NANYANG
TECHNOLOGICAL
UNIVERSITY

VISUALIZATION OF ENGINE DESIGN, SIMULATION AND OPTIMIZATION

YAN CHAO WANG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2019

VISUALIZATION OF ENGINE DESIGN, SIMULATION AND OPTIMIZATION

YAN CHAO WANG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

A thesis submitted to the Nanyang Technological University
in partial fulfilment of the requirement for the degree of
Doctor of Philosophy

2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

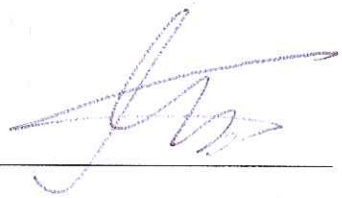
05 August 2019	
Date	Yan Chao WANG

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

05 August 2019

Date



Feng LIN

Authorship Attribution Statement

This thesis contains material from **SIX** paper(s) published in the following peer-reviewed journal(s) / from papers accepted at conferences in which I am listed as an author.

Chapter 3 is published as:

- Qian Zhang, Chung Soo Ahn, Jigang Liu, **Yan Chao Wang**, Feng Lin, Hock Soon Seah, **Interactive geo-temporal data visualization for aircraft engine**, In the *International Workshop on Advanced Image Technology* (IWAIT), SPIE, 2019.

  The contributions of the co-authors are as follows:

  - Miss Zhang wrote the drafts of the manuscript;
  - Mr. Ahn and Dr. Liu assisted to implement the system;
  - I generated the figures and revised the manuscript;
  - Prof. Lin revised the manuscript together with Prof. Seah;

- **Yan-Chao Wang**, Qian Zhang, Feng Lin, and Hock-Soon Seah. **EngineQV: Investigating External Cause of Engine Failures Based on Geo-temporal Association**. In the *IEEE Pacific Visualization Symposium* (PacificVis), IEEE, 2019.

  The contributions of the co-authors are as follows:

  - I wrote the drafts of the manuscript;
  - Miss Zhang assisted to implement the system;
  - Prof. Lin revised the manuscript together with Prof. Seah;

Chapter 4 is published as:

- **Yan-Chao Wang**, Qian Zhang, Feng Lin, Chi-Keong Goh, Xuan Wang, and Hock-Soon Seah. **Histogram equalization and specification for high-dimensional data visualization using radviz**. In the *Computer Graphics International Conference* (CGI), ACM, 2017.

  The contributions of the co-authors are as follows:

  - I proposed the method, implemented the algorithms, and wrote the drafts of the manuscript;
  - Miss Zhang generated the figures;
  - Dr. Goh and Dr. Wang proposed the topic;
  - Prof. Lin revised the manuscript together with Prof. Seah;

- **Yan-Chao Wang**, Qian Zhang, Feng Lin, Chi Keong Goh, Hock Soon Seah, **PolarViz: a discriminating visualization and visual analytics tool for high-dimensional data**, *The Visual Computer*, Springer, 2018.

  The contributions of the co-authors are as follows:

  - I proposed the method, implemented the algorithms, and wrote the drafts of the manuscript;
  - Miss Zhang generated the figures;
  - Dr. Goh proposed the topic;
  - Prof. Lin revised the manuscript together with Prof. Seah;

- **Yan-Chao Wang**, Feng Lin, and Hock-Soon Seah. **Evaluation of Star Coordinate Boundaries**. In the *Computer Graphics International Conference* (CGI), ACM, 2018.

  The contributions of the co-authors are as follows:

  - I proposed the method, implemented the algorithms, and wrote the drafts of the manuscript;
  - Prof. Lin revised the manuscript together with Prof. Seah;

Chapter 5 is published as:

- **Yan-Chao Wang** Feng Lin, and Hock-Soon Seah. **Orthogonal Voronoi Diagram and Treemap**, *arXiv preprint*, arXiv:1904.02348, 2019.

  The contributions of the co-authors are as follows:

  - I proposed the method, implemented the algorithms, and wrote the drafts of the manuscript;
  - Prof. Lin revised the manuscript together with Prof. Seah;

05 August 2019

Date Yan Chao WANG

Acknowledgment

I would like to take this opportunity to express my sincere gratitude to my supervisors Assoc. Prof. Lin Feng and Prof. Seah Hoch Soon for their patient guidance and constant support throughout my Ph.D. study. Prof. Lin gave me valuable comments and suggestions to help me logically and clearly present my work. Meanwhile, Prof. Seah focused on the details of my work, helping me concisely express my ideas. I appreciate all their contributions to make my Ph.D. study productive. Without these help, I cannot complete my Ph.D. study as well as this work.

My great thanks also go to Prof. Chen Lujie in SUTD and Prof. Quan Chenggen in NUS, who led me to start my academic journey. Thanks to staff from Rolls-Royce company, Dr. Wang Xuan, Dr. Goh Chi Keong, Dr. Wang Yi, Dr. Gee Sen Bong, Mr. Rajendran Vinod, for your support throughout my study. I would like to thank my thesis advisory committee members, Prof. Qian Kemao and Prof. Lee Yong Tsui, for their rigorous and brilliant comments.

Thanks to my friends and colleagues from Rolls-Royce@NTU Corp Lab, Dr. Liu Jigang, Miss Zhang Qian, Mr. Ahn Chung Soo, Dr. Xie Xiurui, Miss Li Jingliang, Mr. Liu Yunxiang, Dr. Yan Xiaoqi, Dr. Yao Dezhong, who created a pleasant atmosphere. Also thanks to my friends and colleagues from Multi-plAtform Game Innovation Centre (MAGIC), Dr. Qiu Jie, Dr. Chen Quan, Dr. Koa Ming Di, Mr. Jiang Jie, Dr. Budi, for their great help and accompany. Thanks to all staff of Rolls-Royce@NTU Corp Lab, MAGIC Lab, and SCSE Graduate Office. They helped me smoothly conduct my research study.

Last but not least, I would like to thank my family for their unconditional support and precious love. Special thanks to my wife, Dr. Xing Yidan. She is my spiritual pillar, the pillar of this that I can still feel alive significance. Thanks to my lovely son, Wang Leyi, who taught me the innate sense of responsibility.

05 August 2019

Contents

Acknowledgment **i**

Table of Contents **ii**

Abstract **vii**

List of Tables **ix**

List of Figures **xvi**

List of Algorithms **xvii**

1 Introduction **1**

1.1 Background 1

1.1.1 Digitalization of Engine-associated Procedures 2

1.1.2 Data Analysis of Engine-associated Data 4

1.2 Visualization of Engine-associated Data 6

1.2.1 Specific Visualization Application 6

1.2.2 Generic Visualization Method 7

1.2.3 Challenges of Engine-associated Data Visualization 8

1.3 Contributions and Outline of Thesis 9

1.3.1 Contributions of Thesis 9

1.3.2 Outline of Thesis 10

2 Literature Review **12**

2.1 Geo-temporal Data Visualization 12

2.1.1 Map Oriented Trajectory Visualization 12

2.1.2 Temporal Event Data Visualization 13
2.1.3 Visual Comparison 14
2.2 High-dimensional Data Visualization 14
2.2.1 RadViz 16
2.2.2 Star Coordinate 17
2.3 Hierarchical Data Visualization 18
2.3.1 Explicit Hierarchy Visualization 18
2.3.1.1 Node-link Connection 19
2.3.1.2 Touch Connection 20
2.3.2 Implicit Hierarchy Visualization 20
2.3.2.1 Non-site-based Methods 21
2.3.2.2 Site-based Methods 22

3 EngineQV: Investigating External Cause of Engine Failures Based on Geo-temporal Association 23

3.1 Introduction 23
3.2 Background Study 25
3.2.1 Task Analysis 25
3.2.2 Design Requirements 27
3.3 EngineQV 28
3.3.1 Overview 28
3.3.2 Data Design 29
3.3.2.1 Data Description and Pre-processing 29
3.3.2.2 Data Model 31
3.3.3 Architecture Design 33
3.3.4 Query Design 34
3.3.5 Visual Design 36
3.3.6 Interactions 41
3.3.7 Implementation 41
3.4 Evaluation 42
3.4.1 Use Cases Study 42

3.4.1.1 Use Case 1: Sense Making and Knowledge Discovery of Single Engine . . . 43
3.4.1.2 Use Case 2: Visual Comparison of Multiple Engines . . 46
3.4.2 Expert Feedback . . . 48
3.5 Summary . . . 49

4 Discriminating Visualization for High-dimensional Data 50

4.1 Introduction . . . 50
4.2 Background . . . 52
4.3 PolarViz: A focus+context distortion approach for RadViz . . . 53
4.3.1 Introduction . . . 53
4.3.2 The Clumping Effect . . . 55
4.3.2.1 Clumping Problem in RadViz . . . 55
4.3.2.2 Previous Approach for Clumping Problem . . . 56
4.3.2.3 Formulation in Polar Coordinate . . . 58
4.3.2.4 Visual Analytics in Polar Coordinate . . . 60
4.3.3 Detailed Methods . . . 62
4.3.3.1 Radial Operations . . . 62
4.3.3.2 PolarViz . . . 67
4.3.4 Comparative Studies . . . 69
4.3.4.1 RadViz with Sigmoid Function . . . 70
4.3.4.2 Distortion Methods . . . 75
4.3.5 User Study . . . 76
4.3.6 Discussion . . . 79
4.4 Boundaries of the Star Coordinate . . . 80
4.4.1 Introduction . . . 80
4.4.2 Boundary of SC with standard configuration . . . 81
4.4.2.1 Axis-aligned Bounding Box . . . 81
4.4.2.2 Bounding Polygon . . . 82
4.4.3 Exploration for Boundary of Arbitrary SC . . . 88
4.4.3.1 Axis-aligned Bounding Box . . . 89

4.4.3.2 Bounding Polygon 90
4.4.3.3 Minimum Bounding Box 92
4.4.4 Boundary-Driven Application 92
4.4.4.1 SC with Boundaries 92
4.4.4.2 Association with RadViz 93
4.4.4.3 Clumping Effect in SC 95
4.4.5 Discussion 97
4.5 Summary 98

5 Orthogonal Voronoi Diagram and Treemap for Implicit Hierarchical Visualization 99

5.1 Introduction 99
5.2 Background 101
5.2.1 Voronoi Diagram 101
5.2.2 Weighted Voronoi Diagram 102
5.2.3 Centroidal Voronoi Diagram 103
5.2.4 Voronoi Treemap 103
5.3 Orthogonal Voronoi Treemap 104
5.3.1 Overview 104
5.3.2 Initialization 105
5.3.3 Site Status Update 106
5.3.4 Computation of the Orthogonal Voronoi Diagram 107
5.3.4.1 Distance Function 108
5.3.4.2 Clustering 110
5.3.4.3 Sweepline + skyline algorithm 111
5.3.5 Implementation 113
5.4 Performance 113
5.4.1 Site Pairs and Valid Neighbor 115
5.4.2 Single Iteration Comparison 117
5.4.3 Converge Rate 119
5.4.4 Aspect Ratio 120

5.5 Discussion 123
5.6 Summary 124

6 Conclusions and Future Work **126**

6.1 Conclusions 126
6.2 Future Work 127
6.2.1 Advanced Specific Visualization 128
6.2.2 Novel Generic Visualization Methods 128

List of Publications **130**

Bibliography **132**

Abstract

The development of techniques contributes to the digitalization of engine-related procedures, generating a massive amount of engine-related data generated. Effective and efficient understanding of these datasets would provide domain experts with new insights. However, it is impractical for human beings to go through the raw data to have a clear understanding to conduct data analysis, as these datasets have large-scale size and complex data formats. Hence such data understanding requires closely coupled human and data analysis, where information visualization comes into play. The task on how to utilize information visualization to efficiently understand engine-related datasets can be considered from two aspects. Firstly, visualization can focus on a specific visual analytic target, by integrating heterogeneous datasets and providing meaningful visualization methods. Secondly, visualization can generically provide a tidy and useful view for certain data types. Hence, our work is organized in these two aspects: designing specific visualization system for certain analysis target and proposing novel visualization methods for certain data structures in the engine-related dataset, such as high-dimensional data and hierarchical data.

For specific visualization system, this thesis presents EngineQV, a single page web-based integrated visualization system that integrates multiple geo-temporal engine-associated datasets, with the aim of assisting domain experts to explore the external factors that affect the aircraft engine performance. The system features a dynamic query on the datasets and incorporates several customized interactive visualizations, which provides intuitive exploration and understanding of the data from various aspects. With the system, a user may query a certain group of engines or compare multiple engine groups, identify an issue, and find its potential causes. The functionality and usability of EngineQV are evaluated by several case studies. The validity of this specific system is confirmed by expert feedback.

For generic high-dimensional data visualization, we focus on easing the clumping effect in the two promising plots, RadViz and Star Coordinate. We observe that

with the increase of dataset size and dimensionality, the clumping of projected data points towards the origin in RadViz causes low space utilization and degenerates the visibility of the feature characteristics. Hence, to better evaluate the hidden patterns in the centre region, we proposed a new focus+context distortion approach, termed PolarViz, to manipulate the radial distribution of data points in RadViz. PolarViz cannot be directly used in Star Coordinate, as the Star Coordinate does not have a boundary and PolarViz needs the plot boundary to calculate the relative distance. In this case, we proposed a geometric algorithm to determine an axis-aligned bounding box, minimum bounding box, and bounding polygon of the star coordinate with an evenly distributed configuration and an arbitrarily distributed configuration respectively. These boundaries also can provide an overview of the plot and enable a continuous point rendering that adapts to the screen space boundary.

For generic hierarchical data visualization, we consider providing a tidy and flexible layout for implicit hierarchy visualization. To achieve this, we propose a novel space partitioning strategy. Starting from a new distance function and neighborhood relationship between sites, we divide the space by axis-aligned segments. Then a sweepline+skyline based heuristic algorithm is proposed to allocate the partitioned spaces to form an orthogonal Voronoi diagram with orthogonal rectangles. To the best of our knowledge, it is the first time to use a sweepline-based strategy for the Voronoi treemap. Moreover, we design a novel strategy to initialize the diagram status and modify the status update procedure so that the generation of our plot is more effective and efficient. We show that the proposed algorithm has an $O(nlog(n))$ complexity which is the same as the state-of-the-art Voronoi treemap. To this end, we show, via experiments on artificial and real-world datasets, the performance of our algorithm in terms of computation time, converge rate, and aspect ratio.

Most of our work are implemented in JavaScript and the source code of the proposed PolarViz and the orthogonal Voronoi treemap have been uploaded into the GitHub. The thesis ends with the conclusions of our works and a discussion of future work.

List of Tables

3.1 Data description and symbol list. 32

4.1 Numerical analysis on the clumping effect. The average distance μ, the standard deviation σ, and value of $\mu+3\sigma$ for different dimensionalities n are listed. The last column is the percentage of data points that located in the range of $[0, \mu + 3\sigma)$. Though there is a significant decrease in $\mu + 3\sigma$, the percentage is steady. 61

4.2 The user study process and the corresponding questionnaire are described in this table. 77

5.1 The running time (in ms) for a single iteration. 117

List of Figures

1.1 Engine-Data-Insight Model. 2
1.2 Multiple scenarios of engine digitalization. 3
1.3 Data analysis flow chart in the computational engineering department of advanced technology group at Rolls-Royce Singapore. 5

2.1 The path of Napoleon's march in the Russian campaign of 1812. 13
2.2 High-dimensional Data Visualization. 15
2.3 Explicit Hierarchy Visualization. (a) Node-link connection. (b) Touch connection. 19

3.1 The user interface of EngineQV. 26
3.2 EngineQV comprises a backend (i.e. database and query engines) and a frontend (i.e. query panel and vis panel). The three datasets are stored and handled by query engines in the backend. In the frontend, the query panel allows the user to communicate with the backend and the returned query results will be passed to each visualization component. 29
3.3 EngineQV data model. Three datasets that cover four information types are integrated into the system. Data attributes for building query panel and visualization panel are also highlighted. 31
3.4 Query logic flowchart in Query 1. In the flowchart, *const_rec* refers to the selection constraint list, A_i refers to the ith query component, and a_i refers to the values of A_i. 34

3.5 The geographical visualization includes three layers (the basic map layer, the flight information layer, and the environment-related information layer) with multiple modes to deliver different information. To be specific, we provide boundary lines map (a) and landscape map (b) to give a geographical context in the basic map layer. Besides that, in the flight information layer the user can select which kind of flight information to be illustrated: departure airport (c) or flight route (d). Moreover, the erupted volcano (e) and the volcanic ash (f) can be displayed in the environment-related information layer. Combination of different choices in each layer will deliver different information. 37

3.6 In the statistical visualization, flight distribution visualization (a) and flight statistical visualization (b) are provided. 39

3.7 Temporal Event Visualization displays volcanic ash observation history and maintenance history. 40

3.8 Use case 1: query and visualization of airport air quality. The user may query for airports with high air pollution (a), or compare flights from airports with different PM_i^{from} levels (b). 43

3.9 Use case 1: exploration of the flight routes potentially influenced by the volcanic ash. The user may use Geo-map as an overview to find the influenced flights (a), view the flight route time and frequency in the timetable and histogram plot (b, c), and query results on the volcano eruption events at the related region (d). 44

3.10 Use case 2: visual comparison of multiple engines. The user may start analyzing by looking at the force plot and statistical plots to ignore the outlines (a). After removing EngineRed, comparison of EngineGreen and EngineBlue in timetable shows different patterns from 07th Nov 2017 to 05th Dec 2017 (b). By adjusting the time range, the flight route of different behaviors are identified (c). The geo-map under the airport mode with labels indicates the observation (d). The airport code ICAO displays when hovering a mouse on the airport circle. . . . 46

4.1 The circumscribed circle and bounding polygon of RadViz when plotting 3D, 4D, 5D, and 6D datasets. 54
4.2 We demonstrate the clumping effect by plotting the distribution of 100,000 uniformly sampling 5-dimensional and 50-dimensional data points. The 3D view of the distribution shows that most data points are clumped in the center area in 50D case (b) than that in 5D case (a). 56
4.3 Data points A and B have the same location in RadViz as shown in (a) and (c). However, after the sigmoid weighting operation, the obtained results are quite different as shown in (b) and (d). 57
4.4 Transformation functions of GFV and FET under different distortion factors. 59
4.5 The average distance of each point to the origin in the display decreases as the dimensionality increases in RadViz, as shown in (a). The plot also includes the average distance $\mu \pm$ one standard deviation σ (dashed lines). The graphics in (b) shows line histograms of the distances plotting 10 to 500-dimensional uniformly generated random dataset. The clumping effect becomes severe as the dimensionality increases. [1] . . . 61
4.6 (a) An example with data points located near the circle edge; (b) RadViz plotting after radial equalization. The histogram of original RadViz is plotted in (c). After histogram equalization, the histogram is plotted in (d). 63
4.7 (a) An example with data points located near the circle edge; (b) After radial specification, the RadViz is illustrated. The histogram in (c) is user-specified. After radial specification, the histogram is plotted in (d). 66
4.8 (a) The RadViz plotting after radial movement; (b) The RadViz plotting after radial local equalization. After radial movement, the obtained histogram is plotted in (c). After radial local equalization, the obtained histogram is plotted in (d). 68

4.9 We color clusters and use different symbols for different query-related data points. The original result using basic RadViz (p^{RV}) is shown in (a) while radial equalization result is shown in (b). The result of the RadViz with sigmoid function (p_s^{RV}) is illustrated in (c). The results of two distortion methods, GFV and FET, are illustrated in (d) and (e), respectively. In the end, the proposed PolarViz (p_{polar}^{RV}) is shown in (f). 71

4.10 (a) The position difference of same data point is plotted in p_s^{RV} (Fig. 4.9(c)) and basic RadViz p^{RV} (Fig. 4.9(a)). (b) The attribute values variance of a data point is plotted in p_s^{RV}. (c) The ratio of the largest attribute value to the mean attribute value is plotted in p_s^{RV}. (d) The position difference of same data point is plotted in p_{polar}^{RV} (Fig. 4.9(f)) and basic RadViz p^{RV} (Fig. 4.9(a)). (e) The attribute values variance of a data point is plotted in the PolarViz (p_{polar}^{RV}). (f) The ratio of the largest attribute value to the mean attribute value is plotted in the PolarViz (p_{polar}^{RV}). 72

4.11 The basic RadViz p^{RV} (a), p_s^{RV} (b), GFV with $\lambda_{GFV} = 9$ (c), and p_{polar}^{RV} (d) are used to plot the EngineSim dataset. 74

4.12 In (a), we plot the position difference of data points in p_s^{RV} (Fig. 4.11(b)) and basic RadViz p^{RV} (Fig. 4.11(a)). Then in (b) we compare the position difference between p_{polar}^{RV} (Fig. 4.11(d)) and basic RadViz p^{RV} (Fig. 4.11(a).) 75

4.13 We compare the transformation functions of the view in Fig. 4.9 (f) with Fig. 4.4. 76

4.14 For a 7-dimensional standard SC, the two possible vector orders from point A to point B are illustrated. 83

4.15 Determine the order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ for $\overrightarrow{v_1}$. 84

4.16 The circumscribed circle and bounding polygon of standard SC when plotting 3D, 4D, 5D, and 6D datasets. 85

4.17 Determining the bounding polygon of SC. In (a), the configuration of SC is illustrated. Then the starting point A is confirmed as shown in (b). The third step is to confirm the order of axes vectors. In (c), from the starting point A, the first axes vector is confirmed and plotted. . . 88
4.18 Determining the bounding polygon of SC by visualizing the magnitude and angle of each axes vector with a biplot map. 89
4.19 The bounding polygon, the axis-aligned bounding box, and the minimum bounding box of the 10D SC plots. 89
4.20 The smooth transition between standard SC and RadViz with fixed circumradius on the 'IRIS' dataset. 94
4.21 We plot 100,000 uniformly sampled 10-dimensional data points in SC with general configuration (a, b) and standard configuration (c, d) respectively to demonstrate the clumping effect. 95
4.22 (Top) The average distance of all the projected data points to the origin decreases as the dimensionality increases, which means that the clumping effect becomes more obvious with high dimensional data. (Middle) The distribution of the distance using 10D, 20D, 50D, 100D, 200D, and 500D datasets [2]. (Bottom) Compared with the distribution of data point distance of RadViz, SC's clumping effect is less obvious, but both of them are increasing in dimensionality [1]. 96

5.1 Transform the position of a site in rectangle cell of the treemap into the orthogonal rectangle cell as the initial position. 106
5.2 Two kinds of relative positions for sites s_i and s_j where $x_{\Delta ij} = |x_i - x_j|$ and $y_{\Delta ij} = |y_i - y_j|$. The black solid line is the segmentation based on Euclidean distance while the red solid line is an axis-aligned segmentation. The distance $dist_o(p, s_i)$ and $dist_o(p, s_j)$ in both cases are illustrated. 108

5.3 (a) One example for choosing the reference sites for certain point p in red. A square in pink centered at the point p is built, including at least one horizontal site pair and one vertical site pair. By first considering horizontal site pair s_i and s_j, and then vertical site pair s_i and s_k, point p has smallest distance to site s_i. (b) A more complex case is given in which the orthogonal Voronoi diagram is generated based on the positions of a series of random sites (in red). 110

5.4 Overview of the sweepline + skyline algorithm. (a) Initially, six sites are positioned inside the canvas based on their initial positions. (b) The skyline (in blue) and the sweepline (in pink) are initialized and the first site s_A (left-most) is activated. (c) Sweep the sweepline to meet the second site s_B and determine the relationship of s_A and s_B. Then a horizontal segmentation line L_{AB} is added to the skyline. (d) Sweep the sweepline to site s_C and determine the relationship of s_A and s_C. Since s_A and s_C are horizontal neighbors, a vertical segmentation line is generated. In this case, site s_A is closed. A bounding polygon formed by the skyline and the canvas boundary and the vertical segmentation line is built for s_A. Then the skyline is updated and site s_A is marked as closed status. (e) Sites s_B and s_C are then considered and a horizontal segmentation line is added to the skyline. (f) The process is continued until the last site s_F is considered. All the active sites will be closed and the corresponding bounding polygon will be built based on the current skyline and the canvas boundary. 114

5.5 The average number of site pairs (top) and valid neighbors (bottom) per site. Noted that the x-axis is not linear. 115

5.6 The fitting line and the prediction for the average number of valid neighbors per site. 116

5.7 The running time (in ms) for a single iteration. 117

5.8 The converge rate of the Voronoi treemap, our orthogonal Voronoi treemap with random initial status and with our proposed initial status. Within the 500 iterations, we record the area error of each iteration to indicate the converge rate. 119
5.9 The boxplots of aspect ratio. In these tests, our orthogonal Voronoi treemap (OVT) has a comparable mean aspect ratio with the treemaps (binary treemap and squarified treemap). While the Voronoi treemap has the best aspect ratio in all tests. 121
5.10 The final layouts on the random dataset. 122
5.11 The final layouts on the globalGDP dataset. 123
5.12 (a) The orthogonal Voronoi treemap visualizes the Flare class hierarchy (colored by TreeColors [3]) with random initial status. (b) The orthogonal Voronoi treemap uses the squarified treemap as initial status. . . 124

List of Algorithms

4.1 Radial Equalization 62
4.2 Radial Specification 65
4.3 Bounding polygon of standard SC 87
5.1 Compute Orthogonal Voronoi Treemap (single layer) 105
5.2 Adapt positions and weights. 107
5.3 Generate Weighted Segmentation Line 109
5.4 Compute Orthogonal Voronoi Diagram 112

Chapter 1

Introduction

Human beings can soar in the sky with the help of an aircraft. Engine is the heart of every aircraft, as it powers the aircraft to operate to the certificated standard of safety and efficiency in providing a high power-to-weight ratio. Hence, a large number of engines were designed to meet these requirements in the past and new engines are expected to be designed to provide more powerful and reliable services in the future.

With recent advancements in multiple aspects including novel materials, high-performance sensors, and high-performance computing equipment, traditional engines have a large potential to be improved to increase performance. The improvement can be made by designing a novel engine structure, utilizing novel materials, embedding more multi-functional sensors for accurately tracking the engine status, and simulating using more powerful computing equipment. These strategies provide a large amount of information to domain experts so that they can combine their domain knowledge to get insights and make decisions. However, the combination of a large amount of information and domain knowledge is not easy for domain experts, owing to a large number of datasets. It is time-consuming and inefficient for human beings to go through these datasets. Hence, information visualization comes into play.

1.1 Background

The digitalization of engine-associated procedures generates a large number of datasets, and the development of industry 4.0 speeds up this process. These generated datasets

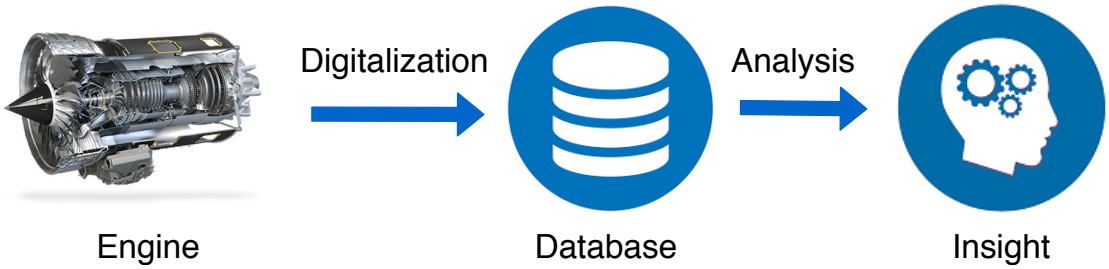


Figure 1.1. Engine-Data-Insight Model.

drive the design of the engine structure, the simulation of the engine design parameters, and the optimization of the engine design. It is impossible for domain experts to plough through these data to get useful insights. At this stage, a data analysis process that can handle these datasets will play a vital role to assist the domain experts. We consider the whole process as Engine-Data-Insight model as shown in Fig. 1.1. In the following, we introduce this Engine-Data-Insight model by first depicting several examples in the digitalization process in Sect. 1.1.1. Then, we introduce the data analysis process in Sect. 1.1.2.

1.1.1 Digitalization of Engine-associated Procedures

The digitalization happens in all the procedures, including engine design, simulation, manufacturing, maintenance, optimization, and so on. It affects different organizations, including designing institutes, production department, maintenance shops, and operating airline companies. Moreover, during this process, the generated datasets usually have different data formats. In the following, we depict several scenarios where the digitalization happens as shown in Fig. 1.2.

Engine Simulation A software simulator can be used to test various engine designs including the profile of fans and the size of other components to get useful feedback. Since this feedback can be used to estimate the performance of a real engine, simulation is quite commonly used by the domain experts. During this process, a large amount of output data will be obtained. These output data usually have high dimensionalities.

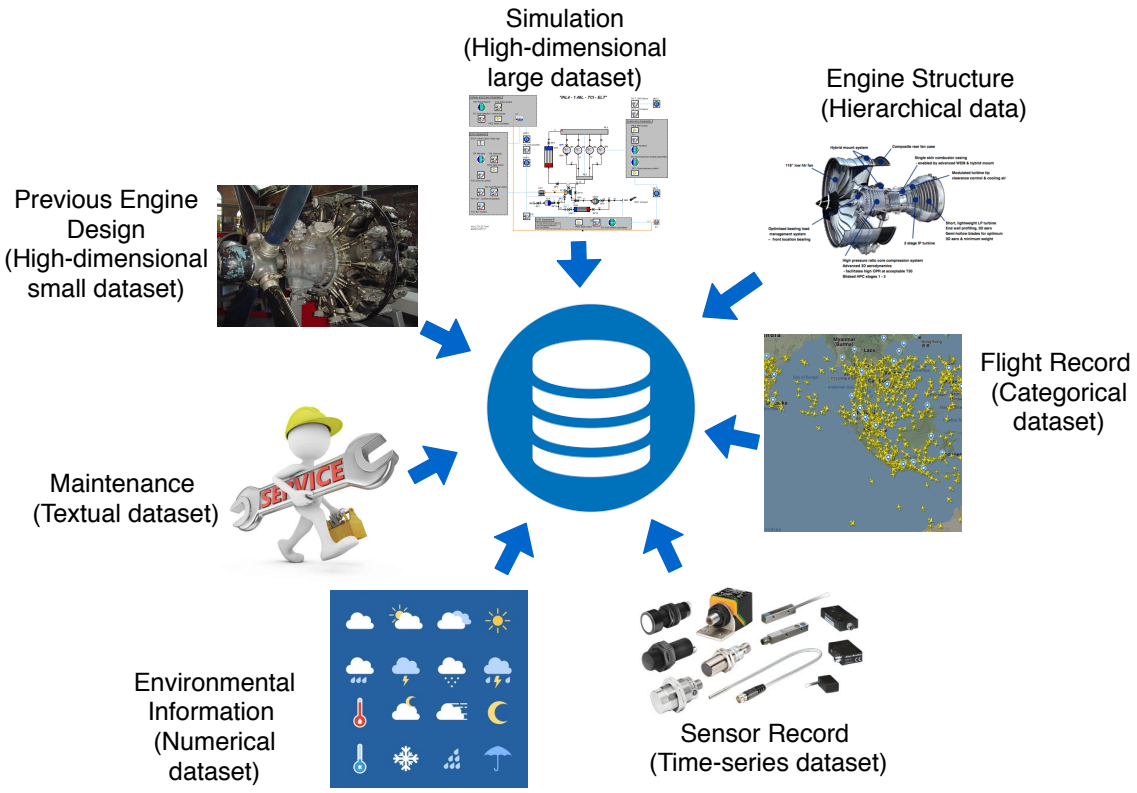


Figure 1.2. Multiple scenarios of engine digitalization.

Previous Engine Design Previous engine designs, including the design of the engine structure and the attributes of each component, are mainly based on the experience of the domain experts. The digitalization of the previous engine designs contributes to the understanding of domain knowledge and to the design of the novel engine. Since previous engine design is limited, the obtained dataset is usually small, but has multiple dimensionalities.

Maintenance Records The maintenance records of an engine record its maintenance situations, including the engine fault type. This information is extremely useful to track the maintenance history of a certain engine, as well as contribute to the design of a new engine by avoiding existing problems. These records are usually written by workers and may follow different formats. Hence, how to associate the maintenance records written by different people is a big challenge. Moreover, since the maintenance is regular, the maintenance dataset usually has thousands of records per engine.

Environmental Information Since air is essential for the operation of engine, environmental information about air, such as volcanic ash, PM10, and so on, plays a

role on the performance of the engine. The aircraft engine may be totally damaged when exposed to volcanic ash. Moreover, the air temperature and the air humidity may also implicitly affect the performance of engine. The recording of environmental information can be divided into two aspects. The ground environmental information is mainly recorded based on the departure and the arrival airports. So, the data is in the categorical format. The erupted volcano and the volcanic ash are usually recorded in the textual format.

