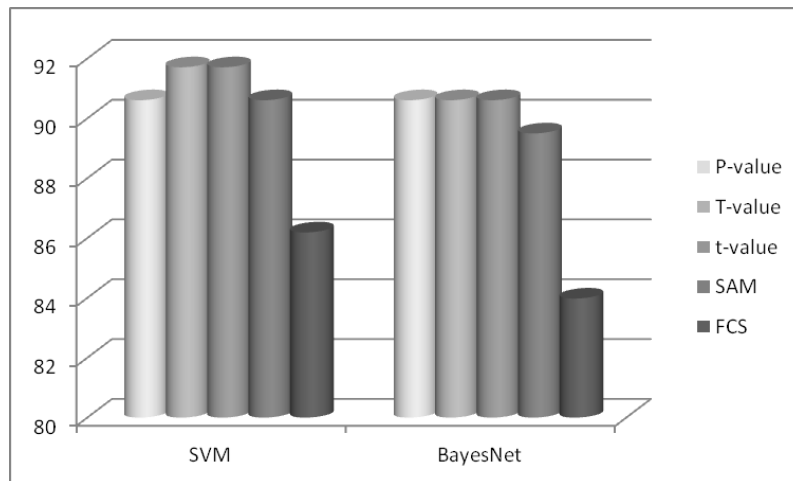


(b) 30 FS of SRBCT

Figure 3.12: Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - SRBCT



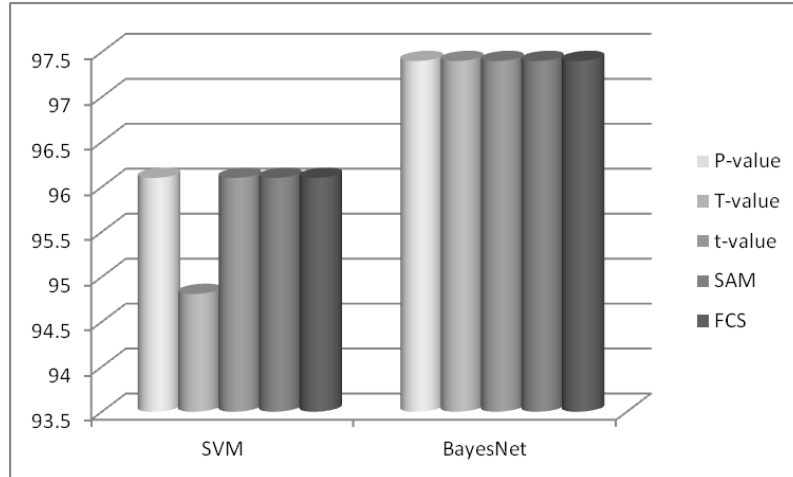
(a) 25 FS of Lung Cancer



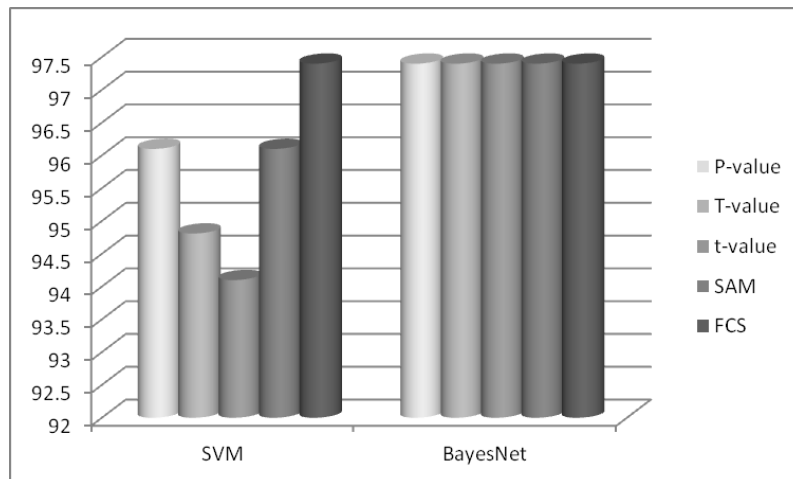
(b) 30 FS of Lung Cancer

Figure 3.13: Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - Lung Cancer





(a) 25 FS of Breast Cancer



(b) 30 FS of Breast Cancer

Figure 3.14: Classifier Accuracy of gene expression dataset on 25 and 30 Feature Space(FS) - Breast Cancer

## 3.9 Summary

Our empirical study of the four datasets shows that the feature space generated by our methods, particularly by the use of P-values, is as good as the feature selection methods based on t-values, SAM and FCS. Towards the broader objective of identifying important biomarkers to distinguish between input classes, in Table 3.3 we enumerate the top 10 genes (or genes attached to probe set in respective microarray experiment) from each of the datasets.

We presented a gene selection strategy to achieve a high classification accuracy. The gene selection strategy exploits the class distinguishing property of genes by testing separability by pairs and triplets. To test for separability we have provided two versions of a linear time algorithm, and demonstrated that the run-time of the incremental version is markedly better than that of the offline version. The importance of the given method lies in the fact that it can be easily extended to higher dimensions, allowing us to test if groups of genes of size greater than 3 can separate the datasets. In the current study, we have limited the separability tests to gene pairs and triplets and used this criterion to rank the genes.

Table 3.3: Top Ten Significant Genes based upon P-values

| Dataset       | Probe Set or image | Gene Name or Description   |
|---------------|--------------------|--|
| Leukemia Data | 39318_at           | Hs.2484 gnl—UG—Hs#S4305 H.sapiens mRNA for Tcell leukemia  |
|               | 36571_at           | Hs.75248 gnl—UG—Hs#S5526 H.sapiens topIIb mRNA for topoisomerase IIb   |
|               | 41462_at           | Hs.11183 gnl—UG—Hs#S1055230 Homo sapiens sorting nexin 2 (SNX2) mRNA, complete cds   |
|               | 266_s_at           | M26692 /FEATURE=exon#1 /DEFINITION=HUMILCKPR02 Homo sapiens lymphocyte-specific protein tyrosine kinase (LCK) gene, exon 1, and downstream promoter region |
|               | 34168_at           | Hs.272537 gnl—UG—Hs#S1611 Human terminal transferase mRNA, complete cds  |
|               | 40285_at           | Hs.58927 gnl—UG—Hs#S876152 Homo sapiens nuclear VCP-like protein NVLp.2 (NVL.2) mRNA, complete cds   |
|               | 40533_at           | Hs.1578 gnl—UG—Hs#S1266737 tg78b04.x1 Homo sapiens cDNA, 3' end  |
|               | 38017_at           | Hs.79630 gnl—UG—Hs#S551444 Human MB-1 gene, complete cds   |
|               | 40282_s_at         | Hs.155597 gnl—UG—Hs#S779 Human adipsin   |
|               | 39520_at           | Hs.100729 gnl—UG—Hs#S1526847 wn60d01.x1 Homo sapiens cDNA, 3'end   |
| Lung Cancer   | 33328_at           |  |
|               | 36533_at           |  |
|               | 33833_at           |  |
|               | 31684_at           |  |
|               | 41388_at           |  |
|               | 1662_r_at          |  |
|               | 33904_at           |  |
|               | 36105_at           |  |
|               | 33245_at           |  |
| 39756_g_at    |                    |  |
| Breast Cancer |                    | Contig53226_RC   |
|               |                    | AI147042_RC  |
|               |                    | NM.000790  |
|               |                    | Contig1789_RC  |
|               |                    | NM.000238  |
|               |                    | AB037821   |
|               |                    | AB033007   |
|               |                    | NM.000353  |
|               | NM.002073          |  |
|               | AF053712           |  |
| SRBCT         | 770394             | Fc fragment of IgG, receptor, transporter, alpha   |
|               | 377461             | caveolin 1, caveolae protein, 22kD   |
|               | 1435862            | antigen identified by monoclonal antibodies 12E7, F21 and O13  |
|               | 814260             | follicular lymphoma variant translocation 1  |
|               | 866702             | protein tyrosine phosphatase, non-receptor type 13 (APO-1/CD95 (Fas)-associated phosphatase)   |
|               | 52076              | olfactomedinrelated ER localized protein   |
|               | 357031             | tumor necrosis factor, alpha-induced protein 6   |
|               | 43733              | glycogenin 2   |
|               | 207274             | Human DNA for insulin-like growth factor II (IGF-2); exon 7 and additional ORF   |
|               | 898219             | mesoderm specific transcript (mouse) homolog   |

# Chapter 4

## On the linear separability of a bichromatic point set with violated constraints

### 4.1 Overview

Let  $S_R$  be a set of red points and  $S_B$  a set of blue points in the plane, with  $n = |S_R| + |S_B|$ . If the sets are linearly separable, a separating line can be found in  $O(n)$  time by the well-known linear programming technique of Megiddo or Dyer. Otherwise, it gives rise to the interesting problem of finding the smallest set  $S_{RB} \subset S_R \cup S_B$  such that  $S_R \setminus S_{RB}$  and  $S_B \setminus S_{RB}$  are linearly separable. In this chapter, we propose an  $O(nk^2)$  time algorithm for this problem, where  $k = |S_{RB}|$ . When  $k = o(\log n)$ , this is better than the so-far-best  $O((n + k^2) \log n)$  time algorithm known for this problem. For  $k = O(1)$ , which holds for the application to gene expression analysis that we have in mind, we have the first linear time algorithm known for this problem.

### 4.2 Introduction

#### 4.2.1 Problem statement

Let  $S_R$  be a set of red points and  $S_B$  a set of blue points in the plane, with  $n = |S_R| + |S_B|$ . Assuming that  $S_R$  and  $S_B$  are “almost” linearly separable, it is an

interesting problem to find a line  $l$  and a set  $S_{RB} \subset S_R \cup S_B$  of minimum size such that the points of the sets  $S_R \setminus S_{RB}$  and  $S_B \setminus S_{RB}$  lie on opposite sides of  $l$ . In this chapter, we study the above problem and show that when  $|S_{RB}| = O(1)$  it can be solved in linear time.

## 4.2.2 Motivation

This problem is motivated by the following fundamental classification problem in machine learning. Given  $n$  sample points (the training set),  $n_1$  from cancer type  $C_1$  (say red points  $S_R, |S_R| = n_1$ ) and  $n_2$  from cancer type  $C_2$  (say blue points  $S_B, |S_B| = n_2$ ), construct a predictor (separating line) that facilitate classification of a new sample point into either  $C_1$  or  $C_2$ . Clarkson [86], Dyer [70], Megiddo [68, 69], Seidel [87], Sharir and Welzl [88] showed that this problem can be solved in linear time if the red and blue points are linearly separable.

However, these algorithms are not designed to handle the case when the point sets are almost linearly separable. Practically, this case arises due to the presence of faulty data points (outliers) as a result of noise or sampling or round-off errors. This variant of the problem was addressed by Matousek [11], Chan [12, 89, 90], Efrat et al. [91], Roos and Widmayer [92]. For a given  $k, k \leq n$ , their methods find a line that separates all but  $k$  of the given points. These  $k$  violations can be points of either color.

### 4.2.3 Prior work

Everett et al. [10] were among the first to investigate this problem, assuming that the red and blue point sets have the same cardinality. Note that such an assumption does not hold for most classification problems. Their proposed dual-space algorithm explores the solution space by constructing in optimal  $O(n \log n + nk)$  time all  $(\leq k)$ -levels of the blue and red lines, and intersecting pairs of blue and red levels to find the minimum  $k$ . As the  $(\leq k)$ -levels in the arrangement of  $n$  lines have combinatorial complexity  $O(nk)$  [93,94], the algorithm has time complexity  $O(n \log n + nk \log k)$ .

Matousek [11] proposed an efficient  $O(n \log n + k^3 \log^2 n)$  time algorithm that finds a line that separates all but  $k$  of given points. This method works differently for (a) the feasible case - when the point sets are completely separable and (b) the infeasible case - when the point sets are not completely separable. Like Everett's algorithm, this method also uses the arrangement of lines in dual space. To find the minimum  $k$ , the smaller levels  $(\leq k)$  in the arrangement have to be searched first.

Nearly a decade later, Chan [12] revisited the result of Everett et. al [10], proposing an improved algorithm that runs in  $O((n + k^2) \log n)$  expected time. His method avoids constructing all  $(\leq k)$ -levels, using instead a concave/convex-chain decomposition technique that involves a small  $O(k)$  number of chains of total size  $O(n)$  [95–97]. This algorithm has same limitation as that of Matousek [11] in requiring an upper

bound on the number of outliers in order to find the minimum.

Megiddo [69], O'Rourke et al. [98], Vapnik [99] studied this problem in higher dimensions with hyperplane or sphere as separator. Arkin et al. [100], Hurtado et al. [101,102] discussed the problem of separability in the plane with separators that are strips or wedges or double-wedges. The problem with convex polygons or simple polygons as a separator was addressed by Edelsbrunner and Preparata [103], Fekete [104] and Mitchell [105].

#### 4.2.4 Our contributions

We propose an output-sensitive algorithm that runs in  $O(nk^2)$  time. If  $k^2 = o(\log n)$  this algorithm is more efficient than existing algorithms; and if  $k = O(1)$  the algorithm runs in linear time as compared to existing  $O(n \log n)$  time algorithms [10–12]. Moreover, the proposed algorithm does not require that the red and blue point sets have the same cardinality as in Everett's algorithm [10], nor does it require an upper bound on the number of violated constraints as in the algorithms of Matousek [11] and Chan [12].

## 4.3 Preliminaries

The problem of finding a linear separator in primal space that minimizes the number of outliers  $k$  can be reduced to a 2-dimensional linear programming problem in dual space [92]. The point sets  $S_R$  and  $S_B$  in primal space transform to line sets  $S_R^*$  and  $S_B^*$  in dual space. If a line  $l$  that separates all but  $k$  points of  $S_R$  and  $S_B$  then in dual there exists a point  $l^*$  that separates all but  $k$  lines of  $S_R^*$  and  $S_B^*$ .

To simplify the formulation in dual space, we make the following *general position* assumptions: (a) the lines of the arrangement have distinct and finite slopes; (b) no three lines are concurrent (see [106], Roos and Widmayer [92]). Note that there are well-known techniques for dealing with the situation where these *general position* assumptions do not hold.

### 4.3.1 Megiddo's algorithm

Megiddo's algorithm solves the following linear program in the  $uv$  plane.

$$\begin{aligned} v &\geq ux_i - y_i & (x_i, y_i) \in S_R \\ v &\leq ux_i - y_i & (x_i, y_i) \in S_B \end{aligned} \tag{4.1}$$

where  $|S_R| = n_1$ ,  $|S_B| = n_2$  and  $n_1 + n_2 = n$ .



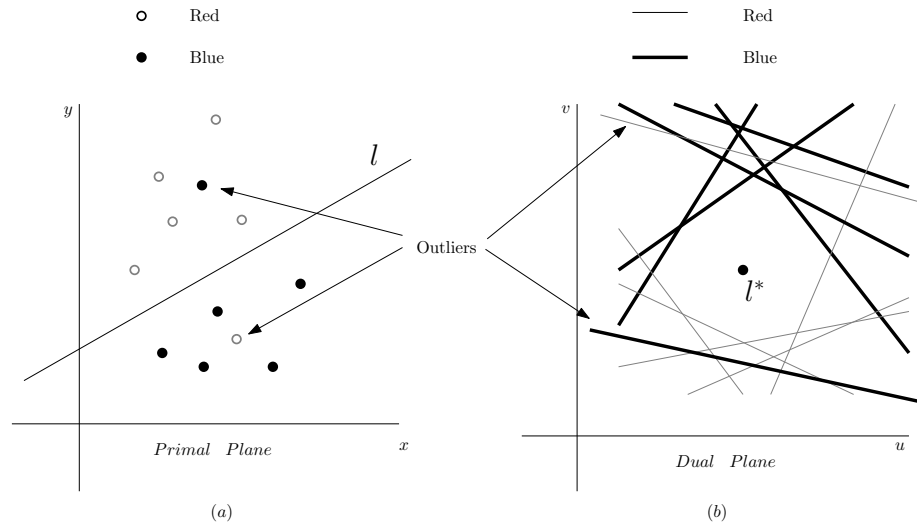


Figure 4.1: Classification of red set (dots (primal) or lines (dual)) from blue set (solid dots (primal) or dark lines (dual)). A separating line  $l$  in the primal space has a feasible point  $l^*$  in the dual space. Two points in primal (lines in dual) are misclassified.

Its output is a point such that all red lines are below this point and all blue lines are above. Call the point-wise minimum (maximum) of the blue (red) lines as *min-curve* (*max-curve*). These are also known as upper and lower envelope respectively. In the feasible case, the *max-curve* intersects the *min-curve*. If they do not intersect the blue and red sets are not completely separable.

Megiddo's algorithm performs  $O(\log n)$  iterations. In each iteration it prunes at least a quarter of the constraints (i.e. lines) that do not determine the boundary of the feasible region. At the same time it reduces the search interval on the  $u$ -axis. Each

iteration involves a vertical test line  $l_T$  that is used to check for feasibility, leading to the following cases:

1. max-curve is below the min-curve at  $l_T$ , indicating separability
2. max-curve is above the min-curve at  $l_T$ , indicating two possibilities
  - (a) there exists a possible feasible region either to the left of  $l_T$  or to its right;
  - (b) or there is inseparability

In case (2.a), the algorithm continues with the next iteration. In the other cases, the algorithm reports separability or inseparability. The inseparable case is determined from the relative slopes of lines belonging to *max-curve* and *min-curve* that intersects the test line  $l_T$ . Call these lines *eccentric-lines*. In view of our general position assumptions, this includes either two lines from each curve (or envelope) or one from one curve and two from the other (see Figure 4.2). Assuming  $|S_R^*| + |S_B^*| > 2$ , we can prove the following result.

**Claim 10** *If  $S_R^*$  and  $S_B^*$  are not completely separable then there exists either 3 or 4 **eccentric-lines**.*

## 4.4 Proposed Algorithm

The following useful theorem underlies our algorithm.

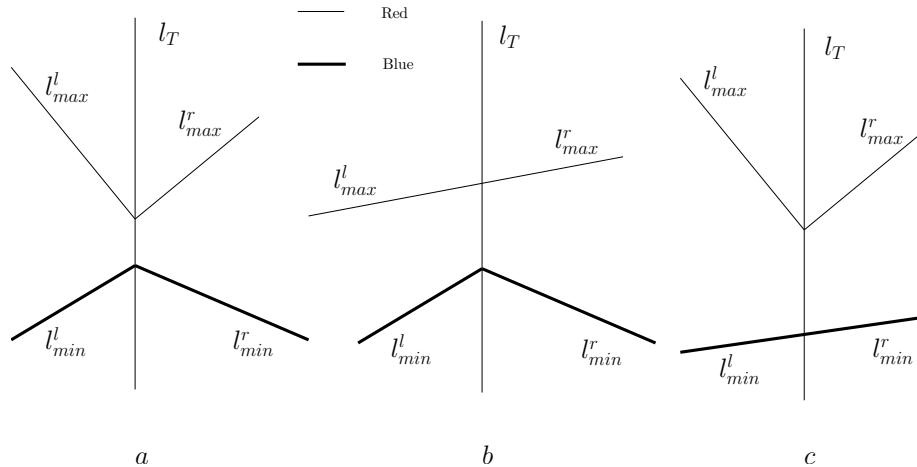


Figure 4.2: Eccentric-lines from max and min curve. In all cases  $\text{slope}(l_{min}^l) \geq \text{slope}(l_{max}^l)$  and  $\text{slope}(l_{max}^r) \geq \text{slope}(l_{min}^r)$ .

**Theorem 5** *If  $S_R^*$  and  $S_B^*$  are not completely separable then at least one of the eccentric-lines is an outlier.*

**Proof:** Assume otherwise. If we run Megiddo's algorithm on an input consisting of these **eccentric-lines** alone, we will arrive at the same situation where the *max-curve* does not intersect the *min-curve*, indicating inseparability.  $\square$

Suppose we have  $k$  outliers that cause inseparability. The algorithm identifies a superset of  $m$  constraints that contains these  $k$  outliers ( $m \leq 4k$  as we will see later). Call this set of  $m$  constraints as the tentative set,  $T$ , and the remaining set of  $n - m$  constraints as the residual set  $D$ .

The proposed algorithm consists of the following main steps.

1. Construct the *tentative-set*  $T$ .
2. Explore the arrangement of the lines in the *tentative-set*,  $T$ , for outlier sets.
3. Validate each outlier set.

The details of the above three steps appear in the following sections.

#### 4.4.1 Construct the *tentative-set* $T$

Initialize the *residual-set*,  $D$ , with all the constraints in the sets  $S_R^*$  and  $S_B^*$  and the *tentative-set*,  $T$ , to empty. In this step, we run Megiddo's algorithm with the constraints of  $D$  as input. If we reach the inseparable case, remove the *eccentric-lines* from  $D$  and add them to  $T$ . Repeat the above process in a loop till Megiddo's algorithm detects separability. Suppose Megiddo's algorithm is called  $i + 1$  times ( $i \geq 0$ ) before it detects separability then  $m \leq 4i$ .

**Theorem 6** *If  $k$  is the minimum number of outliers whose removal from  $S_R \cup S_B$  results in a feasible solution then  $i \leq k \leq m$ .*

**Proof:** The first inequality holds because in each of the first  $i$  runs of Megiddo's algorithm the eccentric-set generated has at least one outlier. The second holds because  $T$  contains all possible outliers. □

## 4.4.2 Explore the arrangement of the lines in the *tentative-set*, $T$ , for outlier sets

### Construction of arrangement

Incrementally construct the arrangement of the lines in  $T$  by simulating Chazelle et al.'s algorithm [54], with some additional book-keeping. We use any reasonable data structure (e.g. a doubly-connected edge list (DCEL)) to represent the planar graph. We enclose the arrangement inside a large bounding box  $B$ . Consider updating the already-constructed arrangement,  $A(j)$ , on  $j$  lines when introducing the  $j + 1$ -th line  $l_{j+1}$ . While updating the data structure, we maintain incident constraints (or lines) of each vertex. Notice that the order of the insertion of the lines is arbitrary. Chazelle et al.'s method takes  $O(j)$  time for the  $j$ -th insertion. Since a line intersects the boundary of  $B$  twice, we can easily determine the entry face in the arrangement  $A(j)$  of  $l_{j+1}$ . The arrangement is updated by walking along the lower part of the zone of  $l_{j+1}$ , denoted by  $zone(l_{j+1})$ . This is the set of faces whose closure intersects  $l_{j+1}$  which is of combinatorial complexity  $O(j)$ . Thus running time of this incremental construction is  $O(m^2)$ , where  $m$  is the number of lines in the *tentative-set*.

### Compute the levels of the arrangement

A point in the arrangement is feasible if it is above the red lines but below the blue lines. Each edge  $e$  in the arrangement is assigned a level  $(a, b)$  where  $a$  is the

number of red lines above  $e$  and  $b$  is the number of blue lines below it. The level  $(a, b)$  represents number of lines  $k$  ( $k = a + b$ ) that are misclassified by a point on  $e$ . The DCEL data structure also maintains the outliers for each edge. The assignment of level to each edge can be done by traversing each line of the arrangement. For a line  $l_j$  calculate the level of the leftmost edge which requires checking of above-below relationship with respect to that edge of all other lines. This takes linear time i.e.  $O(m)$ . To calculate the levels of the other edges on  $l_j$ , we walk along the line using the DCEL, stopping at vertices to update the level information (see Figure 4.3).

**Definition:** A line  $l_j$  crosses a line  $l_i$  from above (from below) if  $l_j$  intersects every vertical line above (below)  $l_i$  before its intersection with  $l_i$ .