Engine Sensor Records Recent engines are embedded with a large number of sensors to record various parameters during the operation. These records are used to track the status of the engines to identify faulty components. The output is time series data and usually have a large number of rows.

Aircraft Flight Records Aircrafts fly around the world. The flight records include the departure time and airport, arrival time and airport, flight route information and so on. This information contributes to the understanding of the engine working environment and frequency. Moreover, the flight records also contribute to the linkage of environmental information and the engine. Usually, the aircraft flight records are categorical data and stored in JSON format.

Engine Structure When it comes to the structure, an aircraft engine, which is also called gas turbine, is mainly formed by fan, compressor, combustor, turbine, and nozzle. For each part, there are thousands of components associated. Usually, these components are organized according to their position or functionality to form a hierarchy structure [4]. The hierarchy contributes to the understanding of the engine structure. Moreover, it contributes to the understanding of the embedded sensors on engine components.

1.1.2 Data Analysis of Engine-associated Data

Analysis of the engine-associated datasets aims to obtain useful insights for the domain experts. This process starts with the collection of user requirements and related

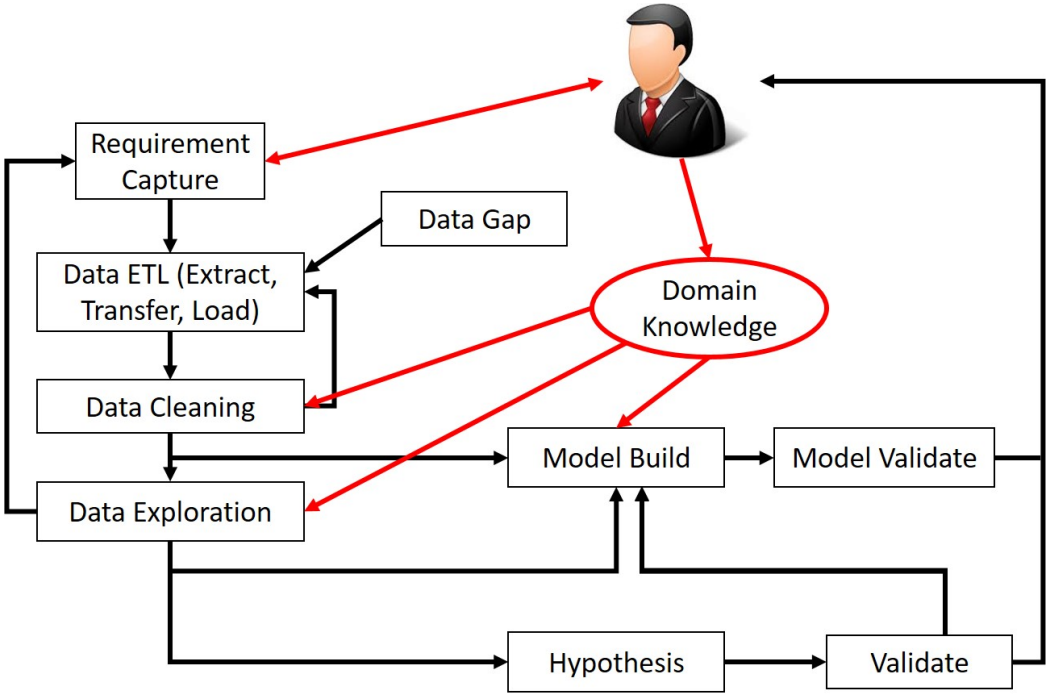


Figure 1.3. Data analysis flow chart in the computational engineering department of advanced technology group at Rolls-Royce Singapore.

datasets and provide operational suggestions at the end to the domain experts. Although many data analysis pipelines have been proposed in the literature [5], here, we describe the data analysis process in the computational engineering department of the advanced technology group at Rolls-Royce Singapore. The flow chart for the data analysis is illustrated in Fig. 1.3.

Customers requirement capture is the very beginning of the whole process. Discussion and communication with customers to collect related datasets and initially explore the dataset, then to adjust analysis target and finally to satisfy their requirement are essential and will be conducted several times during the whole data analysis process.

Among the process, data exploration contributes to the understanding of the datasets which are used in the analysis. With a better understanding of the datasets, the users firstly can determine whether additional datasets are required in the analysis. Secondly, useful hypotheses may be obtained during the exploration. Thirdly, it also contributes to the building of models. Hence, data exploration is a kernel step in the data analysis process. However, this step is usually very time consuming as the size of the datasets is large, and the formats of the datasets are complex. In

this case, proper visualization methods can help to illustrate the datasets and provide interactive operations to play with the datasets.

1.2 Visualization of Engine-associated Data

To explore the datasets, the visualization can be used for a specific analysis target. In this case, the visualization needs to integrate multiple datasets, build a linkage among them, and provide proper exploration operations such as data filtering and data querying. We consider this as **specific visualization application**. On the other hand, the visualization can be utilized to focus on a certain data format. In this case, the visualization needs to provide new visualization methods with better views rather than that in traditional visualization methods. Data exploration operations in this case include distortion, reordering, and so on. We consider this as **generic visualization method**. In the following, we give an introduction to these two kinds of visualization.

1.2.1 Specific Visualization Application

The specific visualization application is also known as the design study, which is a form of problem-driven research. The goal of a design study is to solve the specific real-world problem by analyzing the problem, designing a visualization system to solve the problem, validating the design, and finally reflecting on lessons learned [6]. According to the methodology proposed in [6], the design of a specific visualization application can be organized into three categories: a precondition phase, a core phase, and an analysis phase.

In the precondition phase, the main target is to prepare the visualization application. The first crucial step is to have solid background knowledge of the specific real-world problem to be solved. This needs a promising collaboration with domain experts to increase design efficiency. Meanwhile, background knowledge of the visualization literature is also required in this phase. The research contribution of this phase is the characterization and abstraction of the specific problem. This will contribute to the shared understanding of the problem between researcher and domain experts.

Moreover, it also contributes to the conduct of the whole design process, as well as the evaluation.

In the core phase, the visualization application system is designed, implemented, and evaluated. Based on the preparation in the precondition phase, the researchers abstract the datasets used, and design the visualization system including visual encoding and interaction operations. Once the design is confirmed with the domain experts, implementation of the visualization system will be started by creating software prototypes and choosing the right algorithm to meet the design requirements. After that, the developed system will be evaluated to find out whether it can indeed help the domain experts to solve the target problem. The research contribution of this phase is the visualization system. It should be noted that the visualization system needs to be appropriately validated with evidence that it does assist the domain experts to solve the problem.

In the analysis phase, reflection on how the designed visualization system relates to the research community is done, so that it can be properly added to the body of knowledge and other researchers can benefit from this work. The reflection is also treated as the third type of research contribution of specific visualization application.

1.2.2 Generic Visualization Method

Different from the specific visualization which is problem-driven research, generic visualization is technique-driven research, where the goal is to design novel visualization techniques [6]. The full spectrum of techniques for visualizing data is categorized based on the information visualization pipeline enriched with customized action-driven classifications [7]. There are three main transformation steps: data transformation, visual mapping, and view transformation.

Data transformation is the process of converting data from its original to graphics primitives. It corresponds to data cleaning and analysis-centric methods including dimensionality reduction, regression, clustering, feature extraction and so on.

In the visual mapping step, the analysis result or the original dataset from the data transformation stage is converted into visual structures based on various visual

encoding. Visualization methods based on axes, glyph, pixels and hierarchical representations are the key parts in this stage. Moreover, evaluation of the effectiveness of visualization methods is also discussed.

The view transformation stage directly generates images in the screen space. Hence, as indicated by Bertini et al. [8], the view transformation stage can be treated as the rendering process for the final visualization results. Research fields in this stage include illustrative rendering, color blending, and continuous visual representation.

1.2.3 Challenges of Engine-associated Data Visualization

To achieve an efficient and effective data visualization, there are a few challenges to be addressed.

- **Build the linkage among heterogeneous datasets:** Instead of single data format, datasets with different data format are required to solve an engine-related problem. To discover the insight, properly building the linkage via visualization is challenging. To achieve this, not only the visual encoding but also the data model is required to build the visualization system.

- **Design data exploration operations:** Although data visualization can provide views to the domain experts, exploration operations are still expected to provide more flexibility to the system. These exploration operations usually need the change of the input data of the visualization. In this case, a properly designed interchange strategy between the views (frontend) and the database (backend) is required to support the exploration operations including data filtering, data querying, and so on.

- **Design visualization exploration operations:** For the generated views, visualization exploration operations can modify the existing views to get a clearer understanding. These operations handle the generated views, rather than the data exploration operations which focus on the visualized dataset. Although many visualization exploration operations have been proposed in the literature, more flexible operations are still emerging into this field to satisfy customized

requirements.

- **Propose novel visualization methods:** Nowadays, many accepted data visualization methods are well accepted by the users to generate views for certain data format. Meanwhile, more powerful methods are still sought after in the field. Some researchers focus on the theoretical study of existing methods and then make modifications to overcome their drawbacks. While other researchers try to merge different visualization methods in order to utilize the advantages of them.

1.3 Contributions and Outline of Thesis

1.3.1 Contributions of Thesis

The objective of this thesis is to visualize the engine-associated data generated during the process of engine design, simulation, and optimization, such that the domain experts can benefit from the designed visualization. Based on this, the main contributions of this thesis are summarized as follows:

- We propose a web-based visualization system in this thesis to provide the domain experts with a qualitative analysis of the possible external cause of engine failure. The system features a dynamic query on the engine-associated datasets and incorporates several customized interactive visualizations. The system can explore the effect caused by environmental factors, such as air pollution and volcanic ash. Moreover, via visual comparison, the proposed visualization system can identify potential external factors.

- We propose a focus+context view distortion strategy, termed PolarViz, for high dimensional data visualization. PolarViz manipulates the radial distribution of data point in RadViz plot to satisfy the user requirement. By using PolarViz, the clumping effect of projected points toward the origin in RadViz can be significantly eased. Computational experiments show that PolarViz not only preserves

the advantages of RadViz but also provides a flexible radial modification on the views.

- We design a geometric algorithm to find the boundaries of the star coordinate plot for high-dimensional data visualization. These boundaries not only provide a global view of the plot, but also contribute to the general expression of the star coordinate with arbitrary configuration for subsequent plotting and customized window size. Moreover, based on these boundaries, we reformulate the relationship between RadViz and standard star coordinate. Furthermore, these boundaries help to analyse the clumping effect of the standard star coordinate plot.

- We propose a novel space partitioning strategy for implicit hierarchy visualization such that the new plot not only has a tidy layout like the treemap, but also is amendable to data changes similar to the Voronoi treemap. We first define a novel distance function based on two sites so that the segmentation is axis-aligned. Second, we design a sweepline+skyline algorithm for space partitioning with $O(n \cdot log(n))$ complexity. To the best of our knowledge, this is the first time the sweep line strategy is used for the Voronoi treemap. Third, we design a novel strategy to initialize the diagram status and modify the status update procedure to increase the efficiency and effectiveness of our algorithm.

1.3.2 Outline of Thesis

The thesis is organized into six chapters.

In Chapter 1, the engine-data-insight model is introduced first. Then the visualization of engine-associated data is categorized and introduced. Moreover, we discuss the challenges encountered during the visualization.

In Chapter 2, a literature review on existing visualization methods which relate to the work of this thesis is given. Firstly, visualization applications on geo-temporal data is discussed. Then, visualization methods on high-dimensional data, and hierarchical data, in the literature are introduced.

In Chapter 3, the proposed EngineQV visualization system is presented. The target of the visualization application is first studied as well as its design requirement. After that, the data design, architecture design, query design, and visual design of the system are described, respectively. The system is evaluated by use case study and discussed by the domain experts.

In Chapter 4, novel visualization methods for high-dimensional data are presented. We first propose a focus+context distortion approach, termed PolarViz, to manipulate the distribution of data points in RadViz. The radial operations, including radial equalization, radial specification, radial movement, and radial local equalization, are described. Comparison with existing distortion methods on an engine simulation dataset and a user study to assess the performance of PolarViz are conducted. Second, we propose a geometric algorithm to determine the axis-aligned bounding box, minimum bounding box, and bounding polygon of the star coordinate with various configurations.

In Chapter 5, a novel visualization method, orthogonal Voronoi treemap, for hierarchical data is presented. The new distance function and neighborhood relationship are given first. Then a sweepline+skyline based heuristic algorithm to partition the space is described. Moreover, a novel initialization strategy and modified status update procedure are also described. In the end, the performance of the proposed orthogonal Voronoi treemap is evaluated in terms of computation time, converge rate, and aspect ratio.

In Chapter 6, the works in this thesis are concluded and the directions of future work are given.

Chapter 2

Literature Review

In this chapter, data visualization methods related to our works are reviewed. According to the data types associated with our works, this chapter is divided into three subsections. We first discuss related works about geo-temporal data visualization since the proposed EngineQV focuses on handling geo-temporal datasets. After that, we discuss visualization methods for high-dimensional data, focusing on the RadViz and the star coordinate plot. In the last section, we discuss visualization methods for hierarchical data by considering explicit and implicit hierarchy representations.

2.1 Geo-temporal Data Visualization

In this section, we discuss the previous research work which is closely associated with geo-temporal data visualization. We consider three sub-categories: map-oriented trajectory visualization, temporal event data visualization, and visual comparison used in geo-temporal data visualization.

2.1.1 Map Oriented Trajectory Visualization

The map by Charles Joseph Minard, which portrays the losses suffered by Napoleon's army in the Russian campaign of 1812, probably is the most famous map-oriented trajectory visualization ever drawn [9] (Fig. 2.1). Starting from the Polish-Russian border, the path of Napoleon's march is depicted visually by banding with different widths to indicate the size of the army at each position.

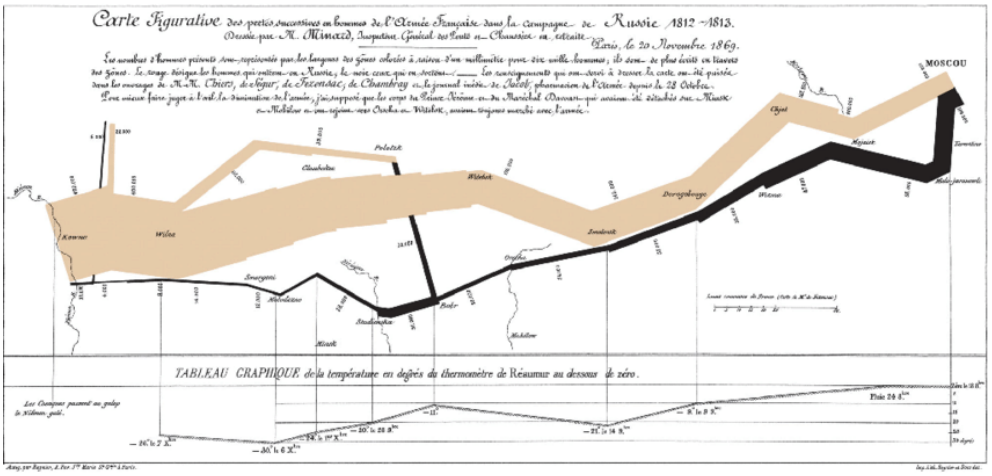


Figure 2.1. The path of Napoleon's march in the Russian campaign of 1812.

Nowadays, map-oriented trajectory visualization has been extensively studied to not only support transportation management [10–12], but also contribute to solving urban-related problems including jam monitoring [13], route recommendation [14], and urban planning [15]. Both in the composite density map [16] or the aggregation flow [17], an accurate and intuitive association with the map is expected especially for route-related management and visualization. Different from these visualizations which focus on the trajectory, our study focuses more on the factors in a geographical area. Hence, we provide an approximate flight route information.

2.1.2 Temporal Event Data Visualization

Temporal data are ubiquitous in many industries to record equipment status and events. Generally, temporal data can be divided into time series data and temporal event data. In this design study, we focus on the visualization of the temporal event data, as the related information is recorded as an event with a time stamp.

Timeline is the most popular visualization method to depict temporal event data. Multiple interactive visualization designs have been proposed to visualize temporal event data based on timeline, such as the TimeSlice [18], the TimeLineCurator [19]. The Gantt chart which illustrates a project schedule in project management can be treated as a form of timeline plot. We adopt the timeline plot to visualize the events with a timestamp in our design. As the records in the datasets are not strictly consistent, information extraction, which is a form of natural language processing

strategy, is required for parsing before dumping it into a database. Specifically, we parse the date information of a record to form a temporal event and then illustrates it in the timeline plot.

2.1.3 Visual Comparison

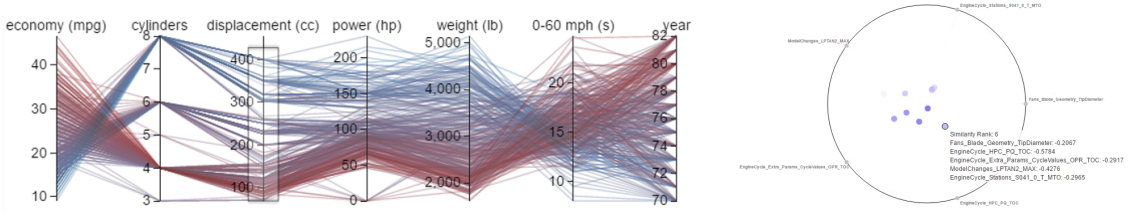
Visual comparison is one of the most common tasks in most visualization designs [20]. The taxonomy of comparative design observes that the range of visual designs for comparison falls into three categories: juxtaposition (i.e. next to each other in either time or space), superposition (i.e. on top of one another), and explicit representation of the relationships [21]. Various visualization designs have been developed by utilizing one of these three categories or their combination [7,22,23]. We adopt both the juxtaposition and the superposition comparison strategies in our design, allowing the user to conduct direct comparative analysis from different aspects. To be specific, for the purpose of comparing the geographical information, we mainly adopt the superposition comparison as it associates with the multi-layer map plot very well. Meanwhile, when it comes to the comparison of some statistic information, the juxtaposition comparison is selected in our design.

2.2 High-dimensional Data Visualization

Visual analysis on the high-dimensional dataset is a big challenge that is known as 'curse of dimensionality' [2]. To visualize high-dimensional dataset, one straightforward choice is to implement dimensionality projection techniques and then plot them in a lower dimension. While the other is to display all the dimensionality information at one time.

Dimensionality reduction that maps high-dimensional data point onto low-dimensional space (two-dimension or three-dimension) is crucial in the high-dimensional information visualization. There are two distinct groups in dimensionality reduction depending on the projection methods: linear projection and non-linear projection. The most popular linear dimensionality reduction methods include Principal Component Analysis [24], and Linear Discriminate Analysis [25]. While for the non-linear case,

Multidimensional Scaling [26], Isomap [27], and Local Linear Embedding [28] are well-known methods adopted to reduce dimensionality. Researchers in the field of machine learning and artificial neural network have also developed methods to present data in low dimensions, such as t-SNE [29, 30].



(a) Parallel Coordinates Plot (b) RadViz

Figure 2.2. High-dimensional Data Visualization.

Besides these traditional dimensionality reduction techniques, some visual mapping approaches also can be used to plot high-dimensional dataset onto low dimension. A fast and straightforward visualization method is Scatter Plot Matrix (SPLOM), which is formed by all possible combinations of bivariate scatterplots. However, scalability is the primary drawback of SPLOM due to the quadratically increase of bivariate scatterplots with respect to the increase of dimensionality. To avoid this problem, parallel coordinates plot (PCP) [31](as shown in Fig. 2.2(a)), which can show all the axes at once is considered. Quite similar to PCP, heatmap [32] also plots all parallel axes at once, but uses color instead to represent value. However, one challenge for PCP and heatmap is determining the order of the axes. RadViz (RV) [33](as shown in Fig. 2.2(b)) and Star Coordinate Plot (SC) [34], which adopt a radial pattern, are well studied visual mapping approaches in recent years. A recent work [1] shows that SC has greater flexibility in visualization compared with RV. However, they share the same challenge with PCP and heatmap. Moreover, for RV and SC, there exists a clumping effect that forces points towards the center of the plot as dimensionality increases [2]. Many criteria have been proposed to judge the quality of visualization when using the methods mentioned above. However, none of them consider utilization for visualization. When mapping high-dimensional dataset into low dimension, obviously, it is not the expectation that most of the data points are mapped into a small region while leaving majority of the space empty. If it ever happens, then it can be

treated as a visualization with low utilization ratio. For SPLOM, PCP and heatmap, they have full utilization when plotting uniform distributed high-dimensional dataset. However, for RV and SC, due to the clumping problem, their utilization ratio decreases as the dimensionality increases.

2.2.1 RadViz

In RadViz, all the dimensionalities of a data point in data space contribute to the final position of the projected data point in the 2D image space. In the basic RadViz, dimensional anchors (DAs) are evenly spaced around the perimeter of a circle. One end of a spring is attached to each dimensional anchor while the other end is attached to a certain data point. The number of springs is equal to the data dimensionality. The equilibrium location of this data point in the RadViz spring system is where the sum of each spring force equals to zero. RadViz is well-explored in terms of cluster representation, outlier detection and so on. The main challenge for RadViz is the placement of dimensional anchors. Various algorithms have been proposed to place dimensional anchors to obtain desired configurations, followed by quality measurement methods for justification [35–39].

Some researchers focus on designing 3D RadViz with the purpose of getting a better configuration [40–42]. In [40], they used the mean value of each dimension as the third coordinate value while in [41] the Euclidean distance of the projected data point to the origin in the high-dimensional space was used as the third coordinate. The 3D RadViz visualization scheme incorporates a third dimension to visualize the shape and convergence by using the distance to a reference hyper-plane. Their 3D RadViz can effectively visualize the Pareto-optimal fronts with more than three objectives and also can be used to evaluate the performance of an algorithm.

Some researchers focus on solving the clumping problem in RadViz, in which a large amount of data points are crowded into a small region. To provide a detailed view of this crowded data region while retaining surrounding context to help keep analysts oriented is exactly a focus+context method [43, 44]. Distortion is one of the focus+context techniques that can transform the display region so that focused regions

are magnified while contextual regions are demagnified. The application of distortion-oriented techniques to data visualization has a relatively long history. The problem arises when using a small display window to view large information systems [45]. This is quite similar to the clumping problem in RadViz. Trials of the distortion techniques include the polyfocal display [46], bifocal display [47], perspective wall [48], and graphical fisheye views [49]. Among them, the fisheye distortion is the most commonly used method due to the use of cameras with wide fields-of-view in computer vision applications. Many polynomial and non-polynomial models of fish-eye radial distortion are proposed to simulate the distortion [50]. The detail can be found in the survey papers [45, 50] and reference therein.

2.2.2 Star Coordinate

Star coordinate (SC) [34] adopts a radial pattern for data analysis. It has emerged as a promising visual mapping approach in recent years. By transforming data point with a combination of radial-distributed axes vectors, SC linearly maps data points with high computational performance.

The main disadvantage of basic SC is overlapping projected data points where different data points in high-dimensional space may be projected close together in SC. To solve this problem, techniques including interaction techniques to control the axes vectors [51], orthogonal projections [52] and heuristics algorithm for dimension ordering [53] are proposed. The key point of this method is to find the desired configuration of these radial-distributed vectors $\overrightarrow{v_i}$. Actually, this general SC with arbitrary configuration is more commonly used in practical applications to generate visual representations.

Star coordinate plotting not only aims to provide an overview of high-dimensional datasets but also often is integrated into the data analysis process with exploratory purpose including cluster representation [34, 35, 54–56], outlier detection [1], data attribute estimation [57]. As SC is a linear projection, a bunch of linear dimensionality reduction methods can be combined with SC to get a better visual interpretation of the clusters in high-dimensional space. Popular methods including principal component

analysis [24], linear discriminate analysis [25, 58], large margin nearest neighbor [59], and neighborhood component analysis [60] can be reproduced by choosing appropriate axes vectors. These have been successfully implemented in order to preserve distance and separate class or clusters in SC [1]. Besides reproducing classification results, optimal sets of axes vectors have been studied to get maximal insight [61] and trial to improve data attribute estimation in SC by implementing data centering is undertaken in recent [57]. Essentially, the kernel is to calculate the magnitudes and angles of radial-distributed axes vectors based on certain exploratory purpose. With different configurations of axes vectors, different insight will be obtained and the range of the plotting may also be different. To enhance the capability of visualization, some works also focus on 3D visualization method [40, 42, 62].

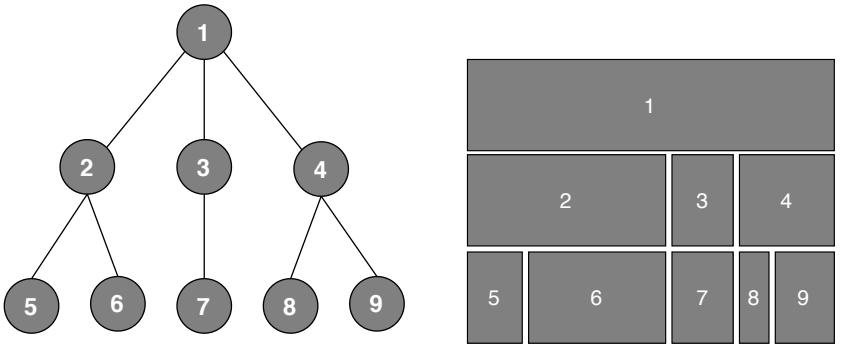
2.3 Hierarchical Data Visualization

Hierarchical data structure exists widely in engine-related datasets including the structure of the engine and the engine-airplane relationship, and in our daily life including file system, software/package class structures, and the organization structure of companies. Thus, a large number of hierarchy visualization methods have been developed to depict the dataset from different aspects [20]. Some of them explicitly show the hierarchical structure as straight lines, arcs, or curves, while others focus on the value within each node and positionally encode the hierarchy by node overlap or inclusion [63]. Hence, in the following, we divide the hierarchical data visualization methods into two categories: explicit hierarchy visualization and implicit hierarchy visualization.

2.3.1 Explicit Hierarchy Visualization

In this section, we give an overview of explicit hierarchy visualization methods. Explicit hierarchy visualization methods use connection diagram, such as node-link diagram, to visualize the parent-child relations of nodes in the hierarchy. Hence, it is easy for the user to capture an immediate perception of the hierarchy, although it is not efficient in display space utilization compared with the implicit hierarchy visual-

ization [64]. According to the connection methods, we divide the methods into two clusters: node-link connection and touch connection. Both of them place nodes on a set of layers in the display area, and nodes from the same level in the hierarchy are placed on one unique layer.



(a) Node-link Connection (b) Touch Connection

Figure 2.3. Explicit Hierarchy Visualization. (a) Node-link connection. (b) Touch connection.

2.3.1.1 Node-link Connection

The node-link connection exists commonly in the explicit hierarchy visualization methods, such as the outline diagram and the tree diagram as shown in Fig. 2.3(a). In these methods, each node is represented by a symbol, such as a circle. They are then placed layer by layer and linked by lines or curves to indicate the hierarchy. The Reingold-Tiford algorithm, which is the first linear time tree layout algorithm proposed by Reingold and Tilford in 1981, is one of the most famous tree layout algorithms [65]. In this algorithm, the sub-hierarchies are recursively drawn independently in a bottom-up direction. After all the nodes have their positions fixed, links are then inserted accordingly. Later this algorithm is modified to make a variation for general trees without violating any of the aesthetics [66] as well as a further modification with linear runtime [67]. Nowadays, many researchers also focus on designing 3D tree layout algorithms to provide a more intuitive view [68–70].

Besides placing nodes on several parallel lines, radial distribution is popular in recent study. By giving more freedom on the placements of nodes, radial distribution can achieve improvements in terms of angular resolution, space utilization, aspect ratio, running time and so on, especially in hyperbolic space [71–75]. These works

focus on the design of the radial tree layout and the view operations including the navigation, traversal, discovery and interactive manipulation of information stored in large hierarchical structures [74, 75]. Since the radial distribution is more space efficient, focus+context distortion is often designed to assist the user to zoom in/out of the structure [73]. Moreover, comparison between two tree layouts is also studied [23].

2.3.1.2 Touch Connection

Famous explicit hierarchy visualization methods that use touch connection are icicle plot [76] and sunburst [77]. In these methods, each node is represented by a region, such as a rectangle or an annulus as shown in Fig. 2.3(b). The sub-hierarchy of the parent node is drawn beside the parent region with smaller area. Children nodes touch their parent node in the plot, such that the hierarchy is obvious.

Pythagoras tree is another example of explicit hierarchy visualization method using touch connection. Pythagoras tree is a fractal technique depicting a binary hierarchy as branching squares. Every sub-hierarchy creates a right triangle and then the Pythagorean theorem is applicable to the areas of the squares created. Later, a generalized Pythagoras tree is proposed to n-arily branching structures for depicting information hierarchies [78]. Instead of triangles in the traditional Pythagoras tree, the generalized layout recursively produces a convex polygonal shape where the corners are placed on a semi-circle. Moreover, the size of the created rectangles can be used to encode other numerical information, such as the associated values and the number of children nodes.

2.3.2 Implicit Hierarchy Visualization

In this section, we give an overview of implicit hierarchy visualization methods. We mainly focus on the canvas subdivision strategies used to generate layouts, instead of including all implicit visualization techniques. Based on whether the sites are referred to during the subdivision, we divide the methods into two clusters: non-site-based methods and site-based methods. Both belong to methods with inclusion edge representation according to the design space definition [79].

2.3.2.1 Non-site-based Methods

Implicit hierarchy visualization methods that partition the whole space without considering the sites are treated as non-site-based methods, such as the treemap. These methods position the data by following some rules or experience in order to get expected configurations, which sometimes are also named as heuristic-based algorithms. Starting from the proposal of the original treemap in 1992 [80], many variants are proposed in the literature [20, 81].

The squarified treemap focuses on the emergence of thin, elongated rectangles in the standard treemaps and presents a new subdivision method such that the resulting rectangles have a lower aspect ratio [82]. The ordered treemap layout is the first type of treemap layout that takes stability into consideration [83]. In their work, two pivot based algorithms (pivot-by-size and pivot-by-middle) are proposed to ensure that items near each other in the original data will be near each other in the final layout. The split algorithm used in the ordered and quantum treemaps [84] is a modification of the squarified treemaps, following a given one-dimensional ordering. The spiral treemap positions the one-dimensional ordering of the input data along the border following a circular arrangement or an S-shape [22]. Different from previous methods, which only consider one dimension, the spatially order treemaps consider two-dimensional consistency by relating node order to Euclidean distance from the parent nodes top-left corner [85]. We observe that the layout generation problem in treemap is quite similar to the two-dimensional (2D) bin packing, which is an optimization problem with a wide range of applications in resource management. This observation was also mentioned by Schulz et al. [79]. Since many heuristic algorithms [86, 87] have been proposed to solve the bin packing problem, how to utilize them into the layout generation in treemap would be an interesting research direction and we find that some researchers have started to do this [88, 89].

Non-rectangular treemaps are also described in the literature. Jigsaw map has nicely shaped regions and stable layout by considering Hilbert curves or H curves [90]. They generate irregular shapes which are not easy to be compared with. A modification then was proposed by splitting the space into rectangles [91]. To relax rectan-

gular constraint, angular treemaps employ a divide-and-conquer method to partition the space into various shapes [92]. Beside that, the treemap layout that produces irregular nested shapes by subdividing the Gosper curve [93] was also proposed.

2.3.2.2 Site-based Methods

Some implicit hierarchy visualization methods partition the space based on a series of pre-defined sites, such as the Voronoi treemap. The Voronoi treemap was originally presented by Balzer et al. [94]. By relaxing the constraint of rectangular shapes, they utilize Voronoi tessellations to generate polygonal subdivisions. They firstly initialize a set of sites with initial weight values and then compute the Voronoi tessellations based on distance functions. By adaptively altering the weight value of each site, it enables a dedicated Voronoi region in the next iteration step. Finally, the computation will be stopped when a good enough layout is reached. Later, the Voronoi treemaps are utilized to visualize dynamic hierarchical data owing to its adjustment ability [95,96]. However, the calculation of these Voronoi treemaps is computationally expensive as a random-sampling strategy is used to compute the Voronoi tessellations. In 2012, Nocaj and Brandes [97] proposed a resolution-independent algorithm by calculating the Voronoi tessellations with power diagrams, such that the new algorithm is faster in both theory and practice. An improvement is then made by setting an initial position for visualizing varying hierarchies [98].

Neighborhood treemap (Nmap) [99], which successively bisects a set of pre-defined sites on the horizontal or vertical directions and then scales the bisections to match the value of each site, is also a site-based method. Although no distance function is used during the segmentation, Nmap also needs sites representing the similarity relationships of data elements to be positioned in the canvas. Thus, Nmap can preserve similarity relationship among data elements very well. However, no evidence shows that Nmap can produce stable layouts with dynamic data. Circle packing [100], as well as the recently proposed bubble treemaps [101], can also be treated as a site-based method since the generation of the layouts is based on the center of each circle.

Chapter 3

EngineQV: Investigating External Cause of Engine Failures Based on Geo-temporal Association

3.1 Introduction

A healthy and reliable engine powers the aircraft to operate to the certificated standard of safety and efficiency. Therefore, it is essential to guarantee the effectiveness of the engines. Identifying the reason for engine failures can be challenging as many factors contribute to the damage of an aircraft engine. To be specific, turbine engine failures can be caused by mechanical problems in the engine itself after certain cycles of flights, or by entirely external factors, such as the environment temperature and the air quality.

Many efforts have been taken to ensure the effectiveness of the aircraft engine. A large number of sensors are available for real-time recording and tracking. These sensor data not only can be utilized for engine simulation [102], but also contribute to the development of aircraft engine prognostic and health management systems. These systems aim to provide an on-line diagnostic and also track the engine health condition over the lifetime by utilizing an off-line trend monitoring algorithm [103–105]. Nevertheless, as increasing research interests are focused on the diagnostic and prognostic algorithms, less research work centers on finding the external cause of

these faults, especially the environmental factors. Many factors, either internal or external, contribute to the aircraft engine's durability and lifespan. As some factors are implicit, exploration of the potential impact factors is directionless and time-consuming. Big data and operational experience can help to make exploration more efficient [106]. The task requires closely coupled human and data analysis, where information visualization comes to play [107].

In this work, we aim to assist domain experts to explore the external factors that affect the aircraft engine performance. We attempt to achieve this by integrating datasets of different formats and providing various exploration strategies and visual illustrations. Guided by the nine-stage design study methodology [6], we first conduct a detailed characterization of the tasks, and abstract design requirements via a background knowledge study. Based on the task analysis and requirements, we develop an integrated visualization system to show the data in a single-page web application. The visualization system utilizes **Engine**-associated datasets and presents multiple visualization forms across various aspects, allowing the user to **Q**uery and **V**isualize different data clusters for further exploration. Hence, we refer it as **EngineQV**. The EngineQV is capable of querying, visualizing and comparing geographical and temporal datasets. Domain experts may benefit from the system by querying engine records and spotting potential issues in multiple plots. To the best of our knowledge, no visualization work on the engine fault has been published. During the software development life cycle, we work closely with domain experts and develop the system iteratively. Furthermore, use case studies and expert feedback are described for the evaluation of the system performance. The content of this work has been published. [1] [2]

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

- We characterize the challenges in finding the potential external factors on the aircraft engine failures and conduct a thorough discussion on the design require-

[1] Yan-Chao Wang, Qian Zhang, Feng Lin, and Hock-Soon Seah. EngineQV: Investigating External Cause of Engine Failures Based on Geo-temporal Association. In the *IEEE Pacific Visualization Symposium* (PacificVis), IEEE, 2019. [108]

[2] Qian Zhang, Chung Soo Ahn, Jigang Liu, Yan Chao Wang, Feng Lin, Hock Soon Seah, Interactive geo-temporal data visualization for aircraft engine, In the *International Workshop on Advanced Image Technology* (IWAIT), SPIE, 2019. [109]

ments for fulfilling the challenges.

- We develop an integrated visualization system for aircraft engine-associated data to support visual-assisted sense-making and use a suite of visualization techniques with customized features to help the user analyze heterogeneous data without a specific requirement of programming skills.

- We showcase an experience of working with domain expert from industry to process the raw data records, iteratively design a visual analytic system, and then evaluate the system through case studies and user interviews.

The rest of this chapter is organized as follows. We analyze the task of the visualization system in Sect. 3.2. Based on the task analysis, we introduce our EngineQV in Sect. 3.3 including data design (Sect. 3.3.2), architecture design (Sect. 3.3.3), query design (Sect. 3.3.4), and visual design (Sect. 3.3.5). The performance of the EngineQV is evaluated in Sect. 3.4 by use cases (Sect. 3.4.1) and expert feedback (Sect. 3.4.2). Finally, a conclusion is made in Sect. 3.5.

3.2 Background Study

We follow a user-centered design process and get domain experts involved in every stage of our software development life cycle. At the requirement capture stage, we conduct background knowledge observations and interviews with the domain experts in order to clarify the user's needs. Analysis on the task is conducted in Sect. 3.2.1 while the requirements are summarized in Sect. 3.2.2.

3.2.1 Task Analysis

The task to explore the external cause of engine failures is challenging. We follow a user-centered design process and get domain experts involved in every stage of our software development life cycle. The domain experts are from the Rolls-Royce company. They are responsible to collect information from multiple internal departments and identify the problems about engines.

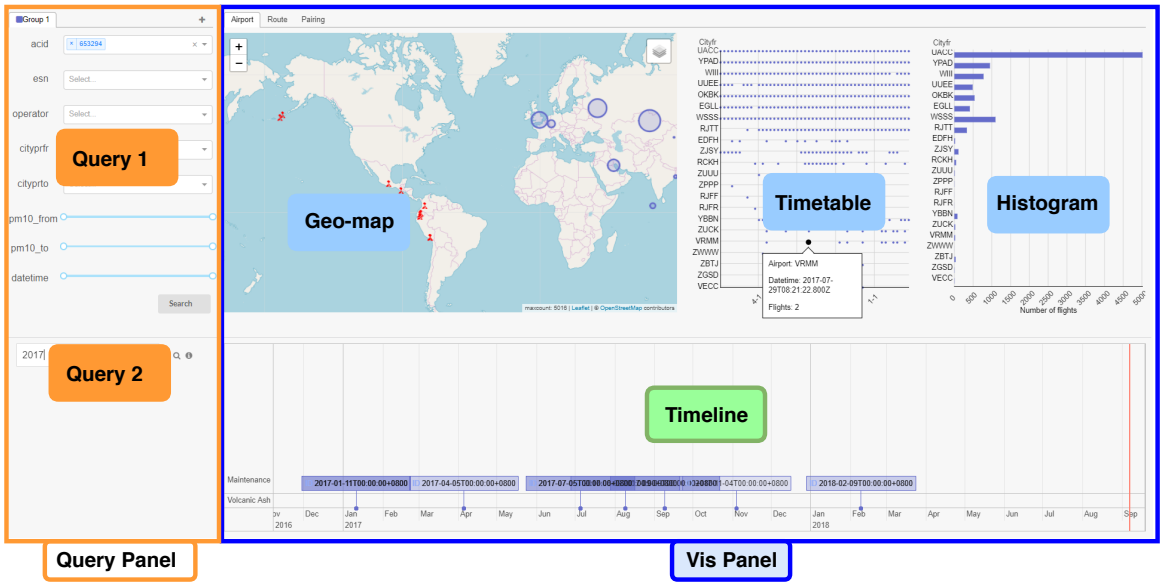


Figure 3.1. The user interface of EngineQV.

To explore the factors, the first challenge is how to handle various datasets despite the heterogeneous data forms. In their daily work, they receive textual data from the maintenance department while some tabular datasets from other departments. Though current commercial software can visualize each kind of dataset, a system integrating these datasets is expected to query on the heterogeneous datasets (**T1**). Moreover, the experts expect that the heterogeneous datasets are properly visualized for intuitive understanding (**T2**).

The second challenge is that even the experts are uncertain about what kind of external factors can cause engine fault and what kind of data should be collected for analyzing. Currently, based on their experience, immediate exposure to volcanic ash require increased maintenance as well as the prolonged exposure to particles. Hence, the proposed visualization system should not only be able to depict whether the engine is affected by these two aspects (**T3**), but also show where and how often the exposure is (**T4**).

Despite displaying existing information, the system should be able to help users to get new insights. This is the third challenge. The common way to do this is comparing the pairing engines in an aircraft, as mentioned by the experts. To achieve this, our system should allow grouping and comparing (**T5**).

3.2.2 Design Requirements

Based on the task analysis above, we propose features of our system and use them as requirements to guide the design and development of the aircraft engine-associated data visualization system:

- **Heterogeneous data operation.** The system should support data query on the various datasets despite the heterogeneous data forms. To meet this requirement, data understanding and specific data model design are expected. Moreover, the system should be devised to communicate the frontend and the backend by collecting query request and returning wanted data. (**T1**)

- **Dynamic query control.** The user should be able to adjust constraints for query dynamically, intuitively, and flexibly. The particular values for a query request should be set intuitively, such as selecting values in drop-down lists, typing keywords in search bar, or setting ranges by scrolling value bars [110]. Meanwhile, the query criteria should be designed on multiple aspects to cover the whole database. (**T1**, **T4**, **T5**)

- **Visualization of the geo-temporal relationship.** Datasets from different sources are integrated into the system. One of the challenges is to find the linkage among the heterogeneous data, i.e. the environment information with geographical coordinates, flight records with both airports and time, and the timestamped engine maintenance records. To provide an intuitive exploration and understanding, a proper visualization strategy is required to show all the datasets. (**T3**, **T4**)

- **Various visual representations.** Besides the geo-temporal relationship visualization, versatile visualization forms are also necessary for presenting the multiple facets of each of the heterogeneous datasets. Detailed information may be displayed upon user's request, in the form of tooltips when the mouse hovers. (**T2**)

- **Data grouping and comparison.** Identifying the causes of engine failures can be difficult, especially with the ambiguous relations among external factors.

Fortunately, the expert user knows the expected good engine life cycle, so the comparison across groups of different engine status and environmental conditions is essential for gaining an insight. Hence, the system should enable the user to create multiple groups and make comparisons. Several criteria contribute to the group building, including the hierarchical organization of airline identifiers, the departure and arrival airports of the flights, and some other environmental information. (**T5**)

- **System scalability.** The system design should be made flexible and allow rooms to accommodate new features and functionalities. For example, adding and visualizing new sources of data may be done without much modification. This design requirement is crucial as there are a large number of external factors and additional datasets and visual representations may be added to the system. (**T1**)

3.3 EngineQV

3.3.1 Overview

The EngineQV system is designed to achieve the above-listed tasks. The system takes the form of a single-page web application, which consists of a user interface (UI) in the frontend and two query engines in the backend. A screenshot of the system UI is shown in Figure 3.1. The dashboard consists of two panels, a query panel on the left and a visualization (Vis) Panel on the right.

In the dashboard, the user can set certain constraints and generate groups in the query panel (**T1, T5**) and the corresponding query engine will return the requested data for visualization in the vis panel (**T2**). Several kinds of visual representations are displayed in the vis panel, including a geo-map showing the geographical flight and environment information (**T3**), a timetable and a histogram showing statistical patterns of flight records (**T4**), a timeline presenting the temporal information of engine maintenance (**T1, T2**).

In the backend, we deploy two query engines for querying the heterogeneous

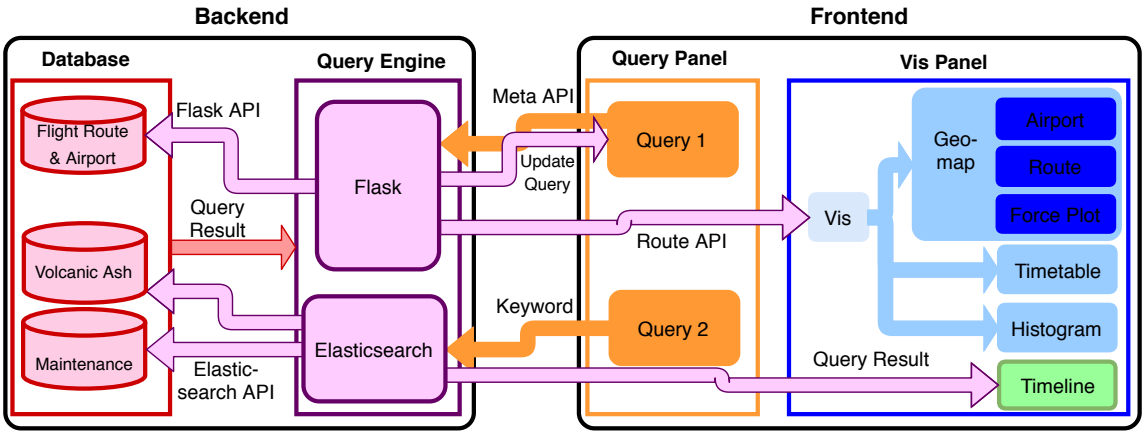


Figure 3.2. EngineQV comprises a backend (i.e. database and query engines) and a frontend (i.e. query panel and vis panel). The three datasets are stored and handled by query engines in the backend. In the frontend, the query panel allows the user to communicate with the backend and the returned query results will be passed to each visualization component.

datasets (**T1**). The numerical and categorical data are handled by the Python-based Flask server. The textual datasets are loaded onto Elasticsearch, a query support engine for text data.

Three datasets are used in EngineQV: the flight route&airport dataset, the volcanic ash dataset and the maintenance datasets. The flight route&airport dataset is tabular with explicit data attributes. It covers the engine flight records and PM_{10} values in the departure and arrival airport. The volcanic ash dataset has information about the erupted volcanoes and the observed volcanic ash. While the maintenance datasets indicate the engine maintenance detail. Both of them are textual data and do not have a uniform schema. In the data pre-processing stage, we extract temporal and geographical information by utilizing regular expression. Some detailed information is combined as one attribute because there are no pre-defined data forms. The system architecture of EngineQV is illustrated in Figure 3.2.

3.3.2 Data Design

3.3.2.1 Data Description and Pre-processing

The EngineQV visualization system integrates multiple datasets for engine failure external factor analysis. After a series of discussion with the domain experts, we choose four kinds of information that we think is crucial for the analysis: aircraft

engine identification information, the flight records, environment-related information, and the engine maintenance information. We use the aircraft engine identification information as a basis and associate all the other temporal and geographic information with it (**T1**).

Aircraft/Engine Identifier The aircraft/engine identification usually consists of three identifiers: airline operator (Operator), aircraft registration identifier (ACID), and aircraft engine serial number (ESN). These are categorical data. We use ESN as the unique identifier to match data attributes across all the engine-associated datasets. With the hierarchical identification information, the user can query different layers of the engine data, such as a single aircraft engine, certain aircraft with multiple engines, or all the engines under an airline operator. Different engine groups can be created, and the user can explore the relations among them. Operators and ACIDs are useful as sometimes the engines of one aircraft may be replaced due to maintenance, so we may use ACID with ESN to keep track of the engines.

Flight Records The flight record data are the main focus of the EngineQV system. It contains important information about how engines may be affected by external factors. We extract the following data attributes for analysis in this system: engine serial number (ESN), departure date and time of the flight (datetime), the departure airport (cityprfr), and the arrival airport (cityprto). For the geo-location of an airport, we search the ICAO airport code [111] to get the accurate latitude and longitude. The dataset is tabular, containing temporal, categorical and geographical data. The data format is consistent in the dataset with no missing values.

Environmental Information With the temporal and geographical information in the flight records, we may see how the environmental factors contribute to the life cycle of an engine. The environment-related data in this system have two parts: the airport surrounding environment data and significant meteorological information across the world. During the takeoff, climb, descent, landing and taxiing phase of a flight, the airport surrounding environment plays a bigger role in influencing the engine status. In this system, we choose the particle pollution PM_{10} [112] near the airport

as an indicator of the weather condition. During the cruise phase, the meteorological information concerns the safety of the aircraft. In particular, here we consider the influence of the volcano eruption and the resulting volcanic ash cloud. The datasets are usually textual, with temporal and geographical information of various formats. We use regular expressions to parse the datasets and obtain the desired data attributes for analysis.

Maintenance Records We also include the engine maintenance data, which contains key information of the engine status. Analyzing the flight history of an under-maintenance engine may help the user better understand the influences of external environmental factors. The engine maintenance data are also textual with no predefined schema. We extract the maintenance event information, including engine serial number, maintenance timestamp, maintenance type, and other detailed maintenance parameters.

3.3.2.2 Data Model

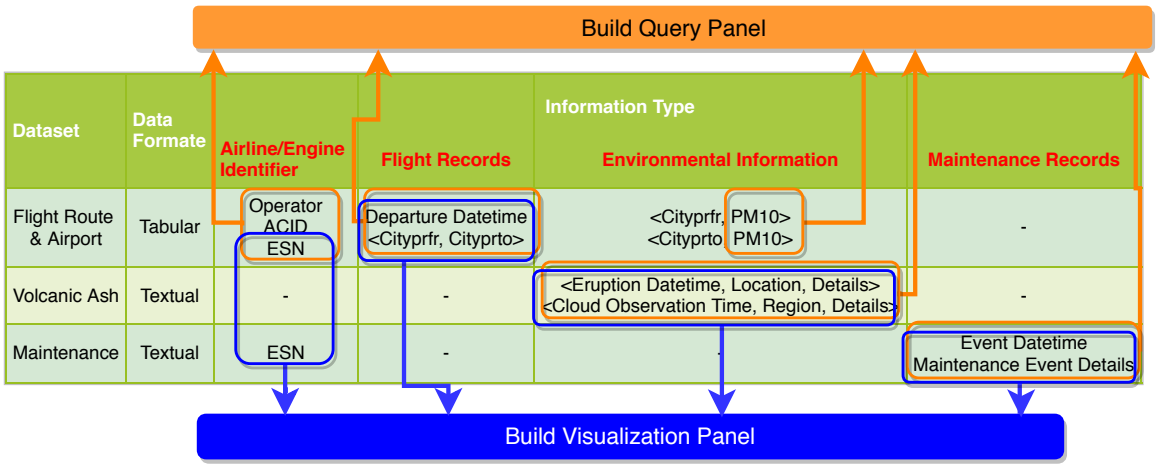


Figure 3.3. EngineQV data model. Three datasets that cover four information types are integrated into the system. Data attributes for building query panel and visualization panel are also highlighted.

To build an integrated visualization system for handling multiple information, we need to find the linkage among datasets and extract certain information for visualization. Three datasets are used in our application. The symbols and attributes of each dataset are listed in Table 3.1. Fig. 3.3 lists the attributes according to the

Table 3.1. Data description and symbol list.

	Symbol	Attribute	Description;
F_i	O_i	operator	Airline operator;
	P_i	ACID	Aircraft registration identifier;
	E_i	ESN	Aircraft engine serial number;
	T_i^{depart}	datetime	Flight departure date and time;
	L_i^{from}	cityprfrom	ICAO of the departure airport;
	PM_i^{from}	pm10_from	PM_{10} value at L_i^{from};
	L_i^{to}	cityprto	ICAO of the arrival airport;
	PM_i^{to}	pm10_to	PM_{10} value at L_i^{to};
V_i	T_i^{erupt}	validstart	Time of volcano eruption;
	$L_i^{volcano}$	position	Latitude and longitude value;
	$D_i^{volcano}$	detail	Volcano description (e.g. region);
	T_i^{ash}	obs_time	Observation time;
	L_i^{ash}	obs_position	Latitude and longitude value;
M_i	E_i	ESN	Aircraft engine serial number;
	T_i^{event}	datetime	Timestamp of the maintenance;
	D_i^m	detail	Maintenance description;

characters to highlight the relationship and explains the data model used in the EngineQV. The flight route&airport dataset (F_i) is tabular with explicit data attributes, but for volcanic ash (V_i) and maintenance (M_i) datasets, the textual data does not have a uniform schema. In the data pre-processing stage, we extract temporal and geographical information by utilizing some natural language processing (NLP) techniques. Some detailed information is combined as one attribute because there are no pre-defined data forms.

We try to build the query panel with all the attributes illustrated in Fig. 3.3. The query result Q_i returns all the data records that satisfied the user's query constraints. Then it can be expressed as:

$$\begin{aligned} Q_i = \{ &< F_i, V_i, M_i > \mid \text{if} < O_i, P_i, E_i, T_i^*, L_i^{from}, PM_i^{from}, L_i^{to}, PM_i^{to}, D_i^* > \\ &\text{satisfies the constraints} < const_rec, keyword > \} \end{aligned} \tag{3.1}$$

The multiple formats of the datasets force the system to involve multiple query components in order to provide wide flexibility to query and view the whole database from multiple facets. In EngineQV, we build drop-down lists for querying categorical data ($O_i, P_i, E_i, L_i^{from}, L_i^{to}$), scroll bars for setting tabular data ranges ($T_i^*, PM_i^{from}, PM_i^{to}$), and a text search bar for typing textual inputs (D_i^*).

We utilize the query result Q_i to build visualization panel Vis_i which consists of multiple visualization forms (Vis_i^*). Following the single responsibility principle, each visualization component only achieves one functionality.

$$Vis_i = \{<Vis_i^{airport}, Vis_i^{route}, Vis_i^{force}, Vis_i^{timetable}, Vis_i^{histogram}, Vis_i^{timeline} > | \text{ if } Q_i \text{ exists.}\} \tag{3.2}$$

3.3.3 Architecture Design

The system architecture of EngineQV is illustrated in Figure 3.2. The query panel in the frontend allows the user to interact with the backend and two query engines are used to handle different kinds of database. Hence, $Q_i = \{Q_i^1, Q_i^2\}$ where Q_i^1 is responsible for tabular and categorical data that meets the constraint *const_rec* in Query 1 while Q_i^2 is for textural data that meets the constraint *keyword* in Query 2. They can be expressed as:

$$\begin{aligned} Q_i^1 &= \{< F_i, V_i > | \text{ if } < O_i, P_i, E_i, T_i^*, L_i^{from}, PM_i^{from}, L_i^{to}, PM_i^{to} > \\ &\qquad \text{satisfies the constraints } < const_rec >\} \\ Q_i^2 &= \{< V_i, M_i > | \text{ if } < D_i^* > \text{ satisfies the constraints } < keyword >\} \end{aligned} \tag{3.3}$$

The major design challenges in this project are:

- Processing datasets from different sources and of various formats and supporting querying in the datasets at the backend.
- Visualizing the data in a single-page dashboard and updating the data according to user interactions at the frontend.

To tackle these two challenges, we first design RESTful APIs for the HTTP request and a logic for updating Query 1 (Sect. 3.3.4), and then build the multiple visualization forms in the vis panel (Sect. 3.3.5). We address the details in the following subsections.

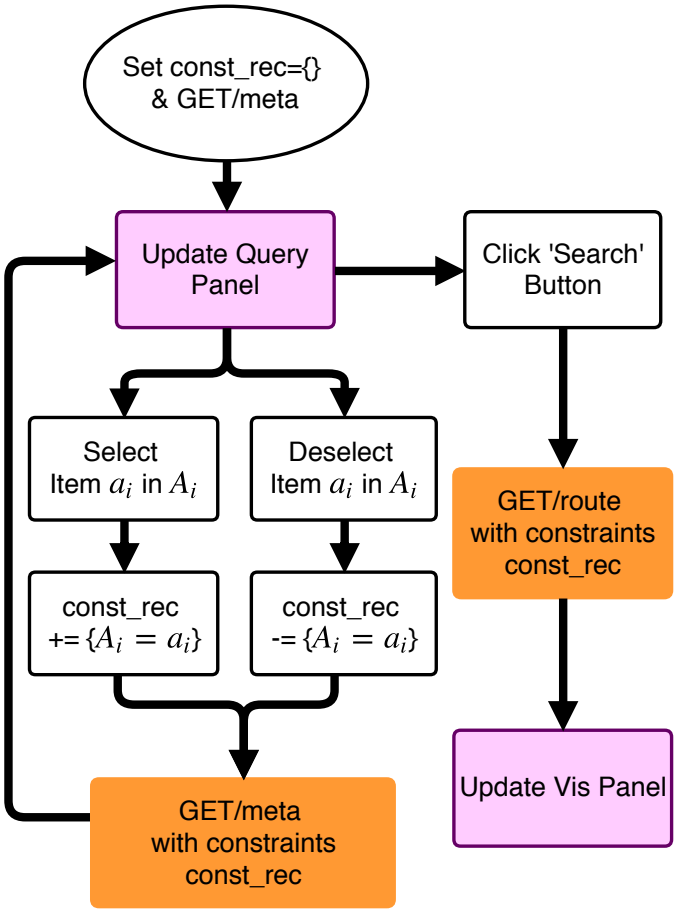


Figure 3.4. Query logic flowchart in Query 1. In the flowchart, *const_rec* refers to the selection constraint list, A_i refers to the ith query component, and a_i refers to the values of A_i.

3.3.4 Query Design

There are two contributions in our query design. Firstly, we design a query logic so that the query components in Query 1 can be dynamically updated according to the user's operations. Secondly, we design the data attribute setting in the text search in Query 2 so that the new text datasets can be added following the existing setting easily.

Query 1 To start with, we define two types of API for the HTTP requests, **meta** and **route**. The **meta** API will update the components in Query 1, returning all unique data attribute values for each constraint. The **route** API will return all data (Q_i^1) satisfying the selected constraints *const_rec*. The query logic is demonstrated in the flowchart in Fig. 3.4.

Once the website is loaded, **meta** API with no constraint (i.e. $const_rec = \{\}$) is sent to the server and all the unique records for each attribute are returned so that the query components (e.g. dropdown lists and scroll bars) are initialized. Then the constraint list *const_rec* is used to record the user's selections.

When a selection is made, **meta** API with updated constraints list *const_rec* is sent to the server and all the satisfied unique records for each attribute are returned to Query 1. Each query component is updated accordingly. Moreover, if a query component is deselected, the selections before this query component selection are removed, and the corresponding query components are set to the default values.

When a search is made, **route** API with *const_rec* is sent to the server and all the satisfied records are returned to the vis panel (i.e. Q_i^1 is obtained and Vis_i is then generated.). If the volume of the data is huge, the API request takes some time to return the results. To avoid collision of operations, the query panel is frozen in this case and unfrozen until the results are returned and the query components are updated.

Multiple groups of query data are made possible by adding new tabs in Query 1. The user may select different constraints in the query panel for each group, and the returned data will be visualized in the vis panel in a different color, the same as it shows in the Query tab.

Query 2 Elasticsearch engine is used to query the textual data and the returned records Q_i^2 will be shown in the vis panel ($Vis_i^{timeline}$). We design the data attribute setting based on the elasticsearch's principle so that additional datasets can be added conveniently. In EngineQV, we set the data type and data name based on the type of the data attributes. The settings of data attributes are written in the "config" files of the project so that the new text datasets can be added following the existing setting. The "config" file is a self-contained document and independent of the application framework. When dumping new dataset into our system, only modification of the "config" file is needed, rather than detailed changes in the application code.

The input is split by space and each is searched in Elasticsearch. If the input is "test" (e.g. $keyword =$ "test"), "test" will be searched in all fields in the textual data

(V_i, M_i). If the *keyword* is in the format of "place=asia&info=good", the details (D_i^*) satisfying place column is "asia" and info column is "good" are returned.

3.3.5 Visual Design

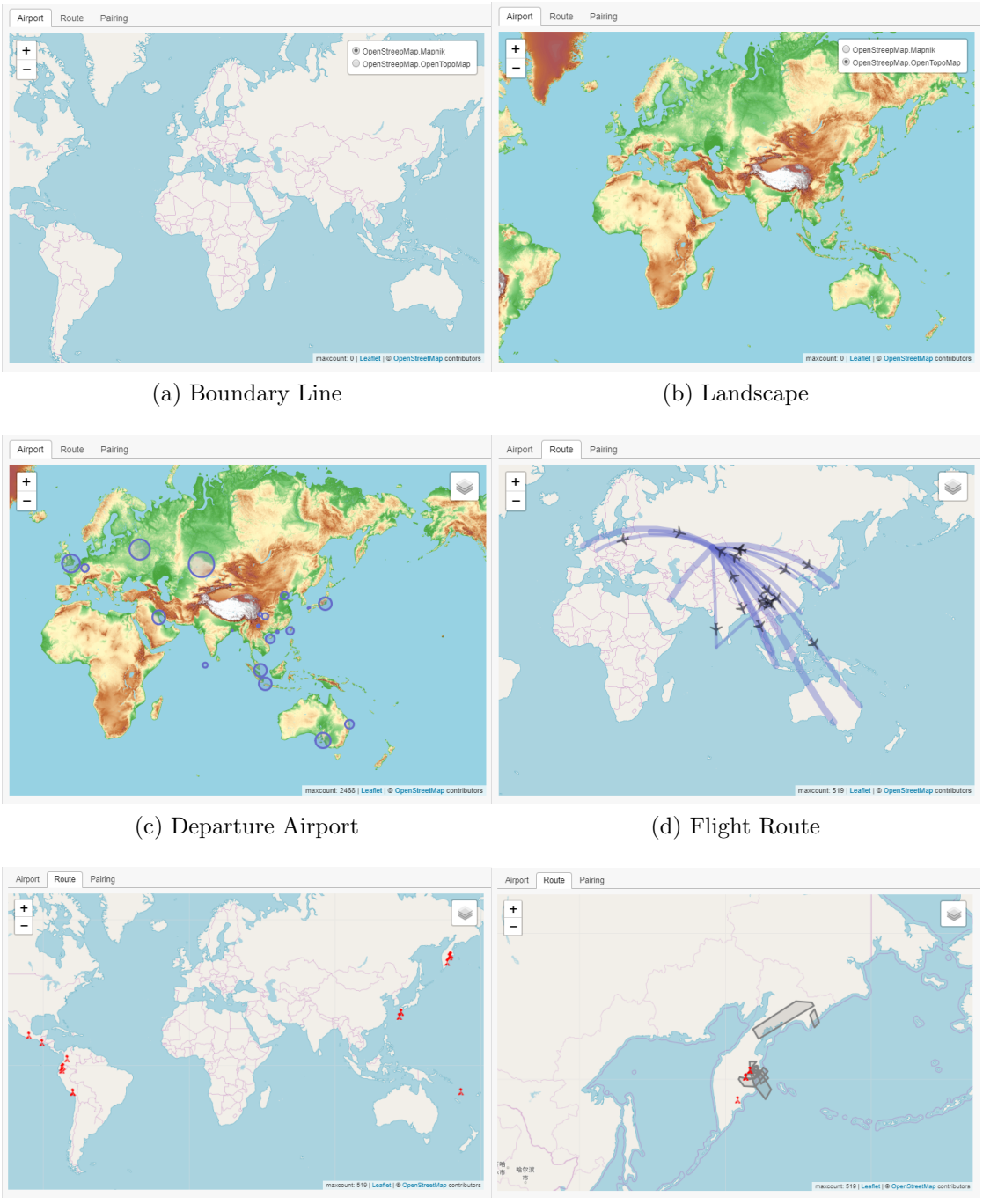
The vis panel Vis_i is responsible for displaying the query results Q_i and involves multiple visualization components (Vis_i^*). The vis panel is divided into three main components for visualizing geographical information (vis:geo), statistical information (vis:timetable and vis:histogram), and temporal event information (timeline:timeline) respectively.

Geographical Visualization A map-based visualization is utilized for displaying the geographical location of the data, linking the datasets with geographical information. As the flight information and the environment-related information are both geographical, it is straightforward to use a map as the carrier and we adopt a multiple layer plotting method in the geo-map to display different information. (**T2**) To be specific, it consists of three layers: the basic map layer (Fig. 3.5 (a, b)), the flight information layer (Fig. 3.5 (c, d)), and the environment-related information layer (including the erupted volcanoes in Fig. 3.5 (e) and the volcanic ash clouds in Fig. 3.5 (f)). This multiple layer design contributes to the usage of the superposition comparison strategy to plot one engine's information on top of one another. The basic map layer displays the general geographical information G.

In the flight information layer, to illustrate different facets of the datasets, two geo-map modes (vis:geo:airport $Vis_i^{airport}$ in Fig. 3.5 (c) and vis:geo:route Vis_i^{route} in Fig. 3.5 (d)) are designed. The airport mode provides airport-based information when the users explore the airport-related pollution data (e.g. PM_{10}) (**T3**). Meanwhile, the flight route mode helps the users to understand the environment around the flight route.

In the airport mode shown in Fig. 3.5 (c), the returned query result is grouped by the departure airport (i.e. L_i^{from}). Then $Vis_i^{airport}$ can be expressed as:

$$Vis_i^{airport} = \{< G, L_i^{from}, V_i > \mid \text{if } Q_i^1 \text{ exists.}\} \tag{3.4}$$



(a) Boundary Line

(b) Landscape

(c) Departure Airport

(d) Flight Route

(e) Volcano

(f) Volcanic Ash

Figure 3.5. The geographical visualization includes three layers (the basic map layer, the flight information layer, and the environment-related information layer) with multiple modes to deliver different information. To be specific, we provide boundary lines map (a) and landscape map (b) to give a geographical context in the basic map layer. Besides that, in the flight information layer the user can select which kind of flight information to be illustrated: departure airport (c) or flight route (d). Moreover, the erupted volcano (e) and the volcanic ash (f) can be displayed in the environment-related information layer. Combination of different choices in each layer will deliver different information.