Suppose  $e_{ij}^L$  and  $e_{ij}^R$  are edges on the line  $l_i$  respectively to the left and to the right of its intersection with  $l_j$ . If  $e_{ij}^L(a, b)$  is the level of the edge  $e_{ij}^L$  then we have the following level change rules.

$$e_{ij}^L(a, b) \rightarrow e_{ij}^R(a - 1, b) \quad \text{if } l_j \in S_R^* \text{ and crosses } l_i \text{ from above}$$

$$e_{ij}^L(a, b) \rightarrow e_{ij}^R(a + 1, b) \quad \text{if } l_j \in S_R^* \text{ and crosses } l_i \text{ from below}$$

$$e_{ij}^L(a, b) \rightarrow e_{ij}^R(a, b + 1) \quad \text{if } l_j \in S_B^* \text{ and crosses } l_i \text{ from above}$$

$$e_{ij}^L(a, b) \rightarrow e_{ij}^R(a, b - 1) \quad \text{if } l_j \in S_B^* \text{ and crosses } l_i \text{ from below}$$

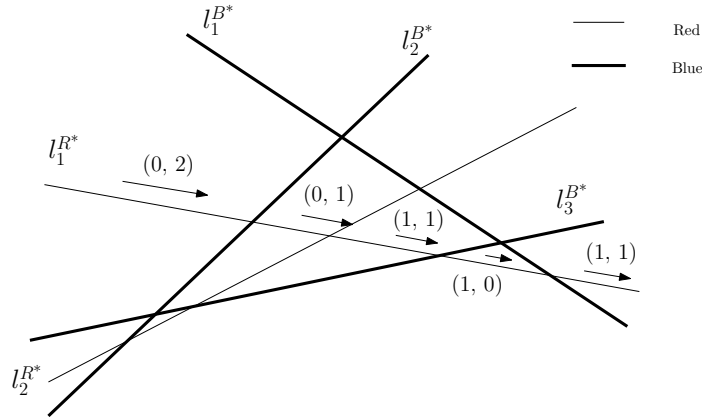


Figure 4.3: The levels to the edges on the line  $l_1^{R*}$  of arrangement.

The assignment of levels to the edges of a line takes  $O(m)$  time. Thus we have an  $O(m^2)$  algorithm for  $m$  lines.

### Search for potential feasible regions

In this step, we scan the faces of the arrangement for potential feasible regions. The number of lines that are misclassified by any feasible point within a face is related to the levels of the edges bounding that face. Every edge in an arrangement has two faces adjacent to it. A face is above (below) an edge if all the points belonging to that face are above (below) the line incident on the edge. Thus if a face is above (below) an edge  $e_{ij}(a, b)$  on a red line then every point in the face misclassifies  $a$  ( $a + 1$ ) red lines and  $b$  ( $b$ ) blue lines. Similarly, if a face is above (below) an edge  $e_{ij}(a, b)$  on a blue line then every point in the face misclassifies  $a$  ( $a$ ) red lines and  $b + 1$  ( $b$ ) blue lines. A face is a potential feasible region if it misclassifies  $k$  lines such that  $i \leq k \leq m$  (see Theorem 6). As we walk along the line  $l_j$  in the arrangement we look for potential

feasible regions in the zone of  $l_j$ . Since the complexity of  $zone(l_j)$  is  $O(m)$  [54], the search along all the lines in the arrangement takes  $O(m^2)$  time. This step can be carried out in parallel with the computation of the levels in the arrangement.

### 4.4.3 Validate the outlier set

We know the outliers for each face in the arrangement. For each potential feasible region (face in the arrangement) with  $k$  outliers, we check for the feasible region with  $n - k$  constraints, belonging to the red and blue sets. We run Megiddo's algorithm for this validation. During the validation we keep track of the minimum  $k$  for which two sets are separable for all but  $k$  of the given constraints. Since we have  $O(k^2)$  faces in the arrangement the validation takes  $O(nk^2)$  time.

**Theorem 7** *If  $S_{RB}$ ,  $|S_{RB}| = k$ , is a set of minimum violations such that  $S_R \setminus S_{RB}$  and  $S_B \setminus S_{RB}$  are linearly separable then there exists a line  $l$  that separates red and blue points in the sets  $(S_R \cup S_B) \setminus S_{RB}$  and  $S_{RB}$  where all red (blue) points of  $(S_R \cup S_B) \setminus S_{RB}$  are on one side of  $l$  and all red (blue) points of  $S_{RB}$  are on opposite side.*

**Proof:** If there exists a line  $l$  that separates red and blue points of  $(S_R \cup S_B) \setminus S_{RB}$  but not  $S_{RB}$ , where all red (blue) points  $(S_R \cup S_B) \setminus S_{RB}$  are on one side of  $l$  and all red (blue) points of  $S_{RB}$  are on opposite side, then  $|S_{RB}|$  is not minimum.  $\square$



A formal description of the above iterative algorithm is given below.

---

**Algorithm** SWkVC (Separability With k Violated Constraints)

**Input:** (a) Line duals  $S_R^*$  and  $S_B^*$  of the point sets  $S_R$  and  $S_B$   
 (b) Upper bound  $K$  to the number of violated constraints, where  $K = O(1)$

**Output:** A minimum set of outliers  $S_{RB}$  of size  $k$  (i.e.  $k = |S_{RB}|$ )

**Step 1:** Initialize  $T = \phi$ ,  $D = S_R^* \cup S_B^*$ ,  $i = 0$ ,  $k = K + 1$ .

**Step 2:** do

call Megiddo( $D$ )  
 if(infeasible)  
    $i = i + 1$   
   if ( $i > K$ )  
     exit and report “not almost-separable”  
   identify the set of **eccentric-lines**,  $S_e$   
    $D = D/S_e$   
    $T = T \cup S_e$   
 while (infeasible)

**Step 3:** if ( $|T| == 0$ )

report “linearly separable with  $k = 0$ ” and exit

**Step 4:** Construct the arrangement of the lines in  $T$

**Step 5:** Assign levels to all the edges of the arrangement

**Step 6:** repeat (for each  $l \in T$ )

repeat (for each face  $F \in zone(l)$  )  
 if ( $(|S_{RB}^*| < k) \ \&\& \ (|S_{RB}^*| \geq i)$ )  
   call Megiddo( $(S_R^* \cup S_B^*) \setminus S_{RB}^*$  )  
   if (feasible)  
      $k = |S_{RB}^*|$   
      $S_{RB} = \text{Primal}(S_{RB}^*)$

**Step 7:** if ( $k \leq K$ )

report  $S_{RB}$  and  $k$

else

report “not almost-separable”

---

**Theorem 8** *The algorithm SW $k$ VC is correct.*

**Proof:** At the end of Step 2, the *tentative-set*,  $T$ , contains all potential outliers. Otherwise Megiddo's algorithm when run with  $D$  would report infeasibility. Otherwise, Megiddo's algorithm with input  $D$  returns infeasibility. If  $i > K$  then from Theorem 6  $k > K$  and the algorithm correctly reports "not almost-separable". Otherwise the algorithm checks for a minimum number of violations such that  $k \leq K$ .

All potential feasible regions with  $k$  violations are contained in the arrangement of  $T$ . From Theorem 6, the minimum  $k$  satisfies the inequality  $i \leq k \leq K$ . The rest of the algorithm scans for faces of the arrangement that satisfies  $i \leq |S_{RB}^*| \leq K$ . Each potential feasible region is validated by Megiddo's algorithm with input  $(S_R^* \cup S_B^*) \setminus S_{RB}^*$  and keeps record of the minimum  $|S_{RB}^*|$  that returns feasibility. From Theorem 7, the algorithm returns a minimum  $|S_{RB}|$  that separates red and blue points of  $(S_R \cup S_B) \setminus S_{RB}$ .

□

#### 4.4.4 Time Complexity

The algorithm takes  $O(nk)$  time for identifying the constraints of  $T$ , where  $n$  is total number constraints and  $k$  is the minimum number of outliers. The construction of the arrangement from  $T$  as well as the scanning of the arrangement for potential feasible regions (or faces), takes  $O(k^2)$  time. The validation step is run for at most  $O(k^2)$  faces, each validation taking  $O(n)$  time. Thus we have an  $O(nk^2)$  algorithm.

## 4.5 Experiment on gene expression datasets

We tested the above algorithm with five publicly available gene expression datasets: (a) Lung Cancer [79] (b) Leukemia Data [80] (c) SRBCT [64] (d) Colon Data [66] (e) Breast Cancer [63] (see Appendix A for detail about the datasets). To satisfy the general position assumption, we prune all the genes that have duplicate values in any two samples. This turns out to be a strong assumption for three out of the five gene expression datasets and prunes most of the genes. Table 4.1 shows numbers of genes in each dataset before and after the pruning.

Table 4.1: Five gene expression datasets before and after pruning

|    | Dataset            | Number of Genes |               | Total Samples |
|----|--------------------|-----------------|---------------|---------------|
|    |                    | before pruning  | after pruning |               |
| 1. | Lung Cancer [79]   | 12533           | 263           | 181(31+150)   |
| 2. | Leukemia [80]      | 12582           | 4702          | 52(24+28)     |
| 3. | SRBCT [64]         | 2308            | 2079          | 43(23+20)     |
| 4. | Colon [66]         | 2000            | 1982          | 62(40+22)     |
| 5. | Breast Cancer [63] | 21682           | 119           | 77(44+33)     |

Finally, we select 100 genes based on the correlation between gene expression values and class labels. This method was first proposed by Golub et al [62]. The correlation metric defined by Nguyen and Rocke [73] and by Golub et al [62] reflects the difference between the class mean relative to standard deviation within the class. High absolute value of this correlation metric favors those genes that are highly expressed in one class as compared to the other class, while their sign indicates the class in which the

gene is highly expressed. We have chosen to select genes based on a  $t$ -statistic defined by Nguyen and Rocke [73].

For  $i$ th gene, a  $t$ -value is computed using the formula

$$t_i = \frac{\mu_1^i - \mu_2^i}{\sqrt{\frac{\sigma_1^{i^2}}{n_1} + \frac{\sigma_2^{i^2}}{n_2}}} \quad (4.2)$$

where  $n_k$ ,  $\mu_k^i$  and  $\sigma_k^{i^2}$  are the sample size, mean and variance of  $i$ th gene respectively of class  $k = 1, 2$ .

The set of 100 genes obtained by taking 50 genes with the largest positive  $t$ -values and another 50 genes with the smallest negative  $t$ -values. In Table 4.2 and Figure 4.4, we show percentage of linear separability for  $k = 0, 1, \dots, 10$  on all possible gene pairs (i.e. 4950 gene pairs) out the 100 genes in each datasets.

In Leukemia and SRBCT, the percentage of linear separable pairs grow 10% to 20% approximately as  $k$  increase from 0 to 5. The datasets (i.e. Leukemia and SRBCT) show a less than 10% growth as  $k$  increases from 6 to 10. The other datasets (i.e. Colon, Lung Cancer and Breast Cancer) show a small growth as  $k$  increases from 0 to 4. These datasets (i.e. Colon, Lung Cancer and Breast Cancer) show a small number of linear separable pairs i.e. 39.03%, 16.85%, 15.47% respectively with  $k = 10$ . Figure 4.5 shows rate of growth of linear separable pairs for each dataset as  $k$  grows

Table 4.2: Almost linear separability for  $k = 0, 1, \dots, 10$

| k  | Percentage of Linear separable pair |          |             |       |               |
|----|-------------------------------------|----------|-------------|-------|---------------|
|    | Colon                               | Leukemia | Lung Cancer | SRBCT | Breast Cancer |
| 0  | 0.00                                | 1.25     | 0.00        | 0.87  | 0.00          |
| 1  | 0.00                                | 11.80    | 0.00        | 10.59 | 0.00          |
| 2  | 0.00                                | 31.90    | 0.04        | 28.67 | 0.10          |
| 3  | 0.00                                | 51.03    | 0.46        | 45.09 | 0.61          |
| 4  | 0.24                                | 66.91    | 1.07        | 60.28 | 2.59          |
| 5  | 1.90                                | 80.08    | 2.00        | 74.55 | 5.62          |
| 6  | 5.56                                | 88.02    | 3.76        | 86.08 | 6.20          |
| 7  | 11.64                               | 92.95    | 6.44        | 92.71 | 8.14          |
| 8  | 20.61                               | 95.98    | 8.87        | 96.57 | 9.52          |
| 9  | 30.46                               | 97.39    | 12.10       | 98.63 | 13.07         |
| 10 | 39.03                               | 98.51    | 16.85       | 99.37 | 15.47         |

from 0 to 10.

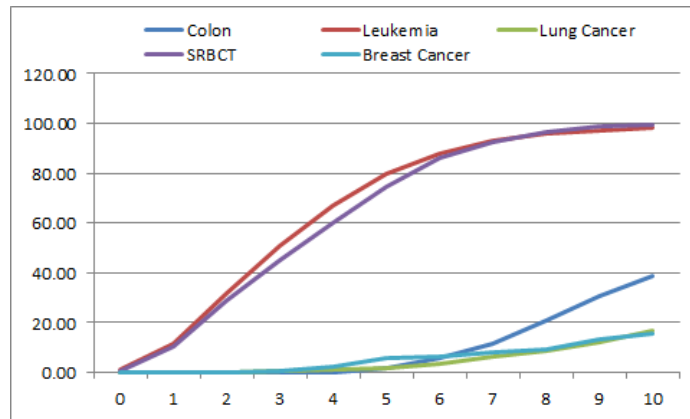
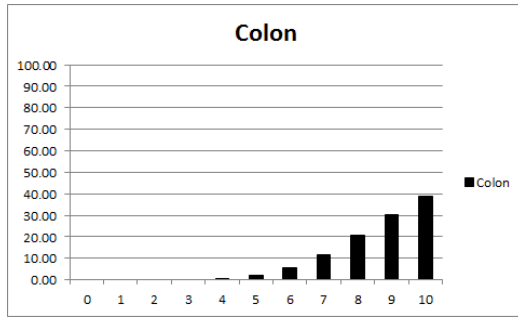
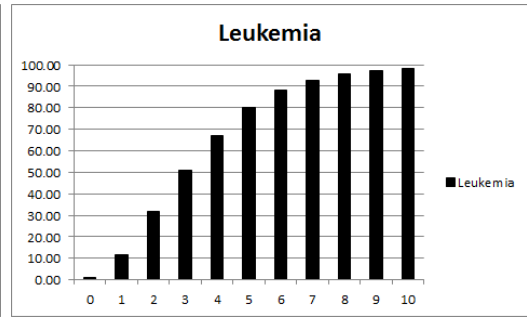


Figure 4.4: Percentage of linear separable pairs vs number of outliers ( $k$ )

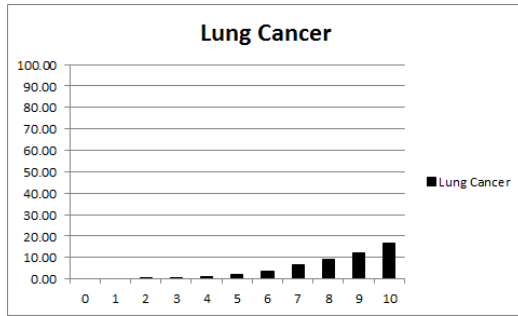
The case when a dataset exhibits a low linearly separable pairs, the proposed geometric tool can be used to select genes for a good classifier by simulating the steps discussed in chapter 3.



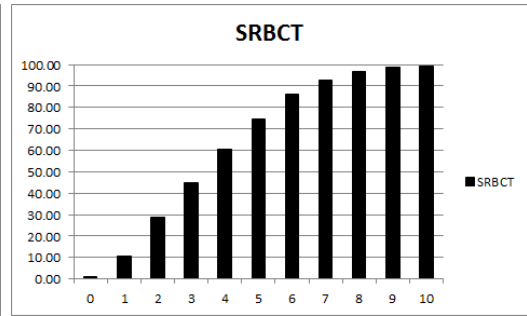
(a) Colon dataset



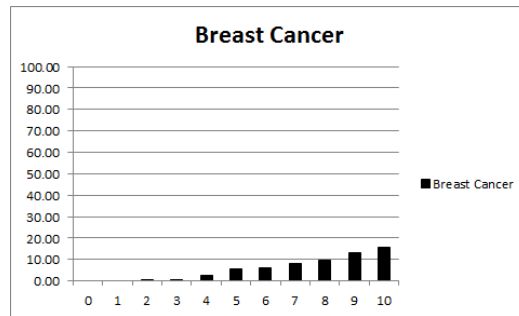
(b) Leukemia dataset



(c) Lung cancer dataset



(d) SRBCT dataset



(e) Breast cancer dataset

Figure 4.5: Rate of growth of linear separable pairs with increase in  $k$

## 4.6 Summary

This chapter presents an efficient technique for detecting linear separability with a minimum number of violations. These violations are due to noise, sampling error or round off error. Practically, if the allowable number of violations  $k$  is  $O(1)$  then to our knowledge, this is the only linear time algorithm known for this problem. When  $k = o(\log^2 n)$  the proposed algorithm is better than the known algorithms by Everett et al. [10], Matussek [11] and Chan [12].

We also tested the above algorithm on five publicly available datasets and report the rate of growth of linear separable pairs as  $k$  increases from 0 to 10. In this line of the study, some promising directions for future works are (a) extend the proposed technique to higher dimensions (b) to demonstrate the use of the proposed geometric tool to select genes for a good classifier.

## Chapter 5

# An eigendecomposition method for protein structure alignment

### 5.1 Overview

The alignment of two protein structures is a fundamental problem in structural bioinformatics. Their structural similarity carries with it the connotation of similar functional behavior that could be exploited in various applications. In this chapter, we model a protein as a polygonal chain of  $\alpha$  carbon residues in three dimension and investigate the application of an eigendecomposition method due to Umeyama to the protein structure alignment problem. This method allows us to reduce the structural alignment problem to an approximate weighted graph matching problem.

The chapter introduces two new algorithms,  $EDAlign_{res}$  and  $EDAlign_{sse}$ , for pairwise protein structure alignment.  $EDAlign_{res}$  identifies the best structural alignment of two equal length proteins by refining the correspondence obtained from eigendecomposition and to maximize similarity measure, TM-score, for the refined correspondence.  $EDAlign_{sse}$ , on the other hand, does not require the input proteins to be of equal length. It works in three stages: (1) identifies a correspondence between secondary structure elements (i.e SSE-pairs); (2) identifies a correspondence



between residues within SSE-pairs; (3) applies a rigid transformation to report structural alignment in space. The latter two steps are repeated until there is no further improvement in the alignment. We report the TM-score and cRMSD as measures of structural similarity. These new methods are able to report sequence and topology independent alignments, with similarity scores that are comparable to those of the state-of-the-art algorithms such as, TM align and SuperPose.

## 5.2 Introduction

Along with DNA and RNA, protein molecules are the main drivers of all life processes at the molecular level. A protein molecule is a linear polypeptide chain, with adjacent pairs of amino acids, joined together by a peptide bond, giving rise to the nomenclature “polypeptide”. In order to perform its particular biological function, the linear polypeptide chain folds into a stable, low-energy 3-dimensional tertiary structure. The latter structure is formed by the joining together of two types of secondary structures, known as  $\alpha$ -helices and  $\beta$ -sheets.

The two important aspects of this process are: (1) how the folding takes place; (2) how does the particular structure it assumes allows it to perform its designated function. The first is well-known as the protein folding problem, predicting how a protein will fold, given the amino acid sequence that makes up its polypeptide chain structure. This problem still awaits a comprehensive solution. The second problem is

that of predicting function from structure. Here a reductionist approach is a popular one: structural comparisons with proteins of known functions. Thus the problem of structural alignments of proteins, which is the subject of this chapter.

As Taylor et al. [107] observed, “The most important things we know about proteins have come therefore not from theory but from observation and comparison of sequences and structures”. In view of the importance of the problem, numerous heuristics have been proposed, consequently giving rise to an extensive literature and several large structural databases of proteins [108–110]. These databases help in the classification of the large space of protein sequences into structurally equivalent classes by means of alignment or structure comparison algorithms.

In order to design an alignment algorithm, it is important to enunciate clearly the protein model that will be used. Some of the earliest alignment algorithms [111,112] assumed a model in which the central  $\alpha$  carbon atom of each residue are joined successively to form a polygonal chain in three dimensions. A more primitive model is to view a protein as a collection of points (again the  $\alpha$  carbon atoms) in three space, which allows one to view the alignment problem as that of matching two point sets. We must point out that in order to draw biologically meaningful conclusions from an alignment, it is important to supplant these models with features of the proteins like hydrophobicity, exposure to solvents, mutual affinities of amino acids etc.

The alignment of two protein structures is the 3-dimensional analogue of linear sequence alignment of peptide or nucleotide sequences. An initial equivalence set can be obtained by various methods such as comparison of distance matrix [113], maximal common subgraph detection [114, 115], geometric hashing [116, 117], local geometry matching [118], spectral matching [119], contact map overlap [120–124] and dynamic programming [42, 125]. This equivalence set is optimized by different methods such as a Monte Carlo algorithm or simulated annealing [113], dynamic programming [42, 125–127], incremental combinatorial extension of the optimal path [128] and genetic algorithm [129]. Indeed the goal is to determine an alignment of protein residues to measure the extent of structural similarity. To quantify this similarity, various measures have been defined and can be broadly classified into four categories: (1) distance map similarity [113, 130–132] (2) root mean square deviation (*RMSD*) [42, 119, 128, 133, 134] (3) contact map overlap [135] (4) universal similarity matrix [136, 137]. A comprehensive list of different similarity measures are discussed by Hasegawa and Holm [138]. Surprisingly, even after so many years of research there is no universally acknowledged definition of similarity score to measure the extent of structural similarity [138, 139].

In [122], alignment of eigenvectors is used for fast overlapping of contact maps. The chapter uses Needleman-Wunch’s algorithm to compute a global alignment of two

protein sequences, where the cost function is derived from an approximation of the contact map  $M$  (of the two protein structures), obtained from the spectral decomposition of  $M$ . Using a graph theoretic approach, Taylor et al. [140] obtained a structural similarity measure by matching pairs of secondary structural elements (SSEs) of the input proteins. The set of matching pairs of SSEs is obtained by a bipartite graph-matching algorithm.

In this chapter we introduce two new algorithms,  $EDAlign_{res}$  and  $EDAlign_{sse}$ , for the protein structure alignment problem. These algorithms rely on a matrix eigen-decomposition approach due to Umeyama [141] for an approximate solution to the weighted graph matching problem.  $EDAlign_{res}$  identifies best structural alignment of two equal length proteins by refining the correspondence obtained from the eigendecomposition technique and to maximize similarity measure, TM-score, for the refined correspondence.  $EDAlign_{sse}$ , on the other hand, does not require the input proteins to be of equal length. It works in three stages: (1) identifies correspondence between secondary structure elements (i.e SSE-pairs); (2) identifies a correspondence between residues within SSE-pairs; (3) applies a rigid transformation to report structural alignment in space. The latter two steps are repeated until there is no improvement in the alignment. These methods are able to provide sequence and topology independent similarities. The reason for this is that the primary equivalence set (residues-pairs for equal length proteins and SSE-pairs for unequal length proteins) depends on the in-

trinsic geometry of  $\alpha$ -the carbon atoms within the tertiary structure that is revealed by eigendecomposition. We report the TM-score and *cRMSD* as measures of the structural similarity. The similarity scores of both the algorithms are comparable to those of the state-of-the-art algorithms such as, TM align and SuperPose.

## 5.3 Preliminaries

### 5.3.1 Notations and Definitions

The following definitions help us formulate the problem precisely.

**Definition:** A protein  $P$  is a sequence of points,  $P = \{p_i | p_i \in R^3, i = 1, 2, 3, \dots, m\}$ , in a 3-dimensional Euclidean space, where  $m(= |P|)$  is the number residues and  $p_i$  represents the coordinates of the central  $\alpha$ -carbon atom of the  $i$ -th residue.

**Definition:** Given two proteins  $P$  and  $Q$  of length  $m$  and  $n$  respectively. An alignment of  $P$  and  $Q$  is:

- a sequence of corresponding pairs of points of  $P$  and  $Q$ ,

$$S(P, Q) = \{(p_{i_1}, q_{j_1}), (p_{i_2}, q_{j_2}), \dots, (p_{i_k}, q_{j_k})\}, \text{ where } 1 \leq i_1 < i_2 < \dots < i_k \leq m$$

and  $1 \leq j_1 \neq j_2 \neq \dots \neq j_k \leq n$ , together with

- a rigid transformation  $t$ ,  $t(Q) = \{t(q_j) = q'_j | q'_j \in R^3, j = 1, 2, 3, \dots, n\}$ , that optimizes some similarity measure for the above correspondence.