For each group, we add a circle with its radius being the count of records and its center being the latitude and longitude of the airport to display the distribution on the map. The geographical view allows the user to perceive in a geographical context. On the other hand, the distribution of the departure flights contributes to the further understanding of the potential airport-related geographical factors (e.g. the climate of the departure airport).

In the flight route mode shown in Fig. 3.5 (d), the returned query result is grouped by using the departure and arrival city pair without order (i.e. $< L_i^{from}, L_i^{to} >$).

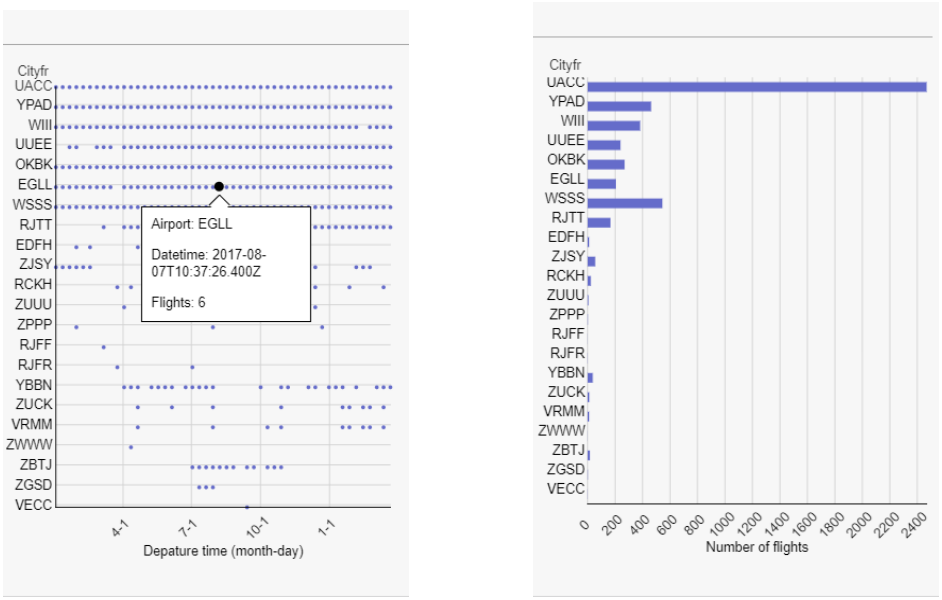
$$Vis_i^{route} = \{< G, L_i^{from}, L_i^{to}, V_i > \mid \text{if } Q_i^1 \text{ exists.}\} \tag{3.5}$$

Then for each group, we link the departure city and the arrival city with the line width being the count of records. For convenience, the direction with more frequency is labeled as forward. The flight route mode also allows the user to perceive in a geographical context the same as the airport mode. Contrast to the airport-related geographical factors provided by the airport mode, the flight route mode contributes to the understanding of potential factors during the cruise phase (e.g. volcanic ash).

In the environment-related information layer, the volcano eruption and the volcanic ash (V_i) always exist no matter which mode is chosen in the flight information layer. The volcano eruption is labeled as a volcano symbol on the map with its latitude and longitude position as shown in Fig. 3.5 (e) while the volcanic ash is represented as a polygon on the map as shown in Fig. 3.5 (f). (**T3**)

Statistical Visualization When and how often an aircraft departures from certain airport or flights along a route are interesting to the experts. This information indicate the probability of the potential influence. Hence, besides the geographical context of the geo-map, concrete quantitative information is provided in the statistical visualization. Two kinds of plots (timetable $Vis_i^{timetable}$ and histogram $Vis_i^{histogram}$) are given. (**T4**)

In the timetable plot, the flight temporal distribution is represented as a timetable in the dot plot, with x-axis showing the flight departure time stamp and y-axis showing



(a) (b)

Figure 3.6. In the statistical visualization, flight distribution visualization (a) and flight statistical visualization (b) are provided.

the airport ICAO code as shown in Fig. 3.6 (a). The timetable provides an overview of the flight records with regard to departure time. With a large amount of the flight records, the plot may get frozen. Therefore, the records are aggregated (i.e. sampled) in order to increase the computation speed. $Vis_i^{timetable}$ can be expressed as:

$$Vis_i^{timetable} = \{< T_i^*, L_i^{from}, L_i^{to} > \mid \text{if } Q_i^1 \text{ exists.}\} \tag{3.6}$$

In the histogram plot, the amount of flight records is represented as a histogram. The x-axis represents the number of flight records while the y-axis shows the airport ICAO code the same as the timetable plot as shown in Fig. 3.6 (b). Although distribution view has been displayed in the geo-map, the histogram plot gives a concrete quantitative information. Then $Vis_i^{histogram}$ can be expressed as:

$$Vis_i^{histogram} = \{< L_i^{from}, L_i^{to} > \mid \text{if } Q_i^1 \text{ exists.}\} \tag{3.7}$$

Multiple variations of $Vis_i^{timetable}$ and $Vis_i^{histogram}$ are provided to associate with other plots. Firstly, since there are two modes in the geo-map, the corresponding statistical information has been provided in $Vis_i^{timetable}$ and $Vis_i^{histogram}$. When under

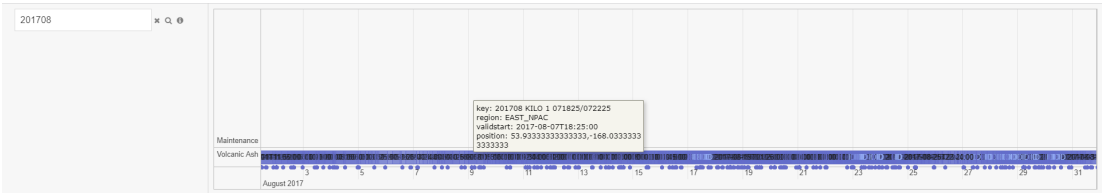


Figure 3.7. Temporal Event Visualization displays volcanic ash observation history and maintenance history.

the airport mode, the y-axis of both plots is the departure airport ICAO codes. When switching to the flight route mode, the y-axis of both plots is the departure and arrival city pair (i.e. $< L_i^{from}, L_i^{to} >$). Secondly, multiple-group information can be displayed simultaneously in these plots. According to the taxonomy of comparative designs [20], here we adopt a juxtaposition comparison strategy to compare multiple groups next to each other. As the EngineQV system is capable of forming multiple groups, displaying multiple groups of information simultaneously provides a direct visual comparison among different groups. (**T5**)

Temporal Event Visualization The textual datasets about the maintenance details are essential as the domain experts frequently check them to track the engine's status (**T1**). Hence, a timeline plot $Vis_i^{timeline}$ as shown in Fig. 3.7 is adopted to display the temporal event information and build a linkage among different datasets. In the EngineQV, we plan to visualize the maintenance information (M_i) and some environment-related information (V_i) in $Vis_i^{timeline}$. The user can query in Query 2 to filter the textual datasets and the returned query result (Q_i^2) is visualized. Each record is represented by a rectangle with the event timestamp. M_i and V_i are shown in two separate categories for clearing view. This plot can be formulated as:

$$Vis_i^{timeline} = \{< T_i^*, D_i^* > \mid \text{if } Q_i^2 \text{ exists.}\} \tag{3.8}$$

Moreover, a search bar is designed in Query 2 (Figure 3.1) so that the users can query by keywords which are familiar to them. Then the returned query result is visualized in the timeline. Each record is represented by a rectangle with the event timestamp. The maintenance information and volcano-related information are shown in two separate categories for clearing view. The timeline also supports multiple-group

comparison by using a side-by-side comparison with different colors for each group.

3.3.6 Interactions

Other interactions provided in EngineQV are summarized as follows.

Details-on-demand The geo-map is displayed at different levels of details. Users can zoom in and out to explore the geographical information, especially the detailed volcanic ash region.

Tooltips Tooltips are provided in each plot to depicted the detail information when hovering a mouse over the plot. For example, in the geo-map airport mode, when hovering over a circle, the number of flights will be depicted. Moreover, when hovering on a rectangle about the volcanic ash in the timeline plot, the detailed information about the volcanic ash will be depicted.

Layer control Since the geo-map consists of multiple layers of information, users can turn on or off each layer in our application. It allows the user to visually compare different information in multiple groups.

3.3.7 Implementation

In the backend of the EngineQV, the categorical and tabular data are stored in an SQL database, and the query will be handled by Flask with SQLAlchemy. On the other hand, the volcanic ash and maintenance datasets are parsed and saved as JSON files. They are then dumped into the Elasticsaerch storage with the specified data types, search fields, and data attribute names.

In the frontend, the React-Redux is used to control the data flow. The visual components are implemented using Leaflet.js, Semiotic.js, and Vis-timeline.js.

3.4 Evaluation

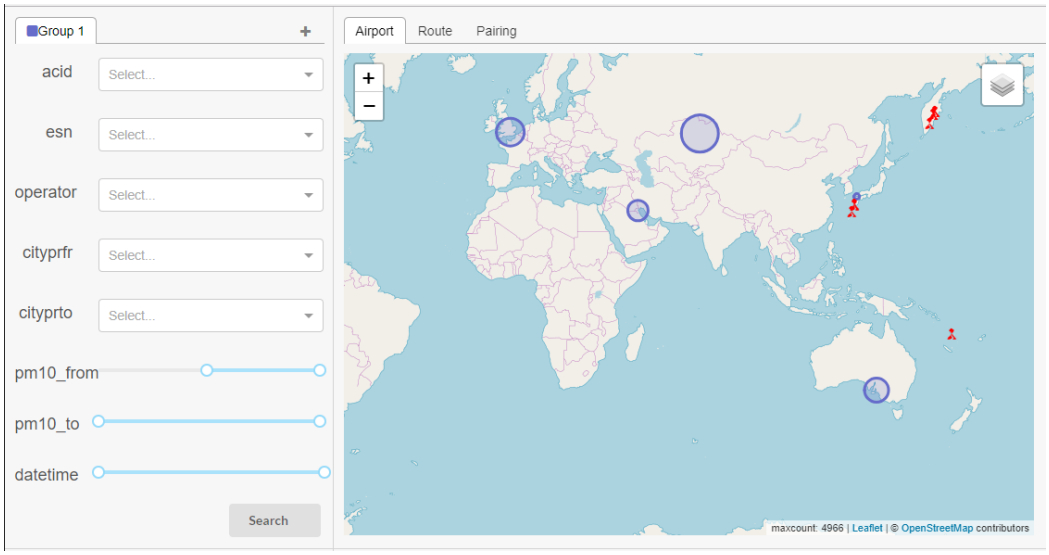
Two groups of domain experts were invited to evaluate the usability and effectiveness of the EngineQV system. One (**G1**, two people) was from the engine operational department. They have no programming experience and use commercial software as their daily tool. The other group (**G2**, three people) was from the data analysis department. They usually use Python to visualize and analyze the data. Both of them are from the same engine company and responsible to explore the external factors that affect the aircraft engine performance. **G2** is the group that works with us during the design while **G1** is the main target user of our system.

We demonstrate two use cases for evaluating the usability and effectiveness of our system. After that presentation and interviews were conducted with **G1** and **G2** experts. In this section, we will first describe the scenarios of use cases based on real-world requirements (Sect. 3.4.1), and then discuss the feedback from experts on our system (Sect. 3.4.2).

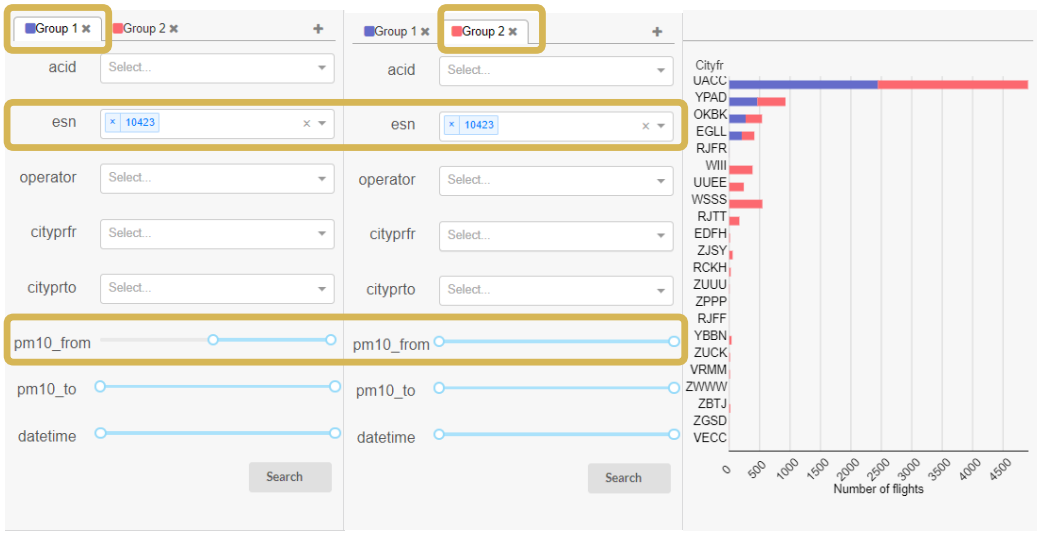
3.4.1 Use Cases Study

In this section, we present our EngineQV system for two application scenarios. The first scenario is when only data relating to a single engine is explored. Exploring the effect caused by environmental factors is the main target in this scenario. The user may utilize the multiple query controls in EngineQV to get the expected view. In the second scenario, we present a visual comparison of multiple engines. The expert user may know the expected/good behaviors of an engine. Comparison of multiple engines may inspire the experts to identify the potential external factors for engine failures.

The flight information dataset and the associated engine maintenance data we use are gathered from an Asia-based airline company from 1st January 2017 to 31st March 2018. We also capture the environment-related data at the corresponding airports and the volcano eruption data around the world during this period.



(a) Airports with high air pollution level ($PM_{10} > 30$)

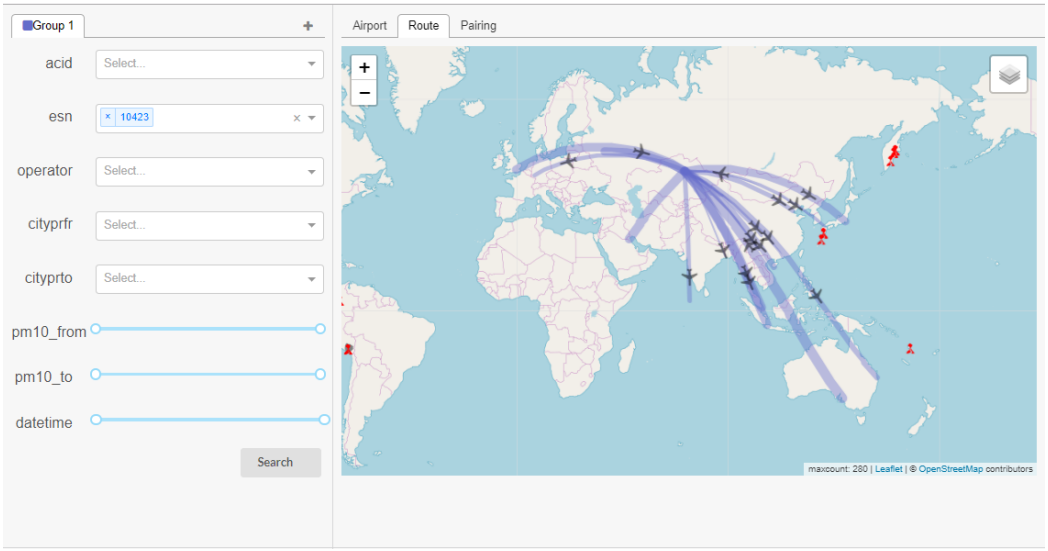


(b) The number of flights from airports with different PM_{10} levels.

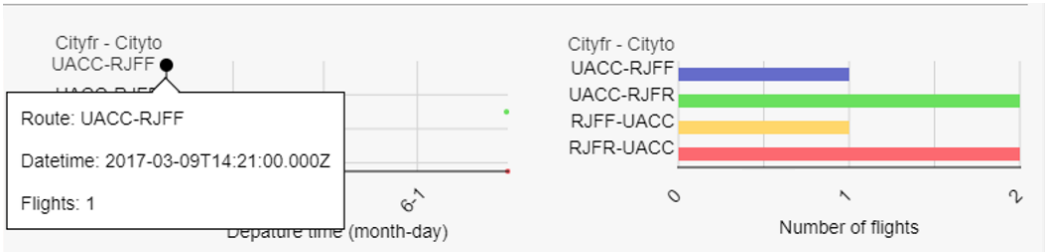
Figure 3.8. Use case 1: query and visualization of airport air quality. The user may query for airports with high air pollution (a), or compare flights from airports with different PM_i^{from} levels (b).

3.4.1.1 Use Case 1: Sense Making and Knowledge Discovery of Single Engine

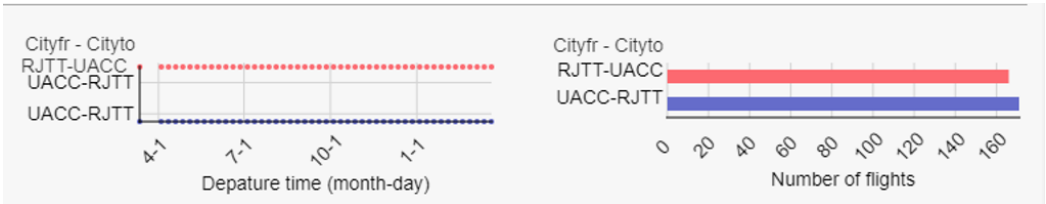
In this scenario, we consider the environmental factors which may affect the performance of an aircraft engine. The exploration is twofold. On one hand, the environmental factors that may influence the aircraft engine during the takeoff and landing phase are explored. The air condition of the departure and arrival airport is considered and the airports with bad air quality need to be pointed out. This gives the user a general idea of which airports have a bad air quality. In this case, we consider



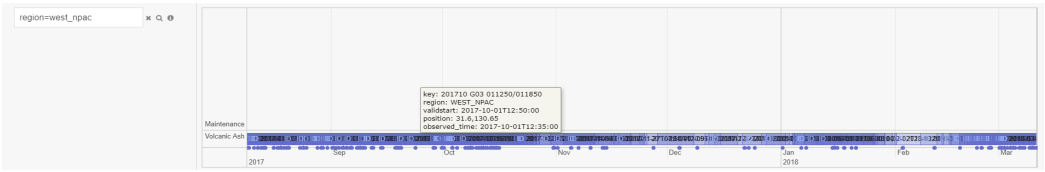
(a) Geo-map (flight route mode)



(b) The flight routes potentially influenced by volcano eruption.



(c) Another flight route that may be influenced.



(d) Search the volcano eruption events in the 'west_npac' region.

Figure 3.9. Use case 1: exploration of the flight routes potentially influenced by the volcanic ash. The user may use Geo-map as an overview to find the influenced flights (a), view the flight route time and frequency in the timetable and histogram plot (b, c), and query results on the volcano eruption events at the related region (d).

$PM_{10} > 30$ as high air pollution level which may cause adverse influence on the aircraft engine. As illustrated in Figure 3.8 (a), we set the lower bound of the PM_i^{from} to 30 and five airports ('UACC', 'YPAP', 'OKBK', 'EGLL', 'RJFR') with bad air quality are illustrated.

Further consideration of the frequency of flights from and to these airports can be

achieved by using two query groups with different PM_i^{from} constraints as illustrated in Figure 3.8 (b). Since the landed aircraft takes off soon, only PM_i^{from} is considered in this system. Moreover, the histogram shows the number of flights from these airports.

On the other hand, the environmental influence during the cruise phase is also taken into consideration. After querying all the flights of a certain engine, the flight route mode of geo-map could provide an intuitive view to help the user to find the geographical distribution of flight routes and the erupted volcanoes (Figure 3.9 (a)). The flight routes that are potentially influenced by the eruption are further analyzed. In this case, the flights between $< UACC, RJFF >$ and the flights between $< UACC, RJFR >$ have the largest possible to be affected. However, after checking the timestamps of the flights and the surrounded erupted volcanoes, we notice that these flight records are before July 2017 while the volcano eruption at the region 'west_npac' (i.e. west_north pacific ocean) is from August 2017 as illustrated in Figure 3.9 (b) and (d). Hence, the exploration may be expanded by considering the flights between $< UACC, RJTT >$ as the volcanic ash dispersal (Figure 3.9 (c)). There are in total, 337 flights operated from 11th March 2017 to 21st March 2018. As the observed volcano eruption in this region starts from August 2017 and lasts till March 2018, the aircraft engine might be influenced by the volcanic ash. The spreading direction of the volcanic ash illustrated on the geo-map puts further evidence on the suspect.

One thing should be noted is that only qualitative analysis can be obtained as the volcano ash is difficult to be observed and may be sustained for a long period of time. Once a qualitative observation is obtained, a quantitative analysis of the influence can be conducted by analyzing the engine sensor data as well as the mechanical check after each flight. With the guideline of the qualitative observation, these time- and cost-consuming checks can be more efficient.

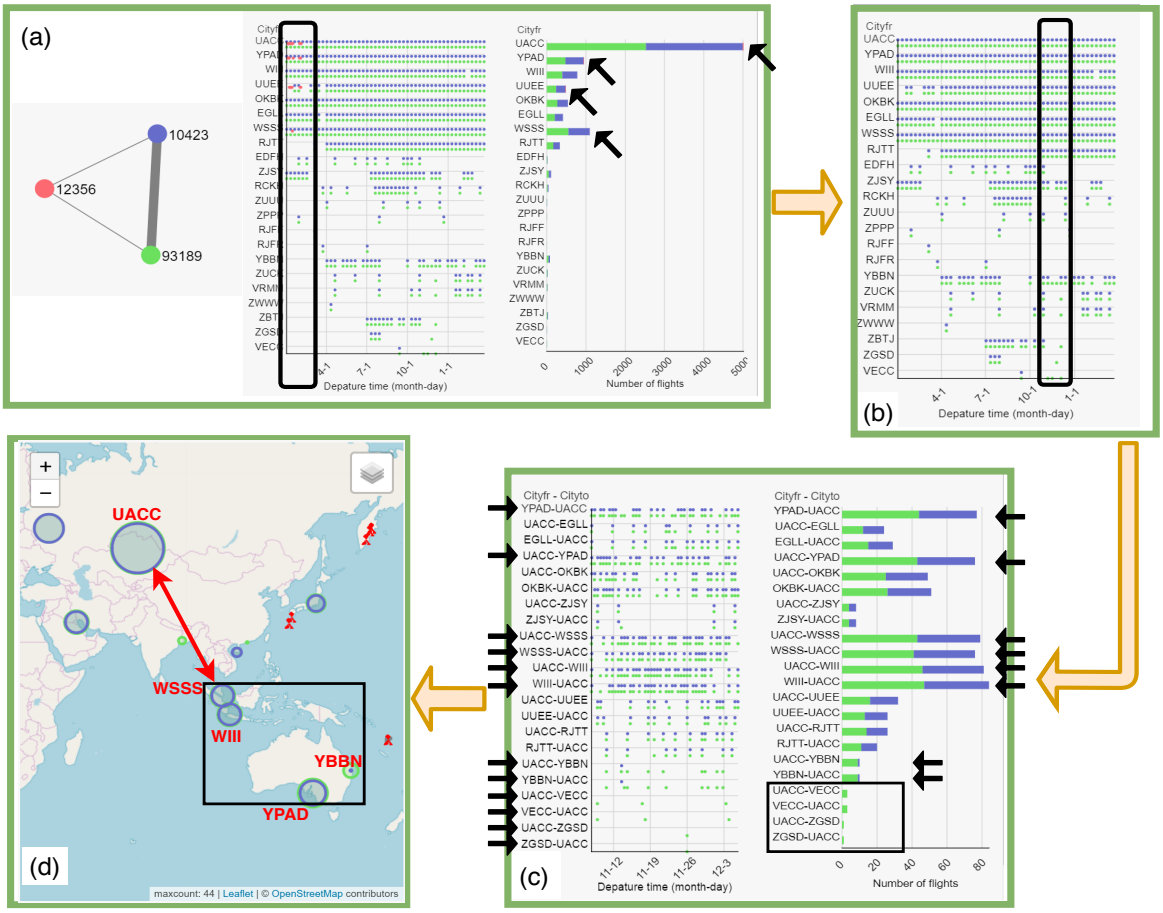


Figure 3.10. Use case 2: visual comparison of multiple engines. The user may start analyzing by looking at the force plot and statistical plots to ignore the outlines (a). After removing EngineRed, comparison of EngineGreen and EngineBlue in timetable shows different patterns from 07th Nov 2017 to 05th Dec 2017 (b). By adjusting the time range, the flight route of different behaviors are identified (c). The geo-map under the airport mode with labels indicates the observation (d). The airport code ICAO displays when hovering a mouse on the airport circle.

3.4.1.2 Use Case 2: Visual Comparison of Multiple Engines

Some factors (e.g. environmental factors) can be identified on the basis of operational experience and some data operations including data visualization, while others may not. One strategy used in this case is to compare similar objects to find the difference. By comparing objects with different status, hypotheses may be put forward before moving towards the truth. In EngineQV, the user can build multiple query groups under different constraints to compare their performance in various plots.

The engines served on a certain aircraft at different periods are considered, as engines on the same aircraft usually have similar flight records including flight route, flight time, and flight frequency. The target of this visual comparison is to compare

the flight information of different engines and find the main difference. The visual comparison process is illustrated in Figure 3.10. In the following descriptions, the engine is named by its group color for better association with the visualization.

Three engines (EngineRed, EngineGreen, and EngineBlue) under the same aircraft are involved in the dataset. When visualizing multiple engines, EngineQV provides a force plot (Vis_i^{force}) to display their pairing relationship. An edge between two engines means that these two engines are pairing in certain flight. The more flights engines are pairing with each other, the stronger the linking edge between them is. As shown in Figure 3.10 (a), the EngineRed has weak linkage with the other two engines. Moreover, the statistical plots show that the EngineRed only appears in a very short period. Hence, analysis focus may shift to the comparison between the EngineGreen and the EngineBlue.

The timetable depicts the temporal distribution of the flight information of the EngineGreen and the EngineBlue in Figure 3.10 (b). There is a huge overlap between these two engines and the main difference is located in November 2017. By referring to the tooltips, we focus on the time interval from Tuesday 07th November 2017 to Tuesday 05th December 2017.

By adjusting the T_i in Query 1 for both engines, $Vis_i^{timetable}$ and $Vis_i^{histogram}$ show more details. After a comparison on the flight's temporal distribution and flights frequency, flights with large dissimilarity were identified in Figure 3.10 (c). As illustrated, the EngineGreen operates more flights in the following six airport pairs $< UACC, YPAD >$, $< UACC, WSSS >$, $< UACC, WIII >$, $< UACC, YBBN >$, $< UACC, VECC >$, and $< UACC, ZGSD >$. As the flights between $< UACC, VECC >$ and the flights between $< UACC, ZGSD >$ have very limited records, lower priority is given to these airports. Finally, the five suspicious airports are labeled in red on the geo-map under the airport mode in Figure 3.10 (d). All the flights are taking off at UACC or landing at UACC. The black rectangle marks the region where the other four airports (WSSS, WIII, YBBN, and YPAD) are in the geo-map. We observe that, in this region, there are no other related airports and no other flights flew over this region in this period.

This observation may be used to find the potential external factors. The domain experts may want to check previous data related to this region during the 'November' as well as the maintenance data in this period to find any interesting points. Others may focus on what happened in this region at that time as our data may be not sufficient. Under this track, we find that the volcano at Bali island erupted in Nov 2017 and our volcanic ash dataset did not include this information. The third hypothesis is that weather difference in these flights may have a negative influence on the aircraft engines, since all of these flights have large latitude span.

3.4.2 Expert Feedback

We presented our system to the two groups of experts, and gave them time to familiarize with the system and explore the dashboard. **G1** experts mentioned that the integrated visualization of multiple datasets makes data analysis much easier. As in their previous workflow, they may have to open and view the data files from different sources separately. Moreover, the system is convenient to query different datasets by setting constraints. The visualization of environmental factors provides a straightforward understanding of the potential influence on the engines. Furthermore, the grouping functionality significantly contributes to their exploration. Most importantly, the **G1** experts mentioned that the hypotheses founded in the case study are very helpful for them.

Both **G1** and **G2** experts agreed that the functionality of EngineQV satisfied their requirements. They were impressed by the ease of use of our system and the intuitive presentation of datasets. **G2** experts mentioned that the visualization design provides an intuitive way to understand multiple datasets and the dynamic query panel allows the user to flexibly explore the database. This also reduces the recurring coding for a dataset overview. **G2** experts also pointed out that with more datasets, the system may have increased lag during operation.

3.5 Summary

In this chapter, we developed an interactive geo-temporal visualization system for analyzing the potential external causes of aircraft engine failures. With requirements from the domain experts considered, the system integrates multiple heterogeneous datasets and present various aspect of the data in versatile visual representations. The user may benefit from the system's dynamic query and intuitive visual exploration, which make insight finding easier and more efficient. This highly tailored system shows how datasets of different sources can be linked and visualized interactively. Finally, the usability of our system is illustrated by two use cases and confirmed with expert feedback.

Chapter 4

Discriminating Visualization for High-dimensional Data

4.1 Introduction

Datasets with multiple dimensionalities exist widely in engine design and engine simulation, with one dimensionality representing one character of the engine. The analysis on these datasets is conducted by considering more than one dimensionality since more information usually bring more insights. Data visualization provides an intuitive way to display these high-dimensional datasets. Visualization of the high-dimensional dataset can be considered as a problem of how to map the data to a lower dimension in a useful way. A suitable data projection method enables one to observe and detect underlying data patterns and distributions in exploratory data analysis. A great deal of efforts has been devoted to this topic and various visualization methods have been proposed based on specific application requirements [7, 113].

Both RadViz [33] and Star coordinate (SC) [34] adopt a radial pattern to display the high-dimensional data. They have emerged as promising visual mapping approaches in recent years. By transforming data point with a combination of radial distributed axes vectors, RadViz and SC map data points with a high computational performance. These characters indicate great potential of RadViz and SC to provide insights on high-dimensional data visualization. Nevertheless, such potential is yet to be translated to capability in real applications due to insufficient works. For instance,

the cluster representation should include outlier detection and attribute estimation, which is still imperfect. A technical challenge is that data points which are far away from each other in high-dimensional space may be mapped to the nearby points in low-dimensional plotting. With the increase in dimensionality, data points tend to clump towards the center of RadViz, known as the clumping problem [2]. The patterns and features are thus buried in the clumping data points, so they cannot be detected by visualization functions such as query result display and outlier detection in our turbine performance assessment. Another challenging problem is that the range of SC for all the possible projected data points is not determined. This range not only enables the plot of additional data points based on current configuration but also supports the analysis of the geometrical properties of SC. Hence, it leads to a core computational problem of SC boundary evaluation. To the best of our knowledge, no research work on the boundary of SC plots has been published.

In this chapter, for the former challenge, we propose a new focus+context distortion approach, termed PolarViz, to manipulate the radial distribution of data points in RadViz. We derive radial equalization to automatically spread out the frequency, and radial specification to shape the distribution based on users requirement. Meanwhile, for the latter challenge, we propose a geometric algorithm to determine the axis-aligned bounding box, minimum bounding box, and bounding polygon of SC with an arbitrary configuration. The content of this work has been published. [1] [2] [3]

The rest of this chapter is organized as follows. We introduce the notation used and the formal expressions of RadViz and SC in Sect. 4.2. In Sect. 4.3, we describe the proposed distortion approach for RadViz to ease the clumping effect and in Sect. 4.4, we describe the geometric algorithm to generate the boundary of SC. Finally, discussion is made in Sect. 4.5.

[1] Yan-Chao Wang, Qian Zhang, Feng Lin, Chi-Keong Goh, Xuan Wang, and Hock-Soon Seah. Histogram equalization and specification for high-dimensional data visualization using radviz. In the *Computer Graphics International Conference* (CGI), ACM, 2017. [114]

[2] Yan-Chao Wang, Qian Zhang, Feng Lin, Chi Keong Goh, Hock Soon Seah, PolarViz: a discriminating visualization and visual analytics tool for high-dimensional data, *The Visual Computer*, Springer, 2018. [107]

[3] Yan-Chao Wang, Feng Lin, and Hock-Soon Seah. **Evaluation of Star Coordinate Boundaries**. In the *Computer Graphics International Conference* (CGI), ACM, 2018. [106]

4.2 Background

In this section, we define the notation used in the following sections and review the background on the RadViz and the Star Coordinate plots.