**Definition:** A residue  $p_i$  of a protein  $P$  is known as a  $k$ -neighbor of another residue  $p_j$  if  $|i - j| = k$ , where  $1 \leq i, j \leq |P|$ .

### 5.3.2 Similarity measures

To measure the extent of structural similarity of two proteins, the root mean square deviation (*RMSD*) is widely used [139, 142]. Two different *RMSD* measures have been proposed in the literature: (1) coordinate root mean square deviation (*cRMSD*) and (2) distance root mean square deviation (*dRMSD*). In the proposed algorithms, *EDAlign<sub>res</sub>* and *EDAlign<sub>sse</sub>*, we obtain a correspondence (i.e. residue-pairs for equal length proteins and SSE-pairs for unequal length proteins) that minimizes the *dRMSD* measure (see equation 5.5) and finally reports an alignment that minimizes the *cRMSD* measure and maximizes the TM-score [41, 42]. For completeness, the *cRMSD* and *dRMSD* measures are defined below.

**Definition:** The similarity measure between two aligned substructures of proteins  $P$  and  $Q$  of length  $k$  can be defined as follows

$$dRMSD = \sqrt{\frac{2}{k^2 - k} \sum_{u=1}^{k-1} \sum_{v=u+1}^k (\|p_{i_u} - p_{i_v}\| - (\|q_{j_u} - q_{j_v}\|))^2}; \text{ and} \quad (5.1)$$

$$cRMSD = \sqrt{\frac{1}{k} \sum_{u=1}^k \|p_{i_u} - t(q_{j_u})\|^2}. \quad (5.2)$$

Since the similarity measures,  $cRMSD$  and  $dRMSD$ , are in terms of absolute distances, a small presence of outliers may result in a poor  $RMSD$  even if the two structures are globally similar. A similar observation has been made by other researchers [42, 139, 143, 144]. To circumvent this problem, Zhang and Skolnick [41] introduced a sequence independent structural alignment measure (TM-score) that is a variation of a measure originally defined by Levitt and Gerstein [145]. A critical assessment of this TM-score has been given by Xu and Zhang [146].

**Definition:** Given two proteins, a template protein  $P$  and a target protein  $Q$ ,  $|P| \geq |Q|$ , the structural similarity is obtained by a spatial superposition of  $P$  and  $Q$  that maximizes the following score

$$\text{TM-score} = \frac{1}{|Q|} \sum_{i=1}^k \frac{1}{1 + \left(\frac{d_i}{d_0}\right)^2}, \quad (5.3)$$

where  $k$  is the number of aligned residues of  $P$  and  $Q$ ;  $d_i$  is the distance between  $i$ -th pair of aligned residues and  $d_0 (= 1.24 \sqrt[3]{|Q| - 15} - 1.8)$  is a normalization factor.

When the value of  $d_0$  in equation (5.3) is set to  $5A^\circ$ , the resulting TM-score is known as a raw TM-score (rTM-score). In  $EDAlign_{sse}$ , to report the TM-score the protein lengths are set to the number of residues in the aligned SSEs, ignoring the residues

in the fragments that connects these SSEs. Despite this, the modified score successfully reveals the extent of similarity between the aligned SSEs. Xu and Zhang [146] observed that two proteins are structurally similar and belong to same fold when the TM-score  $> 0.5$ .

### 5.3.3 Umeyama's matrix eigendecomposition method

The algorithms proposed in this chapter rely on Umeyama's matrix eigendecomposition method for weighted graph matching [141] to generate sequence independent alignments, i.e. residue-pairs for equal length proteins and SSE-pairs for unequal length proteins. To make the chapter self-contained, we briefly describe Umeyama's technique.

Let  $P$  and  $Q$  be two proteins of length  $N$  each. Let  $P_G$  ( $Q_G$ ) be the adjacency matrix corresponding to a weighted graph  $G(H)$  whose vertices are the central  $\alpha$ -carbon atoms of  $P$  ( $Q$ ) and  $w(p_i, p_j)$  ( $w(q_i, q_j)$ ) is the Euclidean distance between  $i$ -th and  $j$ -th residues of  $P$  ( $Q$ ). This reduces the protein structure alignment problem to a weighted undirected graph matching problem. This problem is NP-Complete as this is a special case of largest common subgraph problem [147].

In particular, Umeyama's method seeks to obtain a node correspondence



$$S = \{(p_i, \phi(p_i)) \mid p_i \in P \text{ and } \phi(p_i) \in Q\} \quad (5.4)$$

that minimizes the following distance measure

$$J(\phi) = \sum_{i=1}^N \sum_{j=1}^N ((w(p_i, p_j) - w(\phi(p_i), \phi(p_j))))^2. \quad (5.5)$$

Umeyama showed that the mapping  $\phi()$  can be approximated by a permutation matrix  $\Pi$ , and instead minimizes the following measure:

$$J(\Pi) = \|\Pi P_G \Pi^T - Q_G\|^2 \quad (5.6)$$

where  $\|\cdot\|$  represents the Euclidean norm.

The proposed approximation algorithm is based on Theorem 3 below, which is proved [141] using the next two theorems.

**Theorem 9** *The eigendecompositions of the real symmetric matrices  $P_G$  and  $Q_G$  are given by*

$$\begin{aligned} P_G &= U_P \Lambda_P U_P^T \\ Q_G &= U_Q \Lambda_Q U_Q^T \end{aligned} \quad (5.7)$$

where  $U_P$  ( $U_Q$ ) is an orthogonal matrix and  $\Lambda_P$  ( $\Lambda_Q$ ) is diagonal. The entries of the diagonal matrix  $\Lambda_P$  ( $\Lambda_Q$ ) are the (real) eigenvalues of  $P_G$  ( $Q_G$ ) and the columns of the orthogonal matrix  $U_P$  ( $U_Q$ ) are the eigenvectors of  $P_G$  ( $Q_G$ ).

**Theorem 10** *If  $P_G$  and  $Q_G$  are symmetric matrices then*

$$\|P_G - Q_G\|^2 \geq \sum_{i=1}^n (\lambda_i - \mu_i)^2 \quad (5.8)$$

where  $\lambda_i$  ( $\mu_i$ ),  $i = 1, 2, \dots, n$  are the eigenvalues of  $P_G$  ( $Q_G$ ) with  $\lambda_i \geq \lambda_{i+1}$  ( $\mu_i \geq \mu_{i+1}$ ).

**Theorem 11** *Let  $P_G$  and  $Q_G$  two real symmetric matrices with distinct eigenvalues.*

*If  $O$  is an orthogonal matrix, ranging over the set of all orthogonal matrices, then*

*$\|OP_G O^T - Q_G\|^2$  attains its minimum when*

$$O = U_Q S U_P^T, \quad (5.9)$$

where  $S = \{s_i | s_i = 1 \text{ or } -1, i = 1, 2, \dots, n\}$ .

If there exists a protein homology, without any conformational changes, between  $P$  and  $Q$  then the two weighted graphs  $G$  and  $H$  are isomorphic. Thus from equation (5.6) we have:

$$\Pi P_G \Pi^T = Q_G. \quad (5.10)$$

Since the eigenvalues of two isomorphic graphs  $G$  and  $H$  are the same, from theorem 11 we have

$$OP_GO^T = Q_G. \quad (5.11)$$

Thus

$$OP_GO^T = \Pi P_G \Pi^T$$

$$U_Q S U_P^T U_P \Lambda_P U_P^T U_P S U_Q^T = \Pi U_P \Lambda_P U_P^T \Pi^T$$

$$\Pi U_P = U_Q S$$

$$\Pi = U_Q S U_P^T.$$

Though the matrix  $\Pi$  in the last line above is orthogonal, it is not necessarily a permutation matrix. Umeyama [141] showed that the desired permutation matrix  $\Pi$  can be obtained using the Hungarian method on a suitably defined matrix as below:

$$\Pi = \text{Hungarian}(|U_Q| |U_P^T|), \quad (5.12)$$

where  $|U_P^T|$  and  $|U_Q|$  are matrices whose entries are the absolute values of the corresponding entries of  $U_P^T$  and  $U_Q$ . This enables us to obtain a residue correspondence  $S(P, Q)$ .

## 5.4 Methods

To design algorithm  $EDAlign_{res}$ , we first reformulate the pairwise structural alignment problem as a weighted graph matching problem, and apply the matrix eigendecomposition method due to Umeyama [141] to obtain an equivalence set of residues. Next, we use the primary sequences of the proteins to refine the equivalence set by a two-stage strategy: (1) pruning outliers; (2) replacing outliers (patching). During the *pruning* step, we identify  $\phi(p_i), 1 < i < N (= |P|)$ , as an outlier if it is neither a 1-neighbor of  $\phi(p_{i-1})$  nor of  $\phi(p_{i+1})$ . Similarly  $\phi(p_1)$  ( $\phi(p_N)$ ), is an outlier if it is neither a 1-neighbor of  $\phi(p_2)$  ( $\phi(p_{N-1})$ ) nor a 2-neighbor of  $\phi(p_3)$  ( $\phi(p_{N-2})$ ). Once we have identified all outliers, we substitute each suitably, whenever possible. We call this *patching*.

Thus if  $\phi(p_i), 1 < i < N$ , is an outlier then we identify two non outliers,  $q_h \in \{\phi(p_{i-k}) \mid k = 1, 2\}$  and  $q_j \in \{\phi(p_{i+l}) \mid l = 1, 2\}$  such that  $q_h$  and  $q_l$  are  $(k + l)$ -neighbor along  $Q$ . We replace  $\phi(p_i)$  with residue  $q_{h+k}$  ( $= q_{j-l}$ ). For  $i = N$ , we have  $q_h \in \{\phi(p_{i-k_1}) \mid k_1 = 1, 2, 3, 4\}$  and  $q_j \in \{\phi(p_{i-k_2}) \mid k_2 = 1, 2, 3, 4\}$  such that  $k_1 \neq k_2$  and  $q_h$  and  $q_j$  are  $|k_1 - k_2|$ -neighbor along  $Q$ . Thus  $\phi(p_N)$  can be replace by  $q_{h-k_1}$  ( $= q_{j-k_2}$ ). We replace  $\phi(p_1)$  similarly when it is an outlier.

Finally, the aligned residue order with non outliers are used to obtained an alignment that maximizes the TM-score. This involves an application of Kabsch's method to

get an initial alignment of two proteins in space. The alignment is refined by repetitive application of dynamic programming followed by Kabsch's rotation that only considers the corresponding pairs, separated by a distance  $d_i < d_0$  (see equation 5.3),  $1 \leq i \leq N$ .

We note once again that the applicability  $EDAlign_{res}$  is limited to equal length proteins. In  $EDAlign_{sse}$ , we overcome this limitation by using matrix eigendecomposition to obtain SSE-pairs and subsequently residue-pairs from these. The details are as follows.

**Identifying SEEs:** In this step, we map the residues to secondary structure elements(SSEs) which are limited to  $\alpha$ -helices and  $\beta$ -sheets. Based on the hydrogen bond patterns of secondary structure elements (SSEs), Kabsch and Sander [148] came up with following inequalities for assigning a residue to  $\alpha$ -helix ( $\beta$ -sheet)

$$\left|d_{j,j+k} - \lambda_k^{\alpha(\beta)}\right| < \delta^{\alpha(\beta)}, \quad (j = i - 2, i - 1, i; k = 2, 3, 4) \quad (5.13)$$

The optimized parameters for the above inequalities [42, 148] are  $\lambda_2^\alpha = 5.45A^\circ$ ,  $\lambda_3^\alpha = 5.18A^\circ$ ,  $\lambda_4^\alpha = 6.37A^\circ$ ,  $\delta^\alpha = 2.1A^\circ$ ,  $\lambda_2^\beta = 6.1A^\circ$ ,  $\lambda_3^\beta = 10.4A^\circ$ ,  $\lambda_4^\beta = 13A^\circ$ ,  $\delta^\beta = 1.42A^\circ$ . To identify such structures we have used the DSSP program that implements these inequalities [148, 149].

**Representations of SSEs:** Let  $SSE_i^\alpha$  denote the  $i$ -th  $\alpha$ -helix of a residue chain with  $n$   $\alpha$ -carbon atoms. We use following set of  $\alpha$ -carbon atoms to represent the  $\alpha$ -helix

$$\left\{ C_k | k = 1, 2, 3, m, n - 2, n - 1, n \quad \text{and} \quad m = \frac{n + 1}{2}, n \geq 7 \right\}. \quad (5.14)$$

If  $n$  is even  $C_m$  represents a virtual  $\alpha$ -carbon atom whose coordinates are obtained by averaging the coordinates of  $\alpha$ -carbon atoms  $C_{\frac{n}{2}}$  and  $C_{\frac{n}{2}+1}$ .

Similarly, to represent a  $\beta$ -sheet,  $SSE_i^\beta$ , with  $n$   $\alpha$ -carbon atoms, we use the following representative set of  $\alpha$ -carbon atoms

$$\left\{ C_k | k = 1, m_1, m_2, n \quad \text{and} \quad m_1 = \left\lfloor \frac{n}{2} \right\rfloor, m_2 = \left\lceil \frac{n + 1}{2} \right\rceil, n \geq 4 \right\}. \quad (5.15)$$

Since SSEs such as  $\alpha$ -helices and  $\beta$ -sheets show regular patterns of hydrogen bonds, the above representation does not affect the overall topology. For this reason such structures have even been represented as vectors in some earlier protein structure alignment algorithms [131, 150].

**Identifying SSEs for alignment:** Let protein  $P$  ( $Q$ ) have  $n_1$  ( $m_1$ )  $\alpha$ -helices and  $n_2$  ( $m_2$ )  $\beta$ -sheets. Assume that  $n_1 > m_1$  and  $n_2 > m_2$ . This gives us  $\prod_{i=1}^2 n_i C_{m_i}$ , possible combinations of SSEs from  $P$  that can be aligned with those from  $Q$ . This value becomes impractically large when the differences  $m_i - n_i$  are large. Fortunately,

proteins pairs do not differ much with respect to the number of SSEs. Nevertheless, in cases where the differences exceed a prescribed threshold value, noting that the SSEs are ordered along a protein chain, we allow only the following combinations of SSEs from  $P$  as candidates for alignment with those from  $Q$ .

$$S_{i,j}^P = \left\{ SSE_{i+1}^\alpha, SSE_{i+2}^\alpha, \dots, SSE_{i+m_1}^\alpha, SSE_{j+1}^\beta, SSE_{j+2}^\beta, \dots, SSE_{j+m_2}^\beta \right\}, \quad (5.16)$$

where the values of  $i$ ,  $j$  are in the range of  $[0, n_1 - m_1]$  and  $[0, n_2 - m_2]$ . Such a selection is consistent with other alignment techniques such as dynamic programming and combinatorial extension that also consider residues along the chain with reasonable gaps, and also reduces the number of possible combinations of SSEs to a quadratic order:  $\prod_{i=1}^2 (n_i - m_i)$ . The cases where  $n_1 < m_1$  and  $n_2 > m_2$  or  $n_1 > m_1$  and  $n_2 < m_2$ , with  $(n_1 + n_2) > (m_1 + m_2)$  in each case, can be handled in a similar way.

**Identifying SSE-pairs:** It is reasonable to assume that regions of  $P$  and  $Q$  that are perfectly aligned have an equal number of  $\alpha$ -helices as well as  $\beta$ -sheets in both. Suppose we know the candidate sets  $S_{i,j}^P$  and  $S_{i',j'}^Q$  that are to be aligned. We can represent these sets of SSEs,  $S_{i,j}^P$  and  $S_{i',j'}^Q$ , as complete weighted graphs on their constituent  $\alpha$ -carbon atoms which are input to Umeyama's method. The output of Umeyama's method is a symmetric matrix  $M$  (see equation 5.12), each entry being

the cost of the correspondence between a pair of alpha carbon atoms one in  $S_{i,j}^P$  and the other in  $S_{i',j'}^Q$ . We coalesce the  $\alpha$ -carbon atoms that belong to an SSE into a single entity and create a modified cost matrix,  $M'$ , each entry being the cost of the correspondence of a pair of SSEs, one in  $S_{i,j}^P$  and the other in  $S_{i',j'}^Q$ . When both the SSEs are  $\alpha$ -helices the cost is:

$$M'[SSE_P^\alpha, SSE_Q^\alpha] = \frac{\sum_{C_a \in SSE_P^\alpha, C_b \in SSE_Q^\alpha} M(C_a, C_b)}{49}, \quad (5.17)$$

while if both are  $\beta$ -sheets the cost is:

$$M'[SSE_P^\beta, SSE_Q^\beta] = \frac{\sum_{C_a \in SSE_P^\beta, C_b \in SSE_Q^\beta} M(C_a, C_b)}{16}. \quad (5.18)$$

To avoid a correspondence between an  $\alpha$ -helix and a  $\beta$ -sheet, we set the cost of such a correspondence to zero. Finally, we apply the Hungarian method to this modified cost matrix  $M'$  to get a correspondence that optimizes the total cost. The aligned SSE pairs are used to obtain an initial spatial structural alignment.

**Reordering of residues:** In a structural alignment the order of the SSEs may not be same as the primary sequence order. Therefore we reorder the residues, according to the SSE-pairs obtained from the previous step. To make this precise we set the smaller length protein  $Q$  as the template and arrange its SSEs as these appear along the chain. We order the SSEs of the  $P$  according to their correspondence with



the SSEs of  $Q$ , ignoring those SSEs that do not have a corresponding SSE in  $Q$ . Now a corresponding SSE-pair may align with each other either in forward or reverse direction. To determine the correct direction we have exhaustively checked all possible  $2^m$  combinations where  $m$  is the number of SSEs. Of these combinations, we choose the one with minimum  $J(\phi)$ . Here we also consider the topological ordering of alignment set as one candidate as a majority of protein structural alignment algorithms use the conventional sequence order, primarily for biological reasons [151]. In the above rearrangement, we ignore residues in the loops that connect the SSEs. Finally, we reorder the residues according to their appearance in the ordering of SSEs.

**Apply Dynamic Programming:** To refine the alignment, the residue order obtained from the previous step is input to a dynamic programming [41, 42, 145] algorithm. The entries of the scoring matrix are defined by

$$S(i, j) = \frac{1}{1 + \left(\frac{d_{ij}}{d_0}\right)^2} \quad (5.19)$$

where  $d_{ij}$  is the distance between the  $i$ -th residue in  $P$  and the  $j$ -th residue in  $Q$  and  $d_0$  is scale factor that normalizes the distances between residue pairs of  $P$  and  $Q$  (see equation 5.3). Setting an opening gap penalty of -0.6, and considering pair correspondences that are at distances less than  $d_0$ , we apply Kabsch's method to superimpose  $P$  and  $Q$ . The process is repeated until the alignment becomes stable

with maximum TM-score. Based on our experiments, it takes typically 2-3 steps to get the best alignment - a fact also observed by Zhang and Skolnick [42].

## 5.5 Results and Discussions

To illustrate the proposed methods, we apply both to pairs of proteins in the following two categories [43]: (1) same primary sequence with slightly different tertiary structures; (2) same primary sequence with vastly different tertiary structures.

As can be seen from Table 5.1, the scores computed by both *EDAlign<sub>res</sub>* and *EDAlign<sub>sse</sub>* are as good or better than the scores computed by SuperPose [43] and TM-align [42]. The number of residues aligned by *EDAlign<sub>sse</sub>* is smaller than that of TM-align as it ignores the residues in the loop fragments that join pairs of SSEs and this fact is reflected in the computation of the TM-score. Nevertheless, the TM-score of *EDAlign<sub>sse</sub>* compares remarkably well with that of TM-align. The results also show that *EDAlign<sub>res</sub>* is more successful in detecting the structural similarity of homologous proteins of equal length and therefore might be potentially useful during NMR spectroscopy. To further illustrate the effectiveness of *EDAlign<sub>res</sub>* and its improvement over the basic Umeyama method, we ran this algorithm on NMR models *1m2f\_A.1* – *1m2f\_A.25* to an average model *1m2e\_A*. Table 5.2 also includes results from the refinement stages (i.e. pruning outliers and replacing outliers) of *EDAlign<sub>res</sub>*.

Table 5.1: Pairwise structural alignment of equal length proteins

| Structure                                      | N   | Seq. id | <i>EDAlign<sub>res</sub></i> |             | <i>EDAlign<sub>ssc</sub></i> |             | TM-align         |             | SuperPose   |             |
|--|-----|---------|------------------------------|-------------|------------------------------|-------------|------------------|-------------|-------------|-------------|
|  |     |         | cRMSD                        | TM-score    | cRMSD                        | TM-score*   | cRMSD            | TM-score    | cRMSD       | TM-score    |
| Same sequence and similar structure (pair)     |     |         |                              |             |                              |             |                  |             |             |             |
| Thioredoxin<br>(2TRXA 2TRXB)                   | 108 | 100%    | <b>0.45 - 105</b>            | 0.94        | 0.26 - 53                    | <b>0.98</b> | 0.66 - 108       | <b>0.98</b> | 0.77 - 108  | 0.97        |
| Hemoglobin<br>(4HHBA 1DKEA)                    | 141 | 100%    | <b>0.31 - 141</b>            | <b>0.99</b> | 0.26 - 92                    | <b>0.99</b> | 0.37 - 141       | <b>0.99</b> | 0.37 - 141  | <b>0.99</b> |
| P21 Oncogene<br>(6Q21A 6Q21B)                  | 171 | 100%    | <b>0.39 - 108</b>            | 0.61        | 0.31 - 91                    | <b>0.99</b> | 1.22 - 171       | 0.96        | 1.27 - 171  | 0.96        |
| ~ Same sequence and different structure (pair) |     |         |                              |             |                              |             |                  |             |             |             |
| Calmodulin<br>(1A29 1CLL)                      | 144 | 98.6%   | 22.81 - 144                  | 0.59        | 10.19 - 59                   | <b>0.62</b> | <b>1.91 - 71</b> | 0.51        | 23.83 - 142 | 0.0002      |
| Maltose Bind<br>Prot.<br>(1OMP 1ANF)           | 370 | 100%    | <b>3.04 - 327</b>            | 0.65        | 2.83 - 176                   | 0.76        | 3.42 - 364       | <b>0.82</b> | 8.87 - 369  | 0.79        |

cRMSD values are reported as backbone cRMSD in  $A^o$  - number of aligned  $\alpha$ -carbon atoms

N: Number of residues in protein structure

\*: Ignores the residues in the fragment that connects SSEs

Unlike  $EDAlign_{res}$ ,  $EDAlign_{sse}$  can detect the structural similarity of any pair of proteins. To substantiate this, we have run  $EDAlign_{sse}$  on pairs of proteins (see Table 5.3) that are in the following categories [43]: (1) have modestly dissimilar sequences, lengths and structures (2) have vastly different lengths but similar structures or sequences. Table 5.3 shows that the TMscore of  $EDAlign_{sse}$  is as good as that of TM align and on top of that is able to locate conserved regions between protein pairs.

$EDAlign_{sse}$  is also able to detect structural similarity independent of topological order (see Figure 5.1). To support this claim, we have run  $EDAlign_{sse}$  to compare the protein Apolipoprotein III (PDB ID:1aep) to a theoretical model of four-helix bundle protein (PDB ID:1fx). As a further demonstration of the versatility of  $EDAlign_{sse}$  we consider a difficult case for alignment: (1) Core-binding factor alpha subunit (PDB ID:1e50, Chain A) with Riboflavin synthase alpha chain (PDB ID: 1pkv, Chain A) (2) Hemoglobin (Deoxy) (PDB ID:2hbg) with Tyrosine-protein kinase ABL2 (PDB ID:2ecd). The alignment of SSEs obtained by  $EDAlign_{sse}$  is shown in Figure 5.2. Notably, the two proteins 1e50A and 1pkvA have three aligned

Table 5.2: NMR models 1m2f\_A\_1-1mf\_A\_25, compared to an average model 1m2e\_A

| Model | Umeyama Method |          | <i>EDAlign<sub>res</sub></i> |          |                               |          | TM-align   |             | Superpose  |             |
|-------|----------------|----------|------------------------------|----------|-------------------------------|----------|------------|-------------|------------|-------------|
|       | cRMSD          | TM-score | Pruning Outliers             |          | Replacing Outliers (Patching) |          | cRMSD      | TM-score    | cRMSD      | TM-score    |
|       |                |          | cRMSD                        | TM-score | cRMSD                         | TM-score |            |             |            |             |
| 1     | 1.294 - 135    | 0.93     | 0.607 - 133                  | 0.91     | <b>0.641 - 135</b>            | 0.93     | 1.06 - 135 | <b>0.95</b> | 1.06 - 135 | <b>0.95</b> |
| 2     | 1.032 - 135    | 0.92     | 0.542 - 128                  | 0.89     | <b>0.585 - 131</b>            | 0.9      | 0.91 - 135 | <b>0.96</b> | 0.91 - 135 | <b>0.96</b> |
| 3     | 3.454 - 135    | 0.84     | 0.387 - 111                  | 0.77     | <b>0.424 - 124</b>            | 0.88     | 1.54 - 134 | <b>0.94</b> | 1.83 - 135 | 0.92        |
| 4     | 5.059 - 135    | 0.78     | 0.632 - 95                   | 0.65     | <b>0.698 - 110</b>            | 0.76     | 1.42 - 135 | <b>0.94</b> | 1.42 - 135 | 0.93        |
| 5     | 0.891 - 135    | 0.91     | 0.755 - 134                  | 0.92     | <b>0.755 - 134</b>            | 0.92     | 0.89 - 134 | <b>0.95</b> | 1.11 - 135 | <b>0.95</b> |
| 6     | 6.359 - 135    | 0.75     | 0.657 - 90                   | 0.61     | <b>0.766 - 113</b>            | 0.76     | 1.51 - 135 | <b>0.94</b> | 1.51 - 135 | 0.93        |
| 7     | 0.757 - 135    | 0.92     | 0.685 - 135                  | 0.92     | <b>0.685 - 135</b>            | 0.92     | 1.12 - 135 | <b>0.95</b> | 1.12 - 135 | <b>0.95</b> |
| 8     | 0.814 - 135    | 0.93     | 0.604 - 133                  | 0.92     | <b>0.612 - 135</b>            | 0.93     | 0.79 - 135 | <b>0.97</b> | 0.79 - 135 | <b>0.97</b> |
| 9     | 4.896 - 135    | 0.78     | 0.496 - 98                   | 0.68     | <b>0.548 - 115</b>            | 0.79     | 1.33 - 135 | <b>0.94</b> | 1.34 - 135 | 0.93        |
| 10    | 0.848 - 135    | 0.91     | 0.646 - 133                  | 0.91     | <b>0.705 - 135</b>            | 0.92     | 0.91 - 134 | <b>0.95</b> | 1.09 - 135 | <b>0.95</b> |
| 11    | 0.638 - 135    | 0.93     | 0.629 - 135                  | 0.93     | <b>0.629 - 135</b>            | 0.93     | 0.89 - 135 | <b>0.96</b> | 0.89 - 135 | <b>0.96</b> |
| 12    | 1.411 - 135    | 0.9      | 0.588 - 128                  | 0.88     | <b>0.662 - 135</b>            | 0.92     | 1.09 - 135 | <b>0.95</b> | 1.09 - 135 | <b>0.95</b> |
| 13    | 1.022 - 135    | 0.89     | 0.745 - 133                  | 0.91     | <b>0.786 - 135</b>            | 0.92     | 0.88 - 133 | <b>0.95</b> | 1.38 - 135 | 0.93        |
| 14    | 1.318 - 135    | 0.92     | 0.532 - 130                  | 0.91     | <b>0.629 - 135</b>            | 0.92     | 1.08 - 135 | <b>0.96</b> | 1.08 - 135 | 0.95        |
| 15    | 1.517 - 135    | 0.9      | 0.592 - 129                  | 0.89     | <b>0.731 - 135</b>            | 0.92     | 1.32 - 135 | <b>0.95</b> | 1.32 - 135 | 0.94        |
| 16    | 1.278 - 135    | 0.91     | 0.579 - 129                  | 0.88     | <b>0.594 - 133</b>            | 0.92     | 0.92 - 135 | <b>0.96</b> | 0.92 - 135 | <b>0.96</b> |
| 17    | 0.964 - 135    | 0.92     | 0.594 - 132                  | 0.91     | <b>0.64 - 135</b>             | 0.93     | 1.03 - 135 | <b>0.96</b> | 1.03 - 135 | <b>0.96</b> |
| 18    | 0.94 - 135     | 0.92     | 0.61 - 132                   | 0.91     | <b>0.617 - 135</b>            | 0.93     | 0.97 - 135 | <b>0.96</b> | 0.97 - 135 | <b>0.96</b> |
| 19    | 1.147 - 135    | 0.91     | 0.486 - 127                  | 0.89     | <b>0.541 - 131</b>            | 0.91     | 1.05 - 135 | <b>0.95</b> | 1.05 - 135 | <b>0.95</b> |
| 20    | 0.723 - 135    | 0.92     | 0.636 - 135                  | 0.93     | <b>0.636 - 135</b>            | 0.93     | 1.05 - 135 | <b>0.95</b> | 1.05 - 135 | <b>0.95</b> |
| 21    | 2.704 - 135    | 0.85     | 0.635 - 120                  | 0.82     | <b>0.723 - 135</b>            | 0.93     | 1.28 - 134 | <b>0.95</b> | 1.4 - 135  | 0.94        |
| 22    | 0.827 - 135    | 0.91     | 0.615 - 132                  | 0.91     | <b>0.706 - 135</b>            | 0.92     | 1.16 - 135 | <b>0.95</b> | 1.16 - 135 | <b>0.95</b> |
| 23    | 2.138 - 135    | 0.85     | 0.598 - 117                  | 0.8      | <b>0.718 - 133</b>            | 0.9      | 1.19 - 135 | <b>0.94</b> | 1.19 - 135 | <b>0.94</b> |
| 24    | 4.26 - 135     | 0.81     | 0.66 - 104                   | 0.72     | <b>0.79 - 122</b>             | 0.84     | 1.04 - 135 | <b>0.95</b> | 1.05 - 135 | <b>0.95</b> |
| 25    | 1.198 - 135    | 0.9      | 0.563 - 127                  | 0.88     | <b>0.598 - 131</b>            | 0.9      | 1.07 - 135 | <b>0.95</b> | 1.07 - 135 | <b>0.95</b> |

cRMSD values are reported as backbone cRMSD in  $A^\circ$  - number of aligned  $\alpha$ -carbon atoms

Table 5.3: Pairwise structural alignment of unequal length proteins

| Structure  | N1  | N2  | Seq. id | <i>EDAlign<sub>ssc</sub></i> |             | TM-align   |             | SuperPose      |                |
|--|-----|-----|---------|------------------------------|-------------|------------|-------------|----------------|----------------|
|  |     |     |         | cRMSD                        | TM-score*   | cRMSD      | TM-score    | cRMSD          | TM-score       |
| modestly dissimilar sequence, length and structure |     |     |         |                              |             |            |             |                |                |
| Hemoglobin<br>(4HHBA 4HHBB)                        | 141 | 146 | 43%     | 1.11 - 89                    | 0.88        | 1.41 - 139 | <b>0.9</b>  | 1.61 - 139     | 0.89           |
| Thioredoxin<br>(3TRX 2TRXA)                        | 105 | 108 | 29%     | 5.97 - 38                    | 0.71        | 1.4 - 101  | <b>0.86</b> | 4.72 - 99      | 0.62           |
| Lysozyme/<br>Lactalbumin<br>(1DPX 1A4V)            | 129 | 123 | 36%     | 0.62 - 31                    | <b>0.94</b> | 1.48 - 123 | 0.87        | 1.63 - 121     | 0.86           |
| Calmodulin/<br>TnC<br>(1CLL 5TNC)                  | 144 | 161 | 47%     | 4.64 - 79                    | <b>0.69</b> | 3.49 - 107 | 0.55        | 6.83 - 144     | 0.48           |
| different length but similar structure or sequence |     |     |         |                              |             |            |             |                |                |
| Ubiquitin/<br>Elongin<br>(1UBI 1VCBA)              | 76  | 98  | 26%     | 1.83 - 28                    | 0.81        | 1.57 - 75  | <b>0.86</b> | 3.22 - 72      | 0.54           |
| Thio/<br>Glutaredoxin<br>(3TRX 3GRXA)              | 105 | 82  | 7%      | 7.54 - 31                    | 0.64        | 2.27 - 74  | <b>0.67</b> | 4.64 - 76      | 0.33           |
| Hemoglobins<br>(1ASH 2LHB)                         | 147 | 149 | 17%     | 1.75 - 88                    | <b>0.83</b> | 2.15 - 135 | 0.77        | — <sup>a</sup> | — <sup>a</sup> |
| Thioredoxins<br>(1NHOA 1DE2A)                      | 85  | 87  | 22%     | 1.86 - 18                    | <b>0.81</b> | 3.69 - 72  | 0.47        | 7.77 - 65      | 0.19           |

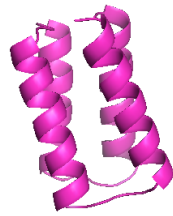
cRMSD values are reported as backbone cRMSD in  $A^\circ$  - number of aligned  $\alpha$ -carbon atoms

N1: Number of residues in 1st protein structure

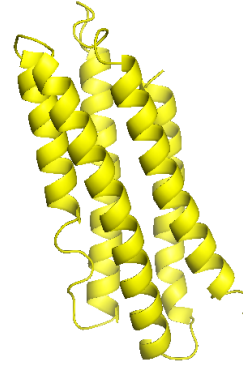
N2: Number of residues in 2nd protein structure

\*: Ignores the residues in the fragment that connects SSEs

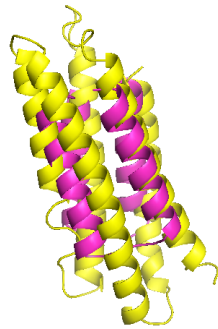
—<sup>a</sup>: Fail to align



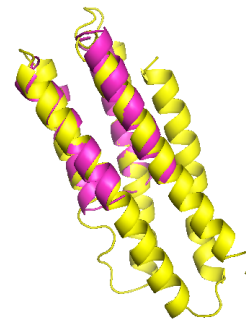
(a) 1flx with four helices  $H_i^a$ ,  $i = 1$  to 4



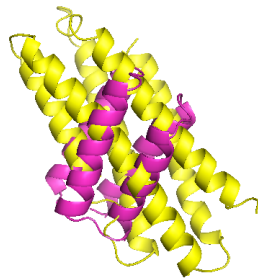
(b) 1aep with five helices  $H_i^b$ ,  $i = 1$  to 5



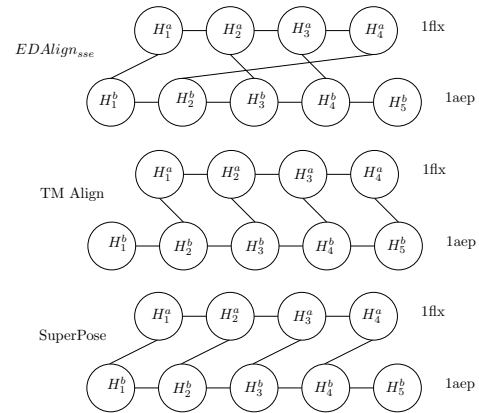
(c) *EDAlign<sub>sse</sub>* (cRMSD: 5.36 - 61,



(d) TM-align (cRMSD: 2.4 - 77, TM-score: 0.68)

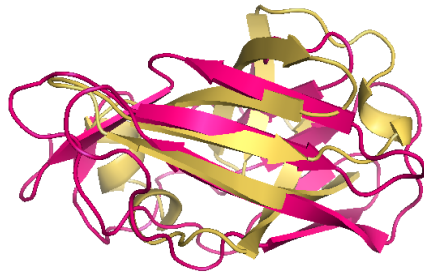


(e) SuperPose (cRMSD: 14.69 - 76, TM-score: 0.09 )

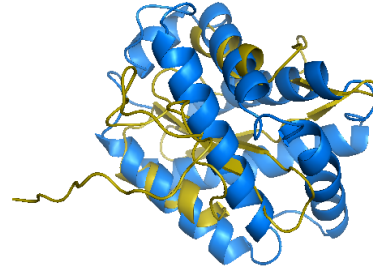


(f) SSE correspondence

Figure 5.1: Structural Alignment of 1flx with 1aep



(a) 1e50A with 1pkvA



(b) 2hbg with 2ecd

Figure 5.2:  $EDAlign_{sse}$  on difficult alignment (a)  $EDAlign_{sse}$ : cRMSD 1.94 - 20 and TM-score\* 0.81 TM-align: cRMSD 4.2 - 68 and TM-score 0.42 SuperPose: cRMSD 13.21 - 72 and TM-score 0.07 (b)  $EDAlign_{sse}$ : cRMSD 2.42 - 16 and TM-score\* 0.77 TM-align: cRMSD 4.23 - 63 and TM-score 0.31 SuperPose: cRMSD 13.77 - 111 and TM-score 0.2

$\beta$ -sheets where as the proteins 2hbg and 2ecd have two aligned  $\alpha$ -helices. On top of that, these pairs do not share any structural similarity. This has also been observed by TM align, as reflected in the TM-scores of 0.42 and 0.31 respectively.

## 5.6 Summary

In this chapter we have exploited matrix eigendecomposition to design two new algorithms,  $EDAlign_{res}$  and  $EDAlign_{sse}$ , for the structural alignment of two proteins. The former outputs an alignment of residue-pairs, while the latter reports aligned SSE-pairs.  $EDAlign_{res}$  can measure the structural similarity of two equal length



proteins only; the more general algorithm, *EDAlign<sub>sse</sub>*, combines eigendecomposition with dynamic programming and TM-score rotation matrix, and is able to handle proteins of unequal lengths. Experimental results show that *EDAlign<sub>sse</sub>* is able to align successfully common SSEs of any pair of proteins and also reveal potential conserved regions. Unlike other dynamic programming approaches, *EDAlign<sub>sse</sub>* is able to detect alignments that are independent of the order of the SSEs.

# Chapter 6

## Conclusions and Future works

We have proposed efficient algorithms for optimizing various distance measures for proximity problems defined on a point set. Many open problems remain; some are listed below.

1. Find an  $O(n^2)$  deterministic algorithm for the all-minimum problem in the line-difference measure.
2. Close the time-complexity gaps for the all-maximum problems in the area and perimeter measures.
3. Improve the upper bound on the time-complexity of the all-minimum problem in the perimeter measure and establish a corresponding lower bound.
4. Whittle away the  $\log n$  factor from the time-complexities of the all-minimum and all-maximum problems in the circumcircle measure.
5. Design an  $O(n^3)$ -time algorithm for the all-minimum problem in the triangle distance measure to improve on the trivial  $O(n^4)$  algorithm; for this last problem, an effective characterization needs to be found as the first step.

6. Design algorithms for the all- $k$ -closest problems in all of the above the measures we have discussed.
7. Along the the line of the study in [27], the problems of constructing the farthest-segment Voronoi diagram of a set of segments that are the edges of a convex polytope, or the farthest- triangle Voronoi diagram of the facets of a triangulated polytope merit further investigation.

We have also proposed efficient incremental geometric tools, in both two and three dimensions, to test the linear separability of two point sets (colored red and blue respectively). We have highlighted the effectiveness of the “incremental” algorithm over the “offline” algorithm for 5 different gene expression data sets. The dissertation also quantifies the number of separable gene-pairs and gene-triplets on 4 gene expression data-sets. The effective testing of linear separability of gene-triplets was left as an open problem by Unger and Chor [31]. To achieve the larger objective of finding bio-marker genes, we have propose a gene selection method that works as well as other known feature selection method like as  $t$ -values, FCS (Fisher Criterion Score) and SAM (Significance Analysis of Microarrays).

The dissertation also extends the idea of “linear separability” to “almost linear separability”. This an oft-recurring problem while collecting sample data due to noises or sampling errors or round off errors. We propose an  $O(nk^2)$  time algorithm for this problem. For  $k = O(1)$ , we have the first linear time algorithm known for this

problem. There are several promising directions for further work, some of which are listed below.

1. Extend the incremental algorithm from the linear separability paradigm to non-linear separability.
2. Design an efficient algorithm for testing polygonal separability and almost polygonal separability

We have shown how geometric optimization can play a role in designing algorithms for testing protein structure similarity. We have proposed two heuristics for pairwise protein structure alignment that uses an eigendecomposition technique due to Umeyama [141]. Of the proposed algorithms, (a) *EDAlign<sub>res</sub>* identifies structural similarity for equal length proteins; (b) *EDAlign<sub>sse</sub>*, on the other hand, does not require the input proteins to be of equal length. We have used the TM-score and cRMSD as measures of structural similarity. The algorithms are able to report sequence and topology independent alignments, with similarity scores that are comparable to those of the state-of-the-art algorithms such as TM align [42] and SuperPose [43]. Some promising directions for further work in the domain of structural similarities are:

1. Develop protein classification algorithm that take into account the overall structure of the proteins.
2. Design an efficient algorithm for finding motifs along a protein chain.

3. Find pharmacophore (i.e. a common sub-structure) in ligands.
4. Develop an algorithm for finding a ligand with a given pharmacophore.

# References

- [1] imgur: the simple image sharer. <http://i.imgur.com/N33uY3a.jpg>.
- [2] bitesizebio: Introduction to DNA Microarrays. <http://bitesizebio.com/7206/introduction-to-dna-microarrays/>.
- [3] W. S. Noble, “What is a support vector machine?,” *Nature biotechnology*, vol. 24, no. 12, pp. 1565–1567, 2006.
- [4] D. H. Shin, J. Y. Lee, K. Y. Hwang, K. Kyu Kim, and S. W. Suh, “High-resolution crystal structure of the non-specific lipid-transfer protein from maize seedlings,” *Structure*, vol. 3, no. 2, pp. 189–199, 1995.
- [5] J. Y. Lee, K. Min, H. Cha, D. H. Shin, K. Y. Hwang, and S. W. Suh, “Rice non-specific lipid transfer protein: the 1.6 Å crystal structure in the unliganded state reveals a small hydrophobic cavity,” *Journal of molecular biology*, vol. 276, no. 2, pp. 437–448, 1998.
- [6] scioly: Science Olympiad Student Center. [http://scioly.org/wiki/index.php/Protein\\_Modeling](http://scioly.org/wiki/index.php/Protein_Modeling).
- [7] scioly: Science Olympiad Student Center. [http://scioly.org/wiki/index.php/File:AASK\\_Sidechain\\_List.pdf](http://scioly.org/wiki/index.php/File:AASK_Sidechain_List.pdf).
- [8] M.I.Shamos, *Computational Geometry*. PhD thesis, Yale University, 1978.

- [9] P. K. Agarwal and M. Sharir, “Efficient algorithms for geometric optimization,” *ACM Comput. Surv.*, vol. 30, pp. 412–458, Dec. 1998.
- [10] H. Everett, J.-M. Robert, and M. V. Kreveld, “An optimal algorithm for computing (k)-levels, with applications,” *International Journal of Computational Geometry and Applications*, vol. 06, no. 03, pp. 247–261, 1996.
- [11] Matouek, J., “On geometric optimization with few violated constraints,” *Discrete and Computational Geometry*, vol. 14, no. 1, pp. 365–384, 1995.
- [12] T. M. Chan, “Low-dimensional linear programming with violations,” *SIAM J. Comput.*, vol. 34, pp. 879–893, Apr. 2005.
- [13] P. K. Agarwal and J. Erickson, “Geometric range searching and its relatives,” in *Advances in Discrete and Computational Geometry* (B. Chazelle, J. E. Goodman, and R. Pollack, eds.), vol. 23, p. 156, American Mathematical Society, 1998.
- [14] G. Toussaint, “Computational geometric problems in pattern recognition,” in *Pattern Recognition Theory and Applications* (J. Kittler, K. Fu, and L.-F. Pau, eds.), vol. 81 of *NATO Advanced Study Institutes Series*, pp. 73–91, Springer Netherlands, 1982.
- [15] J. Vince, *Geometry for Computer Graphics*. Springer, 2005.

- [16] D. Marsh and D. L. Marshall, *Applied Geometry for Computer Graphics*. London, UK, UK: Springer-Verlag, 1st ed., 1999.
- [17] T. Pavlidis, *Algorithms for Graphics and Image Processing*. Springer, 1982.
- [18] J. L. Bentley, M. I. Shamos, J. L. Bentley, and M. I. Shamos, “A problem in multivariate statistics: Algorithm, data structure, and applications,” in *In Proceedings of the Fifteenth Allerton Conference on Communication, Control, and Computing*, 1977.
- [19] M. I. Shamos, *Geometry a ~ Statistics: Problems at the Interface*. Academic Press, 1977.
- [20] J. T. Schwartz, M. Sharir, and J. E. Hopcroft, eds., *Planning, Geometry, and Complexity of Robot Motion*. Norwood, NJ, USA: Ablex Publishing Corp., 1986.
- [21] M. I. Shamos and D. Hoey, “Geometric intersection problems,” in *Foundations of Computer Science, 1976., 17th Annual Symposium on*, pp. 208–215, Oct 1976.
- [22] J. A. Bennell, K. A. Dowsland, and W. B. Dowsland, “The irregular cutting-stock problem a new procedure for deriving the no-fit polygon,” *Computers and Operations Research*, vol. 28, no. 3, pp. 271 – 287, 2001.



- [23] J.-M. Robert and G. Toussaint, “Computational geometry and facility location,” in *Proc. International Conference on Operations Research and Management Science*, pp. 11–15, 1990.
- [24] G. Barequet, M. T. Dickerson, and R. L. S. Drysdale, “2-Point site Voronoi diagrams,” *Discrete Appl. Math.*, vol. 122, no. 1-3, pp. 37–54, 2002.
- [25] K. Duffy, C. McAloney, H. Meijer, and D. Rappaport, “Closest segments,” in *Proc. of CCCG 2005*, pp. 229–231, 2005.
- [26] A. Mukhopadhyay, S. Chatterjee, and B. Lafreniere, “On the all-farthest-segments problem for a planar set of points,” *Information Processing Letters*, vol. 100, pp. 120–123, 2006.
- [27] F. Aurenhammer, R. Drysdale, and H. Krasser, “Farthest line segment Voronoi diagrams,” *Information Processing Letters*, vol. 100, no. 6, pp. 220–225, 2006.
- [28] O. Daescu, J. Luo, and D. M. Mount, “Proximity problems on line segments spanned by points,” *Computational Geometry: Theory and Applications*, vol. 33, pp. 115–129, 2006.
- [29] S. Bespamyatnikh and M. Segal, “Selecting distances in arrangement of hyperplanes spanned by points,” *Journal of Discrete Algorithms*, vol. 2, pp. 333–345, 2004.

- [30] M. Journée, A. Teschendorff, P.-A. Absil, S. Tavaré, and R. Sepulchre, “Geometric optimization methods for the analysis of gene expression data,” in *Principal Manifolds for Data Visualization and Dimension Reduction* (D. C. W. Alexander N. Gorban, Balázs Kégl and A. Y. Zinovyev, eds.), vol. 58 of *Lecture Notes in Computational Science and Engineering*, ch. 12, pp. 271–292, Springer Berlin Heidelberg, 2007.
- [31] G. Unger and B. Chor, “Linear separability of gene expression data sets,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 7, pp. 375–381, 2010.
- [32] M. S. Alam, S. Panigrahi, P. Bhabak, and A. Mukhopadhyay, “A multi-gene linear separability of gene expression data in linear time,” in *Short Abstracts in ISBRA 2010: 6th International Symposium on Bioinformatics Research and Applications, May 23-26*, (Connecticut, USA), pp. 51–54, 2010.
- [33] A. Ben-Israel and Y. Levin, “The geometry of linear separability in data sets,” *Linear Algebra and its Applications*, vol. 416, no. 1, pp. 75 – 87, 2006. Special Issue devoted to the Haifa 2005 conference on matrix theory.
- [34] M. R. Anderberg, *Cluster Analysis for Applications*. New York: Academic Press, 1973.