The radial visualization method, RadViz [33], non-linearly maps high-dimensional data points into low-dimensional space by taking all dimensionalities into consideration. RadViz described using a spring model is well accepted and has broad applications [36, 37, 41, 42, 114–118]. The mapping is based on the standard assumption that the values of each data point should be normalized into the interval of $[0, 1]$ first. For dataset $D = (d_1, \ldots, d_j, \ldots, d_m)$ containing m data points, each data point $d_j = (d_{1,j}, \ldots, d_{i,j}, \ldots, d_{n,j})$ has n-dimensionalities. Let $d'_{i,j}$ be the normalization result for $d_{i,j}$. This normalization step for D transfers data point $d_{i,j}$ to $d'_{i,j}$ for $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. The normalization equation for $d_{i,j}$ is

$$d'_{i,j} = \frac{d_{i,j} - min_i}{max_i - min_i} \tag{4.1}$$

where $min_i = \min\{d_{i,j}\}$ and $max_i = \max\{d_{i,j}\}, \forall j$. Each dimensionality axis is represented by using a low-dimensional vector $\overrightarrow{v}$. Formally, the expression of RadViz can be defined as:

$$p^{RV}(x_j, y_j) = \left(\frac{\sum_{i=1}^{n}(d'_{i,j} \cdot L_i \cdot \cos\theta_i)}{\sum_{i=1}^{n} d'_{i,j}}, \frac{\sum_{i=1}^{n}(d'_{i,j} \cdot L_i \cdot \sin\theta_i)}{\sum_{i=1}^{n} d'_{i,j}} \right), \tag{4.2}$$

where (x_j, y_j) is the projected data point p^{RV} in two-dimensional space and (L_i, θ_i) is the magnitude and angle of the i^{th} vector $\overrightarrow{v_i}$.

Another radial visualization method often compared with RadViz is SC [34]. The main difference between SC and RadViz is that RadViz defines a non-linear mapping. So linear transformations which can be combined with SC are difficult to be implemented in RadViz. Then the projected point $p^{SC}(x_j, y_j)$ in 2D SC is simply the linear combination of these radial distributed vectors, where the linear coefficients are the

attributes of each data point. The expression of standard SC can be defined as:

$$p^{SC}(x_j, y_j) = \left(\sum_{i=1}^{n} (d'_{i,j} \cdot L_i \cdot \cos\theta_i), \sum_{i=1}^{n} (d'_{i,j} \cdot L_i \cdot \sin\theta_i) \right), \tag{4.3}$$

where (x_j, y_j) is the projected data point p^{SC} in two-dimensional space and (L_i, θ_i) is the magnitude and angle of the i^{th} vector $\overrightarrow{v_i}$. For standard SC, all the radial distributed vectors have unit magnitude and are evenly distributed which means that the angle between successive vectors is the same.

Then the relationship between SC and RadViz can be expressed as:

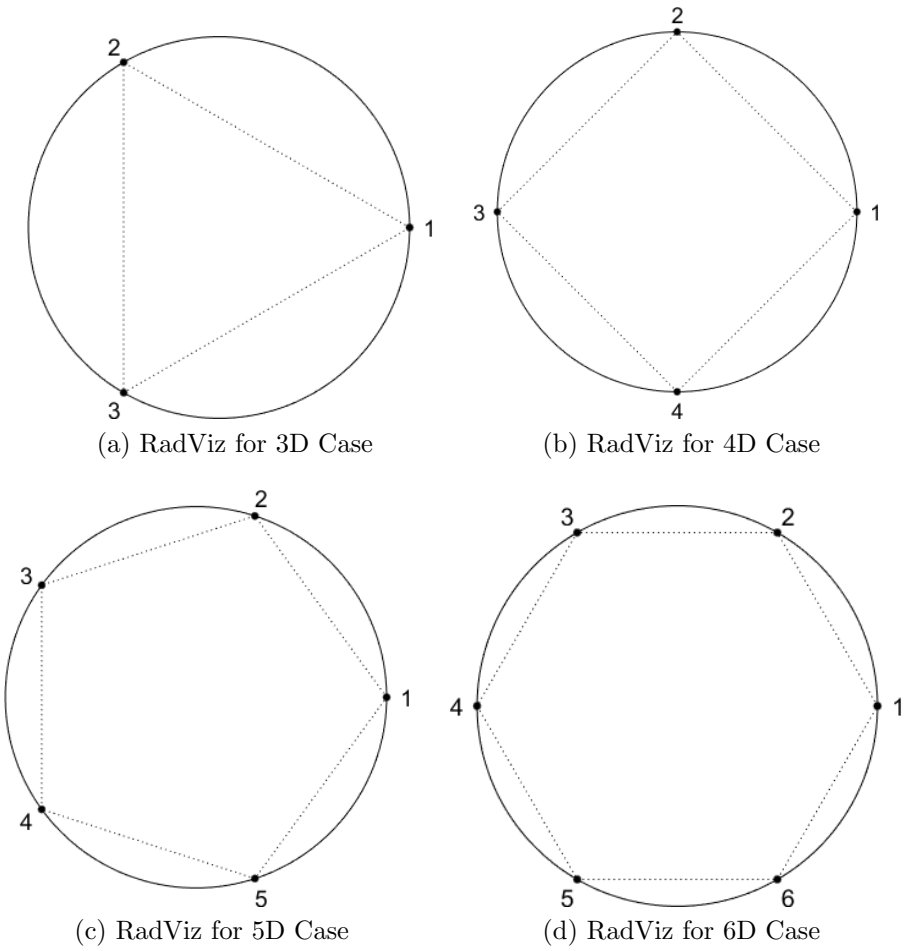
$$p^{RV}(x_j, y_j) = \frac{1}{\sum_{i=1}^{n} d'_{i,j}} \cdot p_j^{SC}(x_j, y_j). \tag{4.4}$$

The non-linear factor $1/\sum_{i=1}^{n} d'_{i,j}$ in Eq. 4.4 brings new properties into RadViz. The theoretical justification for the properties of RadViz lays a foundation for future visualization research [2]. In RadViz, each dimensionality is associated with a point on a unit circle which is called a dimensional anchor. One property is that RadViz maps each high-dimensional data point to a point that is within the convex hull of the dimensional anchors while all dimensional anchors are located on a circle with unit radius [2]. The circumscribed circle of RadViz is fixed for any plotting even the location change of dimensional anchors. It is convenient to give the user an overview of the whole high-dimensional space by using RadViz. Examples are given in Fig. 4.1.

4.3 PolarViz: A focus+context distortion approach for RadViz

4.3.1 Introduction

In this section, we propose a focus+context visualization distortion techniques to ease the clumping problem and increase space utilization in RadViz. The clumping problem was analyzed before and one example is the RadViz with sigmoid function [119]. However, the method changes the values of the original dataset and destroys



(a) RadViz for 3D Case (b) RadViz for 4D Case

(c) RadViz for 5D Case (d) RadViz for 6D Case

Figure 4.1. The circumscribed circle and bounding polygon of RadViz when plotting 3D, 4D, 5D, and 6D datasets.

the cluster information of the dataset.

Our method plots high-dimensional data points into RadViz and analyzes the clumping problem by using a polar coordinate system instead of the Cartesian coordinate system. Based on the new representation, the radial distance shows its potential to alleviate the clumping problem of RadViz. Hence, we propose the idea on the modification of the radial distance distribution. For the distribution of the radial distance, we observe a similar problem in the image processing field. For images with low contrast, it seems that the intensity of pixels is clumping into a small range. Hence, similar to histogram-based operations in image processing, we propose a series of radial operations to manipulate the radius of each point.

The main contributions of this section are:

1. we present the RadViz by using a polar coordinate system. The clumping problem of RadViz can be clearly analyzed and visualized in the polar coordinate

system;

2. we define the radial operations including radial equalization and radial specification for the radial distance of RadViz in order to adjust the RadViz plots;

3. we generalize the radial operations by considering the cumulative distribution function (CDF). Any modification on the CDF can be visualized in the new view.

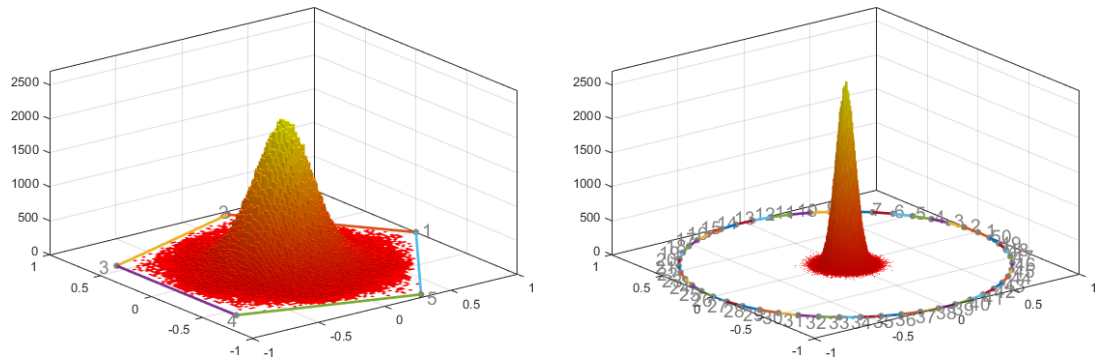
The rest of the section is organized as follows. In Sect. 4.3.2, we first introduce the clumping problem in RadViz in the Cartesian coordinate system as well as previous approaches and then we consider the clumping problem in the polar coordinate system and our new approach. The detailed methodology is described in Sect. 4.3.3 including the radial operations in Sect. 4.3.3.1 and the generalization in Sect. 4.3.3.2. Applications using the proposed method for data analysis and comparisons with other methods are discussed in Sect. 4.3.4. User study to evaluate the usability of the proposed method is described in Sect. 4.3.5. Finally, technical discussion is presented in Sect. 4.3.6.

4.3.2 The Clumping Effect

4.3.2.1 Clumping Problem in RadViz

The mapping in Eq. 4.2 is nonlinear. Due to that, an effect that data points tend to clutter in the center of the plotting exists and is often known as the clumping problem as shown in Fig. 4.2. RadViz's clumping problem was first observed and analyzed in [2], and the effect of diametrically opposed dimensional anchors under the spring-force analogy is considered as the main reason.

To ease the clumping effect by changing the order of dimensional anchors (DAs) is an inefficient way. Firstly, for general cases, the clumping effect exists under every configuration. Secondly, the exploration for the optimal result by exhaustively searching all possible configurations is very time-consuming with a complexity of $\mathcal{O}(n!)$ and practically intractable for not too large n. More importantly, the order of DAs (as well as the orientation of DAs) is often used by researchers to get a better plotting



(a) 5D dataset (b) 50D dataset

Figure 4.2. We demonstrate the clumping effect by plotting the distribution of 100,000 uniformly sampling 5-dimensional and 50-dimensional data points. The 3D view of the distribution shows that most data points are clumped in the center area in 50D case (b) than that in 5D case (a).

with more information [37]. Hence, it is highly inefficient to change the order of DAs to tackle the clumping effect.

4.3.2.2 Previous Approach for Clumping Problem

The clumping problem in RadViz was analyzed in [119]. They proposed a filtering mechanism to cancel the forces in RadViz and reduce clutter in the center region. Their approach was used for visualization of multi-task, multi-label classification and applications with validation were given.

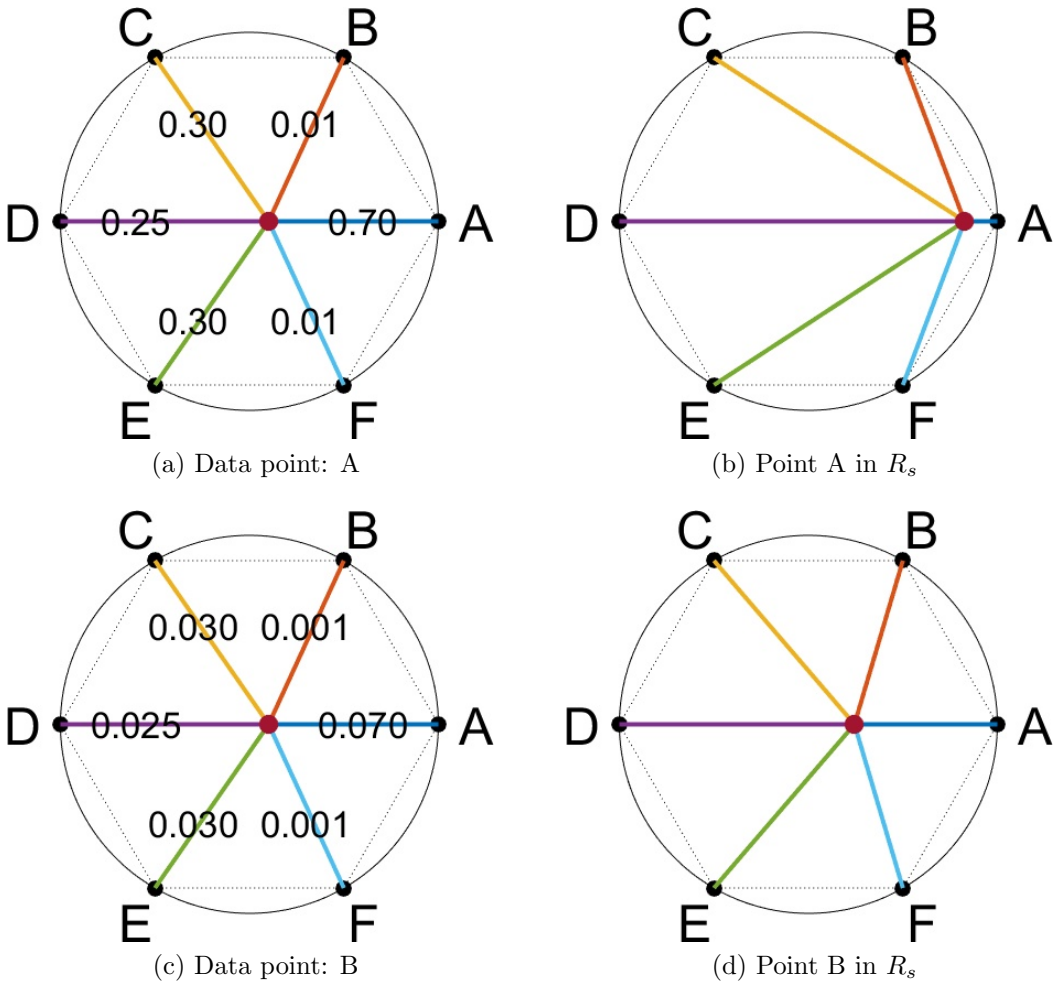
The filtering mechanism is a sigmoid weighting method. The filter operates by multiplying each dimension value $d'_{i,j}$ with a zero-one normalized sigmoid function

$$\widehat{\sigma}(x,s,t) = \begin{cases} \dfrac{\sigma(x)-\sigma(0)}{\sigma(1)-\sigma(0)} & if \quad \sigma(1) \neq \sigma(0), \\ 1 & otherwise, \end{cases} \tag{4.5}$$

with

$$\sigma(x) = \frac{1}{1+\exp(-s(x+t))}.$$

Let d^s_j be the data point after sigmoid function. Then the original data point d'_j will be changed to $d^s_j = d'_j \cdot \widehat{\sigma}\left(d'_{i,j}, s, t\right)$. The control parameters s and t are used to build the threshold. Finally, following the expression in Eq. 4.2, the RadViz with



(a) Data point: A (b) Point A in R_s

(c) Data point: B (d) Point B in R_s

Figure 4.3. Data points A and B have the same location in RadViz as shown in (a) and (c). However, after the sigmoid weighting operation, the obtained results are quite different as shown in (b) and (d).

sigmoid function p_s^{RV} can be expressed as:

$$p_s^{RV}(x_j, y_j) = \left(\frac{\sum_{i=1}^{n} (d_{i,j}^s \cdot L_i \cdot \cos\theta_i)}{\sum_{i=1}^{n} d_{i,j}^s}, \frac{\sum_{i=1}^{n} (d_{i,j}^s \cdot L_i \cdot \sin\theta_i)}{\sum_{i=1}^{n} d_{i,j}^s} \right), \tag{4.6}$$

where $d_j^s = d_j^{'} \cdot \widehat{\sigma}\left(d_{i,j}^{'}, s, t\right)$.

However, as the high dimensional data points are changed and the projection is non-linear, the obtained RadViz plotting cannot be estimated. An example is shown in Fig. 4.3. In Fig. 4.3 (a) and (c), two data points A and B are plotted in RadViz at the same location and their dimension values are labeled. After the sigmoid weighting operation, data point A is moved towards the circle farther than data point B. For data point A, the variance of each dimension is higher than that of data point B. Sigmoid weighting function has less influence on higher values while

reduces lower values significantly. Hence, it is obvious that the cluster information of original high dimensional dataset will be destroyed. The experiment results when handling benchmark dataset and our engine-related datasets in Sect. 4.3.4 also show this drawback of sigmoid function.

Distortion methods operated in the image space can be used to ease the clumping problem in RadViz, though they are not designed for this purpose. Fisheye distortion designed upon the observation from fisheye camera lens is the most commonly used distortion method and we take the graphical fisheye views (GFV) [49] and fisheye transform (FET) [120] as examples. The transformation functions of these two methods are displayed as follows:

$$T_{GFV} = \frac{(1+\lambda)\,x}{\lambda x + 1}, \quad T_{FET} = \frac{\ln(1+\lambda x)}{\ln(1+\lambda)} \tag{4.7}$$

where distortion factor λ controls the amount of the distortion and x is the distance from a point under consideration to the point of focus. x is a normalized distance which can have a value between 0 and 1. To better illustrate their difference, we plot two transformation functions under different distortion factors in Fig. 4.4.

The distortion models in these distortion methods are fixed. Though the user can change the distortion factor and interactively select the focus, the user cannot modify the transformation function on data distribution. Compared with our proposed method in which the user can change the transformation model according to the data distribution, the difference is significant.

4.3.2.3 Formulation in Polar Coordinate

As RadViz is a radial configuration, the projected data points can also be presented in the polar coordinate system. We use radial distance r to present the radial coordinate and orientation θ to present the angular coordinate. Then the data point $p^{RV}(x_j, y_j)$ in Cartesian coordinates can be converted to $p^{RV}(r_j, \theta_j)$ in the polar coordinates with $0 \leq r_j \leq 1$ and θ_j in the interval $(-\pi, \pi]$ by:

$$p^{RV}(r_j, \theta_j) = \left(\sqrt{x_j^2 + y_j^2}, atan2\,(x_j, y_j)\right), \tag{4.8}$$

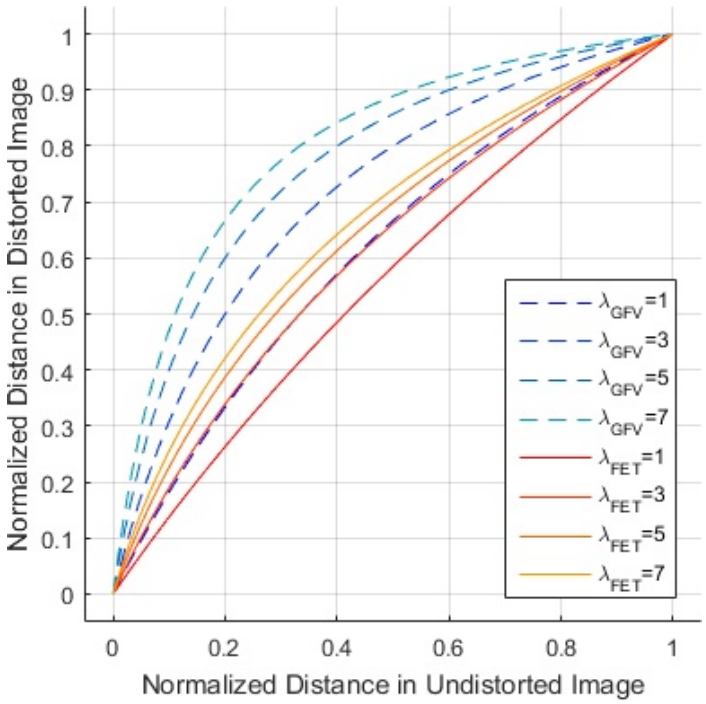


Figure 4.4. Transformation functions of GFV and FET under different distortion factors.

where $atan2\,(y,x)$ is defined as

$$atan2\,(y,x) = \begin{cases} \arctan\,(y/x) & \text{if} \quad x > 0 \\ \arctan\,(y/x) + \pi & \text{if} \quad x < 0 \quad y \geq 0 \\ \arctan\,(y/x) - \pi & \text{if} \quad x < 0 \quad y < 0 \\ \pi/2 & \text{if} \quad x = 0 \quad y > 0 \\ -\pi/2 & \text{if} \quad x = 0 \quad y < 0 \\ undefined & \text{if} \quad x = 0 \quad y = 0 \end{cases}. \tag{4.9}$$

The orientation θ_j of a data point in RadViz indicates the dimension in which the original high-dimensional data point has larger deviation. Meanwhile, the radial distance r_j shows the relative extent of the deviation. RadViz is designed not for numerical analysis but to gain insights from the plotting. Users can have an overall understanding on the whole dataset in the initial exploration, and then they may know where and how to conduct further numerical analysis.

4.3.2.4 Visual Analytics in Polar Coordinate

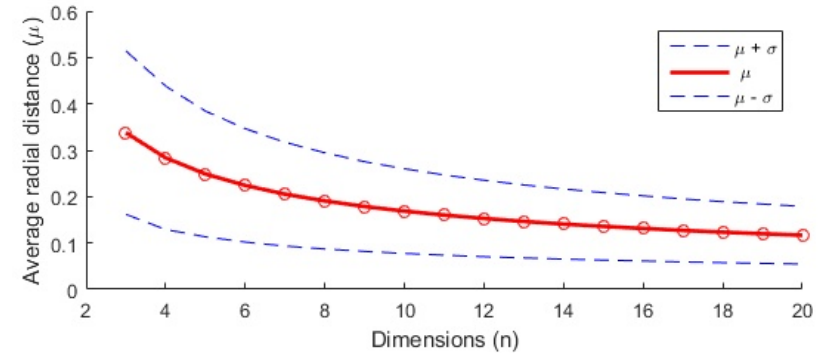
The clumping problem can be further explained by considering the radial distance of each projected data points in image space. The average radial distance μ and the standard deviation σ can be expressed as:

$$\mu = \frac{\sum_{j=1}^{m} r_j}{m}, \qquad \sigma = \sqrt{\frac{\sum_{j=1}^{m} (r_j - \mu)^2}{m}}, \tag{4.10}$$

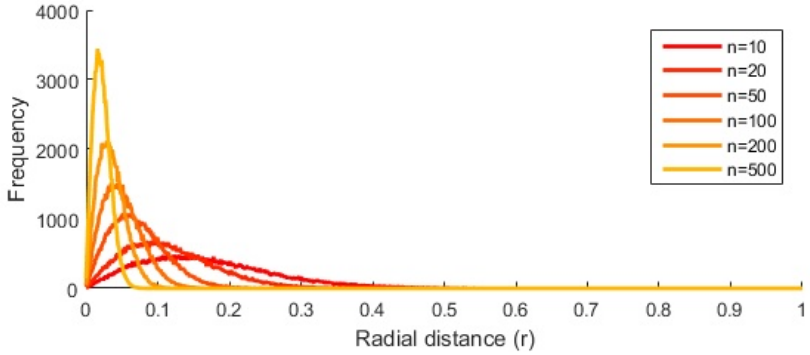
where m is the amount of data points.

Firstly, how μ and σ change under different dimensionality are explored. The average radial distances μ, $\mu + \sigma$, and $\mu - \sigma$ when plotting datasets with different dimensionality are illustrated in Fig. 4.5 (a) by considering the maximum range as one unit. The average distances decrease as the dimensionality increases in RadViz. This is one strong illustration to show the clumping effect. Secondly, the distributions of distances to the origin under different dimensionality are also plotted in Fig. 4.5 (b). As the dimensionality increases, it can be seen that the peak of the plot is skewing to the left, which means that more data points are clumping towards the center. Besides the analysis in Fig. 4.5 (a) and (b), numerical details including the average distance μ, the standard deviation σ, and $\mu + 3\sigma$ are listed in Table 4.1. The percentage of data points located in the range of $[0, \mu + 3\sigma)$ is very steady and slightly increasing as the dimensionality increases. However, we should notice the significant decrease of the value of $\mu + 3\sigma$ as the dimensionality increases. When plotting the 50D dataset, around 99.4% data points in the uniformly distributed random dataset are plotted in the range of $[0, \mu + 3\sigma)$. When it comes to 500D dataset, the range that covers around 99.4% data points is only 0.059.

According to the analysis in Fig. 4.5 and Table 4.1, the clumping problem of RadViz can be clearly expressed by using the radial distance r, but not the orientation θ. Hence, regarding the clumping problem of RadViz, we can convert it to how to handle the radial distance r in polar coordinate system. Based on this hypothesis, we consider the operations on r to ease the clumping problem in RadViz while keeping the orientation θ unchanged. We come to the ideas on the modification of r by changing the distribution of radial distance r.



(a)



(b)

Figure 4.5. The average distance of each point to the origin in the display decreases as the dimensionality increases in RadViz, as shown in (a). The plot also includes the average distance $\mu \pm$ one standard deviation σ (dashed lines). The graphics in (b) shows line histograms of the distances plotting 10 to 500-dimensional uniformly generated random dataset. The clumping effect becomes severe as the dimensionality increases. [1]

Table 4.1. Numerical analysis on the clumping effect. The average distance μ, the standard deviation σ, and value of $\mu + 3\sigma$ for different dimensionalities n are listed. The last column is the percentage of data points that located in the range of $[0, \mu+3\sigma)$. Though there is a significant decrease in $\mu + 3\sigma$, the percentage is steady.

n	μ	σ	$\mu + 3\sigma$	%
10	0.168	0.091	0.442	99.301
20	0.117	0.062	0.304	99.356
50	0.073	0.039	0.189	99.381
100	0.051	0.027	0.132	99.429
200	0.036	0.019	0.093	99.403
500	0.023	0.012	0.059	99.442

This idea is motivated by one observation: the distance distribution in Fig. 4.5 (b) is similar to the intensity histogram of a low-contrast image. In the image processing field, histogram equalization is widely used to solve this problem [121]. Hence, a straightforward and interesting idea is to use the histogram equalization as well as other histogram operations (histogram specification, histogram local equalization, and so on) to solve the clumping problem by manipulating the projected data point distribution in the basic RadViz. The work in [122] also used probability distribution histogram to enhance visualization. Their method extends a dimension to multiple new dimensions based on the histogram, and finds the optimal placement of dimension anchors for good visual clustering. In comparison, we manipulate the distribution of data points with the histogram. Without creating new dimensions and reordering DAs, our method allows a more intuitive interpretation of the layout. The detailed methodology is described in Sect. 4.3.3.1 and then generalized in Sect. 4.3.3.2.

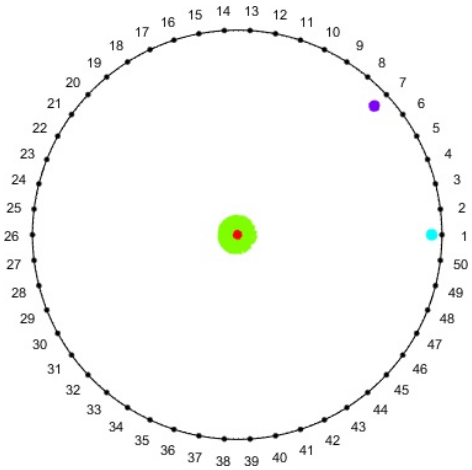
4.3.3 Detailed Methods

4.3.3.1 Radial Operations

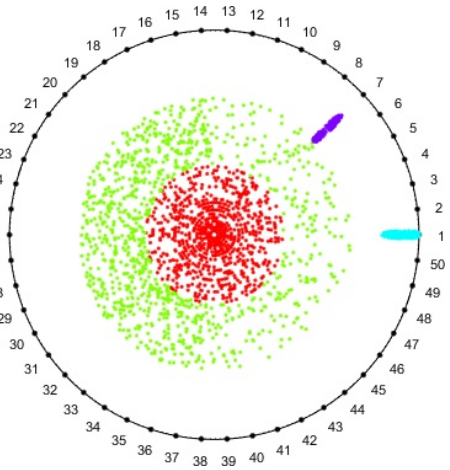
After projecting high dimensional data points into RadViz in the polar coordinate system, the radial distance r_j of data point $d_j^{'}$ will be modified according to the new distribution, while the polar angle θ_j remains unchanged. We define our proposed methods as radial operations, including radial equalization, radial specification, radial movement, and radial local equalization.

Algorithm 4.1 Radial Equalization

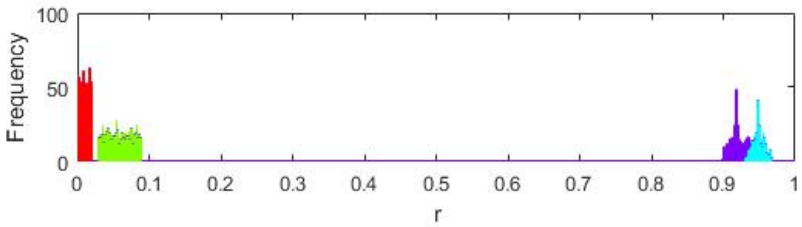
1: Dataset $D = (d_1, \cdots, d_m)$;
2: Normalize D to get $D^{'} = (d_1^{'}, \cdots, d_m^{'})$;
3: Plot $D^{'}$ in RadViz $p^{RV}(r, \theta)$;
4: Set maximum histogram level L;
5: Calculate histogram H by only considering r;
6: Histogram equalization: $H^{'} = T_{equal}(H)$;
7: Calculate the corresponding distance $r^{'}$ using $H^{'}$;
8: Calculate the new RadViz $p^{RV^{'}}(r^{'}, \theta)$;
9: Plot $p^{RV^{'}}(r^{'}, \theta)$;



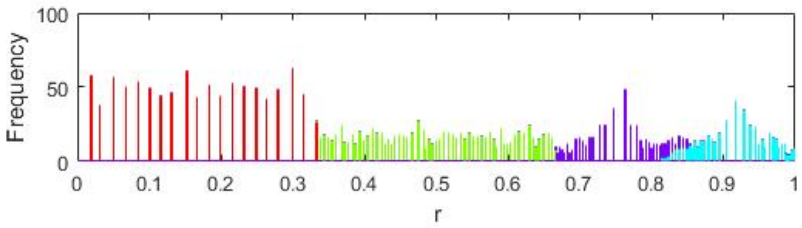
(a) Basic RadViz



(b) Radial Equalization



(c) Histogram Before Equalization



(d) Histogram After Equalization

Figure 4.6. (a) An example with data points located near the circle edge; (b) RadViz plotting after radial equalization. The histogram of original RadViz is plotted in (c). After histogram equalization, the histogram is plotted in (d).

Radial Equalization We propose a radial equalization technique here by combining histogram equalization and basic RadViz. For the RadViz $p^{RV}(r,\theta)$ in the polar coordinate system, the orientation θ is preserved. The distribution of the radius r of all projected data points is shown in the histogram, and then a histogram equalization method is implemented to manipulate the radial distribution.

The pixel values of an image are discrete in the range of $[0, 255]$. However, in basic RadViz, when using the distance to the origin as the criteria to plot a histogram, the values are continuous. Hence, firstly, we need to specify the bins to discretize the distance values. The polar coordinate system is adopted to represent the basic RadViz in which the distance value can be obtained directly. For a n-dimensional data point d_j in dataset D, its corresponding projected data point in polar coordinate is $p^{RV}(r_j, \theta_j)$. The whole distance range ($[0, 1]$) is digitized into L bins ($\{H_0, H_1, \cdots, H_{L-1}\}$). L is set as 1000 in this paper. Let h_k denotes the total number of projected data points whose r_j are located in the range of $[H_k, H_{k+1})$, then the probability density function (PDF) is

$$p(H_k) = \frac{h_k}{m}, \quad 0 \leq k < L, \tag{4.11}$$

where m is the number of data points in dataset D. The cumulative distribution function (CDF) is defined as

$$c(H_k) = \sum_{0}^{k} p(H_k), \quad 0 \leq k < L. \tag{4.12}$$

Thus the transform function of histogram equalization $T(x)$ can be defined as

$$T_{equal}(H_k) = H_0 + (H_{L-i} - H_0) \cdot c(H_k). \tag{4.13}$$

Suppose $p^{RV'}(r_j', \theta_j)$ is defined as the RadViz with equalized radial distance r_j', then

$$p^{RV'}(r_j', \theta_j) = (T_{equal}(r_j), atan2(x_j, y_j)). \tag{4.14}$$

The equalization process is described in Algorithm 4.1. To better visualize the proposed algorithm, we design a dataset with four clusters ('red', 'green', 'blue', and

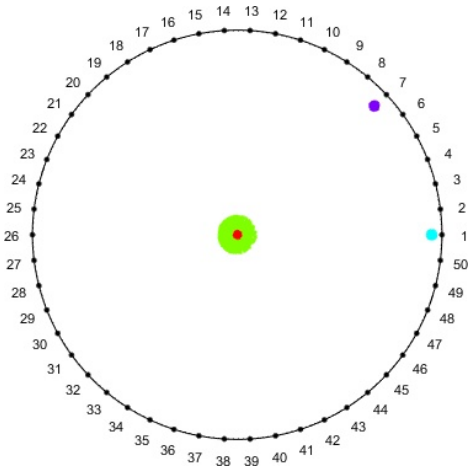
'purple') as shown in Fig. 4.6 (a). For this synthetic example that cannot be handled by stretching, the radial equalization method is used to ease the clumping and increase the space utilization. Fig. 4.6 (b) illustrates the result after employing the radial equalization method. The distance histograms of Fig. 4.6 (a) and (b) are plotted in Fig. 4.6 (c) and (d), respectively.