- [35] A. Riva, A.-S. Carpentier, B. Torrsani, and A. Hnaut, “Comments on selected fundamental aspects of microarray analysis,” *Computational Biology and Chemistry*, vol. 29, no. 5, pp. 319 – 336, 2005.
- [36] J.-C. Kader, “Lipid-transfer proteins in plants,” *Annual Review of Plant Physiology and Plant Molecular Biology*, vol. 47, no. 1, pp. 627–654, 1996. PMID: 15012303.
- [37] H. Berman, K. Henrick, H. Nakamura, and J. L. Markley, “The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data.,” *Nucleic acids research*, vol. 35, pp. D301–D303, Jan. 2007.
- [38] C. Branden and J. Tooze, *Introduction to protein structure*. Garland, 2 ed., 1998.
- [39] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, “The protein data bank,” *Nucleic Acids Res*, vol. 28, pp. 235–242, 2000.
- [40] Schrödinger, LLC, “The PyMOL molecular graphics system, version 1.3r1.” August 2010.
- [41] Y. Zhang and J. Skolnick, “Scoring function for automated assessment of protein structure template quality,” *Proteins: Structure, Function, and Bioinformatics*, vol. 57, no. 4, pp. 702–710, 2004.

- [42] Y. Zhang and J. Skolnick, “TM-align: A protein structure alignment algorithm based on TM-score.,” *Nucleic Acids Research*, vol. 33, pp. 2302–2309, 2005.
- [43] R. Maiti, G. H. V. Domselaar, H. Zhang, and D. S. Wishart, “SuperPose: a simple server for sophisticated structural superposition.,” *Nucleic Acids Research*, vol. 32, no. Web-Server-Issue, pp. 590–594, 2004.
- [44] A. Mukhopadhyay and S. C. Panigrahi, “All-maximum and all-minimum problems under some measures,” *Journal of Discrete Algorithms*, vol. 21, no. 0, pp. 18 – 31, 2013.
- [45] O. Daescu and J. Luo, “Proximity problems on line segments spanned by points,” in *Proc. of 14th Annual Fall Workshop on Computational Geometry*, pp. 9–10, 2004.
- [46] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. Springer-Verlag, 1985.
- [47] S. Khuller and Y. Matias, “A simple randomized sieve algorithm for the closest-pair problem,” *Information and Computation*, vol. 118, pp. 34–37, 1995.
- [48] M. Overmars and E. Welzl, “New methods for computing visibility graphs,” in *Proceedings of the fourth annual symposium on Computational geometry*, pp. 164–171, 1988.

- [49] H. Edelsbrunner and L. Guibas, “Topologically sweeping an arrangement,” *Journal of Computer and System Sciences*, vol. 38, no. 1, pp. 165–194, 1989.
- [50] A. Gajentaan and M. Overmars, “On a class of  $o(n^2)$  hard problems in computational geometry,” *Computational Geometry: Theory and Applications*, vol. 5, no. 3, pp. 165–185, 1995.
- [51] G. S. Brodal and R. Jacob, “Dynamic planar convex hull,” in *FOCS '02: Proceedings of the 43rd Symposium on Foundations of Computer Science*, (Washington, DC, USA), pp. 617–626, IEEE Computer Society, 2002.
- [52] H. Edelsbrunner, J. O’Rourke, and R. Seidel, “Constructing arrangements of lines and hyperplanes with applications,” *SIAM J. Comput.*, vol. 15, pp. 341–363, 1986.
- [53] R.L.Drysdale and J.W.Jaromczyk, “A note on lower bounds for the maximum area and maximum perimeter k-gon problems,” *Information Processing Letters*, vol. 32, pp. 301–303, 1989.
- [54] B. Chazelle, L. J. Guibas, and D. T. Lee, “The power of geometric duality,” *BIT*, vol. 25, pp. 76–90, 1985.
- [55] H. Edelsbrunner, M. Sharir, and R. Seidel, “On the zone theorem for hyperplane arrangements,” *SIAM J. Comput.*, vol. 22, no. 2, pp. 418–429, 1993.

- [56] A. Aggarwal and J. Park, “Notes on searching in multidimensional monotone arrays,” in *SFCS '88: Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, (Washington, DC, USA), pp. 497–512, IEEE Computer Society, 1988.
- [57] J. E. Boyce, D. P. Dobkin, R. L. S. Drysdale, and L. J. Guibas, “Finding extremal polygons,” *SIAM J. Computing*, vol. 14, no. 1, pp. 134–147, 1985.
- [58] H. Coxeter, *Introduction to Gometry, Second Edition*. John Wiley, 1969.
- [59] R. Drysdale and A. Mukhopadhyay, “An  $o(n \log n)$  algorithm for the all-farthest-segments problem for a planar set of points,” *Information Processing Letters*, vol. 105, pp. 47–51, 2008.
- [60] J. Matousek and O. Schwarzkopf, “On ray shooting in convex polytopes,” *Discrete Comput. Geom.*, vol. 10, pp. 215–232, 1993.
- [61] S. Panigrahi, M. Alam, and A. Mukhopadhyay, “An incremental linear programming based tool for analyzing gene expression data,” in *Computational Science and Its Applications ICCSA 2013* (B. Murgante, S. Misra, M. Carlini, C. Torre, H.-Q. Nguyen, D. Taniar, B. Apduhan, and O. Gervasi, eds.), vol. 7975 of *Lecture Notes in Computer Science*, pp. 48–64, Springer Berlin Heidelberg, 2013.

- [62] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring,” *Science*, vol. 286, pp. 531–537, Oct. 1999.
- [63] L. J. van ’t Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. M. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend, “Gene expression profiling predicts clinical outcome of breast cancer,” *Nature*, vol. 415, pp. 530–536, Jan. 2002.
- [64] J. Khan, J. S. Wei, M. Ringnér, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, and P. S. Meltzer, “Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks.,” *Nature medicine*, vol. 7, pp. 673–679, June 2001.
- [65] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini, “Tissue classification with gene expression profiles,” *Journal of Computational Biology*, vol. 7, no. 3-4, pp. 559–583, 2000.
- [66] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, “Broad patterns of gene expression revealed by clustering analysis of tu-

- mor and normal colon tissues probed by oligonucleotide arrays.,” in *Proceedings of the National Academy of Science USA*, vol. 96, pp. 6745–6750, 1999.
- [67] S. Kim, E. R. Dougherty, J. Barrera, Y. Chen, M. L. Bittner, and J. M. Trent, “Strong feature sets from small samples.,” *Journal of Computational Biology*, vol. 9, no. 1, pp. 127–146, 2002.
- [68] N. Megiddo, “Linear-time algorithms for linear programming in  $R^3$  and related problems,” *SIAM Journal of Computation*, vol. 12, no. 4, pp. 759–776, 1983.
- [69] N. Megiddo, “Linear programming in linear time when the dimension is fixed,” *J. ACM*, vol. 31, pp. 114–127, 1984.
- [70] M. E. Dyer, “Linear time algorithms for two- and three-variable linear programs,” *SIAM J. Comput.*, vol. 13, no. 1, pp. 31–45, 1984.
- [71] D. Anastassiou, “Computational analysis of the synergy among multiple interacting genes.,” *Molecular systems biology*, vol. 3, Feb. 2007.
- [72] J. O’Rourke, *Computational Geometry in C*. New York, NY, USA: Cambridge University Press, 2nd ed., 1998.
- [73] D. V. Nguyen and D. M. Rocke, “Tumor classification by partial least squares using microarray gene expression data,” *Bioinformatics*, vol. 18, no. 1, pp. 39–50, 2002.



- [74] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York, NY, USA: Oxford University Press, 1995.
- [75] D. Zhang, S. Chen, and Z.-H. Zhou, "Constraint score: A new filter method for feature selection with pairwise constraints," *Pattern Recogn.*, vol. 41, pp. 1440–1451, May 2008.
- [76] V. G. Tusher, R. Tibshirani, and G. Chu, "Significance analysis of microarrays applied to the ionizing radiation response," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, pp. 5116–5121, Apr. 2001.
- [77] C. Gil, L. Jun, N. Balasubramanian, T. Robert, and T. Virginia, "*Significance Analysis of Microarrays*" *Users guide and technical document*. Stanford CA 94305: Stanford University.
- [78] Y. Lu and J. Han, "Cancer classification using gene expression data," *Inf. Syst.*, vol. 28, no. 4, pp. 243–268, 2003.
- [79] G. J. Gordon, R. V. Jensen, L.-L. Hsiao, S. R. Gullans, J. E. Blumenstock, S. Ramaswamy, W. G. Richards, D. J. S. , and R. Bueno, "Translation of mi-croarray data into clinically relevant cancer diagnostic tests using gene expres- sion ratios in lung cancer and mesothelioma," *Cancer Research*, vol. 62, pp. 4963–4967, Sept. 2002.

- [80] S. A. Armstrong, J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Minden, S. E. Sallan, E. S. Lander, T. R. Golub, and S. J. Korsmeyer, “Mll translocations specify a distinct gene expression profile that distinguishes a unique leukemia,” *Nature Genetics*, vol. 30, pp. 41 – 47, Jan 2002.
- [81] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, “The *WEKA* data mining software: An update,” in *SIGKDD Explorations*, vol. 11, pp. 11 –18, 2009.
- [82] J. C. Platt, *Fast training of support vector machines using sequential minimal optimization*, pp. 185–208. Cambridge, MA, USA: MIT Press, 1999.
- [83] R. R. Bouckaert, “Bayesian Network Classifiers in *WEKA*,” *Internal Notes*, vol. 11, no. 3, pp. 1–23, 2004.
- [84] G. F. Cooper and E. Herskovits, “A Bayesian Method for the Induction of Probabilistic Networks from Data,” *Machine Learning*, vol. 9, pp. 309–347, 1992.
- [85] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, (San Francisco, CA, USA), pp. 1137–1143, Morgan Kaufmann Publishers Inc., 1995.

- [86] K. L. Clarkson, “Las vegas algorithms for linear and integer programming when the dimension is small,” *J. ACM*, vol. 42, pp. 488–499, Mar. 1995.
- [87] R. Seidel, “Small-dimensional linear programming and convex hulls made easy,” *Discrete and Computational Geometry*, vol. 6, no. 1, pp. 423–434, 1991.
- [88] M. Sharir and E. Welzl, “A combinatorial bound for linear programming and related problems,” in *STACS 92* (A. Finkel and M. Jantzen, eds.), vol. 577 of *Lecture Notes in Computer Science*, pp. 567–579, Springer Berlin Heidelberg, 1992.
- [89] T. M. Chan, “Fixed-dimensional linear programming queries made easy,” in *Proceedings of the Twelfth Annual Symposium on Computational Geometry, SCG '96*, (New York, NY, USA), pp. 284–290, ACM, 1996.
- [90] T. M. Chan, “Geometric applications of a randomized optimization technique,” *Discrete and Computational Geometry*, vol. 22, no. 4, pp. 547–567, 1999.
- [91] A. Efrat, M. Lindenbaum, and M. Sharir, “Finding maximally consistent sets of halfspaces.,” in *CCCG*, pp. 432–436, University of Waterloo, 1993.
- [92] T. Roos and P. Widmayer, “k-violation linear programming,” in *System Modelling and Optimization* (J. Henry and J.-P. Yvon, eds.), vol. 197 of *Lecture Notes in Control and Information Sciences*, pp. 855–862, Springer Berlin Heidelberg, 1994.

- [93] J. E. Goodman and R. Pollack, “On the number of  $k$ -subsets of a set of  $n$  points in the plane,” *Journal of Combinatorial Theory, Series A*, vol. 36, no. 1, pp. 101–104, 1984.
- [94] N. Alon and E. Gyri, “The number of small semispaces of a finite set of points in the plane,” *Journal of Combinatorial Theory, Series A*, vol. 41, no. 1, pp. 154–157, 1986.
- [95] P. K. Agarwal, B. Aronov, T. M. Chan, and M. Sharir, “On levels in arrangements of lines, segments, planes, and triangles,” *Discrete and Computational Geometry*, vol. 19, no. 3, pp. 315–331, 1998.
- [96] T. K. Dey, “Improved bounds on planar  $k$ -sets and  $k$ -levels,” *Discrete and Computational Geometry*, vol. 19, pp. 156–161, 1997.
- [97] M. Sharir, S. Smorodinsky, and G. Tardos, “An improved bound for  $k$ -sets in three dimensions,” in *Proceedings of the Sixteenth Annual Symposium on Computational Geometry*, SCG ’00, (New York, NY, USA), pp. 43–49, ACM, 2000.
- [98] J. O’Rourke, S. Rao Kosaraju, and N. Megiddo, “Computing circular separability,” *Discrete and Computational Geometry*, vol. 1, no. 1, pp. 105–113, 1986.
- [99] V. N. Vapnik, *The nature of statistical learning theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995.

- [100] E. M. Arkin, F. Hurtado, J. S. B. Mitchell, C. Seara, and S. S. Skiena, “Some lower bounds on geometric separability problems,” *International Journal of Computational Geometry and Applications*, vol. 16, no. 01, pp. 1–26, 2006.
- [101] F. Hurtado, M. Mora, P. A. Ramos, and C. Seara, “Two problems on separability with lines and polygonals,” in *Proc. 15th European Workshop on Computational Geometry*, pp. 33–35, Citeseer, 1999.
- [102] F. Hurtado, M. Noy, P. A. Ramos, and C. Seara, “Separating objects in the plane by wedges and strips,” *Discrete Applied Mathematics*, vol. 109, no. 12, pp. 109 – 138, 2001. 14th European Workshop on Computational Geometry.
- [103] H. Edelsbrunner and F. Preparata, “Minimum polygonal separation,” *Information and Computation*, vol. 77, no. 3, pp. 218 – 232, 1988.
- [104] S. Fekete, “On the complexity of min-link red-blue separation,” *Manuscript, Department of Applied Mathematics, SUNY Stony Brook, NY*, 1992.
- [105] J. S. Mitchell, “Approximation algorithms for geometric separation problems,” in *Technical Report, Department of Applied Mathematics, SUNY Stony Brook, NY*, Citeseer, 1993.
- [106] H. Edelsbrunner and L. J. Guibas, “Topologically sweeping an arrangement,” *Journal of Computer and System Sciences*, vol. 38, no. 1, pp. 165 – 194, 1989.

- [107] W. Taylor, A. May, N. Brown, and A. Aszódi, “Protein structure: geometry, topology and classification,” *Reports on Progress in Physics*, vol. 64, no. 4, pp. 517–590, 2001.
- [108] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia, “SCOP: a structural classification of proteins database for the investigation of sequences and structures.,” *Journal of molecular biology*, vol. 247, pp. 536–540, Apr. 1995.
- [109] C. Orengo, A. Michie, S. Jones, D. Jones, M. Swindells, and J. Thornton, “CATH - a hierarchic classification of protein domain structures.,” *Structure*, vol. 5, pp. 1093–1108, Aug. 1997.
- [110] L. Holm and C. Sander, “Dali/FSSP classification of three-dimensional protein folds,” *Nucleic Acids Research*, vol. 25, no. 1, pp. 231–234, 1997.
- [111] W. Kabsch, “A solution for the best rotation to relate two sets of vectors,” *Acta Crystallographica Section A*, vol. 32, pp. 922–923, Sep 1976.
- [112] A. D. McLachlan, “Rapid comparison of protein structures,” *Acta Crystallographica Section A*, vol. 38, pp. 871–873, Nov. 1982.
- [113] L. Holm and C. Sander, “Protein Structure Comparison by Alignment of Distance Matrices,” *Journal of Molecular Biology*, vol. 233, pp. 123–138, Sept. 1993.

- [114] P. J. Artymiuk, A. R. Poirrette, H. M. Grindley, D. W. Rice, and P. Willett, “A graph-theoretic approach to the identification of three-dimensional patterns of amino acid side-chains in protein structures,” *Journal of Molecular Biology*, vol. 243, no. 2, pp. 327 – 344, 1994.
- [115] P. J. Artymiuk, R. V. Spriggs, and P. Willett, “Graph theoretic methods for the analysis of structural relationships in biological macromolecules: Research articles,” *J. Am. Soc. Inf. Sci. Technol.*, vol. 56, pp. 518–528, Mar. 2005.
- [116] N. Siew, A. Elofsson, L. Rychlewski, and D. Fischer, “MaxSub: an automated measure for the assessment of protein structure prediction quality,” *Bioinformatics (Oxford, England)*, vol. 16, pp. 776–785, Sept. 2000.
- [117] R. Nussinov and H. J. Wolfson, “Efficient detection of three-dimensional structural motifs in biological macromolecules by computer vision techniques,” *Proceedings of the National Academy of Sciences*, vol. 88, no. 23, pp. 10495–10499, 1991.
- [118] T. D. Wu, T. Hastie, S. C. Schmidler, and D. L. Brutlag, “Regression analysis of multiple protein structures,” in *RECOMB*, pp. 276–284, 1998.
- [119] Y. Shibberu and A. Holder, “A spectral approach to protein structure alignment,” *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 8, pp. 867–875, 2011.

- [120] R. Andonov, N. Malod-Dognin, and N. Yanev, “Maximum contact map overlap revisited.,” *J Comput Biol*, vol. 18, pp. 27–41, Jan. 2011.
- [121] R. Andonov, N. Yanev, and N. Malod-dognin, “An efficient lagrangian relaxation for the contact map overlap problem,” in *In WABI 08: Proceedings of the 8th international workshop on Algorithms in Bioinformatics*, pp. 162–173, Springer-Verlag, 2008.
- [122] P. Di Lena, P. Fariselli, L. Margara, M. Vassura, and R. Casadio, “Fast overlapping of protein contact maps by alignment of eigenvectors,” *Bioinformatics*, vol. 26, no. 18, pp. 2250–2258, 2010.
- [123] W. Xie and N. V. Sahinidis, “A reduction-based exact algorithm for the contact map overlap problem,” *Journal of Computational Biology*, vol. 14, no. 5, pp. 637–654, 2007.
- [124] M. Vendruscolo, E. Kussell, and E. Domany, “Recovery of protein structure from contact maps,” *Folding and Design*, vol. 2, no. 5, pp. 295 – 306, 1997.
- [125] W. R. Taylor, “Protein structure comparison using iterated double dynamic programming.,” *Protein Sci*, vol. 8, pp. 654–665, Mar. 1999.
- [126] M. Gerstein and M. Levitt, “Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures,” in *In Proc. Fourth Int. Conf. on Intell. Sys. Mol. Biol*, pp. 59–67, AAAI Press, 1996.



- [127] M. Gerstein and M. Levitt, “Comprehensive assessment of automatic structural alignment against a manual standard, the scop classification of proteins,” *Protein Science*, vol. 7, no. 2, pp. 445–456, 1998.
- [128] I. N. Shindyalov and P. E. Bourne, “Protein structure alignment by incremental combinatorial extension (CE) of the optimal path,” *Protein Engineering*, vol. 11, pp. 739–747, Sept. 1998.
- [129] J. D. Szustakowski and Z. Weng, “Protein structure alignment using a genetic algorithm,” *Proteins: Structure, Function, and Bioinformatics*, vol. 38, no. 4, pp. 428–440, 2000.
- [130] W. R. Taylor and C. A. Orengo, “Protein structure alignment,” *Journal of Molecular Biology*, vol. 208, no. 1, pp. 1 – 22, 1989.
- [131] N. N. Alexandrov, K. Takahashi, and N. G., “Common spatial arrangements of backbone fragments in homologous and non-homologous proteins,” *Journal of Molecular Biology*, vol. 225, no. 1, pp. 5 – 9, 1992.
- [132] C. A. Orengo and W. R. Taylor, “SSAP: Sequential structure alignment program for protein structure comparison.,” in *Computer Methods for Macromolecular Sequence Analysis* (R. F. Doolittle, ed.), vol. 266 of *Methods in Enzymology*, pp. 617 – 635, Academic Press, 1996.

- [133] L. Chen, L.-Y. Wu, Y. Wang, S. Zhang, and X.-S. Zhang, “Revealing divergent evolution, identifying circular permutations and detecting active-sites by protein structure comparison,” 2006.
- [134] A. Zemla, “LGA: a method for finding 3D similarities in protein structures.,” *Nucleic Acids Research*, vol. 31, no. 13, pp. 3370–3374, 2003.
- [135] A. Godzik, A. Kolinski, and J. Skolnick, “Topology fingerprint approach to the inverse protein folding problem,” *Journal of Molecular Biology*, vol. 227, no. 1, pp. 227 – 238, 1992.
- [136] N. Krasnogor and D. A. Pelta, “Measuring the similarity of protein structures by means of the universal similarity metric,” *Bioinformatics*, vol. 20, no. 7, pp. 1015–1021, 2004.
- [137] S. Rahmati and J. I. Glasgow, “Comparing protein contact maps via universal similarity metric: an improvement in the noise-tolerance.,” *I. J. Computational Biology and Drug Design*, vol. 2, no. 2, pp. 149–167, 2009.
- [138] H. Hasegawa and L. Holm, “Advances and pitfalls of protein structural alignment,” *Current Opinion in Structural Biology*, vol. 19, no. 3, pp. 341 – 348, 2009.
- [139] P. Koehl, “Protein structure similarities,” *Current Opinion in Structural Biology*, vol. 11, no. 3, pp. 348 – 353, 2001.

- [140] W. R. Taylor, “Protein structure comparison using bipartite graph matching and its application to protein structure classification,” *Mol. Cell Proteomics*, pp. 334–339, 2002.
- [141] S. Umeyama, “An eigendecomposition approach to weighted graph matching problems,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 10, pp. 695–703, Sept. 1988.
- [142] R. Kolodny and N. Linial, “Approximate protein structural alignment in polynomial time,” *Proc Natl Acad Sci U S A*, vol. 101, pp. 12201–12206, Aug. 2004.
- [143] S. Wang, J. Ma, J. Peng, and J. Xu, “Protein structure alignment beyond spatial proximity,” *Scientific reports*, vol. 3, Mar. 2013.
- [144] S. C. Li, “The difficulty of protein structure alignment under the RMSD.,” *Algorithms for Molecular Biology*, vol. 8, p. 1, 2013.
- [145] M. Levitt and M. Gerstein, “A unified statistical framework for sequence comparison and structure comparison,” *Proceedings of the National Academy of Sciences*, vol. 95, pp. 5913–5920, May 1998.
- [146] J. Xu and Y. Zhang, “How significant is a protein structure similarity with TM-score = 0.5?,” *Bioinformatics*, vol. 26, no. 7, pp. 889–895, 2010.

- [147] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.
- [148] W. Kabsch and C. Sander, “Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features,” *Biopolymers*, vol. 22, pp. 2577–2637, Dec. 1983.
- [149] R. P. Joosten, T. A. H. te Beek, E. Krieger, M. L. Hekkelman, R. W. W. Hooft, R. S. 0002, C. Sander, and G. Vriend, “A series of pdb related databases for everyday needs.,” *Nucleic Acids Research*, vol. 39, no. Database-Issue, pp. 411–419, 2011.
- [150] K. Mizuguchi and N. G, “Comparison of spatial arrangements of secondary structural elements in proteins,” *Protein Engineering*, vol. 8, no. 4, pp. 353–362, 1995.
- [151] A. Abyzov and V. Ilyin, “A comprehensive analysis of non-sequential alignments between all protein structures,” *BMC Structural Biology*, vol. 7, no. 1, pp. 1–20, 2007.