Radial equalization can automatically calculate the transformation function to reshape the distribution. This eases the burden of users. In some cases, however, the configuration obtained by the radial equalization may not be as desired by the users and the original patterns may be destroyed. The radial specification can address this issue, which allows the user to specify a histogram distribution. By employing these techniques, users can see a specific part of the dataset and explore the patterns or features.

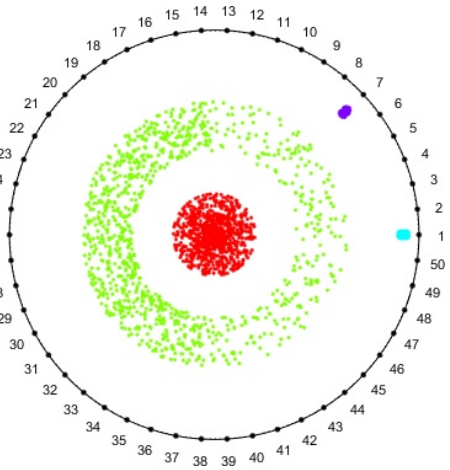
Algorithm 4.2 Radial Specification

1: Dataset $D = (d_1, \cdots, d_m)$;
2: Normalize D to get $D^{'} = (d_1^{'}, \cdots, d_m^{'})$;
3: Plot $D^{'}$ in RadViz $p^{RV}(r, \theta)$;
4: Set maximum histogram level L;
5: Calculate histogram H and the CDF $c(H)$ by only considering r for the original RadViz;
6: Calculate the CDF $c^{'}(H^{'})$ for the user specified histogram;
7: Find the histogram level $H_j^{'}$ for which $c(H_i) = c^{'}(H_j^{'})$;
8: Then the histogram specification transformation function: $H_j^{'} = T_{specify}(H_i)$;
9: Calculate the corresponding distance $r^{'}$ using $H^{'}$;
10: Calculate the new RadViz $p^{RV^{'}}(r^{'}, \theta)$;
11: Plot $p^{RV^{'}}(r^{'}, \theta)$;

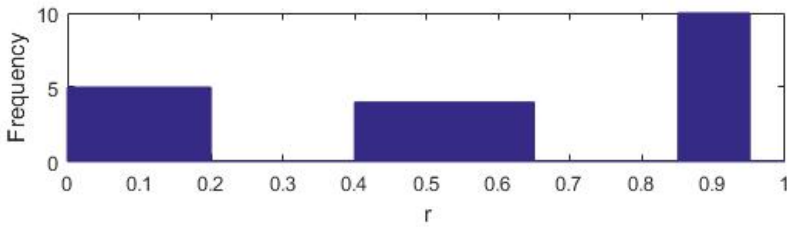
Radial Specification The radial specification technique accepts a user-specified histogram distribution as an input to reshape the RadViz. When the clustering of projected data points is not clear or users have special requirements on the distribution, the radial specification can generate a required configuration. The dataset in Fig. 4.6 (a) is again used here as an example to show the radial specification results. In Fig. 4.7 (c), a specified histogram is given and the corresponding plotting is illustrated in Fig. 4.7 (b). Compared with the radial equalization result in Fig. 4.6 (b), which plots clusters together, the radial specification results in Fig. 4.7 (b) shows the



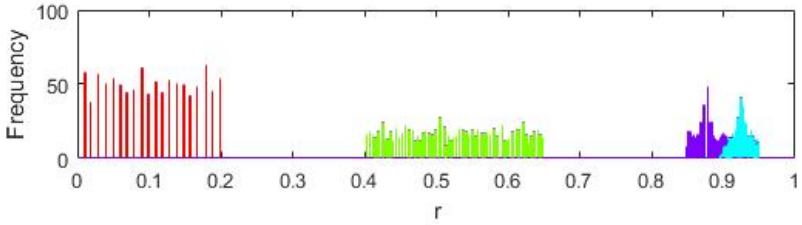
(a) Basic RadViz



(b) Radial Specification



(c) User Specified Histogram



(d) Histogram After Histogram Specification

Figure 4.7. (a) An example with data points located near the circle edge; (b) After radial specification, the RadViz is illustrated. The histogram in (c) is user-specified. After radial specification, the histogram is plotted in (d).

pattern in a visually clearer way.

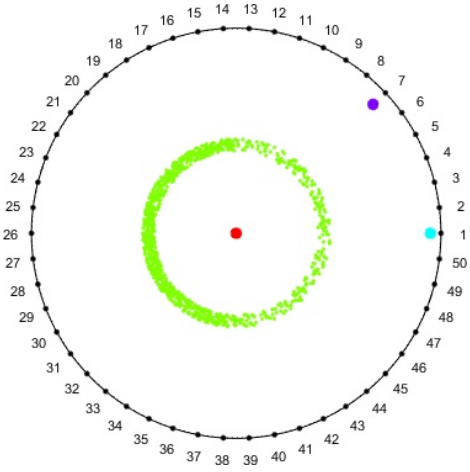
The advantage of the radial specification is that it displays the data points in RadViz exactly as the user's desire, and the radial specification operation will not destroy the relative distance of points to the origin. Let $H^{'}$ denotes the user specified histogram and $c^{'}(H)$ be the corresponding CDF. Then the specification process can be described in Algorithm 4.2. Compared with the radial equalization operation in Algorithm 4.1, the radial equalization is a special case of radial specification by using a uniform specified histogram as user input.

Radial Movement Besides bringing in a new histogram, the user also can edit the current histogram. Radial movement operation allows the user to select a range of bars in the histogram, and move the selected bars to another location. No collision with other bars is allowed during this movement in order to preserve the relative relationship. Based on the histogram in Fig. 4.6 (c), the middle cluster is selected and moved to the position of around the 400th bin and the result is shown in Fig. 4.8 (c). With this modified histogram, the original RadViz in Fig. 4.6 (a) is changed into Fig. 4.8 (a). As shown, cluster 'green' is now separated from cluster 'red' via the radial movement.

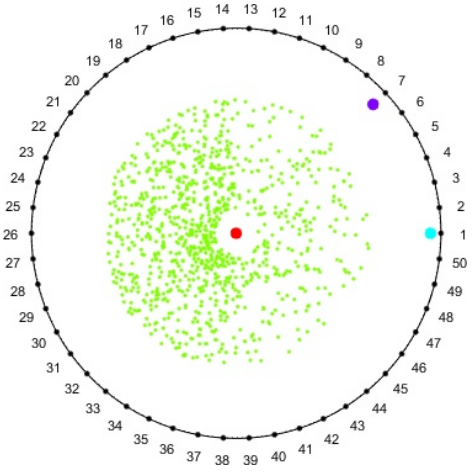
Radial Local Equalization Radial local equalization is conducted in the histogram by selecting a partial range and then implementing radial equalization in this range only. This operation is used to increase local contrast. In the example shown in Fig. 4.6 (a), after radial movement, we implement radial local equalization on cluster 'green' to increase the local contrast. The range $[100, 650]$ is selected for local equalization. The resulting histogram and RadViz plotting are shown in Figs. 4.8 (d) and (b) respectively.

4.3.3.2 PolarViz

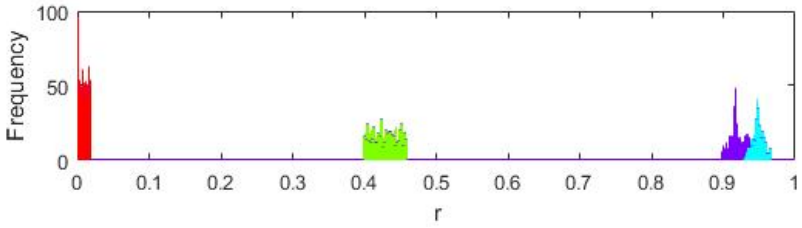
The proposed histogram-based radial operations in the previous four subsections can flexibly control the radial distribution by importing specified distribution or editing current distribution or both. Although these operations are defined in the histogram,



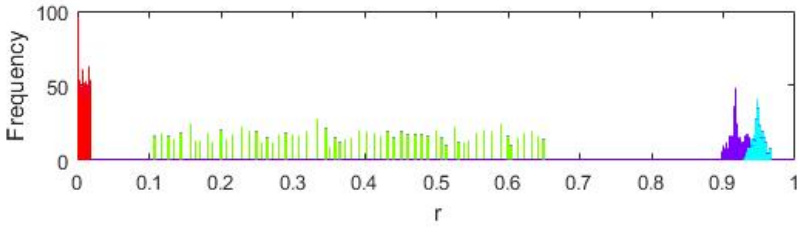
(a) Radial Movement



(b) Radial Local Equalization



(c) Histogram After Radial Movement



(d) Histogram After Radial Local Equalization

Figure 4.8. (a) The RadViz plotting after radial movement; (b) The RadViz plotting after radial local equalization. After radial movement, the obtained histogram is plotted in (c). After radial local equalization, the obtained histogram is plotted in (d).

the CDF is used for calculation. All the operations can be treated as the matching between the CDF before transformation and the CDF after transformation. Hence, we can generalize the radial operation by considering the matching of different CDFs. Let $c(H)$ be the CDF of the current RadViz while $c^{'}(H)$ being the new CDF, then the histogram level $H_j^{'}$ in $c^{'}(H)$ for which $c(H_i) = c^{'}(H_j^{'})$ is found. The result of this matching is the generalized radial transformation function.

Based on this, we propose the PolarViz plotting which can be treated as a series of radial operations to modify the radial distribution of RadViz. Let $T(\cdot)$ be the operator of the generalized radial distribution transformation, then the PolarViz p_{polar}^{RV} can be expressed as

$$p_{polar}^{RV}(r_j^p, \theta_j^p) = \left(T\left(r_j\right), atan2\left(x_j, y_j\right)\right). \tag{4.15}$$

4.3.4 Comparative Studies

The comparison with other methods that can be used to solve the clumping problem is described in this section. As the clumping problem is formed during the projection from data space to image space, the comparative studies are divided into two parts: comparison with methods operated in the data space (Sect. 4.3.4.1) and comparison with methods operated in the image space (Sect. 4.3.4.2).

For the methods operated in the data space, we execute the Ono's method [119] in which the data point in the high-dimensional space is filtered by a sigmoid function. In the experiment, we set the parameter of the RadViz with sigmoid function (p_s^{RV}) with $s = 15$ and $t = -0.5$.

For the methods operated in the image space, distortion methods for radial layout (i.e. GFV [49] and FET [120]) are implemented for comparison. For GFV, we set λ_{GFV} to 9 while setting λ_{FET} for FET to 7 in our experiment. The values of λ_{GFV} and λ_{FET} can be changed as well as the s and t for p_s^{RV}.

Two datasets are used in the comparison experiment. The first one is the four-dimensional dataset 'IRIS' in which three clusters are involved. While the second dataset is a simulation dataset generated by the EngineSim. The EngineSim devel-

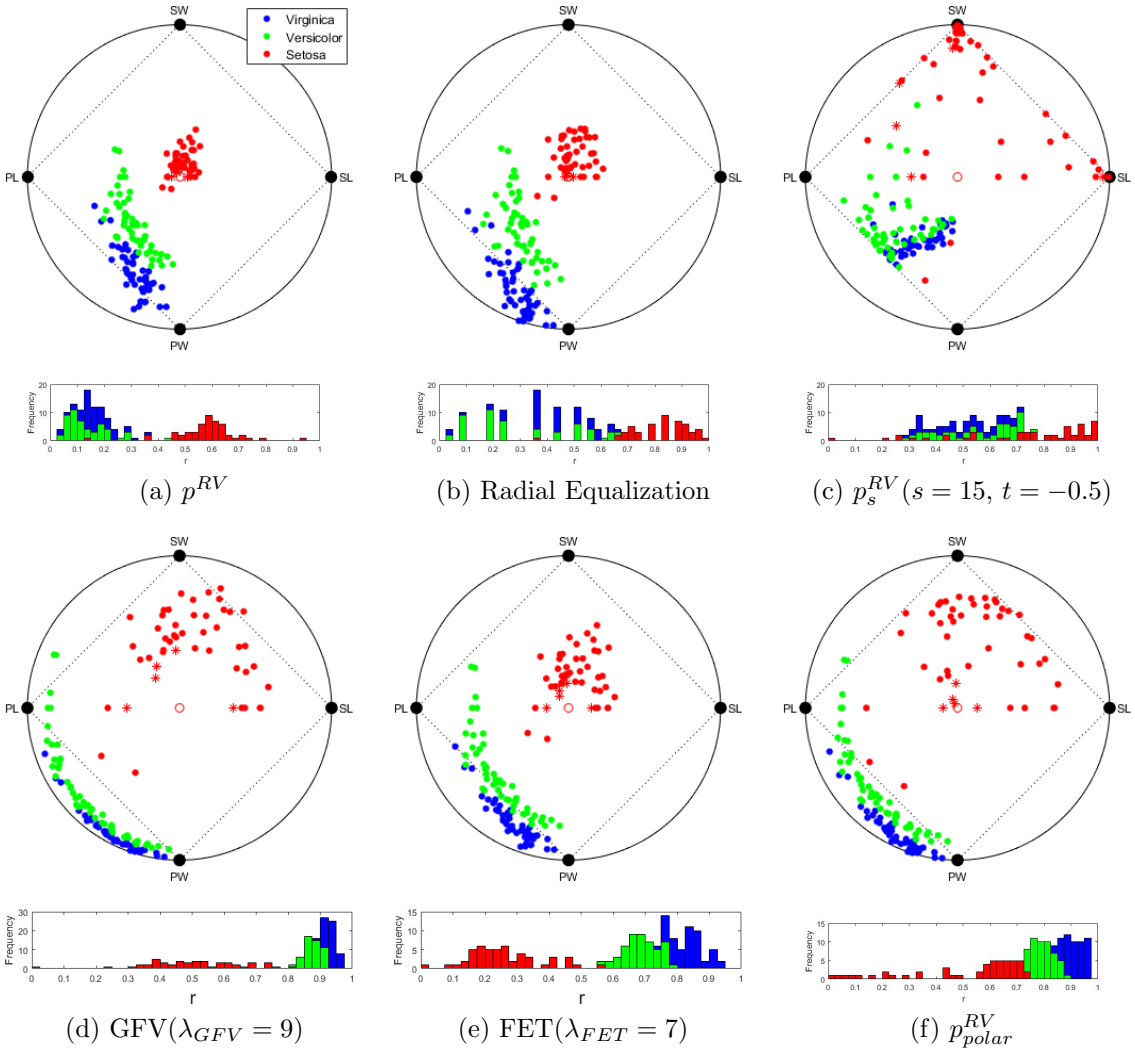
oped by NASA [123] is a turbine engine simulator used in our project to assess the performance of an engine. We designed 3444 input data points that have 16 parameters for the Turbo Fun engine simulator and then obtained 3444 output data points. The 3444 output data points with 38 dimensionalities are used here.

With these two datasets, we aim to visualize the center region of the views after projection into image space. The center point is padded with zero value for each dimension and few nearest data points are highlighted with different symbols. Due to the clumping problem, these points are hard to view in the basic RadViz plot. Different methods are compared at this stage to provide a detailed view of the center region while retaining the surrounding context to help keep analysts oriented [43].

4.3.4.1 RadViz with Sigmoid Function

Iris As the filter in p_s^{RV} is nonlinear and the projection method in RadViz is also nonlinear, the resulting view of p_s^{RV} will have quite different distribution when compared with the basic RadViz. By using the benchmark dataset 'IRIS', we plot the views obtained from the basic RadViz (p^{RV}) and p_s^{RV} and the PolarViz (p_{polar}^{RV}) in Fig. 4.9 (a) and (c) and (f), respectively.

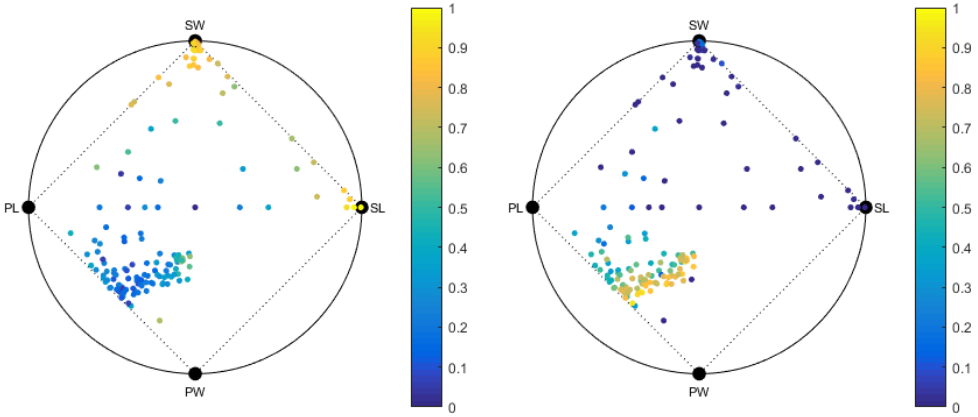
Compared with the radial equalization operation on the basic RadViz result (Fig. 4.9 (b)), p_s^{RV} does not adhere to the meaning of RadViz plot in the image space. Although p_s^{RV} can significantly solve the clumping problem in this case, the position relationship among data points in the image space is destroyed. In Fig. 4.9(c), the neighbors of the center point are not the nearest data points to the center anymore. To better understand the position change in these plots, we compare the data point position difference between p_s^{RV} and p^{RV} in Fig. 4.10 (a) while plotting the position difference between p_{polar}^{RV} and p^{RV} in Fig. 4.10 (d). The position difference is calculated by considering the Euclidean distance of the same data point in different views. For each comparison, the difference is normalized into the interval of $[0, 1]$ and then colorized accordingly. The color of data point in Fig. 4.10 (a) shows that the position change in p_s^{RV} is irregular. Meanwhile, in Fig. 4.10 (d), a gradient ramp in the radial direction can be observed which shows a more regular radial modification.



(a) p^{RV} (b) Radial Equalization (c) $p_s^{RV}(s = 15,\ t = -0.5)$

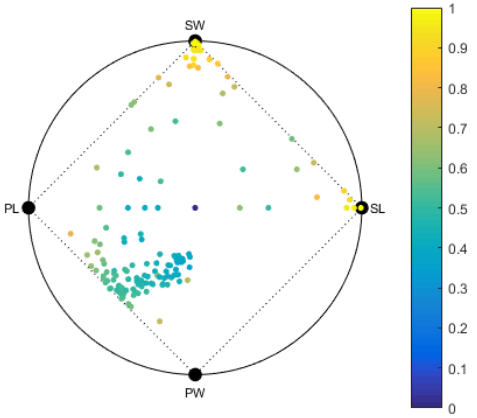
(d) GFV($\lambda_{GFV} = 9$) (e) FET($\lambda_{FET} = 7$) (f) p_{polar}^{RV}

Figure 4.9. We color clusters and use different symbols for different query-related data points. The original result using basic RadViz (p^{RV}) is shown in (a) while radial equalization result is shown in (b). The result of the RadViz with sigmoid function (p_s^{RV}) is illustrated in (c). The results of two distortion methods, GFV and FET, are illustrated in (d) and (e), respectively. In the end, the proposed PolarViz (p_{polar}^{RV}) is shown in (f).

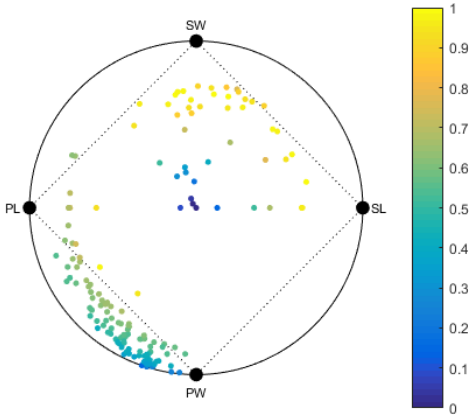


(a) Position difference ($p_s^{RV} - p^{RV}$)

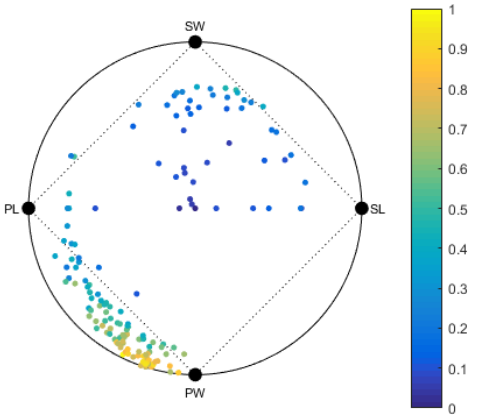
(b) Data attributes variance (p_s^{RV})



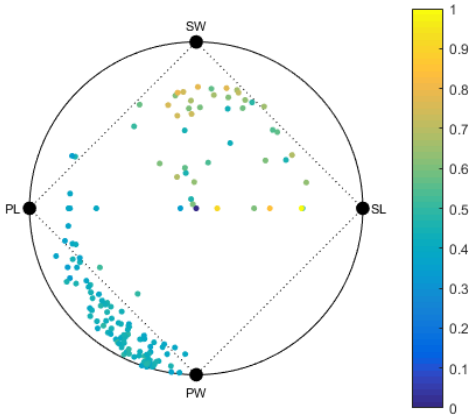
(c) Attribute ratio (p_s^{RV})



(d) Position difference ($p_{polar}^{RV} - p^{RV}$)



(e) Data attributes variance (p_{polar}^{RV})



(f) Attribute ratio (p_{polar}^{RV})

Figure 4.10. (a) The position difference of same data point is plotted in p_s^{RV} (Fig. 4.9(c)) and basic RadViz p^{RV} (Fig. 4.9(a)). (b) The attribute values variance of a data point is plotted in p_s^{RV}. (c) The ratio of the largest attribute value to the mean attribute value is plotted in p_s^{RV}. (d) The position difference of same data point is plotted in p_{polar}^{RV} (Fig. 4.9(f)) and basic RadViz p^{RV} (Fig. 4.9(a)). (e) The attribute values variance of a data point is plotted in the PolarViz (p_{polar}^{RV}). (f) The ratio of the largest attribute value to the mean attribute value is plotted in the PolarViz (p_{polar}^{RV}).

It does not mean that p_s^{RV} is useless in data visualization, as the design of p_s^{RV} is to highlight the attributes with large values [119]. In this case, we consider p_s^{RV} from two perspectives. Firstly, we calculate the attribute value variance for each data point in p_s^{RV} and p_{polar}^{RV} (Figs. 4.10 (b) and (e)). Secondly, we pay attention to the ratio of the largest attribute value to the attribute mean for each data point (Figs. 4.10 (c) and (f)). The attribute variance and attribute ratio for each data point d_j is calculated as follow:

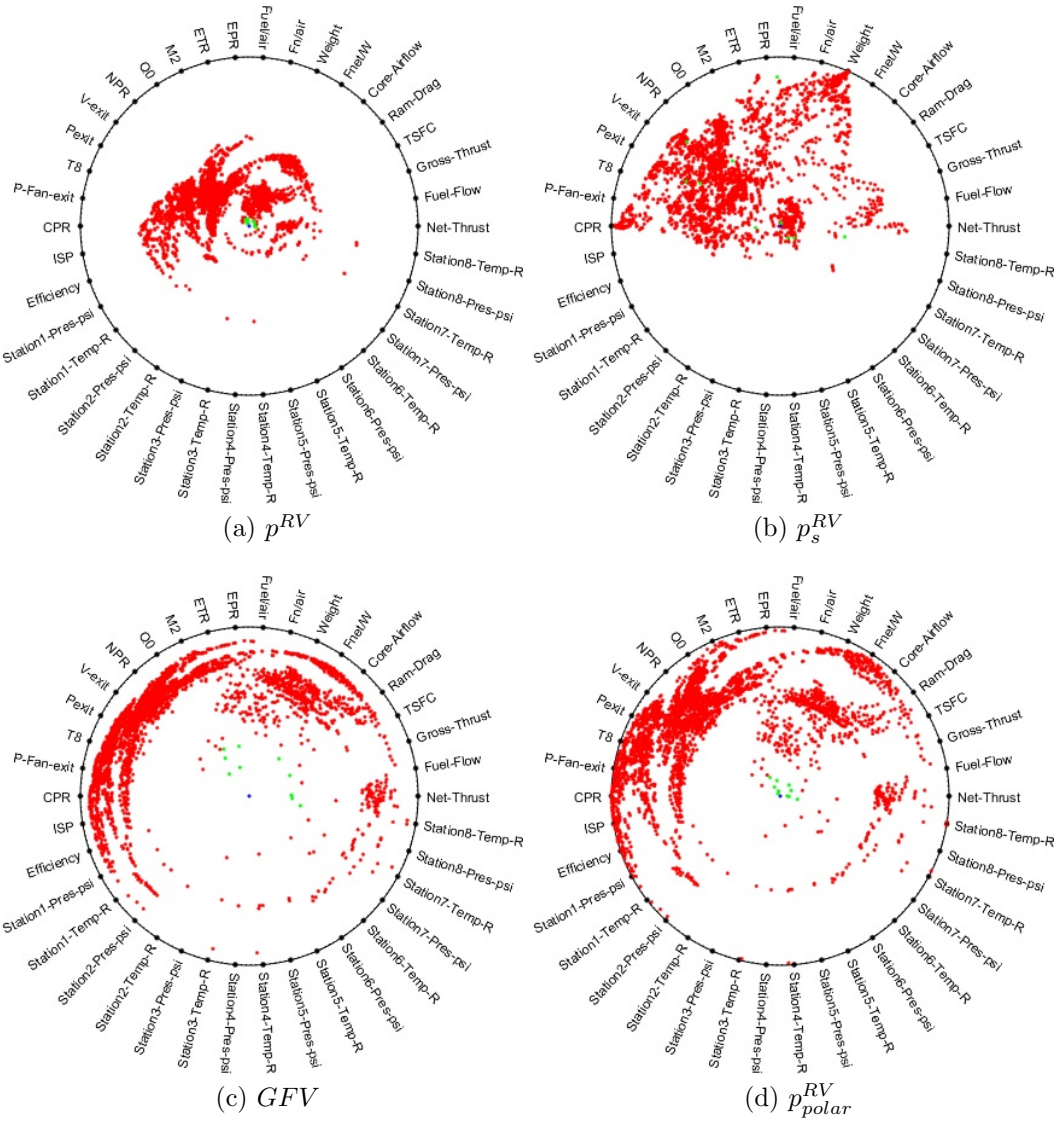
$$AttributeVariance = \sqrt{\frac{1}{n}\sum_{i=0}^{n}\left(d_{i,j} - \overline{d}_j\right)^2} \tag{4.16}$$

$$AttributeRatio = \frac{\max\left\{d_{i,j}\right\}, \forall i}{\sum_{i=0}^{n} d_{i,j}} \tag{4.17}$$

where $\overline{d}_j = \frac{1}{n}\sum_{i=0}^{n} d_{i,j}$. The basic RadViz plots the data point which has small attribute variance near the center and plots the data point with large attribute variance away from the center. This character is preserved in p_{polar}^{RV} as shown in Fig. 4.10 (e). It seems that the attribute variance in p_s^{RV} is not associated with that in p_{polar}^{RV} according to Fig. 4.10 (b). The main property of p_s^{RV} is that after using the sigmoid function to filter the original dataset, the attribute with large values previously will have larger results while small values will have smaller results. Hence, we further calculate the ratio of the attribute with largest value to the mean value in each data point. p_s^{RV} shows clear insight in Fig. 4.10 (c). p_s^{RV} plots data points which have largest values in attribute 'SL' and 'SW' near these two DAs respectively. The dots located between 'PL' and 'PW' indicate that these data points have two attributes with similar values. Meanwhile, p_{polar}^{RV} in Fig. 4.10 (f) cannot show this information.

NASA EngineSim Another example is using the EngineSime simulation dataset. Compared with 'IRIS', there is no cluster label in the EngineSim dataset. We plot the visualization result of p^{RV}, p_s^{RV}, and p_{polar}^{RV} in Figs. 4.11 (a), (b), and (d), respectively. As more data points are involved in the EngineSim dataset, the result of p_s^{RV} still has many points crowded in the center region after the sigmoid filtering. In this case, the advantage of p_{polar}^{RV} over p_s^{RV} is more obvious.

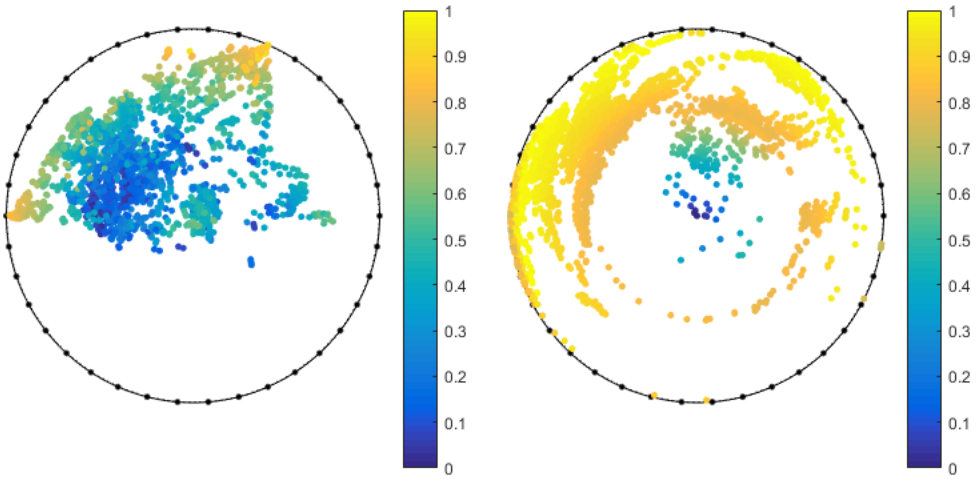
We also plot the position difference of these three plots and the results are shown



(a) p^{RV} (b) p_s^{RV}

(c) GFV (d) p_{polar}^{RV}

Figure 4.11. The basic RadViz p^{RV} (a), p_s^{RV} (b), GFV with $\lambda_{GFV} = 9$ (c), and p_{polar}^{RV} (d) are used to plot the EngineSim dataset.



(a) Position difference ($p_s^{RV} - p^{RV}$) (b) Position difference ($p_{polar}^{RV} - p^{RV}$)

Figure 4.12. In (a), we plot the position difference of data points in p_s^{RV} (Fig. 4.11(b)) and basic RadViz p^{RV} (Fig. 4.11(a)). Then in (b) we compare the position difference between p_{polar}^{RV} (Fig. 4.11(d)) and basic RadViz p^{RV} (Fig. 4.11(a).)

in Fig. 4.12. Similarly, like the result in Fig. 4.12, the position change in p_s^{RV} does not adhere to the meaning of RadViz plot.

4.3.4.2 Distortion Methods

Techniques, such as distortion methods (GFV and FET), that can be used to solve the clumping problem in the image space are also compared with the proposed PolarViz using the two datasets, the 'IRIS' and the EngineSim dataset. The results of the distortion methods on the two datasets are plotted in Fig. 4.9 (d), (e) and Fig. 4.11 (c), respectively.