# Appendix A

## Offline implementation

### A.1 Code for linearly separating gene pairs

**Algorithm:** LinearlySeparatingGenePairs

**Input:** Line duals  $S_R^*$  and  $S_B^*$  of the point sets  $S_R$  and  $S_B$ .

**Output:** LP feasible or infeasible.

- 1: if  $|S_R^*| \leq 1$  and  $|S_B^*| \leq 1$  then use the trivial method to solve the problem.
- 2: Pair up the lines in each set, ignoring a odd residual line in each, under the assumption that if two lines are parallel then one of them can be eliminated immediately as that will not determine the feasible region.
- 3: Find a test line,  $\bar{U}$ , through the point of intersection with median u-coordinate.
- 4: Check for the feasibility solution of one dimensional linear program along test line  $\bar{U}$ . We distinguish with following cases :
  - Case 1:* Line  $\bar{U}$  pass through feasible region. Report the solution and halt.
  - Case 2:* Line  $\bar{U}$  detects inseparability. Report the solution and halt.
  - Case 3:* If line does not pass through the feasible region then find the side of feasible region with respect to test line  $\bar{U}$ .
    - Case 3.1:* If the solution lies to the left of  $\bar{U}$  restrict the feasible range to the left of  $\bar{U}$ .
    - Case 3.2:* If the solution lies to the right of  $\bar{U}$  restrict the feasible range to the right of  $\bar{U}$ .
- 5: Update  $S_R^*$  and  $S_B^*$  by eliminating a line from each pair whose intersection does not lie in the feasible region.
- 6: Repeat the algorithm for updated set of  $S_R^*$  and  $S_B^*$ .

## A.2 Code for linearly separating gene triplets

**Algorithm:** LinearlySeparatingGeneTriplets

**Input:** Plane duals  $S_R^*$  and  $S_B^*$  of the point sets  $S_R$  and  $S_B$ .

**Output:** LP feasible or infeasible.

- 1: if  $|S_R^*| + |S_B^*| \leq 4$  then use a brute-force method to solve the problem.
- 2: Pair up the planes in each set, ignoring a odd residual plane in each, under the assumption that if two planes are parallel then one of them can be eliminated immediately as that will not determine the feasible region. Take the projection of paired constraints on uv-plane.
- 3: Transform the coordinate system such that half the lines will have negative slope and half of the lines will have positive slope.
- 4: Pair the lines such that in every pair we will have one line from positive slope and other line from negative slope. Identify, the point of intersection of non disjoint pairs; say  $(u_{ij}, v_{ij})$ . If two lines are parallel (must be parallel to u-axis) then identify  $v_{ij}$  to be the mean of their v-coordinates.
- 5: Let  $v_m$  be the median of all  $v_{ij}$ . Solve a 2D linear programming problem along the test line  $v = v_m$ .
  - Case 1:* If feasible then report the separability and halt the program.
  - Case 2:* If not feasible obtain a point on the lower envelop where we realize a minimum difference between upper and lower envelop. Identify the set of constraints tight to this point and solve two 2D linear programs to determine which side of test line  $v = v_m$  the feasible region lies.
    - Case 2.1:* If any one of 2D linear program is feasible then identify the side of feasible solution and continue with step 6.
    - Case 2.2:* If both 2D linear program are not feasible or both are feasible then report the inseparability and halt.
- 6: Identify a test line  $u = u_m$ , where  $u_m$  be the median of all  $u_{ij}$  on the feasible side of test line  $v = v_m$ . Solve a 2D linear programming problem along the test line  $u = u_m$ .
  - Case 1:* If feasible then report the separability and halt the program.
  - Case 2:* If not feasible obtain a point on the lower envelop where we realize a minimum difference between upper and lower envelop. Identify the set of constraints tight to this point and solve two 2D linear programs to determine which side of test line  $u = u_m$  the feasible region lies.
    - Case 2.1:* If any one of 2D linear program is feasible then identify the side of feasible solution and continue with step 7.
    - Case 2.2:* If both 2D linear program are not feasible or both are feasible then report the inseparability and halt.

- 7: Update  $S_R^*$  and  $S_B^*$ ; by eliminating a constraint that does not pass through feasible quadrant defined  $v = v_m$  and  $u = u_m$ .
- 8: Repeat the algorithm for the updated set of  $S_R^*$  and  $S_B^*$ .

## A.3 Data Sets

We tested our algorithms on the following 5 publicly available gene expression data sets.

**Colon Data:** The Colon Data, published by Alon *et al.* [66], consists of gene expression values of 2000 genes. The data set was generated using Affymetrix oligonucleotide arrays. The sample set consists of 40 colon tumor samples and 22 normal colon tissue samples for a total of 62 samples. The dataset is publicly available at <http://microarray.princeton.edu/oncology/affydata/index.html>

**Leukemia Data:** This data set was published by Scott *et al.* [80]. The data set contains three kind of leukemia samples compared to the binary class leukemia dataset. The data set consists of 72 leukemia samples with 24 ALL (Acute lymphoblastic leukemia), 20 MLL (Mixed-lineage leukemia) and 28 AML (Acute Myeloid leukemia). We considered only ALL and AML samples for our study. The gene expression intensities were obtained from Affymetrix high density oligonucleotide micro arrays and there are 12582 genes in the data set. The dataset is publicly available at <http://research.dfci.harvard.edu/korsmeyer/MLL.htm>

**Breast Cancer:** This data set was published by Van't Veer *et al.* [63] and consists of gene expressions of 24481 genes from cDNA experiments. The data set contains 34 samples from patients who developed distant metastases within five years of treatment and 44 samples from the patients who continued to be disease-free for a period of at least five years. The raw gene expression data set generated from cDNA microarray usually contains missing values. We chose to remove one sample (number 54 in original data) that contained many missing values. We also removed all genes that had missing values for any of the samples. The final data set we used contained expression levels of 21682 genes. The dataset is publicly available at <http://www.rii.com/publications/2002/vantveer.htm>

**Lung Cancer:** This data set was published by Gordon *et al.* [79]. It consists of 31 malignant pleural mesothelioma (MPM) and 150 adenocarcinoma (ADCA) tumors, a total of 181 samples. Each sample is described by 12533 genes generated from Affymetrix high density oligonucleotide micro arrays. The dataset is publicly available at <http://www.chestsurg.org/publications/2002-microarray.aspx>

**SRBCT:** This data set was published by Khan *et al.* [64] consists of five classes of small round blue-cell tumors (SRBCT). We chose two 23 samples from Ewing family of tumors (EWS) and 20 rhabdomyosarcoma (RMS), a total of 43 samples.



We excluded the sample group 'TEST' in which there are additional EWS and RMS samples. The gene expression intensities were obtained from Affymetrix high density oligonucleotide micro arrays for 2308 genes. The dataset is publicly available at <http://research.nhgri.nih.gov/microarray>

# Appendix B

## A sample PDB file

Find the sample PDB file (PDB id: 1GCN) downloaded from <http://www.rcsb.org>

```
HEADER      HORMONE                               17-OCT-77   1GCN
TITLE       X-RAY ANALYSIS OF GLUCAGON AND ITS RELATIONSHIP TO RECEPTOR
TITLE       2 BINDING
COMPND      MOL_ID: 1;
COMPND      2 MOLECULE: GLUCAGON;
COMPND      3 CHAIN: A;
COMPND      4 ENGINEERED: YES
SOURCE      MOL_ID: 1;
SOURCE      2 ORGANISM_SCIENTIFIC: SUS SCROFA;
SOURCE      3 ORGANISM_COMMON: PIG;
SOURCE      4 ORGANISM_TAXID: 9823
KEYWDS      HORMONE
EXPDTA      X-RAY DIFFRACTION
AUTHOR      T.L.BLUNDELL,K.SASAKI,S.DOCKERILL,I.J.TICKLE
REVDAT      6  24-FEB-09 1GCN   1      VERSN
REVDAT      5  30-SEP-83 1GCN   1      REVDAT
REVDAT      4  31-DEC-80 1GCN   1      REMARK
REVDAT      3  22-OCT-79 1GCN   3      ATOM
REVDAT      2  29-AUG-79 1GCN   3      CRYST1
REVDAT      1  28-NOV-77 1GCN   0
JRNL        AUTH   K.SASAKI,S.DOCKERILL,D.A.ADAMIAK,I.J.TICKLE,
JRNL        AUTH 2 T.BLUNDELL
JRNL        TITL   X-RAY ANALYSIS OF GLUCAGON AND ITS RELATIONSHIP TO
JRNL        TITL 2 RECEPTOR BINDING.
JRNL        REF    NATURE                               V. 257   751 1975
JRNL        REFN                               ISSN 0028-0836
JRNL        PMID   171582
JRNL        DOI    10.1038/257751A0
REMARK      1
REMARK      1 REFERENCE 1
```

REMARK 1 EDIT M.O.DAYHOFF  
REMARK 1 REF ATLAS OF PROTEIN SEQUENCE V. 5 125 1976  
REMARK 1 REF 2 AND STRUCTURE,SUPPLEMENT 2  
REMARK 1 PUBL NATIONAL BIOMEDICAL RESEARCH FOUNDATION, SILVER  
REMARK 1 PUBL 2 SPRING,MD.  
REMARK 1 REFN ISSN 0-912466-05-7  
REMARK 2  
REMARK 2 RESOLUTION. 3.00 ANGSTROMS.  
REMARK 3  
REMARK 3 REFINEMENT.  
REMARK 3 PROGRAM : NULL  
REMARK 3 AUTHORS : NULL  
REMARK 3  
REMARK 3 DATA USED IN REFINEMENT.  
REMARK 3 RESOLUTION RANGE HIGH (ANGSTROMS) : 3.00  
REMARK 3 RESOLUTION RANGE LOW (ANGSTROMS) : NULL  
REMARK 3 DATA CUTOFF (SIGMA(F)) : NULL  
REMARK 3 DATA CUTOFF HIGH (ABS(F)) : NULL  
REMARK 3 DATA CUTOFF LOW (ABS(F)) : NULL  
REMARK 3 COMPLETENESS (WORKING+TEST) (%) : NULL  
REMARK 3 NUMBER OF REFLECTIONS : NULL  
REMARK 3  
REMARK 3 FIT TO DATA USED IN REFINEMENT.  
REMARK 3 CROSS-VALIDATION METHOD : NULL  
REMARK 3 FREE R VALUE TEST SET SELECTION : NULL  
REMARK 3 R VALUE (WORKING SET) : NULL  
REMARK 3 FREE R VALUE : NULL  
REMARK 3 FREE R VALUE TEST SET SIZE (%) : NULL  
REMARK 3 FREE R VALUE TEST SET COUNT : NULL  
REMARK 3 ESTIMATED ERROR OF FREE R VALUE : NULL  
REMARK 3  
REMARK 3 FIT IN THE HIGHEST RESOLUTION BIN.  
REMARK 3 TOTAL NUMBER OF BINS USED : NULL  
REMARK 3 BIN RESOLUTION RANGE HIGH (A) : NULL  
REMARK 3 BIN RESOLUTION RANGE LOW (A) : NULL  
REMARK 3 BIN COMPLETENESS (WORKING+TEST) (%) : NULL  
REMARK 3 REFLECTIONS IN BIN (WORKING SET) : NULL  
REMARK 3 BIN R VALUE (WORKING SET) : NULL  
REMARK 3 BIN FREE R VALUE : NULL  
REMARK 3 BIN FREE R VALUE TEST SET SIZE (%) : NULL  
REMARK 3 BIN FREE R VALUE TEST SET COUNT : NULL  
REMARK 3 ESTIMATED ERROR OF BIN FREE R VALUE : NULL  
REMARK 3  
REMARK 3 NUMBER OF NON-HYDROGEN ATOMS USED IN REFINEMENT.

REMARK 3 PROTEIN ATOMS : 246  
REMARK 3 NUCLEIC ACID ATOMS : 0  
REMARK 3 HETEROGEN ATOMS : 0  
REMARK 3 SOLVENT ATOMS : 0  
REMARK 3  
REMARK 3 B VALUES.  
REMARK 3 FROM WILSON PLOT (A\*\*2) : NULL  
REMARK 3 MEAN B VALUE (OVERALL, A\*\*2) : NULL  
REMARK 3 OVERALL ANISOTROPIC B VALUE.  
REMARK 3 B11 (A\*\*2) : NULL  
REMARK 3 B22 (A\*\*2) : NULL  
REMARK 3 B33 (A\*\*2) : NULL  
REMARK 3 B12 (A\*\*2) : NULL  
REMARK 3 B13 (A\*\*2) : NULL  
REMARK 3 B23 (A\*\*2) : NULL  
REMARK 3  
REMARK 3 ESTIMATED COORDINATE ERROR.  
REMARK 3 ESD FROM LUZZATI PLOT (A) : NULL  
REMARK 3 ESD FROM SIGMAA (A) : NULL  
REMARK 3 LOW RESOLUTION CUTOFF (A) : NULL  
REMARK 3  
REMARK 3 CROSS-VALIDATED ESTIMATED COORDINATE ERROR.  
REMARK 3 ESD FROM C-V LUZZATI PLOT (A) : NULL  
REMARK 3 ESD FROM C-V SIGMAA (A) : NULL  
REMARK 3  
REMARK 3 RMS DEVIATIONS FROM IDEAL VALUES.  
REMARK 3 BOND LENGTHS (A) : NULL  
REMARK 3 BOND ANGLES (DEGREES) : NULL  
REMARK 3 DIHEDRAL ANGLES (DEGREES) : NULL  
REMARK 3 IMPROPER ANGLES (DEGREES) : NULL  
REMARK 3  
REMARK 3 ISOTROPIC THERMAL MODEL : NULL  
REMARK 3  
REMARK 3 ISOTROPIC THERMAL FACTOR RESTRAINTS. RMS SIGMA  
REMARK 3 MAIN-CHAIN BOND (A\*\*2) : NULL ; NULL  
REMARK 3 MAIN-CHAIN ANGLE (A\*\*2) : NULL ; NULL  
REMARK 3 SIDE-CHAIN BOND (A\*\*2) : NULL ; NULL  
REMARK 3 SIDE-CHAIN ANGLE (A\*\*2) : NULL ; NULL  
REMARK 3  
REMARK 3 NCS MODEL : NULL  
REMARK 3  
REMARK 3 NCS RESTRAINTS. RMS SIGMA/WEIGHT  
REMARK 3 GROUP 1 POSITIONAL (A) : NULL ; NULL  
REMARK 3 GROUP 1 B-FACTOR (A\*\*2) : NULL ; NULL

REMARK 3  
 REMARK 3 PARAMETER FILE 1 : NULL  
 REMARK 3 TOPOLOGY FILE 1 : NULL  
 REMARK 3  
 REMARK 3 OTHER REFINEMENT REMARKS: NULL  
 REMARK 4  
 REMARK 4 1GCN COMPLIES WITH FORMAT V. 3.15, 01-DEC-08  
 REMARK 100  
 REMARK 100 THIS ENTRY HAS BEEN PROCESSED BY BNL.  
 REMARK 200  
 REMARK 200 EXPERIMENTAL DETAILS  
 REMARK 200 EXPERIMENT TYPE : X-RAY DIFFRACTION  
 REMARK 200 DATE OF DATA COLLECTION : NULL  
 REMARK 200 TEMPERATURE (KELVIN) : NULL  
 REMARK 200 PH : NULL  
 REMARK 200 NUMBER OF CRYSTALS USED : NULL  
 REMARK 200  
 REMARK 200 SYNCHROTRON (Y/N) : NULL  
 REMARK 200 RADIATION SOURCE : NULL  
 REMARK 200 BEAMLINE : NULL  
 REMARK 200 X-RAY GENERATOR MODEL : NULL  
 REMARK 200 MONOCHROMATIC OR LAUE (M/L) : NULL  
 REMARK 200 WAVELENGTH OR RANGE (A) : NULL  
 REMARK 200 MONOCHROMATOR : NULL  
 REMARK 200 OPTICS : NULL  
 REMARK 200  
 REMARK 200 DETECTOR TYPE : NULL  
 REMARK 200 DETECTOR MANUFACTURER : NULL  
 REMARK 200 INTENSITY-INTEGRATION SOFTWARE : NULL  
 REMARK 200 DATA SCALING SOFTWARE : NULL  
 REMARK 200  
 REMARK 200 NUMBER OF UNIQUE REFLECTIONS : NULL  
 REMARK 200 RESOLUTION RANGE HIGH (A) : NULL  
 REMARK 200 RESOLUTION RANGE LOW (A) : NULL  
 REMARK 200 REJECTION CRITERIA (SIGMA(I)) : NULL  
 REMARK 200  
 REMARK 200 OVERALL.  
 REMARK 200 COMPLETENESS FOR RANGE (%) : NULL  
 REMARK 200 DATA REDUNDANCY : NULL  
 REMARK 200 R MERGE (I) : NULL  
 REMARK 200 R SYM (I) : NULL  
 REMARK 200 <I/SIGMA(I)> FOR THE DATA SET : NULL  
 REMARK 200  
 REMARK 200 IN THE HIGHEST RESOLUTION SHELL.

REMARK 200 HIGHEST RESOLUTION SHELL, RANGE HIGH (A) : NULL  
 REMARK 200 HIGHEST RESOLUTION SHELL, RANGE LOW (A) : NULL  
 REMARK 200 COMPLETENESS FOR SHELL (%) : NULL  
 REMARK 200 DATA REDUNDANCY IN SHELL : NULL  
 REMARK 200 R MERGE FOR SHELL (I) : NULL  
 REMARK 200 R SYM FOR SHELL (I) : NULL  
 REMARK 200 <I/SIGMA(I)> FOR SHELL : NULL  
 REMARK 200  
 REMARK 200 DIFFRACTION PROTOCOL: NULL  
 REMARK 200 METHOD USED TO DETERMINE THE STRUCTURE: NULL  
 REMARK 200 SOFTWARE USED: NULL  
 REMARK 200 STARTING MODEL: NULL  
 REMARK 200  
 REMARK 200 REMARK: NULL  
 REMARK 280  
 REMARK 280 CRYSTAL  
 REMARK 280 SOLVENT CONTENT, VS (%): 50.74  
 REMARK 280 MATTHEWS COEFFICIENT, VM (ANGSTROMS\*\*3/DA): 2.50  
 REMARK 280  
 REMARK 280 CRYSTALLIZATION CONDITIONS: NULL  
 REMARK 290  
 REMARK 290 CRYSTALLOGRAPHIC SYMMETRY  
 REMARK 290 SYMMETRY OPERATORS FOR SPACE GROUP: P 21 3  
 REMARK 290  

| SYMOP | SYMMETRY OPERATOR |
|-------|-------------------|
| 1555  | X,Y,Z             |
| 2555  | -X+1/2,-Y,Z+1/2   |
| 3555  | -X,Y+1/2,-Z+1/2   |
| 4555  | X+1/2,-Y+1/2,-Z   |
| 5555  | Z,X,Y             |
| 6555  | Z+1/2,-X+1/2,-Y   |
| 7555  | -Z+1/2,-X,Y+1/2   |
| 8555  | -Z,X+1/2,-Y+1/2   |
| 9555  | Y,Z,X             |
| 10555 | -Y,Z+1/2,-X+1/2   |
| 11555 | Y+1/2,-Z+1/2,-X   |
| 12555 | -Y+1/2,-Z,X+1/2   |

  
 REMARK 290  
 REMARK 290 WHERE NNN -> OPERATOR NUMBER  
 REMARK 290 MMM -> TRANSLATION VECTOR  
 REMARK 290  
 REMARK 290 CRYSTALLOGRAPHIC SYMMETRY TRANSFORMATIONS  
 REMARK 290 THE FOLLOWING TRANSFORMATIONS OPERATE ON THE ATOM/HETATM

REMARK 290 RECORDS IN THIS ENTRY TO PRODUCE CRYSTALLOGRAPHICALLY  
REMARK 290 RELATED MOLECULES.

|            |        |    |           |           |           |          |
|------------|--------|----|-----------|-----------|-----------|----------|
| REMARK 290 | SMTRY1 | 1  | 1.000000  | 0.000000  | 0.000000  | 0.000000 |
| REMARK 290 | SMTRY2 | 1  | 0.000000  | 1.000000  | 0.000000  | 0.000000 |
| REMARK 290 | SMTRY3 | 1  | 0.000000  | 0.000000  | 1.000000  | 0.000000 |
| REMARK 290 | SMTRY1 | 2  | -1.000000 | 0.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY2 | 2  | 0.000000  | -1.000000 | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY3 | 2  | 0.000000  | 0.000000  | 1.000000  | 23.55000 |
| REMARK 290 | SMTRY1 | 3  | -1.000000 | 0.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY2 | 3  | 0.000000  | 1.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY3 | 3  | 0.000000  | 0.000000  | -1.000000 | 23.55000 |
| REMARK 290 | SMTRY1 | 4  | 1.000000  | 0.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY2 | 4  | 0.000000  | -1.000000 | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY3 | 4  | 0.000000  | 0.000000  | -1.000000 | 0.00000  |
| REMARK 290 | SMTRY1 | 5  | 0.000000  | 0.000000  | 1.000000  | 0.00000  |
| REMARK 290 | SMTRY2 | 5  | 1.000000  | 0.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY3 | 5  | 0.000000  | 1.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY1 | 6  | 0.000000  | 0.000000  | 1.000000  | 23.55000 |
| REMARK 290 | SMTRY2 | 6  | -1.000000 | 0.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY3 | 6  | 0.000000  | -1.000000 | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY1 | 7  | 0.000000  | 0.000000  | -1.000000 | 23.55000 |
| REMARK 290 | SMTRY2 | 7  | -1.000000 | 0.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY3 | 7  | 0.000000  | 1.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY1 | 8  | 0.000000  | 0.000000  | -1.000000 | 0.00000  |
| REMARK 290 | SMTRY2 | 8  | 1.000000  | 0.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY3 | 8  | 0.000000  | -1.000000 | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY1 | 9  | 0.000000  | 1.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY2 | 9  | 0.000000  | 0.000000  | 1.000000  | 0.00000  |
| REMARK 290 | SMTRY3 | 9  | 1.000000  | 0.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY1 | 10 | 0.000000  | -1.000000 | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY2 | 10 | 0.000000  | 0.000000  | 1.000000  | 23.55000 |
| REMARK 290 | SMTRY3 | 10 | -1.000000 | 0.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY1 | 11 | 0.000000  | 1.000000  | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY2 | 11 | 0.000000  | 0.000000  | -1.000000 | 23.55000 |
| REMARK 290 | SMTRY3 | 11 | -1.000000 | 0.000000  | 0.000000  | 0.00000  |
| REMARK 290 | SMTRY1 | 12 | 0.000000  | -1.000000 | 0.000000  | 23.55000 |
| REMARK 290 | SMTRY2 | 12 | 0.000000  | 0.000000  | -1.000000 | 0.00000  |
| REMARK 290 | SMTRY3 | 12 | 1.000000  | 0.000000  | 0.000000  | 23.55000 |

REMARK 290  
REMARK 290 REMARK: NULL  
REMARK 300  
REMARK 300 BIOMOLECULE: 1  
REMARK 300 SEE REMARK 350 FOR THE AUTHOR PROVIDED AND/OR PROGRAM  
REMARK 300 GENERATED ASSEMBLY INFORMATION FOR THE STRUCTURE IN

REMARK 300 THIS ENTRY. THE REMARK MAY ALSO PROVIDE INFORMATION ON  
REMARK 300 BURIED SURFACE AREA.  
REMARK 350  
REMARK 350 COORDINATES FOR A COMPLETE MULTIMER REPRESENTING THE KNOWN  
REMARK 350 BIOLOGICALLY SIGNIFICANT OLIGOMERIZATION STATE OF THE  
REMARK 350 MOLECULE CAN BE GENERATED BY APPLYING BIOMT TRANSFORMATIONS  
REMARK 350 GIVEN BELOW. BOTH NON-CRYSTALLOGRAPHIC AND  
REMARK 350 CRYSTALLOGRAPHIC OPERATIONS ARE GIVEN.  
REMARK 350  
REMARK 350 BIOMOLECULE: 1  
REMARK 350 AUTHOR DETERMINED BIOLOGICAL UNIT: MONOMERIC  
REMARK 350 APPLY THE FOLLOWING TO CHAINS: A  
REMARK 350 BIOMT1 1 1.000000 0.000000 0.000000 0.000000  
REMARK 350 BIOMT2 1 0.000000 1.000000 0.000000 0.000000  
REMARK 350 BIOMT3 1 0.000000 0.000000 1.000000 0.000000  
REMARK 500  
REMARK 500 GEOMETRY AND STEREOCHEMISTRY  
REMARK 500 SUBTOPIC: COVALENT BOND LENGTHS  
REMARK 500  
REMARK 500 THE STEREOCHEMICAL PARAMETERS OF THE FOLLOWING RESIDUES  
REMARK 500 HAVE VALUES WHICH DEVIATE FROM EXPECTED VALUES BY MORE  
REMARK 500 THAN 6\*RMSD (M=MODEL NUMBER; RES=RESIDUE NAME; C=CHAIN  
REMARK 500 IDENTIFIER; SSEQ=SEQUENCE NUMBER; I=INSERTION CODE).  
REMARK 500  
REMARK 500 STANDARD TABLE:  
REMARK 500 FORMAT: (10X,I3,1X,2(A3,1X,A1,I4,A1,1X,A4,3X),1X,F6.3)  
REMARK 500  
REMARK 500 EXPECTED VALUES PROTEIN: ENGH AND HUBER, 1999  
REMARK 500 EXPECTED VALUES NUCLEIC ACID: CLOWNEY ET AL 1996  
REMARK 500  
REMARK 500 M RES CSSEQI ATM1 RES CSSEQI ATM2 DEVIATION  
REMARK 500 TYR A 10 CZ TYR A 10 OH -0.387  
REMARK 500 TRP A 25 CD1 TRP A 25 NE1 0.287  
REMARK 500 TRP A 25 NE1 TRP A 25 CE2 0.109  
REMARK 500  
REMARK 500 REMARK: NULL  
REMARK 500  
REMARK 500 GEOMETRY AND STEREOCHEMISTRY  
REMARK 500 SUBTOPIC: COVALENT BOND ANGLES  
REMARK 500  
REMARK 500 THE STEREOCHEMICAL PARAMETERS OF THE FOLLOWING RESIDUES  
REMARK 500 HAVE VALUES WHICH DEVIATE FROM EXPECTED VALUES BY MORE  
REMARK 500 THAN 6\*RMSD (M=MODEL NUMBER; RES=RESIDUE NAME; C=CHAIN  
REMARK 500 IDENTIFIER; SSEQ=SEQUENCE NUMBER; I=INSERTION CODE).