The distortion methods are also operated along the radial direction. Hence, they can produce similar results as the PolarViz when dealing with the clumping problem. However, the major difference is that in the distortion methods the transformation functions are fixed. Once the distortion parameter is determined, the user cannot focus only on one region while keeping other parts unchanged. The distortion generated is hard to control in these distortion methods, making them difficult to meet the various requirements. Take the result shown in Fig. 4.11 (c) as an example. The center region is too distorted while most of the data points are pushed towards the bounding circle. In this case, as the transformation function is fixed, the user cannot modify the view. In contrast, p_{polar}^{RV} has more space for the rest of data points while the center region

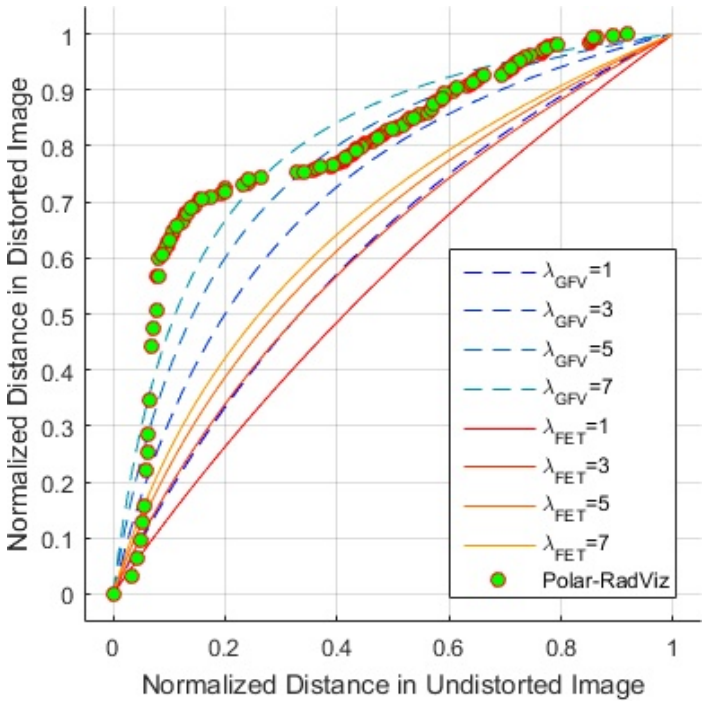


Figure 4.13. We compare the transformation functions of the view in Fig. 4.9 (f) with Fig. 4.4.

is not distorted. The user can specify the distribution to satisfy the requirement. The transformation function of p^{RV}_{polar} as well as those of the distortion methods with different parameters are plotted in Fig. 4.13.

The attribute variance, as well as the attribute ratio, are not calculated in the comparison between the distortion methods and the PolarViz. The PolarViz and the distortion methods are operated in the image space. Hence, they will not change the original data values. The attribute variance and the attribute ratio for each data point are preserved during plotting.

The number of bins used in the radial operations is of great importance. The more bins used, the more accurate the result is. In Fig. 4.9 (b) and (f) when plotting the radial equalization result and p^{RV}_{polar}, we use 1000 bins. Meanwhile, to better illustrate the overlapping of different clusters, we use 40 bins in the corresponding histograms.

4.3.5 User Study

We performed a user study to assess the usability of the proposed PolarViz. The user study was divided into two parts. In the first part, participants in the user study were taught some background knowledge about the RadViz, the clumping problem, and

Table 4.2. The user study process and the corresponding questionnaire are described in this table.

User Study	**Questionnaire**
· Introduce the RadViz plot by using the spring-force model; Then using the equations in the Cartesian coordinate system; Then using the equation in the Polar coordinate system.	**Q1**: Sort the description models (the spring-force model, the equations in the Cartesian coordinate, and the equations in the polar coordinate) of RadViz from 'easy-to-understand' to 'difficult-to-understand'.
· Introduce the clumping problem in RadViz by using the Cartesian coordinate system; Then a polar coordinate system.	**Q2**: Sort the description models (the Cartesian coordinate system and the polar coordinate system) of the clumping problem in RadViz from 'easy-to-understand' to 'difficult-to-understand'.
· Introduce the parameters of each method (RadViz p^{RV}, p_s^{RV}, GFV, FET, and p_{polar}^{RV}).	
· Participants were asked to modify the parameters of each method to achieve a focus+context purpose.	**Q3**: Sort the methods for the clumping problem regarding to performance and operation difficulty from easy to difficult.

the methods for solving the clumping problem. In the second part, participants were asked to explore different methods by themselves with a certain target and then sort these methods based on their performance.

In the background knowledge introduction part, we first introduced the RadViz plot by using the spring-force model and then using equations in the Cartesian coordinate system and then followed by equations in a polar coordinate system. Secondly, we introduced the clumping problem in RadViz by using the Cartesian coordinate system and a polar coordinate system, respectively. Finally, we briefly introduced the parameters of each method without the theory. In the second part, three datasets including the 'IRIS' dataset, the EngineSim dataset, and a random dataset were used. We plotted the three datasets into the basic RadViz and colorized the center point and its nearest neighbors. The RadViz plot is used as a reference. We then plotted

the three datasets by using p_s^{RV}, GFV, FET, and p_{polar}^{RV} respectively. Participants were asked to modify the parameters of each method to achieve a focus+context purpose. It requires a focus on the center region as well as a clear view of the highlighted data points. Meanwhile, the pattern of the rest should be preserved as much as possible. The user study content is summarized in the left column of Table 4.2.

Fourteen graduate students and researcher staff (11 male, 3 female) participated in our user study. None of them reported to be familiar with information visualization before, and 11 of them reported to be familiar with the general spring-force model as well as the equation expression, and 10 of them knew about the fish-eye distortion. The average length of the study was about 30 minutes as around 15 minutes were taken for the background knowledge introduction. Three questions were asked in the user study. In the first question, participants were asked to sort the description models of RadViz from 'easy-to-understand' to 'difficult-to-understand'. In the second question, participants were asked to sort the description model of the clumping problem in RadViz from 'easy-to-understand' to 'difficult-to-understand'. In the third question, participants were asked to sort the methods for the clumping problem regarding to performance and operation difficulty. The questionnaires are summarized in the right column of Table 4.2. The overall opinion indicated that the proposed p_{polar}^{RV} outperformed the other methods. Specifically, the opinions for each question are as follows:

Q1: All the participants favored the spring-force model. 13 of them mentioned that the equations of the RadViz in the Cartesian coordinate are not difficult to understand while only 1 participant supported the equation expression in the polar coordinate rather than the Cartesian coordinate. One reason is that using the spring-force model to explain the RadViz plot makes sense. Meanwhile, the evidence that most of the participants were familiar with the spring-force model as well as the equation expression in the Cartesian coordinate may be another reason.

Q2: 10 participants voted the clumping problem analysis in the polar coordinate system while 4 participants thought the analysis in the Cartesian coordinate

system and in the polar coordinate system are the same.

Q3: 11 participants voted p_{polar}^{RV} as the easiest method while 3 participants favored one of the distortion methods (GFV or FET). The reason why we put GFV and FET together is that during the study, comments provided by some participants indicate that they failed to find the difference between GFV and FET. Hence, we treated the GFV and FET as one method. In this case, p_{polar}^{RV} was 11 times in the first position and 3 times in the second position. Meanwhile, the GFV or FET methods were 3 times in the first position and 11 times in the second position. The p_s^{RV} was 14 times in the third position which means that all the participants found it difficult to meet the objective.

The user study results support the observation that the proposed p_{polar}^{RV}, which uses the histogram to modify the radial distribution, is easy to understand and provide more flexibility.

4.3.6 Discussion

Regarding operation accuracy and computational performance, the proposed PolarViz method not only has a strong relationship with the dimensionality n and the total amount of data points m, but also largely depends on the number of bins L when doing digitization. To plot PolarViz, firstly the high-dimensional dataset needs to be projected into a low dimension space. The computational complexity of this mapping is $O(mn)$. Secondly, once we have the mapping result, the complexity of digitization is $O(m)$. Then no matter how many data points we have, after digitization there are L bins. Thirdly, for the radial operations, the complexity to handle all these bins is $O(L)$ and the complexity of re-plotting data points is $O(m)$. Hence, the total complexity of PolarViz is $O(mn + m + L + m)$. The value of n, m, and L will influence the final complexity of PolarViz.

Nevertheless, the proposed techniques have a limitation that the current distortion is only focusing on the center region of the plot. If the PolarViz is implemented in another region, confusion on the plot understanding and radial operations may arise.

4.4 Boundaries of the Star Coordinate

4.4.1 Introduction

In this section, we propose a generic solution to the boundary evaluation of SC plot, which has standard and arbitrary configurations, by introducing optimal configuration for special datasets. To start, we propose a geometric algorithm to determine the boundary of SC with standard configuration which means all the axes are evenly distributed. The boundary includes the axis-aligned bounding box and bounding polygon. The detailed algorithm on how to calculate the bounding polygon, is illustrated by an example of a 7-dimensional dataset. Then, the algorithms to calculate the axis-aligned bounding box, the bounding polygon, and the minimum bounding box of SC with arbitrary configurations, as well as geometric character analysis, are described. These boundaries enable the plot of subsequent data points based on the current configuration while not having to dynamically adjust the display window. Moreover, the determination of the boundary benefits the understanding of SC plot in several ways. Firstly, based on the determined boundaries, we amend the expression of SC. By using the new expression, not only subsequent data points can be plotted appropriately, but also the window size can be customized with arbitrary configurations. Secondly, the SC plot can be associated with the well accepted visualization method RadViz [33]. The boundaries, providing a fair comparison between RadViz and the SC by fixing the plotting size, are helpful to reformulate the relationship between these two popular plotting methods and make use of the advantages of them. Finally, the boundaries enable the property analysis of SC plot as the dimensionality increases. One example is the analysis on clumping effect which is often known as the effect that forces data points towards the center of the plotting during dimension reduction. As the range of SC changes along the dimensionality, the determination of boundary makes the clumping effect analysis clear.

The main contributions of this section are:

- we are first to calculate the boundaries of SC plot with standard and arbitrary configurations as well as some geometric properties of the boundaries;

- we amend the expression of SC with an arbitrary configuration for subsequent plotting and customized window size;
- we reformulate the relationship between RadViz and standard SC by considering the SC boundary;
- we analyze the clumping effect of SC with a standard configuration.

In the following, we describes the boundaries of SC with standard configurations as well as related analysis in Sect. 4.4.2, and the boundaries of SC with arbitrary configurations in Sect. 4.4.3. Applications to utilize the findings based on the boundaries are given in Sect. 4.4.4.

4.4.2 Boundary of SC with standard configuration

The boundaries of SC with standard configuration, including axis-aligned bounding box and bounding polygon, are studied in this section. The SC with standard configuration is defined, such that the magnitude of radially even distributed axes vector is one. Thus, following Eq. 4.3, in standard SC, $L_i = 1$ for $\forall i$. The value of each angle θ_i is

$$\theta_i = \frac{2\pi}{n} \cdot (i-1) + \theta_1 \qquad i \in \{1, \ldots, n\}, \tag{4.18}$$

where θ_1 refers to the angle of the first axis vector. The range of θ_i is $[0, 2\pi)$. As *sin* and *cos* functions are periodic, θ_i other values can be converted into this interval. In this paper, without loss of generality, the assumption that in standard SC the angle of the $\overrightarrow{v_1}$ is equal to zero ($\theta_1 = 0$) is made.

4.4.2.1 Axis-aligned Bounding Box

The bounding box that aligns with the axes of the Cartesian coordinate system is known as the axis-aligned bounding box (AABB). AABB as the simplest bounding volume is commonly used to contain more complex objects. We study the AABB of standard SC first.

Proposition 4.4.1. *Let* $[X_{\max}, X_{\min}, Y_{\max}, Y_{\min}]$ *be the right, left, top, and bottom side of the AABB which contains the whole SC plotting. In standard SC when plotting an*

n-dimensional dataset, the four sides of the AABB coordinates are

$$\begin{cases} X_{\max} = \sum_{i \in I} \cos\theta_i \quad with \quad I = \{i \in \mathbb{N} | \cos\theta_i \geq 0\} \\ X_{\min} = \sum_{i \in I} \cos\theta_i \quad with \quad I = \{i \in \mathbb{N} | \cos\theta_i \leq 0\} \\ Y_{\max} = \sum_{i \in I} \sin\theta_i \quad with \quad I = \{i \in \mathbb{N} | \sin\theta_i \geq 0\} \\ Y_{\min} = \sum_{i \in I} \sin\theta_i \quad with \quad I = \{i \in \mathbb{N} | \sin\theta_i \leq 0\} \end{cases}, \tag{4.19}$$

where $\mathbb{N} = \{1, \cdots, n\}$.

Proof. Here, we take $X_{\max}$ as an example. To get the maximum value in the positive direction, all vectors $\overrightarrow{v_i}$ are projected onto the x-axis. According to Eq. 4.3, if $\cos\theta_i > 0$, then set the corresponding value $d'_{i,j}$ to one. Otherwise, set the $d'_{i,j}$ to zero. The values of $X_{\min}$, $Y_{\max}$, and $Y_{\min}$ can be obtained in a similar way. □

Corollary 4.4.1. *In standard SC, AABB can also be determined as*

$$\begin{cases} X_{\max} = \frac{1}{2} \sum_{i=1}^{n} |\cos\theta_i| \\ X_{\min} = -\frac{1}{2} \sum_{i=1}^{n} |\cos\theta_i| \\ Y_{\max} = \frac{1}{2} \sum_{i=1}^{n} |\sin\theta_i| \\ Y_{\min} = -\frac{1}{2} \sum_{i=1}^{n} |\sin\theta_i| \end{cases}. \tag{4.20}$$

Proof. As it is an evenly distributed configuration in standard SC, the high-dimensional data point $d'_j = \{1, \ldots, 1\}$ will be projected to the center of SC regardless of the dataset dimensionality. Hence, $X_{\max} = -X_{\min}$ and $Y_{\max} = -Y_{\min}$. □

According to the statement in Corollary 4.4.1, when plotting dataset with higher dimensionality, the area of the SC AABB is significantly larger than that for lower dimensionality dataset. With unit axes vector $\overrightarrow{v}$, for the 4D case, the AABB is $[1, -1, 1, -1]$ while it is $[2, -2, \sqrt{3}, -\sqrt{3}]$ for the 6D case based on Eq. 4.20.

4.4.2.2 Bounding Polygon

The AABB of standard SC is helpful to determine the size of the display window. It encloses the exact plotting region of SC. The more complex bounding polygon

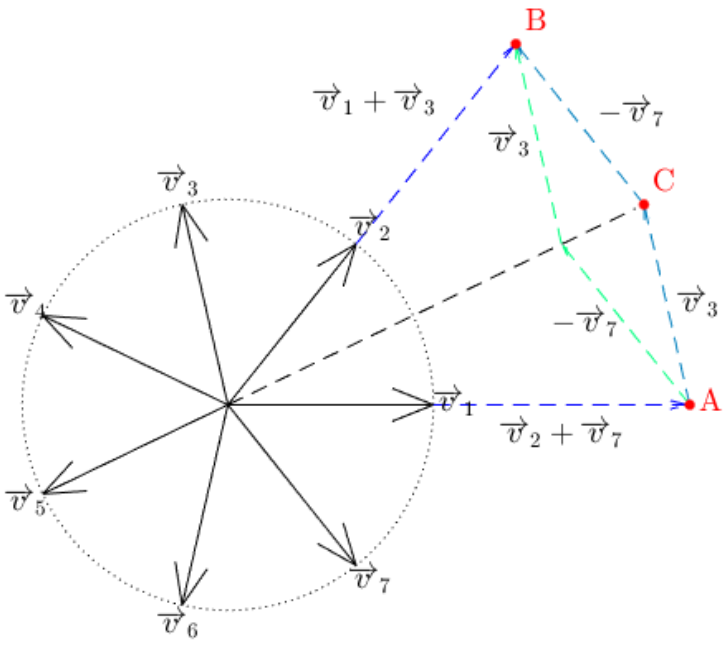


Figure 4.14. For a 7-dimensional standard SC, the two possible vector orders from point A to point B are illustrated.

that covers all projected data points in standard SC is studied. It is well known that in RadViz all the data points are located in a convex hull formed by all the dimensionality anchors [2]. However, in standard SC, a polygon region that encloses all the possible projected data points has not been studied before. The determination of this bounding polygon region will provide support to the property analysis of SC.

7D Standard SC Here, our analysis is based on standard SC and the case to plot a 7-dimensional (7D) dataset is taken as an example to illustrate the procedure.

As standard SC is symmetric, instead of finding the whole bounding polygon directly, our analysis starts from the boundary between two successive axes vectors. In Fig. 4.14, $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ are chosen to be studied first. In Fig. 4.14, the farthest point along $\overrightarrow{v_1}$, A, equals to $\overrightarrow{v_1} + (\overrightarrow{v_2} + \overrightarrow{v_7})$. Meanwhile, the farthest point along $\overrightarrow{v_2}$, B, equals to $\overrightarrow{v_2} + (\overrightarrow{v_1} + \overrightarrow{v_3})$. For points A and B, according to Eq. 4.3 the corresponding high-dimensional data points can be $d'_A = (1, 1, 0, 0, 0, 0, 1)$ and $d'_B = (1, 1, 1, 0, 0, 0, 0)$. Then the polygon region of star coordinate between the direction of $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ can be considered as the movement from point A to point B. There are two possible solutions for this movement which may lead to a different track. The first solution

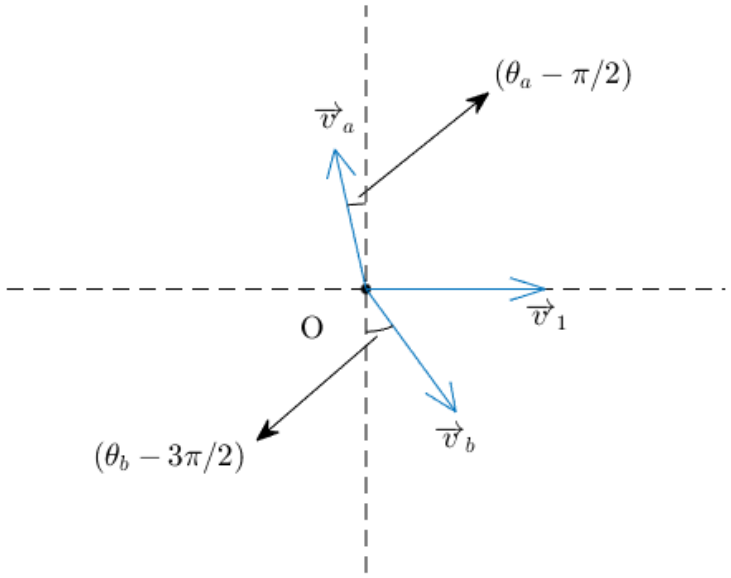


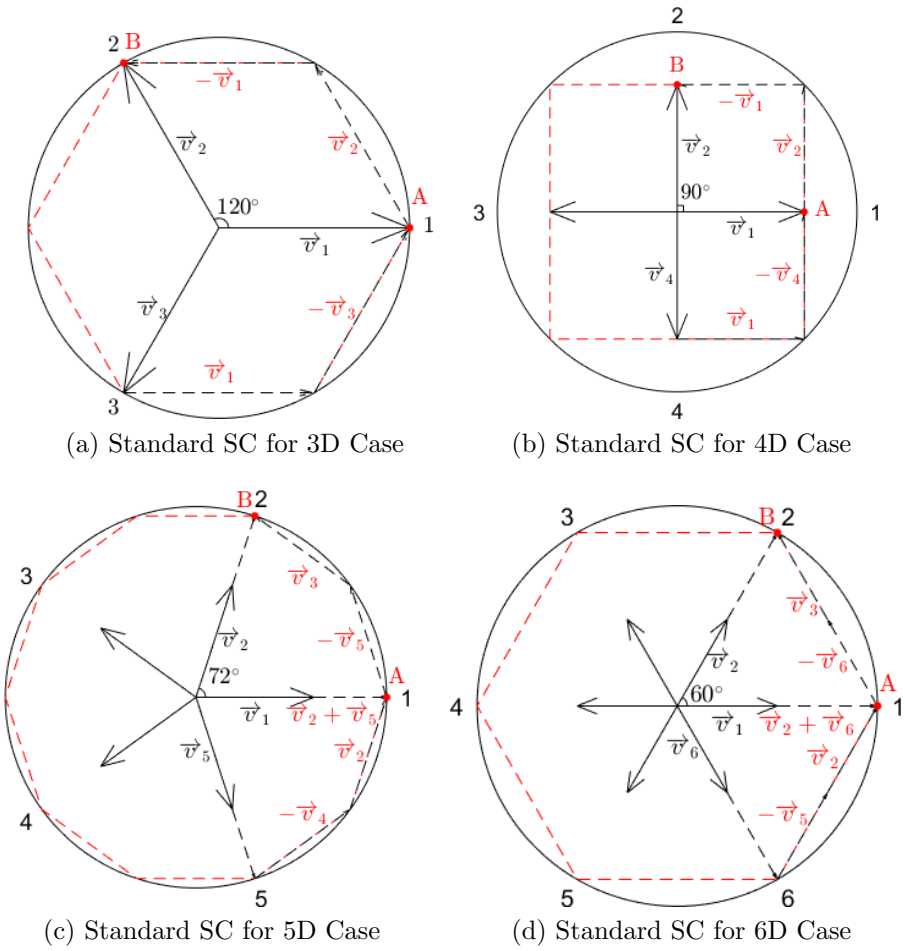
Figure 4.15. Determine the order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ for $\overrightarrow{v_1}$.

is $A + \overrightarrow{v_3} - \overrightarrow{v_7} = B$ while the second one is $A - \overrightarrow{v_7} + \overrightarrow{v_3} = B$. These two possible solutions are plotted in Fig. 4.14 with different colors.

Here, in this case, the first solution is the correct result as $\overrightarrow{v_3}$ plays a more important role at the beginning. This result can be explained more clearly by using the point C in Fig. 4.14. Point C can be obtained by $\overrightarrow{v_1} + \overrightarrow{v_2} + \overrightarrow{v_3} + \overrightarrow{v_7}$. Hence, $\overrightarrow{v_3}$ connects point A with point C. Similarly, $-\overrightarrow{v_7}$ connects point C with point B. Thus, the bounding polygon between the direction of $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ is $(\overrightarrow{v_3}, -\overrightarrow{v_7})$.

Using the rotational symmetry property of standard SC, the rest part of the polygon region can be generated easily. In all, the polygon region is formed by vectors $(\overrightarrow{v_3}, -\overrightarrow{v_7}, \overrightarrow{v_4}, -\overrightarrow{v_1}, \overrightarrow{v_5}, -\overrightarrow{v_2}, \overrightarrow{v_6}, -\overrightarrow{v_3}, \overrightarrow{v_7}, -\overrightarrow{v_4}, \overrightarrow{v_1}, -\overrightarrow{v_5}, \overrightarrow{v_2}, -\overrightarrow{v_6})$ which are successively connected.

n-D Standard SC For the case with an n-dimensional dataset, the bounding polygon can be determined similarly as shown in Fig. 4.14. In the n-dimensional case, we start from $\overrightarrow{v_1}$ to find the bounding polygon. Let V_i be axes vectors that have positive dot product with $\overrightarrow{v_i}$ and $\overline{V_i}$ is complement of V_i. The $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ for $\overrightarrow{v_i}$ is defined as $\overrightarrow{v_a} \in (\overline{V_i} \cap V_{i+1})$ and $\overrightarrow{v_b} \in (V_i \cap \overline{V_{i+1}})$. Then, the problem to find the bounding polygon can be divided into three steps: determine the position of the starting point, find the



(a) Standard SC for 3D Case (b) Standard SC for 4D Case

(c) Standard SC for 5D Case (d) Standard SC for 6D Case

Figure 4.16. The circumscribed circle and bounding polygon of standard SC when plotting 3D, 4D, 5D, and 6D datasets.

$\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ for each $\overrightarrow{v_i}$, and calculate the order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$.

- Determine the position of the starting point. The farthest point along $\overrightarrow{v_1}$ can be calculated as the sum of axes vectors in V_1. In the 7D case in Sect. 4.4.2.2, $V_1 = \{\overrightarrow{v_1}, \overrightarrow{v_2}, \overrightarrow{v_7}\}$.

- Find the $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ for each $\overrightarrow{v_i}$. For $i = 1$, according to the definition of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$, they can be expressed as $\overrightarrow{v_a} \in (\overline{V_1} \cap V_2)$ and $\overrightarrow{v_b} \in (V_1 \cap \overline{V_2})$. In the 7D case in Sect. 4.4.2.2, $V_2 = \{\overrightarrow{v_1}, \overrightarrow{v_2}, \overrightarrow{v_3}\}$. Then $\overrightarrow{v_a} = \overrightarrow{v_3}$ and $\overrightarrow{v_b} = \overrightarrow{v_7}$. Proposition 4.4.2 proves that for standard SC $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ are unique. The determination of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ in n-D standard SC is described in Proposition 4.4.3.

- Calculate the order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$. The influence on $\overrightarrow{v_1}$ will be analyzed to determine the order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$. Proposition 4.4.4 gives the criteria used for this determination.

Proposition 4.4.2. *There is one and only one vector $\overrightarrow{v_i}$ that satisfies the requirement $\overline{V_1} \cap V_2$. Similarly, there is one and only one vector $\overrightarrow{v_i}$ that belongs to the set of $V_1 \cap \overline{V_2}$.*

Proof. This is due to the symmetric configuration of standard SC. □

Proposition 4.4.3. *Let θ_1 be the angle of axes vector v_1. Then based on θ_1, the $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ in standard SC can be expressed as:*

$$\theta_a = \begin{cases} \frac{\pi}{2} + \theta_1, & \text{if } n = 4\mathbb{Z}^+, \\ (int\left[\frac{n}{4}\right] + 1) \cdot \frac{2\pi}{n} + \theta_1, & \text{otherwise}, \end{cases}$$

$$\theta_b = \begin{cases} 2\pi - (\frac{n}{4} - 1) \cdot \frac{2\pi}{n} + \theta_1, & \text{if } n = 4\mathbb{Z}^+, \\ 2\pi - (int\left[\frac{n}{4}\right]) \cdot \frac{2\pi}{n} + \theta_1, & \text{otherwise}, \end{cases}$$

where $\mathbb{Z}^+$ refers to a positive integer, $int[x]$ gives the largest integer less than or equal to x.

Proof. Let $\theta_\Delta = \theta_i - \theta_1$, $i = 1, \ldots, n$ and θ_i is defined as in Eq. 4.18. As i increases, θ_a is the first one that satisfies the condition: $\theta_\Delta = \theta_a - \theta_1 \geq \pi/2$. While for θ_b, it is the first one that satisfies the condition: $\theta_\Delta = \theta_a - \theta_1 > 3\pi/2$. In low-dimensional cases, it is possible that $\theta_b = \theta_1$. □

Proposition 4.4.4. *The influences of θ_a and θ_b on θ_1 determine the vector order. If*

$$\theta_b - 3\pi/2 > \theta_a - \pi/2,$$

then the vector order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ is ($\overrightarrow{v_a}$,$-\overrightarrow{v_b}$, $\cdots$). Otherwise, the order should be ($-\overrightarrow{v_b}$,$\overrightarrow{v_a}$, $\cdots$).

Proof. The included angle between θ_a and $\pi/2$ and the included angle between θ_b and $3\pi/2$ are compared (as shown in Fig. 4.15). □

Once these three issues are solved, the remaining vectors can be arranged successively, and the bounding polygon obtained. A pseudo-code is given in Algorithm 4.3.

Algorithm 4.3 Bounding polygon of standard SC

1: Rotate all the axes vectors so that the angle θ_1 of $\overrightarrow{v_1}$ is equal to 0;
2: Build V_1;
3: Calculate the starting point A;
4: Initialize the bounding polygon $BP = []$;
5: For $i = 1 : n$
6: Build V_i for $\overrightarrow{v_i}$;
7: If $i == n$
8: $V_{i+1} = V_1$;
9: Else
10: Build V_{i+1} for $\overrightarrow{v_i}$;
11: Find the corresponding $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ for $\overrightarrow{v_i}$ as $\overrightarrow{v_a} \in (\overline{V_i} \cap V_{i+1})$ and $\overrightarrow{v_b} \in (V_i \cap \overline{V_{i+1}})$;
12: Calculate the order of $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ according to Proposition 4.4.4;
13: Adding $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ into BP in the calculated order;
14: End;

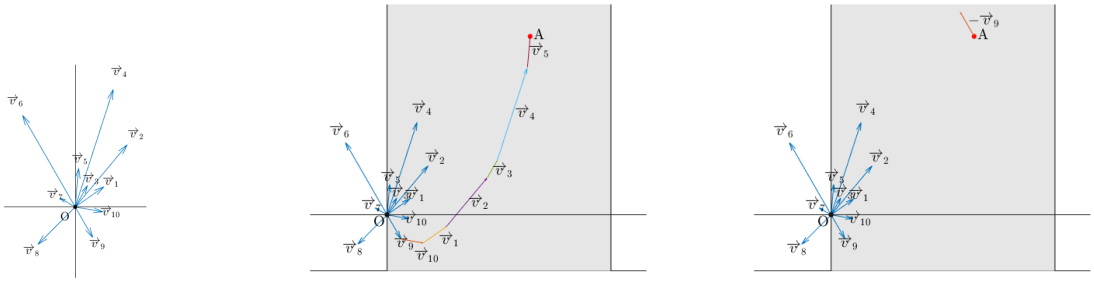
Geometrical Property Here, we show that the projected data points in standard SC lie within a convex hull.

Proposition 4.4.5. *When transforming high-dimensional data point $d_j^{'}$ to $p^{SC}(x_j, y_j)$, the bounding polygon of standard SC is a convex polygon.*

Proof. The convexity of a bounding polygon of standard SC can be proved by considering the definition of convexity. In the set of vector V, if for any two vectors $\overrightarrow{v_1}, \overrightarrow{v_2} \in V$ and any $t \in [0, 1]$, the vector $t\overrightarrow{v_1} + (1-t)\overrightarrow{v_2}$ is also in V, then it is said to be convex [124]. According to Eq. 4.3, all the projected data points in standard SC are the linear combination of unit vectors and the linear coefficients are in the interval of $[0, 1]$. Hence, the bounding polygon is a convex polygon. □

The convex polygon in RadViz is a regular polygon and the number of sides is equal to the number of dimensionality anchors [2]. However, in standard SC, though the convex polygon is also a regular polygon, the number of sides in this convex hull is not always equal to the number of dimensionality anchors. It depends on the dimensionality of the dataset.

Proposition 4.4.6. *Let R be the circumradius of the regular polygon, s be the side length, and N be the number of sides. We set the magnitude of axes vector to 1. If the dimensionality n is even, then $s = 2$ and $N = n$ and $R = 1/\sin(\pi/n)$. If n is odd, then $s = 1$ and $N = 2n$ and $R = 1/(2\sin(\pi/(2n)))$.*



(a) SC cases (b) Determine the starting point A (c) Determine the order of axes vectors

Figure 4.17. Determining the bounding polygon of SC. In (a), the configuration of SC is illustrated. Then the starting point A is confirmed as shown in (b). The third step is to confirm the order of axes vectors. In (c), from the starting point A, the first axes vector is confirmed and plotted.

Proof. By following the procedure described in Sect. 4.4.2.2, the obtained bounding polygon is formed by $2n$ vectors. However, in few cases, $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ may be parallel. In Fig. 4.16, we illustrate the cases when using standard SC to plot 3D, 4D, 5D, and 6D datasets. In the 3D (Fig. 4.16(a)) and 5D (Fig. 4.16(c)) cases, $\overrightarrow{v_b}$ is not parallel with $\overrightarrow{v_a}$ or $\overrightarrow{v_{a-1}}$. Hence, when $n = 4\mathbb{Z}^+ - 1$ and $n = 4\mathbb{Z}^+ + 1$, the convex hull is a regular polygon with $N = 2n$. In the 4D case (Fig. 4.16(b)), though the obtained $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ are not parallel, $\overrightarrow{v_b}$ is parallel with $\overrightarrow{v_{a-1}}$. While for the 6D case (Fig. 4.16(d)), the obtained $\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ are parallel. Hence, when $n = 4\mathbb{Z}^+$ and $n = 4\mathbb{Z}^+ + 2$, the convex hull is a regular polygon with $N = n$ and the side length $s = 2$. □

One issue to note is that the vertices of the bounding polygon obtained when $n = 4\mathbb{Z}^+$ and $n = 4\mathbb{Z}^+ + 2$ have different relative positions. In $n = 4\mathbb{Z}^+$ case, the vertices are not located along the axes vectors. While in $n = 4\mathbb{Z}^+ + 2$ case, vertices are located along the direction of the axes vectors.

4.4.3 Exploration for Boundary of Arbitrary SC

The boundaries of SC with arbitrary distribution configurations (AABB, bounding polygon, and minimum bounding box) are studied in this section. The magnitude L_i of vector $\overrightarrow{v_i}$ can be any non-negative value and $\theta_i \in [0, 2\pi)$. The study of the boundaries of arbitrary SC is more meaningful because finding the optimal configuration of axes vectors is the key point when using SC to perform data analysis.