REMARK 500  
REMARK 500 STANDARD TABLE:  
REMARK 500 FORMAT: (10X,I3,1X,A3,1X,A1,I4,A1,3(1X,A4,2X),12X,F5.1)  
REMARK 500  
REMARK 500 EXPECTED VALUES PROTEIN: ENGH AND HUBER, 1999  
REMARK 500 EXPECTED VALUES NUCLEIC ACID: CLOWNEY ET AL 1996  
REMARK 500  
REMARK 500 M RES CSSEQI ATM1 ATM2 ATM3  
REMARK 500 TRP A 25 CG - CD1 - NE1 ANGL. DEV. = 6.7 DEGREES  
REMARK 500 TRP A 25 CD1 - NE1 - CE2 ANGL. DEV. = -21.5 DEGREES  
REMARK 500 TRP A 25 NE1 - CE2 - CZ2 ANGL. DEV. = -11.0 DEGREES  
REMARK 500 TRP A 25 NE1 - CE2 - CD2 ANGL. DEV. = 9.6 DEGREES  
REMARK 500  
REMARK 500 REMARK: NULL  
REMARK 500  
REMARK 500 GEOMETRY AND STEREOCHEMISTRY  
REMARK 500 SUBTOPIC: TORSION ANGLES  
REMARK 500  
REMARK 500 TORSION ANGLES OUTSIDE THE EXPECTED RAMACHANDRAN REGIONS:  
REMARK 500 (M=MODEL NUMBER; RES=RESIDUE NAME; C=CHAIN IDENTIFIER;  
REMARK 500 SSEQ=SEQUENCE NUMBER; I=INSERTION CODE).  
REMARK 500  
REMARK 500 STANDARD TABLE:  
REMARK 500 FORMAT: (10X,I3,1X,A3,1X,A1,I4,A1,4X,F7.2,3X,F7.2)  
REMARK 500  
REMARK 500 EXPECTED VALUES: GJ KLEYWEGT AND TA JONES (1996). PHI/PSI-  
REMARK 500 CHOLGY: RAMACHANDRAN REVISITED. STRUCTURE 4, 1395 - 1400  
REMARK 500  
REMARK 500 M RES CSSEQI PSI PHI  
REMARK 500 SER A 2 -57.57 -21.14  
REMARK 500 THR A 5 54.62 -63.85  
REMARK 500 SER A 11 9.62 -51.97  
REMARK 500 MET A 27 -93.98 -145.30  
REMARK 500 ASN A 28 64.02 15.67  
REMARK 500  
REMARK 500 REMARK: NULL  
REMARK 500  
REMARK 500 GEOMETRY AND STEREOCHEMISTRY  
REMARK 500 SUBTOPIC: PLANAR GROUPS  
REMARK 500  
REMARK 500 PLANAR GROUPS IN THE FOLLOWING RESIDUES HAVE A TOTAL  
REMARK 500 RMS DISTANCE OF ALL ATOMS FROM THE BEST-FIT PLANE  
REMARK 500 BY MORE THAN AN EXPECTED VALUE OF 6\*RMSD, WITH AN  
REMARK 500 RMSD 0.02 ANGSTROMS, OR AT LEAST ONE ATOM HAS

REMARK 500 AN RMSD GREATER THAN THIS VALUE  
REMARK 500 (M=MODEL NUMBER; RES=RESIDUE NAME; C=CHAIN IDENTIFIER;  
REMARK 500 SSEQ=SEQUENCE NUMBER; I=INSERTION CODE).  
REMARK 500  
REMARK 500 M RES CSSEQI RMS TYPE  
REMARK 500 ASN A 28 0.08 SIDE\_CHAIN  
REMARK 500  
REMARK 500 REMARK: NULL  
REMARK 500  
REMARK 500 GEOMETRY AND STEREOCHEMISTRY  
REMARK 500 SUBTOPIC: MAIN CHAIN PLANARITY  
REMARK 500  
REMARK 500 THE FOLLOWING RESIDUES HAVE A PSEUDO PLANARITY  
REMARK 500 TORSION, C(I) - CA(I) - N(I+1) - O(I), GREATER  
REMARK 500 10.0 DEGREES. (M=MODEL NUMBER; RES=RESIDUE NAME;  
REMARK 500 C=CHAIN IDENTIFIER; SSEQ=SEQUENCE NUMBER;  
REMARK 500 I=INSERTION CODE).  
REMARK 500  
REMARK 500 M RES CSSEQI ANGLE  
REMARK 500 HIS A 1 19.48  
REMARK 500 GLN A 3 -15.78  
REMARK 500 GLY A 4 -17.23  
REMARK 500 THR A 5 -10.38  
REMARK 500 PHE A 6 -12.06  
REMARK 500 THR A 7 -14.66  
REMARK 500 SER A 11 -15.10  
REMARK 500 LYS A 12 14.46  
REMARK 500 ALA A 19 -10.92  
REMARK 500 GLN A 20 -13.40  
REMARK 500 VAL A 23 -15.87  
REMARK 500 LEU A 26 -14.56  
REMARK 500 MET A 27 -16.22  
REMARK 500  
REMARK 500 REMARK: NULL  
DBREF 1GCN A 1 29 UNP P01274 GLUC\_PIG 33 61  
SEQRES 1 A 29 HIS SER GLN GLY THR PHE THR SER ASP TYR SER LYS TYR  
SEQRES 2 A 29 LEU ASP SER ARG ARG ALA GLN ASP PHE VAL GLN TRP LEU  
SEQRES 3 A 29 MET ASN THR  
HELIX 1 A PHE A 6 LEU A 26 1  
CRYST1 47.100 47.100 47.100 90.00 90.00 90.00 P 21 3 12  
ORIGX1 0.021231 0.000000 0.000000 0.000000  
ORIGX2 0.000000 0.021231 0.000000 0.000000  
ORIGX3 0.000000 0.000000 0.021231 0.000000  
SCALE1 0.021231 0.000000 0.000000 0.000000

21

| SCALE2 | 0.000000 | 0.021231 | 0.000000 | 0.000000 |   |        |        |        |      |       |   |
|--------|----------|----------|----------|----------|---|--------|--------|--------|------|-------|---|
| SCALE3 | 0.000000 | 0.000000 | 0.021231 | 0.000000 |   |        |        |        |      |       |   |
| ATOM   | 1        | N        | HIS      | A        | 1 | 49.668 | 24.248 | 10.436 | 1.00 | 25.00 | N |
| ATOM   | 2        | CA       | HIS      | A        | 1 | 50.197 | 25.578 | 10.784 | 1.00 | 16.00 | C |
| ATOM   | 3        | C        | HIS      | A        | 1 | 49.169 | 26.701 | 10.917 | 1.00 | 16.00 | C |
| ATOM   | 4        | O        | HIS      | A        | 1 | 48.241 | 26.524 | 11.749 | 1.00 | 16.00 | O |
| ATOM   | 5        | CB       | HIS      | A        | 1 | 51.312 | 26.048 | 9.843  | 1.00 | 16.00 | C |
| ATOM   | 6        | CG       | HIS      | A        | 1 | 50.958 | 26.068 | 8.340  | 1.00 | 16.00 | C |
| ATOM   | 7        | ND1      | HIS      | A        | 1 | 49.636 | 26.144 | 7.860  | 1.00 | 16.00 | N |
| ATOM   | 8        | CD2      | HIS      | A        | 1 | 51.797 | 26.043 | 7.286  | 1.00 | 16.00 | C |
| ATOM   | 9        | CE1      | HIS      | A        | 1 | 49.691 | 26.152 | 6.454  | 1.00 | 17.00 | C |
| ATOM   | 10       | NE2      | HIS      | A        | 1 | 51.046 | 26.090 | 6.098  | 1.00 | 17.00 | N |
| ATOM   | 11       | N        | SER      | A        | 2 | 49.788 | 27.850 | 10.784 | 1.00 | 16.00 | N |
| ATOM   | 12       | CA       | SER      | A        | 2 | 49.138 | 29.147 | 10.620 | 1.00 | 15.00 | C |
| ATOM   | 13       | C        | SER      | A        | 2 | 47.713 | 29.006 | 10.110 | 1.00 | 15.00 | C |
| ATOM   | 14       | O        | SER      | A        | 2 | 46.740 | 29.251 | 10.864 | 1.00 | 15.00 | O |
| ATOM   | 15       | CB       | SER      | A        | 2 | 49.875 | 29.930 | 9.569  | 1.00 | 16.00 | C |
| ATOM   | 16       | OG       | SER      | A        | 2 | 49.145 | 31.057 | 9.176  | 1.00 | 19.00 | O |
| ATOM   | 17       | N        | GLN      | A        | 3 | 47.620 | 28.367 | 8.973  | 1.00 | 15.00 | N |
| ATOM   | 18       | CA       | GLN      | A        | 3 | 46.287 | 28.193 | 8.308  | 1.00 | 14.00 | C |
| ATOM   | 19       | C        | GLN      | A        | 3 | 45.406 | 27.172 | 8.963  | 1.00 | 14.00 | C |
| ATOM   | 20       | O        | GLN      | A        | 3 | 44.198 | 27.508 | 9.014  | 1.00 | 14.00 | O |
| ATOM   | 21       | CB       | GLN      | A        | 3 | 46.489 | 27.963 | 6.806  | 1.00 | 18.00 | C |
| ATOM   | 22       | CG       | GLN      | A        | 3 | 45.138 | 27.800 | 6.111  | 1.00 | 21.00 | C |
| ATOM   | 23       | CD       | GLN      | A        | 3 | 45.304 | 27.952 | 4.603  | 1.00 | 24.00 | C |
| ATOM   | 24       | OE1      | GLN      | A        | 3 | 46.432 | 28.202 | 4.112  | 1.00 | 24.00 | O |
| ATOM   | 25       | NE2      | GLN      | A        | 3 | 44.233 | 27.647 | 3.897  | 1.00 | 26.00 | N |
| ATOM   | 26       | N        | GLY      | A        | 4 | 46.014 | 26.394 | 9.871  | 1.00 | 14.00 | N |
| ATOM   | 27       | CA       | GLY      | A        | 4 | 45.422 | 25.287 | 10.680 | 1.00 | 14.00 | C |
| ATOM   | 28       | C        | GLY      | A        | 4 | 43.892 | 25.215 | 10.719 | 1.00 | 14.00 | C |
| ATOM   | 29       | O        | GLY      | A        | 4 | 43.287 | 26.155 | 11.288 | 1.00 | 14.00 | O |
| ATOM   | 30       | N        | THR      | A        | 5 | 43.406 | 23.993 | 10.767 | 1.00 | 14.00 | N |
| ATOM   | 31       | CA       | THR      | A        | 5 | 42.004 | 23.642 | 10.443 | 1.00 | 12.00 | C |
| ATOM   | 32       | C        | THR      | A        | 5 | 40.788 | 24.146 | 11.252 | 1.00 | 12.00 | C |
| ATOM   | 33       | O        | THR      | A        | 5 | 39.804 | 23.384 | 11.410 | 1.00 | 12.00 | O |
| ATOM   | 34       | CB       | THR      | A        | 5 | 41.934 | 22.202 | 9.889  | 1.00 | 14.00 | C |
| ATOM   | 35       | OG1      | THR      | A        | 5 | 41.080 | 21.317 | 10.609 | 1.00 | 15.00 | O |
| ATOM   | 36       | CG2      | THR      | A        | 5 | 43.317 | 21.556 | 9.849  | 1.00 | 15.00 | C |
| ATOM   | 37       | N        | PHE      | A        | 6 | 40.628 | 25.463 | 11.441 | 1.00 | 12.00 | N |
| ATOM   | 38       | CA       | PHE      | A        | 6 | 39.381 | 25.950 | 12.104 | 1.00 | 12.00 | C |
| ATOM   | 39       | C        | PHE      | A        | 6 | 38.156 | 25.684 | 11.232 | 1.00 | 12.00 | C |
| ATOM   | 40       | O        | PHE      | A        | 6 | 37.231 | 25.002 | 11.719 | 1.00 | 12.00 | O |
| ATOM   | 41       | CB       | PHE      | A        | 6 | 39.407 | 27.425 | 12.584 | 1.00 | 12.00 | C |
| ATOM   | 42       | CG       | PHE      | A        | 6 | 38.187 | 27.923 | 13.430 | 1.00 | 12.00 | C |

|      |    |     |     |   |    |        |        |        |      |       |   |
|------|----|-----|-----|---|----|--------|--------|--------|------|-------|---|
| ATOM | 43 | CD1 | PHE | A | 6  | 36.889 | 27.518 | 13.163 | 1.00 | 12.00 | C |
| ATOM | 44 | CD2 | PHE | A | 6  | 38.386 | 28.862 | 14.419 | 1.00 | 12.00 | C |
| ATOM | 45 | CE1 | PHE | A | 6  | 35.813 | 27.967 | 13.909 | 1.00 | 12.00 | C |
| ATOM | 46 | CE2 | PHE | A | 6  | 37.306 | 29.328 | 15.177 | 1.00 | 12.00 | C |
| ATOM | 47 | CZ  | PHE | A | 6  | 36.019 | 28.871 | 14.928 | 1.00 | 12.00 | C |
| ATOM | 48 | N   | THR | A | 7  | 38.341 | 25.794 | 9.956  | 1.00 | 12.00 | N |
| ATOM | 49 | CA  | THR | A | 7  | 37.249 | 25.666 | 8.991  | 1.00 | 12.00 | C |
| ATOM | 50 | C   | THR | A | 7  | 36.324 | 24.452 | 9.101  | 1.00 | 12.00 | C |
| ATOM | 51 | O   | THR | A | 7  | 35.111 | 24.637 | 9.387  | 1.00 | 12.00 | O |
| ATOM | 52 | CB  | THR | A | 7  | 37.884 | 25.743 | 7.628  | 1.00 | 13.00 | C |
| ATOM | 53 | OG1 | THR | A | 7  | 37.940 | 27.122 | 7.317  | 1.00 | 14.00 | O |
| ATOM | 54 | CG2 | THR | A | 7  | 37.073 | 25.003 | 6.585  | 1.00 | 14.00 | C |
| ATOM | 55 | N   | SER | A | 8  | 36.964 | 23.356 | 9.442  | 1.00 | 12.00 | N |
| ATOM | 56 | CA  | SER | A | 8  | 36.286 | 22.063 | 9.486  | 1.00 | 12.00 | C |
| ATOM | 57 | C   | SER | A | 8  | 35.575 | 21.813 | 10.813 | 1.00 | 11.00 | C |
| ATOM | 58 | O   | SER | A | 8  | 35.203 | 20.650 | 11.111 | 1.00 | 10.00 | O |
| ATOM | 59 | CB  | SER | A | 8  | 37.291 | 20.958 | 9.189  | 1.00 | 16.00 | C |
| ATOM | 60 | OG  | SER | A | 8  | 37.917 | 21.247 | 7.943  | 1.00 | 20.00 | O |
| ATOM | 61 | N   | ASP | A | 9  | 35.723 | 22.783 | 11.694 | 1.00 | 10.00 | N |
| ATOM | 62 | CA  | ASP | A | 9  | 35.004 | 22.803 | 12.977 | 1.00 | 10.00 | C |
| ATOM | 63 | C   | ASP | A | 9  | 33.532 | 23.121 | 12.749 | 1.00 | 10.00 | C |
| ATOM | 64 | O   | ASP | A | 9  | 32.645 | 22.360 | 13.210 | 1.00 | 10.00 | O |
| ATOM | 65 | CB  | ASP | A | 9  | 35.556 | 23.874 | 13.919 | 1.00 | 11.00 | C |
| ATOM | 66 | CG  | ASP | A | 9  | 36.280 | 23.230 | 15.096 | 1.00 | 13.00 | C |
| ATOM | 67 | OD1 | ASP | A | 9  | 36.088 | 22.010 | 15.324 | 1.00 | 16.00 | O |
| ATOM | 68 | OD2 | ASP | A | 9  | 36.821 | 23.974 | 15.951 | 1.00 | 16.00 | O |
| ATOM | 69 | N   | TYR | A | 10 | 33.316 | 24.220 | 12.040 | 1.00 | 10.00 | N |
| ATOM | 70 | CA  | TYR | A | 10 | 31.967 | 24.742 | 11.748 | 1.00 | 10.00 | C |
| ATOM | 71 | C   | TYR | A | 10 | 31.203 | 23.973 | 10.685 | 1.00 | 10.00 | C |
| ATOM | 72 | O   | TYR | A | 10 | 29.980 | 23.772 | 10.885 | 1.00 | 10.00 | O |
| ATOM | 73 | CB  | TYR | A | 10 | 31.951 | 26.230 | 11.367 | 1.00 | 10.00 | C |
| ATOM | 74 | CG  | TYR | A | 10 | 30.613 | 26.678 | 10.713 | 1.00 | 10.00 | C |
| ATOM | 75 | CD1 | TYR | A | 10 | 30.563 | 26.886 | 9.350  | 1.00 | 10.00 | C |
| ATOM | 76 | CD2 | TYR | A | 10 | 29.463 | 26.824 | 11.461 | 1.00 | 10.00 | C |
| ATOM | 77 | CE1 | TYR | A | 10 | 29.377 | 27.275 | 8.733  | 1.00 | 10.00 | C |
| ATOM | 78 | CE2 | TYR | A | 10 | 28.272 | 27.214 | 10.848 | 1.00 | 10.00 | C |
| ATOM | 79 | CZ  | TYR | A | 10 | 28.226 | 27.452 | 9.483  | 1.00 | 10.00 | C |
| ATOM | 80 | OH  | TYR | A | 10 | 27.365 | 27.683 | 9.060  | 1.00 | 11.00 | O |
| ATOM | 81 | N   | SER | A | 11 | 31.796 | 23.909 | 9.491  | 1.00 | 10.00 | N |
| ATOM | 82 | CA  | SER | A | 11 | 31.146 | 23.418 | 8.250  | 1.00 | 10.00 | C |
| ATOM | 83 | C   | SER | A | 11 | 30.463 | 22.048 | 8.303  | 1.00 | 10.00 | C |
| ATOM | 84 | O   | SER | A | 11 | 29.615 | 21.759 | 7.422  | 1.00 | 10.00 | O |
| ATOM | 85 | CB  | SER | A | 11 | 32.004 | 23.615 | 6.998  | 1.00 | 14.00 | C |
| ATOM | 86 | OG  | SER | A | 11 | 32.013 | 24.995 | 6.632  | 1.00 | 19.00 | O |

|      |     |     |     |   |    |        |        |        |      |       |   |
|------|-----|-----|-----|---|----|--------|--------|--------|------|-------|---|
| ATOM | 87  | N   | LYS | A | 12 | 30.402 | 21.619 | 9.544  | 1.00 | 10.00 | N |
| ATOM | 88  | CA  | LYS | A | 12 | 29.792 | 20.460 | 10.189 | 1.00 | 9.00  | C |
| ATOM | 89  | C   | LYS | A | 12 | 28.494 | 20.817 | 10.932 | 1.00 | 9.00  | C |
| ATOM | 90  | O   | LYS | A | 12 | 27.597 | 19.943 | 10.980 | 1.00 | 9.00  | O |
| ATOM | 91  | CB  | LYS | A | 12 | 30.811 | 20.013 | 11.224 | 1.00 | 10.00 | C |
| ATOM | 92  | CG  | LYS | A | 12 | 30.482 | 18.661 | 11.833 | 1.00 | 14.00 | C |
| ATOM | 93  | CD  | LYS | A | 12 | 31.413 | 18.365 | 12.999 | 1.00 | 18.00 | C |
| ATOM | 94  | CE  | LYS | A | 12 | 31.243 | 16.937 | 13.498 | 1.00 | 22.00 | C |
| ATOM | 95  | NZ  | LYS | A | 12 | 32.121 | 16.717 | 14.652 | 1.00 | 26.00 | N |
| ATOM | 96  | N   | TYR | A | 13 | 28.583 | 21.742 | 11.894 | 1.00 | 9.00  | N |
| ATOM | 97  | CA  | TYR | A | 13 | 27.396 | 22.283 | 12.612 | 1.00 | 8.00  | C |
| ATOM | 98  | C   | TYR | A | 13 | 26.214 | 22.497 | 11.670 | 1.00 | 8.00  | C |
| ATOM | 99  | O   | TYR | A | 13 | 25.037 | 22.245 | 12.029 | 1.00 | 8.00  | O |
| ATOM | 100 | CB  | TYR | A | 13 | 27.730 | 23.578 | 13.385 | 1.00 | 8.00  | C |
| ATOM | 101 | CG  | TYR | A | 13 | 26.516 | 24.500 | 13.692 | 1.00 | 8.00  | C |
| ATOM | 102 | CD1 | TYR | A | 13 | 25.798 | 24.377 | 14.867 | 1.00 | 8.00  | C |
| ATOM | 103 | CD2 | TYR | A | 13 | 26.185 | 25.498 | 12.796 | 1.00 | 8.00  | C |
| ATOM | 104 | CE1 | TYR | A | 13 | 24.713 | 25.228 | 15.120 | 1.00 | 8.00  | C |
| ATOM | 105 | CE2 | TYR | A | 13 | 25.108 | 26.342 | 13.035 | 1.00 | 8.00  | C |
| ATOM | 106 | CZ  | TYR | A | 13 | 24.370 | 26.210 | 14.196 | 1.00 | 8.00  | C |
| ATOM | 107 | OH  | TYR | A | 13 | 23.202 | 26.933 | 14.347 | 1.00 | 10.00 | O |
| ATOM | 108 | N   | LEU | A | 14 | 26.522 | 22.993 | 10.494 | 1.00 | 8.00  | N |
| ATOM | 109 | CA  | LEU | A | 14 | 25.461 | 23.263 | 9.523  | 1.00 | 8.00  | C |
| ATOM | 110 | C   | LEU | A | 14 | 24.912 | 21.978 | 8.907  | 1.00 | 8.00  | C |
| ATOM | 111 | O   | LEU | A | 14 | 24.122 | 22.025 | 7.933  | 1.00 | 8.00  | O |
| ATOM | 112 | CB  | LEU | A | 14 | 25.923 | 24.242 | 8.447  | 1.00 | 13.00 | C |
| ATOM | 113 | CG  | LEU | A | 14 | 25.064 | 25.509 | 8.412  | 1.00 | 19.00 | C |
| ATOM | 114 | CD1 | LEU | A | 14 | 25.564 | 26.496 | 7.505  | 1.00 | 25.00 | C |
| ATOM | 115 | CD2 | LEU | A | 14 | 23.582 | 25.209 | 8.199  | 1.00 | 25.00 | C |
| ATOM | 116 | N   | ASP | A | 15 | 25.556 | 20.886 | 9.263  | 1.00 | 8.00  | N |
| ATOM | 117 | CA  | ASP | A | 15 | 25.075 | 19.552 | 8.885  | 1.00 | 8.00  | C |
| ATOM | 118 | C   | ASP | A | 15 | 24.208 | 19.002 | 10.009 | 1.00 | 8.00  | C |
| ATOM | 119 | O   | ASP | A | 15 | 23.550 | 17.940 | 9.861  | 1.00 | 8.00  | O |
| ATOM | 120 | CB  | ASP | A | 15 | 26.246 | 18.601 | 8.644  | 1.00 | 11.00 | C |
| ATOM | 121 | CG  | ASP | A | 15 | 26.260 | 18.121 | 7.196  | 1.00 | 16.00 | C |
| ATOM | 122 | OD1 | ASP | A | 15 | 26.021 | 18.946 | 6.280  | 1.00 | 21.00 | O |
| ATOM | 123 | OD2 | ASP | A | 15 | 26.732 | 16.984 | 6.946  | 1.00 | 21.00 | O |
| ATOM | 124 | N   | SER | A | 16 | 24.015 | 19.861 | 10.986 | 1.00 | 8.00  | N |
| ATOM | 125 | CA  | SER | A | 16 | 23.180 | 19.548 | 12.149 | 1.00 | 7.00  | C |
| ATOM | 126 | C   | SER | A | 16 | 21.923 | 20.414 | 12.167 | 1.00 | 7.00  | C |
| ATOM | 127 | O   | SER | A | 16 | 20.841 | 19.941 | 12.598 | 1.00 | 7.00  | O |
| ATOM | 128 | CB  | SER | A | 16 | 23.981 | 19.746 | 13.437 | 1.00 | 9.00  | C |
| ATOM | 129 | OG  | SER | A | 16 | 23.327 | 19.102 | 14.524 | 1.00 | 11.00 | O |
| ATOM | 130 | N   | ARG | A | 17 | 22.037 | 21.605 | 11.597 | 1.00 | 7.00  | N |

|      |     |     |     |   |    |        |        |        |      |       |   |
|------|-----|-----|-----|---|----|--------|--------|--------|------|-------|---|
| ATOM | 131 | CA  | ARG | A | 17 | 20.875 | 22.504 | 11.583 | 1.00 | 6.00  | C |
| ATOM | 132 | C   | ARG | A | 17 | 19.868 | 22.156 | 10.491 | 1.00 | 6.00  | C |
| ATOM | 133 | O   | ARG | A | 17 | 18.665 | 22.015 | 10.809 | 1.00 | 6.00  | O |
| ATOM | 134 | CB  | ARG | A | 17 | 21.214 | 23.997 | 11.557 | 1.00 | 7.00  | C |
| ATOM | 135 | CG  | ARG | A | 17 | 20.010 | 24.800 | 12.063 | 1.00 | 9.00  | C |
| ATOM | 136 | CD  | ARG | A | 17 | 19.570 | 25.929 | 11.132 | 1.00 | 11.00 | C |
| ATOM | 137 | NE  | ARG | A | 17 | 20.149 | 27.218 | 11.537 | 1.00 | 12.00 | N |
| ATOM | 138 | CZ  | ARG | A | 17 | 19.828 | 28.351 | 10.936 | 1.00 | 13.00 | C |
| ATOM | 139 | NH1 | ARG | A | 17 | 19.319 | 28.304 | 9.720  | 1.00 | 14.00 | N |
| ATOM | 140 | NH2 | ARG | A | 17 | 20.351 | 29.485 | 11.362 | 1.00 | 14.00 | N |
| ATOM | 141 | N   | ARG | A | 18 | 20.378 | 21.725 | 9.348  | 1.00 | 6.00  | N |
| ATOM | 142 | CA  | ARG | A | 18 | 19.530 | 21.258 | 8.235  | 1.00 | 5.00  | C |
| ATOM | 143 | C   | ARG | A | 18 | 19.148 | 19.796 | 8.478  | 1.00 | 5.00  | C |
| ATOM | 144 | O   | ARG | A | 18 | 18.326 | 19.189 | 7.741  | 1.00 | 5.00  | O |
| ATOM | 145 | CB  | ARG | A | 18 | 20.237 | 21.481 | 6.888  | 1.00 | 8.00  | C |
| ATOM | 146 | CG  | ARG | A | 18 | 19.384 | 21.236 | 5.634  | 1.00 | 9.00  | C |
| ATOM | 147 | CD  | ARG | A | 18 | 19.623 | 19.860 | 5.005  | 1.00 | 11.00 | C |
| ATOM | 148 | NE  | ARG | A | 18 | 20.029 | 19.997 | 3.600  | 1.00 | 12.00 | N |
| ATOM | 149 | CZ  | ARG | A | 18 | 19.398 | 19.415 | 2.597  | 1.00 | 13.00 | C |
| ATOM | 150 | NH1 | ARG | A | 18 | 18.483 | 18.493 | 2.835  | 1.00 | 14.00 | N |
| ATOM | 151 | NH2 | ARG | A | 18 | 19.831 | 19.597 | 1.364  | 1.00 | 14.00 | N |
| ATOM | 152 | N   | ALA | A | 19 | 19.560 | 19.319 | 9.623  | 1.00 | 6.00  | N |
| ATOM | 153 | CA  | ALA | A | 19 | 19.126 | 17.991 | 10.053 | 1.00 | 6.00  | C |
| ATOM | 154 | C   | ALA | A | 19 | 18.002 | 18.136 | 11.071 | 1.00 | 6.00  | C |
| ATOM | 155 | O   | ALA | A | 19 | 16.933 | 17.494 | 10.922 | 1.00 | 7.00  | O |
| ATOM | 156 | CB  | ALA | A | 19 | 20.285 | 17.187 | 10.629 | 1.00 | 15.00 | C |
| ATOM | 157 | N   | GLN | A | 20 | 18.094 | 19.241 | 11.783 | 1.00 | 7.00  | N |
| ATOM | 158 | CA  | GLN | A | 20 | 17.013 | 19.632 | 12.689 | 1.00 | 7.00  | C |
| ATOM | 159 | C   | GLN | A | 20 | 15.897 | 20.314 | 11.905 | 1.00 | 7.00  | C |
| ATOM | 160 | O   | GLN | A | 20 | 14.701 | 20.031 | 12.162 | 1.00 | 7.00  | O |
| ATOM | 161 | CB  | GLN | A | 20 | 17.513 | 20.538 | 13.821 | 1.00 | 11.00 | C |
| ATOM | 162 | CG  | GLN | A | 20 | 16.699 | 21.829 | 13.936 | 1.00 | 16.00 | C |
| ATOM | 163 | CD  | GLN | A | 20 | 16.591 | 22.277 | 15.393 | 1.00 | 22.00 | C |
| ATOM | 164 | OE1 | GLN | A | 20 | 17.533 | 22.060 | 16.194 | 1.00 | 24.00 | O |
| ATOM | 165 | NE2 | GLN | A | 20 | 15.356 | 22.544 | 15.773 | 1.00 | 24.00 | N |
| ATOM | 166 | N   | ASP | A | 21 | 16.292 | 20.724 | 10.714 | 1.00 | 7.00  | N |
| ATOM | 167 | CA  | ASP | A | 21 | 15.405 | 21.490 | 9.835  | 1.00 | 7.00  | C |
| ATOM | 168 | C   | ASP | A | 21 | 14.451 | 20.565 | 9.120  | 1.00 | 7.00  | C |
| ATOM | 169 | O   | ASP | A | 21 | 13.245 | 20.850 | 8.962  | 1.00 | 7.00  | O |
| ATOM | 170 | CB  | ASP | A | 21 | 16.212 | 22.278 | 8.809  | 1.00 | 14.00 | C |
| ATOM | 171 | CG  | ASP | A | 21 | 15.427 | 23.525 | 8.413  | 1.00 | 21.00 | C |
| ATOM | 172 | OD1 | ASP | A | 21 | 15.031 | 24.298 | 9.321  | 1.00 | 28.00 | O |
| ATOM | 173 | OD2 | ASP | A | 21 | 15.316 | 23.827 | 7.200  | 1.00 | 28.00 | O |
| ATOM | 174 | N   | PHE | A | 22 | 14.987 | 19.373 | 8.843  | 1.00 | 7.00  | N |

|      |     |     |     |   |    |        |        |        |      |       |   |
|------|-----|-----|-----|---|----|--------|--------|--------|------|-------|---|
| ATOM | 175 | CA  | PHE | A | 22 | 14.216 | 18.253 | 8.289  | 1.00 | 7.00  | C |
| ATOM | 176 | C   | PHE | A | 22 | 13.098 | 17.860 | 9.246  | 1.00 | 7.00  | C |
| ATOM | 177 | O   | PHE | A | 22 | 11.956 | 17.556 | 8.818  | 1.00 | 7.00  | O |
| ATOM | 178 | CB  | PHE | A | 22 | 15.134 | 17.038 | 8.105  | 1.00 | 8.00  | C |
| ATOM | 179 | CG  | PHE | A | 22 | 14.349 | 15.761 | 7.724  | 1.00 | 10.00 | C |
| ATOM | 180 | CD1 | PHE | A | 22 | 14.022 | 15.527 | 6.410  | 1.00 | 12.00 | C |
| ATOM | 181 | CD2 | PHE | A | 22 | 13.992 | 14.842 | 8.689  | 1.00 | 12.00 | C |
| ATOM | 182 | CE1 | PHE | A | 22 | 13.302 | 14.391 | 6.050  | 1.00 | 14.00 | C |
| ATOM | 183 | CE2 | PHE | A | 22 | 13.269 | 13.708 | 8.340  | 1.00 | 14.00 | C |
| ATOM | 184 | CZ  | PHE | A | 22 | 12.917 | 13.483 | 7.018  | 1.00 | 16.00 | C |
| ATOM | 185 | N   | VAL | A | 23 | 13.455 | 17.883 | 10.517 | 1.00 | 7.00  | N |
| ATOM | 186 | CA  | VAL | A | 23 | 12.574 | 17.403 | 11.589 | 1.00 | 7.00  | C |
| ATOM | 187 | C   | VAL | A | 23 | 11.283 | 18.205 | 11.729 | 1.00 | 7.00  | C |
| ATOM | 188 | O   | VAL | A | 23 | 10.233 | 17.600 | 12.052 | 1.00 | 7.00  | O |
| ATOM | 189 | CB  | VAL | A | 23 | 13.339 | 17.278 | 12.906 | 1.00 | 10.00 | C |
| ATOM | 190 | CG1 | VAL | A | 23 | 12.441 | 17.004 | 14.108 | 1.00 | 13.00 | C |
| ATOM | 191 | CG2 | VAL | A | 23 | 14.455 | 16.248 | 12.794 | 1.00 | 13.00 | C |
| ATOM | 192 | N   | GLN | A | 24 | 11.255 | 19.253 | 10.941 | 1.00 | 8.00  | N |
| ATOM | 193 | CA  | GLN | A | 24 | 10.082 | 20.114 | 10.818 | 1.00 | 8.00  | C |
| ATOM | 194 | C   | GLN | A | 24 | 9.158  | 19.638 | 9.692  | 1.00 | 8.00  | C |
| ATOM | 195 | O   | GLN | A | 24 | 7.959  | 19.990 | 9.663  | 1.00 | 8.00  | O |
| ATOM | 196 | CB  | GLN | A | 24 | 10.575 | 21.521 | 10.498 | 1.00 | 14.00 | C |
| ATOM | 197 | CG  | GLN | A | 24 | 9.505  | 22.591 | 10.661 | 1.00 | 20.00 | C |
| ATOM | 198 | CD  | GLN | A | 24 | 9.964  | 23.862 | 9.956  | 1.00 | 26.00 | C |
| ATOM | 199 | OE1 | GLN | A | 24 | 10.079 | 24.941 | 10.587 | 1.00 | 32.00 | O |
| ATOM | 200 | NE2 | GLN | A | 24 | 10.086 | 23.739 | 8.649  | 1.00 | 32.00 | N |
| ATOM | 201 | N   | TRP | A | 25 | 9.723  | 19.074 | 8.651  | 1.00 | 8.00  | N |
| ATOM | 202 | CA  | TRP | A | 25 | 8.899  | 18.676 | 7.495  | 1.00 | 9.00  | C |
| ATOM | 203 | C   | TRP | A | 25 | 8.118  | 17.395 | 7.751  | 1.00 | 9.00  | C |
| ATOM | 204 | O   | TRP | A | 25 | 6.860  | 17.395 | 7.725  | 1.00 | 9.00  | O |
| ATOM | 205 | CB  | TRP | A | 25 | 9.761  | 18.442 | 6.262  | 1.00 | 11.00 | C |
| ATOM | 206 | CG  | TRP | A | 25 | 8.871  | 18.331 | 5.004  | 1.00 | 12.00 | C |
| ATOM | 207 | CD1 | TRP | A | 25 | 8.097  | 19.279 | 4.442  | 1.00 | 12.00 | C |
| ATOM | 208 | CD2 | TRP | A | 25 | 8.640  | 17.180 | 4.249  | 1.00 | 12.00 | C |
| ATOM | 209 | NE1 | TRP | A | 25 | 7.041  | 18.780 | 3.259  | 1.00 | 12.00 | N |
| ATOM | 210 | CE2 | TRP | A | 25 | 7.873  | 17.564 | 3.121  | 1.00 | 12.00 | C |
| ATOM | 211 | CE3 | TRP | A | 25 | 9.124  | 15.884 | 4.378  | 1.00 | 12.00 | C |
| ATOM | 212 | CZ2 | TRP | A | 25 | 7.726  | 16.765 | 2.003  | 1.00 | 12.00 | C |
| ATOM | 213 | CZ3 | TRP | A | 25 | 8.870  | 15.038 | 3.296  | 1.00 | 12.00 | C |
| ATOM | 214 | CH2 | TRP | A | 25 | 8.216  | 15.469 | 2.140  | 1.00 | 12.00 | C |
| ATOM | 215 | N   | LEU | A | 26 | 8.857  | 16.484 | 8.346  | 1.00 | 9.00  | N |
| ATOM | 216 | CA  | LEU | A | 26 | 8.377  | 15.159 | 8.741  | 1.00 | 10.00 | C |
| ATOM | 217 | C   | LEU | A | 26 | 7.534  | 15.279 | 10.012 | 1.00 | 11.00 | C |
| ATOM | 218 | O   | LEU | A | 26 | 6.755  | 14.347 | 10.331 | 1.00 | 11.00 | O |

|        |     |     |     |   |    |        |        |        |      |       |     |   |   |   |
|--------|-----|-----|-----|---|----|--------|--------|--------|------|-------|-----|---|---|---|
| ATOM   | 219 | CB  | LEU | A | 26 | 9.611  | 14.267 | 8.924  | 1.00 | 10.00 |     | C |   |   |
| ATOM   | 220 | CG  | LEU | A | 26 | 9.342  | 12.810 | 9.303  | 1.00 | 10.00 |     | C |   |   |
| ATOM   | 221 | CD1 | LEU | A | 26 | 8.223  | 12.149 | 8.505  | 1.00 | 10.00 |     | C |   |   |
| ATOM   | 222 | CD2 | LEU | A | 26 | 10.637 | 11.982 | 9.250  | 1.00 | 10.00 |     | C |   |   |
| ATOM   | 223 | N   | MET | A | 27 | 7.281  | 16.544 | 10.320 | 1.00 | 11.00 |     | N |   |   |
| ATOM   | 224 | CA  | MET | A | 27 | 6.446  | 16.959 | 11.451 | 1.00 | 11.00 |     | C |   |   |
| ATOM   | 225 | C   | MET | A | 27 | 5.607  | 18.227 | 11.219 | 1.00 | 13.00 |     | C |   |   |
| ATOM   | 226 | O   | MET | A | 27 | 4.823  | 18.240 | 10.244 | 1.00 | 13.00 |     | O |   |   |
| ATOM   | 227 | CB  | MET | A | 27 | 7.327  | 17.118 | 12.679 | 1.00 | 11.00 |     | C |   |   |
| ATOM   | 228 | CG  | MET | A | 27 | 6.518  | 17.289 | 13.953 | 1.00 | 11.00 |     | C |   |   |
| ATOM   | 229 | SD  | MET | A | 27 | 7.301  | 18.326 | 15.196 | 1.00 | 11.00 |     | S |   |   |
| ATOM   | 230 | CE  | MET | A | 27 | 5.833  | 18.677 | 16.178 | 1.00 | 11.00 |     | C |   |   |
| ATOM   | 231 | N   | ASN | A | 28 | 6.147  | 19.366 | 11.620 | 1.00 | 14.00 |     | N |   |   |
| ATOM   | 232 | CA  | ASN | A | 28 | 5.399  | 20.637 | 11.728 | 1.00 | 14.00 |     | C |   |   |
| ATOM   | 233 | C   | ASN | A | 28 | 3.878  | 20.587 | 11.716 | 1.00 | 17.00 |     | C |   |   |
| ATOM   | 234 | O   | ASN | A | 28 | 3.252  | 21.114 | 10.763 | 1.00 | 19.00 |     | O |   |   |
| ATOM   | 235 | CB  | ASN | A | 28 | 5.874  | 21.774 | 10.843 | 1.00 | 14.00 |     | C |   |   |
| ATOM   | 236 | CG  | ASN | A | 28 | 6.246  | 22.905 | 11.791 | 1.00 | 14.00 |     | C |   |   |
| ATOM   | 237 | OD1 | ASN | A | 28 | 6.929  | 22.629 | 12.807 | 1.00 | 14.00 |     | O |   |   |
| ATOM   | 238 | ND2 | ASN | A | 28 | 6.271  | 24.085 | 11.229 | 1.00 | 14.00 |     | N |   |   |
| ATOM   | 239 | N   | THR | A | 29 | 3.391  | 19.940 | 12.762 | 1.00 | 21.00 |     | N |   |   |
| ATOM   | 240 | CA  | THR | A | 29 | 2.014  | 19.761 | 13.283 | 1.00 | 21.00 |     | C |   |   |
| ATOM   | 241 | C   | THR | A | 29 | 0.826  | 19.943 | 12.332 | 1.00 | 23.00 |     | C |   |   |
| ATOM   | 242 | O   | THR | A | 29 | 0.932  | 19.600 | 11.133 | 1.00 | 30.00 |     | O |   |   |
| ATOM   | 243 | CB  | THR | A | 29 | 1.845  | 20.667 | 14.505 | 1.00 | 21.00 |     | C |   |   |
| ATOM   | 244 | OG1 | THR | A | 29 | 1.214  | 21.893 | 14.153 | 1.00 | 21.00 |     | O |   |   |
| ATOM   | 245 | CG2 | THR | A | 29 | 3.180  | 20.968 | 15.185 | 1.00 | 21.00 |     | C |   |   |
| ATOM   | 246 | OXT | THR | A | 29 | -0.317 | 20.109 | 12.824 | 1.00 | 25.00 |     | O |   |   |
| TER    | 247 |     | THR | A | 29 |        |        |        |      |       |     |   |   |   |
| MASTER |     | 344 |     | 1 | 0  | 1      | 0      | 0      | 0    | 6     | 246 | 1 | 0 | 3 |
| END    |     |     |     |   |    |        |        |        |      |       |     |   |   |   |



# Appendix C

## Co-authors' approval letters

University of Windsor Mail - Permission request



Satish Panigrahi <panigra@uwindsor.ca>

---

### Permission request

2 messages

---

Satish Panigrahi <panigra@uwindsor.ca>

Tue, Sep 9, 2014 at 4:17 PM

To: Asish Mukhopadhyay <asishm@cs.uwindsor.ca>, asishm@uwindsor.ca

Dear Sir

I request your permission as a co-author to include following published materials in my Ph.D. dissertation.

- 1) Asish Mukhopadhyay, Satish Panigrahi. "All maximum and all-minimum problems under some measures"; Journal of Discrete Algorithms, Volume 21, Pages 18 - 31, 2013.
- 2) Satish Panigrahi, Md. Shafiul Alam, and Asish Mukhopadhyay. "An incremental linear programming based tool for analyzing gene expression data"; In Beniamino Murgante, Sanjay Misra, Maurizio Carlini, Carmelo M. Torre, Hong-Quang Nguyen, David Taniar, Bernady O. Apduhan, and Osvaldo Gervasi (Eds.), Computational Science and Its Applications ICCSA 2013, volume 7975 of Lecture Notes in Computer Science, pages 48 - 64, 2013.
- 3) Satish Panigrahi and Asish Mukhopadhyay. "An eigendecomposition method for protein structure alignment", In M. Basu, Y. Pan, and J. Wang (Eds.), Bioinformatics Research and Applications, 10th International Symposium, ISBRA 2014, volume 8492 of Lecture Notes in Bioinformatics, pages 24 - 37, 2014.

Sincerely

Satish Chandra Panigrahi

---

Asish Mukhopadhyay <asish.mukerji@gmail.com>

Tue, Sep 9, 2014 at 4:33 PM

To: Satish Panigrahi <panigra@uwindsor.ca>

Dear Satish Panigrahi:

You have my permission to include the above publications in which I am a co-author to be included in your thesis.

best wishes

asish m

[Quoted text hidden]



Satish Panigrahi <panigra@uwindsor.ca>

---

**Permission request**

2 messages

---

**Satish Panigrahi** <panigra@uwindsor.ca>  
To: Md Alam <alam9@uwindsor.ca>

Tue, Sep 9, 2014 at 4:20 PM

Dear Dr. Alam

I request your permission as a co-author to include following published material in my Ph.D. dissertation.

"Satish Panigrahi, Md. Shafiu Alam, and Asish Mukhopadhyay. "An incremental linear programming based tool for analyzing gene expression data"; In Beniamino Murgante, Sanjay Misra, Maurizio Carlini, Carmelo M. Torre, Hong-Quang Nguyen, David Taniar, Bernady O. Apduhan, and Osvaldo Gervasi (Eds.), Computational Science and Its Applications ICCSA 2013, volume 7975 of Lecture Notes in Computer Science, pages 48 - 64, 2013."

Sincerely

Satish Chandra Panigrahi

---

**Md Alam** <alam9@uwindsor.ca>  
To: Satish Panigrahi <panigra@uwindsor.ca>

Wed, Sep 10, 2014 at 6:50 PM

Dear Mr. Panigrahi,

My permission is granted.

Best regards,

Md. Shafiu Alam  
(Quoted text hidden)

# VITA AUCTORIS

**NAME** : Satish Chandra Panigrahi

**BIRTH YEAR** : 1979

**BIRTH PLACE** : INDIA

## **EDUCATION**

**2014** : **PHD Computer Science**

School of Computer Science

University of Windsor, Windsor, ON, Canada

**2006** : **Masters of Engineering**

Department of Computer Science and Engineering

Birla Institute of Technology, Mesra, Ranchi, India

**2001** : **Bachelors of Engineering**

Department of Computer Science and Engineering

Institute of Technical Education and Research, Bhubaneswar, India