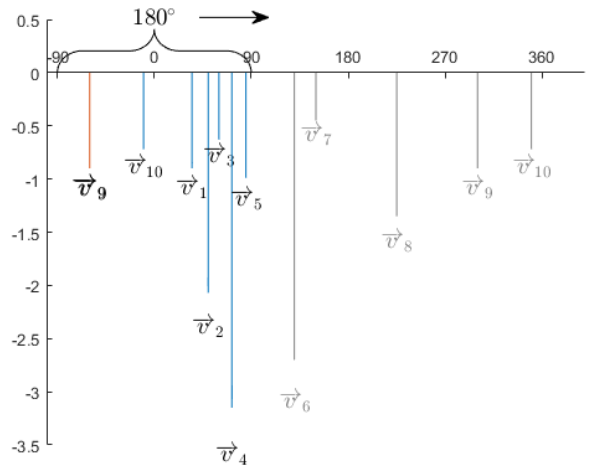
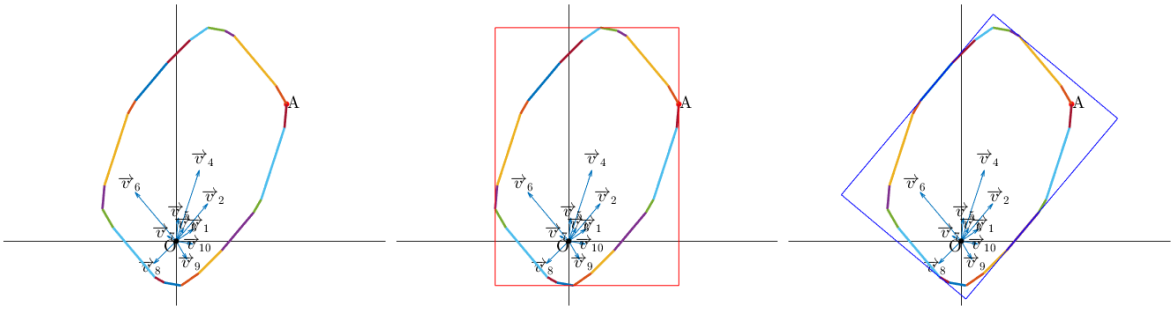


Figure 4.18. Determining the bounding polygon of SC by visualizing the magnitude and angle of each axes vector with a biplot map.



(a) With bounding polygon (b) With axis-aligned bounding box (c) With minimum bounding box

Figure 4.19. The bounding polygon, the axis-aligned bounding box, and the minimum bounding box of the 10D SC plots.

4.4.3.1 Axis-aligned Bounding Box

For arbitrary SC, the magnitude of axes vectors $\overrightarrow{v_i}$ may not be of unit length. Hence, L_i should be considered when calculating the AABB.

Proposition 4.4.7. *Let* $[X_{\max}, X_{\min}, Y_{\max}, Y_{\min}]$ *be the right, left, top, and bottom side of the AABB which contain the whole SC plotting. In arbitrary SC when plotting*

n-dimensional dataset, the AABB coordinates are

$$\begin{cases} X_{\max} = \sum_{i \in I} L_i \cdot \cos\theta_i & with \quad I = \{i \in \mathbb{N} | \cos\theta_i \geq 0\} \\ X_{\min} = \sum_{i \in I} L_i \cdot \cos\theta_i & with \quad I = \{i \in \mathbb{N} | \cos\theta_i \leq 0\} \\ Y_{\max} = \sum_{i \in I} L_i \cdot \sin\theta_i & with \quad I = \{i \in \mathbb{N} | \sin\theta_i \geq 0\} \\ Y_{\min} = \sum_{i \in I} L_i \cdot \sin\theta_i & with \quad I = \{i \in \mathbb{N} | \sin\theta_i \leq 0\} \end{cases}, \tag{4.21}$$

where $\mathbb{N} = \{1, \cdots, n\}$.

Proof. By following a similar strategy as in Proposition 4.4.1, all vectors $\overrightarrow{v_i}$ are projected onto the x-axis when calculating $X_{\max}$. When $\cos\theta_i > 0$, set the corresponding value $d'_{i,j}$ to one. Otherwise, set the $d'_{i,j}$ to zero. As a matter of fact, Eq. 4.21 is the generalized expression of Eq. 4.19. □

In arbitrary SC, the configuration may not be symmetric. Hence, the high-dimensional data point $d'_j = \{1, \ldots, 1\}$ may not be projected to the center of SC. In this case, Corollary 4.4.1 is invalid for arbitrary SC. Hence, the AABB of standard SC can be treated as a special case of the arbitrary SC's AABB.

4.4.3.2 Bounding Polygon

The algorithm to determine the bounding polygon of arbitrary SC is described here. Compared with standard SC cases, the bounding polygon determination of the arbitrary SC is much more complex. For arbitrary SC case, there are two main parts: determine the position of the starting point A and determine the order of axes vectors.

- Determine the position of the starting point A. The starting point is the place where we start to plot the bounding polygon. A configuration of an arbitrary SC plotting 10D dataset is shown in Fig. 4.17 (a) and the corresponding axes vectors are also plotted in a biplot as shown in Fig. 4.18. We use 0 *rad* as the initial orientation. The starting point A is the farthest data point along the direction of 0 *rad*. The position of starting point A is equal to the sum of all the axes vectors in the shadow area in Fig. 4.17 (b). These axes vectors $V_{initial}$ have

a positive value when projecting into the direction of 0 *rad*. This procedure can also be visualized in the biplot in Fig. 4.18. With the initial orientation 0 *rad*, axes vectors that are located in an interval of $[-\pi/2, \pi/2]$ are the related vectors. In this case, to build the starting point A, $V_{initial} = \{\overrightarrow{v_9}, \overrightarrow{v_{10}}, \overrightarrow{v_1}, \overrightarrow{v_2}, \overrightarrow{v_3}, \overrightarrow{v_4}, \overrightarrow{v_5}\}$ are used as shown in Fig. 4.17 (b).

- Determine the order of axes vectors one by one. As the bounding polygon is formed by axes vectors, the order is very important. Starting from point A, $\overrightarrow{v_6}$ and $\overrightarrow{v_9}$ are selected to compare their influence on the initial orientation $0rad$. Similar to Proposition 4.4.4, we compare the value of $(\theta_6 - \pi/2)$ and $(\theta_9 - 3\pi/2)$. In this case, $\theta_9 - 3\pi/2 < \theta_6 - \pi/2$, then the bounding polygon starts from $(-\overrightarrow{v_9}, \cdots)$ as shown in Fig. 4.17 (c). Another explanation is based on the biplot in Fig. 4.18. At the very beginning, the interval of $[-\pi/2, \pi/2]$ is centered at the initial orientation 0 *rad* to get the starting point A. Then to determine the order of the axes vectors, we can imagine the movement of the interval along the positive direction of the x-axis. In the initial moment, only $\overrightarrow{v_9}$, $\overrightarrow{v_{10}}$, $\overrightarrow{v_1}$, $\overrightarrow{v_2}$, $\overrightarrow{v_3}$, $\overrightarrow{v_4}$, and $\overrightarrow{v_5}$ are enclosed in the interval. During the movement,

  - if an axes vector $\overrightarrow{v_a}$ breaks from the interval, then the bounding polygon will have this axes vector but with the opposite direction $-\overrightarrow{v_a}$. In this case, the bounding polygon will have $(-\overrightarrow{v_9}, \cdots)$.
  - if an axes vector $\overrightarrow{v_a}$ falls into the interval, then the bounding polygon will have this axes vector with the same direction $\overrightarrow{v_a}$. In this case, the bounding polygon will have $(-\overrightarrow{v_9}, \overrightarrow{v_6}, \cdots)$.
  - if an axes vector $\overrightarrow{v_a}$ breaks from the interval and another axes vector $\overrightarrow{v_b}$ falls into the interval at the same time, then the bounding polygon will have $-\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ together. In this case, the order of $-\overrightarrow{v_a}$ and $\overrightarrow{v_b}$ has no influence on the bounding polygon as they are collinear.

  The interval keeps moving until the bounding polygon has formed a closed path and the ending point has the same position with the starting point A.

Proposition 4.4.8. *When transforming high-dimensional data point d_j' to $p^{SC}(x_j, y_j)$, the bounding polygon of arbitrary SC is a convex polygon.*

Proof. The proof of this proposition is the same as that in Proposition 4.4.8. Though the configuration of the SC has changed, a projected data point is still the linear combination of these axes vectors. Hence, the bounding polygon for any SC configuration is still convex. □

4.4.3.3 Minimum Bounding Box

The straight forward application of the bounding polygon is to find the minimum bounding box [125] enclosing the whole range of SC and then to rotate the whole view so as to maximize the display view of an arbitrary SC plotting. We follow the algorithm proposed in Sect. 4.4.3.1 and Sect. 4.4.3.2 to calculate the boundary of the 10D case as illustrated in Fig. 4.17 (a). The obtained bounding polygon and the axis-aligned bounding box are displayed in Fig. 4.19 (a) and (b). With the axis-aligned bounding box, a user can have the global view of the whole plotting. However, in some cases, the configuration of SC may lead to an odd bounding polygon. In such cases, the ratio of the bounding polygon area and the AABB area is very low. Hence, based on the obtained bounding polygon, we can calculate the relevant minimum bounding box to maximize the space utilization.

4.4.4 Boundary-Driven Application

4.4.4.1 SC with Boundaries

For standard SC, the relationship between the magnitude and circumradius is described in Proposition 4.4.6. Based on this relationship, we fix the circumscribed circle radius of standard SC as a unit, then the magnitude of each axes vector can be obtained.

Let R be the circumradius of the bounding polygon of standard SC and we set $R = 1$. If the dimensionality n is even, then $L_i = \sin(\pi/n)$. If n is odd, then $L_i = 2\sin(\pi/(2n))$.

Hence, the expression of the standard SC in Eq. 4.3 can be modified as:

$$p^{SC-Even}(x_j, y_j) = (\sin(\pi/n) \cdot \sum_{i=1}^{n}(d'_{i,j} \cdot \cos\theta_i), \sin(\pi/n) \cdot \sum_{i=1}^{n}(d'_{i,j} \cdot \sin\theta_i)),$$

$$p^{SC-Odd}(x_j, y_j) = (2\sin(\pi/2n) \cdot \sum_{i=1}^{n}(d'_{i,j} \cdot \cos\theta_i), 2\sin(\pi/2n) \cdot \sum_{i=1}^{n}(d'_{i,j} \cdot \sin\theta_i)).$$

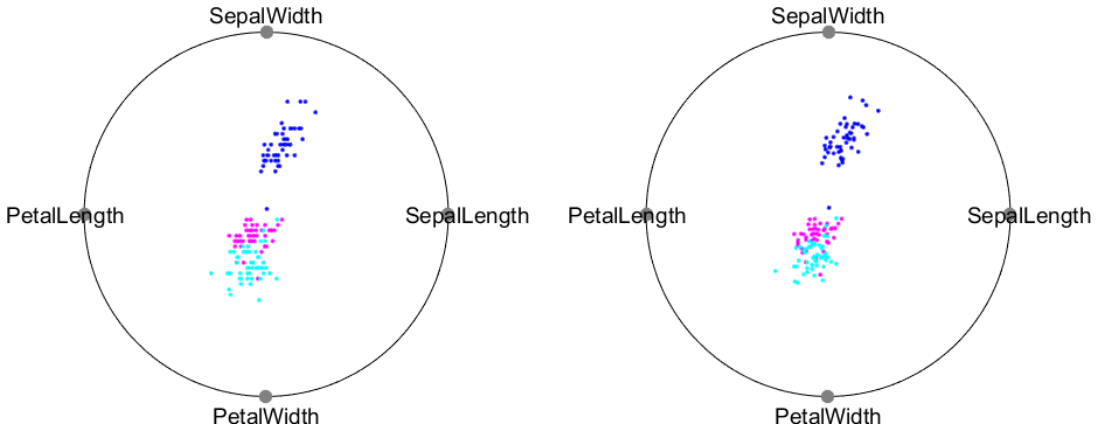
For arbitrary SC, the width and the height of the bounding box may not be the same. Hence, we set the larger one to one. In this case, the expression of the arbitrary SC in Eq. 4.3 can be modified as:

$$p^{SC}(x_j, y_j) = (\frac{\sum_{i=1}^{n}(d'_{i,j} \cdot L_i \cdot \cos\theta_i)}{max(X_{max} - X_{min}, Y_{max} - Y_{min})}, \frac{\sum_{i=1}^{n}(d'_{i,j} \cdot L_i \cdot \sin\theta_i)}{max(X_{max} - X_{min}, Y_{max} - Y_{min})}).$$

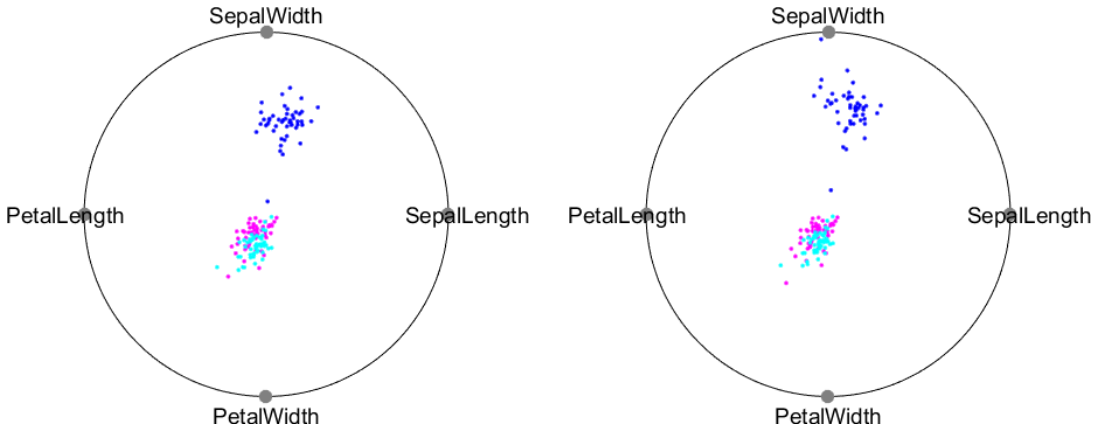
4.4.4.2 Association with RadViz

RadViz is often similarly compared with SC in recent works [1,61]. The key difference between them is the non-linear factor as shown in Eq. 4.4. However, the sizes of these two kinds of plotting are not considered. In their comparison, the magnitude of axes vectors in both RadViz and SC are fixed to be one, i.e. $L_i = 1$. In standard SC, as shown in Sect. 4.4.2.1 and Sect. 4.4.2.2, if the magnitude of axes vectors is fixed, then the range of standard SC has a positive correlation with the dataset dimensionality n. Meanwhile, in RadViz, all the projected data points are located within the convex hull formed by the dimensionality anchors while all dimensionality anchors are located on a circle with a unit radius. As they did not consider the range of SC, they just use a view that can cover all the projected data points. Hence, it is not fair to compare a plotting which has a whole range with another plotting which only has partial range.

We fix the circumscribed circle radius of both RadViz and standard SC in order to compare these two plottings fairly. For RadViz, the radius of the circumcircle for all the dimensionality is the same. So no change will be made in the expression of RadViz in Eq. 4.2. Based on the same circumradius, the configurations of RadViz and standard SC when plotting 3D, 4D, 5D, and 6D datasets are compared in Fig. 4.16. The bounding polygons of RadViz and standard SC have a larger difference in lower



(a) $c = 0$ (b) $c = 0.25$



(c) $c = 0.75$ (d) $c = 1$

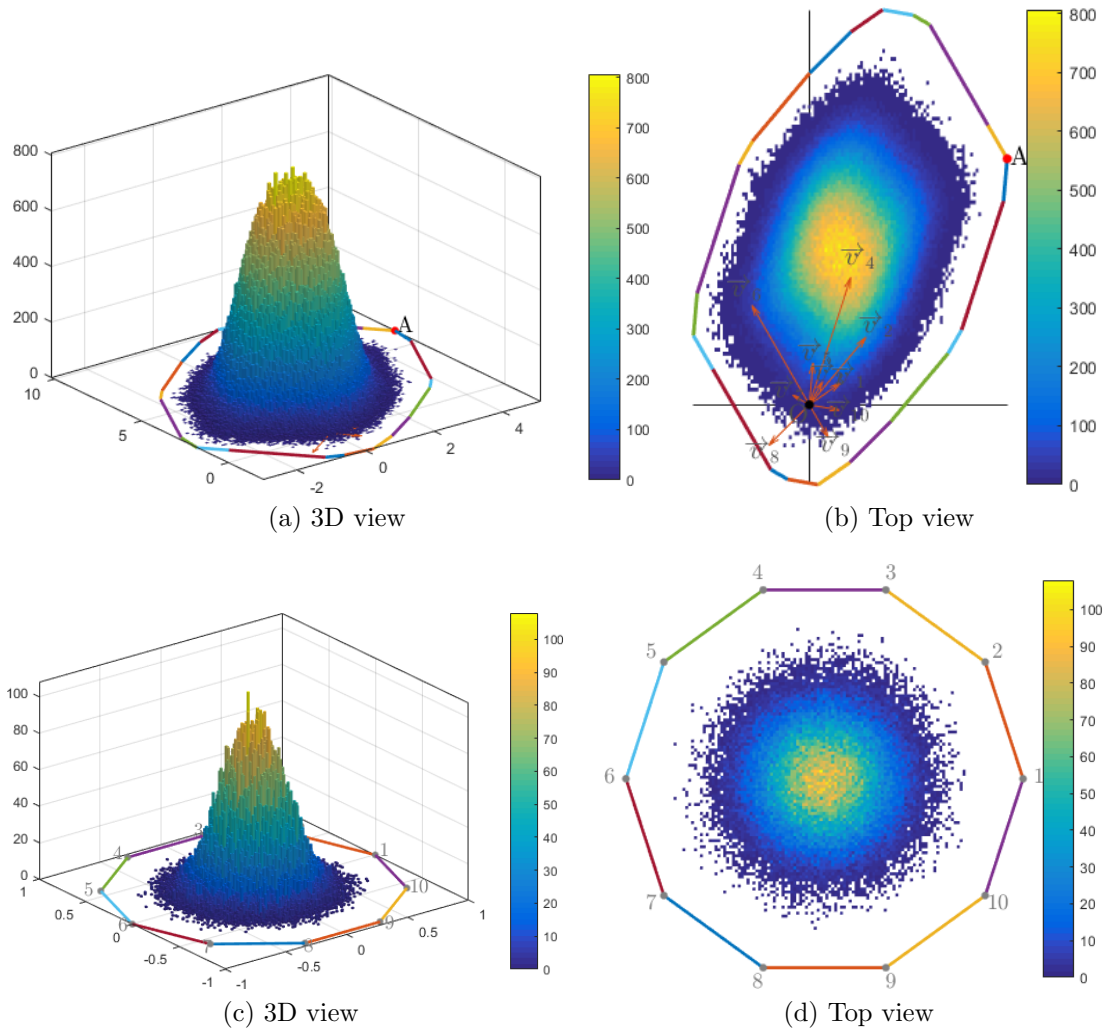
Figure 4.20. The smooth transition between standard SC and RadViz with fixed circumradius on the 'IRIS' dataset.

dimensionality cases. With increasing dimensionality, the bounding polygons of both RadViz and standard SC approach towards the circumscribed circle.

The general projective maps (GPM) designed by Lehmann and Theisel is a linear interpolation between RadViz and SC [61]. However, the expression in [61] does not consider the range difference between RadViz and SC. Here, we reorganize the definition of GPM by combining Eq. 4.3 and Eq. 4.2 yielding the expression is

$$p^{GPM}(x_j, y_j) = \left(\frac{\sum_{i=1}^{n}(d'_{i,j} \cdot L_i \cdot \cos\theta_i)}{(1-c)\cdot 1 + c \cdot \sum_{i=1}^{n} d'_{i,j}}, \frac{\sum_{i=1}^{n}(d'_{i,j} \cdot L_i \cdot \sin\theta_i)}{(1-c)\cdot 1 + c \cdot \sum_{i=1}^{n} d'_{i,j}}\right), \tag{4.22}$$

where $c \in [0, 1]$. If $c = 1$, then GPM equals RadViz. Otherwise if $c = 0$, then it gives the SC. According to Eq. 4.22, the circumradius of GPM is set to be L_i and fixed along with different c. In this case, the smooth transition between RadViz and standard SC can be correctly expressed. As an example, Fig. 4.20 illustrates the



(a) 3D view (b) Top view

(c) 3D view (d) Top view

Figure 4.21. We plot 100,000 uniformly sampled 10-dimensional data points in SC with general configuration (a, b) and standard configuration (c, d) respectively to demonstrate the clumping effect.

smooth transition on the 'IRIS' dataset.

4.4.4.3 Clumping Effect in SC

The determination of SC boundaries contributes to the feature analysis in which a global view is provided. One case is the clumping effect of SC plotting. The clumping effect is known as the effect that forces data points towards the center of the plotting. The clumping problem of radial visualization method was first observed and analyzed and the effect of diametrically opposed dimensional anchors in RadViz is believed to be the main reason [2]. However, they did not analyze the clumping effect under different dimensionality and did not compare the clumping effect between RadViz and standard SC plots. The comparative study conducted by Rubio-Sánchez et al. [1]

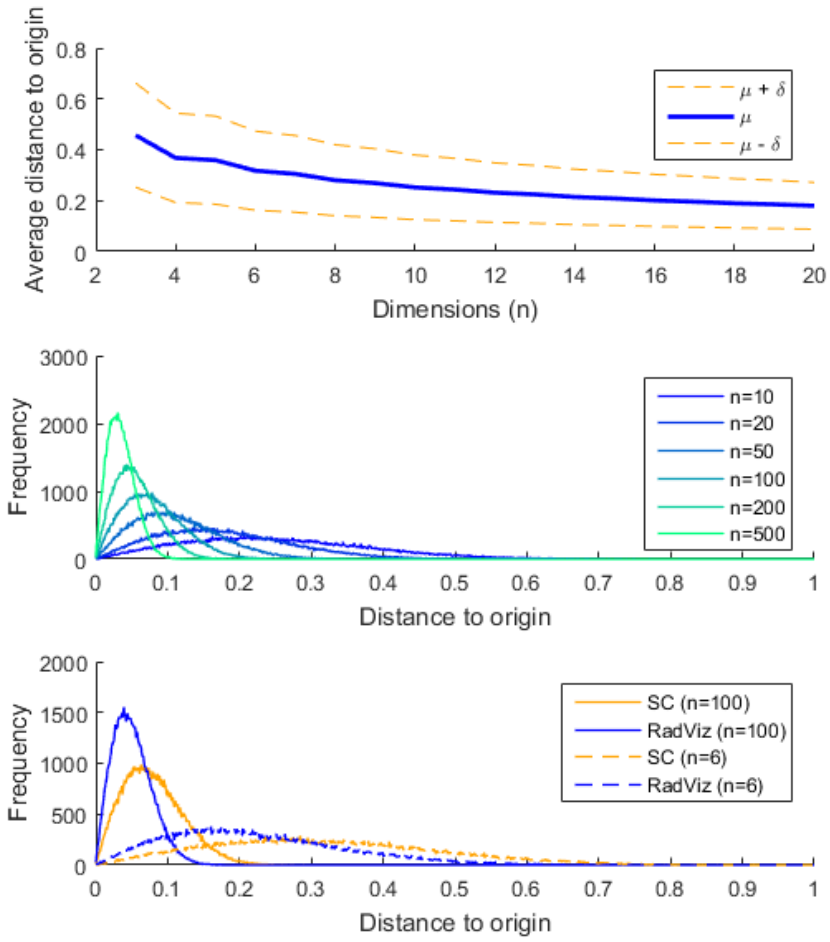


Figure 4.22. (Top) The average distance of all the projected data points to the origin decreases as the dimensionality increases, which means that the clumping effect becomes more obvious with high dimensional data. (Middle) The distribution of the distance using 10D, 20D, 50D, 100D, 200D, and 500D datasets [2]. (Bottom) Compared with the distribution of data point distance of RadViz, SC's clumping effect is less obvious, but both of them are increasing in dimensionality [1].

illustrated the comparison when plotting 6D and 100D datasets, but it did not consider the range change of SC along with the dimensionality changes. Here, we analyze the clumping effect of the standard SC under different dimensionality and fairly compare the clumping effect on RadViz and standard SC. Fig. 4.21 illustrates the clumping effect of SC with an arbitrary configuration and a standard configuration, respectively. The clumping effect can be clearly observed in these two kinds of SC configuration and the configuration difference has little effect on the clumping effect.

A quantitative analysis on the clumping effect on standard SC is conducted. The average distances of projected data points to the origin and the distributions of distances to the origin under different dimensions are calculated in order to provide

different views on the clumping effect of standard SC as shown in Fig. 4.22. The circumscribed circle radius is set to be one for different dimensions. The observation from Fig. 4.22 (Middle) is that with increasing dimensionality, the peak of the plotting is skewing to the left which indicates that an increasing number of data points are clumping towards the origin in standard SC. The residual space is sparsely populated, but it does not mean that it is empty.

The difference between our analysis and the previous result on the clumping effect of SC [1] is twofold. Firstly, the average distance decreases as the number of dimensionality increases in SC in our analysis while an inverse result was obtained in [1]. The previous analysis did not consider the range increase of SC as the dimensionality increases. Hence, the incorrect conclusion was obtained. Secondly, the comparison between the RadViz's clumping effect and the SC's clumping effect is different. In our analysis, the clumping effect of both RadViz and SC are more comprehensive for high-dimensional dataset as shown in Fig. 4.22 (Bottom). However, without the global view, the previous analysis failed to fairly compare the performance of RadViz and SC.

4.4.5 Discussion

Verification of the SC Boundary The verification of the SC plots boundaries, especially bounding polygon, can be conducted by many methods. One is to find the object-oriented minimum bounding boxes for each orientation and then the union of all these object-oriented minimum bounding box is the bounding polygon. This is due to the convexity of the boundary polygon as proved previously.

Computational Complexity The computational complexity of our proposed algorithms is low. For the bounding polygon determination of arbitrary SC configuration, the result can be obtained in $O(n)$ time. Meanwhile, the minimum bounding box can be calculated in linear time. Hence, it is possible for the SC plots to dynamically adjust the view orientation to maximize space utilization.

4.5 Summary

In this chapter, we first focus on the clumping problem of RadViz by providing a series of radial-related operations to uncover hidden patterns and increases the space utilization. We can manipulate the distribution of data points with histogram by radial equalization and other radial specified operations. The latter shows the flexibility of our technique for different user requirements. Experimental results indicate the advantage of the PolarViz. Compared with other methods, the PolarViz not only preserves the pros of RadViz but also provides a flexible radial modification on the views.

On the other hand, we proposed boundary determination algorithms for both standard and arbitrary SC configurations. These boundaries provide users and researchers a global view on the SC plots that has never been done before. Now, it is possible for SC to plot subsequent data points without an additional operation to adjust the view and features such as clumping effect can also be analyzed. Moreover, equipped with the boundaries, the clumping effect of standard SC and arbitrary SC can be clearly visualized which can provide a clearer understanding of the plot. It is believed that better understanding of the plots and the features will be helpful for further concrete numerical analysis.

Chapter 5

Orthogonal Voronoi Diagram and Treemap for Implicit Hierarchical Visualization

5.1 Introduction

When visualizing engine-related hierarchical datasets, domain experts pay a lot attention to the value associated with each node in the hierarchy. The implicit hierarchical data visualization methods, which focus on the value within each node and positionally encode the hierarhcy by node overlap or inclusion are suitable for such purpose. Moreover, implicit hierarhcial data visualization methods utilize space more efficiently. Treemap [80] and Voronoi treemap [94] which are formed by nested rectangles and polygons respectively are the most two popular implicit hierarchy visualization methods. However, they adopt distinct strategies to generate the plots. Treemap method divides the empty canvas into rectangular sub-regions so that the area is associated with the relative sizes of the respective sub-hierarchies. In contrast, Voronoi treemap partitions the canvas into polygon shapes based on distance to specified sites. After that, the position and weight of the sites may be iteratively adapted in order to adjust the area of the sub-regions.

In our opinion, the treemap and the Voronoi treemap make different choices in terms of shape simplicity and adjustment ability. To be specific, by using rectangular

shapes, the treemap looks more comfortable for the viewers and is much tidier than Voronoi treemap with nested polygons. Moreover, rectangle shapes make it easier to visually compare the areas of different regions than polygon shapes [99]. However, when the hierarchical data changes even slightly, the partitioning of empty canvas in treemap needs to be regenerated and the new layout may be largely different from the previous one. It usually leads to bad layout stability. In contrast, Voronoi treemap can adjust its layout via slight modification on the status of its sites.

We aim to find a good balance between shape simplicity and adjustment ability with the proposed novel hierarchy visualization method: orthogonal Voronoi diagram and treemap. The orthogonal layout has been proved to be suitable for human beings in explicit hierarchy visualization (node-link diagram) and network visualization [126, 127]. It would be an interesting topic to evaluate different treemap layouts in implicit hierarchy visualization. A treemap with orthoconvex and L-shape was proposed by dividing existing treemap element to preserve an aspect ratio constraint [128]. However, the orthoconvex treemap cannot be adjusted flexibly.

Our orthogonal Voronoi diagram partitions the empty canvas into nested orthogonal rectangles. Hence, the generated layout is not only flexible to a diversified data value, but also much tidier than the Voronoi treemap with nested polygons. To achieve this, we first define a new distance calculation strategy in order to generate axis-aligned segmentation among the sites, which refer to a set of pre-defined points. In the strategy, a new distance function is defined by considering the relative positions of two sites rather than a single site. Then a horizontal or vertical segmentation line is generated between these two sites. After that, we design a sweepline + skyline heuristic algorithm to partition the canvas based on the new distance calculation strategy to generate an orthogonal Voronoi diagram. The proposed algorithm is motivated by the sweep line algorithm for Voronoi diagram generation [129] and the skyline strategy in handling cutting and packing problems [86]. By iteratively alerting the status of the original sites and calling the sweepline + skyline algorithm, the area of the orthogonal rectangular sub-regions will match their corresponding values. An orthogonal Voronoi treemap will be obtained if this process is recursively continued layer by layer until the

whole hierarchical structure is traversed. Moreover, we also design an initialization strategy to improve algorithm performance. The proposed algorithm is fast, simple, and resolution-independent. The content of this work has been published. [1]

The main contribution of this work is threefold:

- We define a novel distance function based on two sites so that the segmentation is axis-aligned.
- We design a sweepline + skyline algorithm for space partitioning with $O(n \cdot \log(n))$ complexity. To the best of our knowledge, this is the first time the sweep line strategy is used for the Voronoi treemap.
- We design a novel strategy to initialize the diagram status and modify the status update procedure to increase the efficiency and effectiveness of our algorithm.

The rest of this chapter is organized as follows. The background knowledge is introduced in Sect. 5.2 before the description of our methodology. In Sect. 5.3, we describe the proposed orthogonal space partitioning algorithm to generate the orthogonal Voronoi treemap. The performance of our algorithm is discussed in Sect. 5.4. Finally, discussion and conclusion are made in Sect. 5.5 and Sect. 5.6.

5.2 Background

In this section, we review the background on the Voronoi treemap, including Voronoi diagram, weighted Voronoi diagram, centroidal Voronoi diagram, and Voronoi treemap since we adopt similar layout generation strategy with the Voronoi plots. In the description, we follow the notation used in Nocaj and Brandes's work [97].

5.2.1 Voronoi Diagram

A Voronoi diagram (also called a Voronoi tessellation, or a Voronoi partition) is a partitioning of a plane into sub-regions based on distances to a set of points within the plane. These sub-regions are often called *Voronoi cell* (or *cell*) and these points

[1] Yan-Chao Wang, Feng Lin, and Hock-Soon Seah. Orthogonal Voronoi Diagram and Treemap, *arXiv preprint*, arXiv:1904.02348, 2019. [130]

in the plane are called *sites*. In this chapter, we only consider the partitioning in a bounded region (e.g. a rectangle) rather than the whole 2D plane.

Formally, given a bounded region $\Omega \subset R^2$ and a set of n sites $S = \{s_1, s_2, ..., s_n\}$, the Voronoi diagram divides Ω into a set of Voronoi cells $\upsilon(s_i)$, one for each site s_i. Then the cell $\upsilon(s_i)$ can be expressed as

$$\upsilon(s_i) = \{p \in \Omega \mid dist(p, s_i) < dist(p, s).\ \forall s \in S, s \neq s_i\}, \tag{5.1}$$

where $dist(p, s_i)$ is the distance between point $p = (x_p, y_p)$ and site $s_i = (x_{s_i}, y_{s_i})$. The distance can be the Euclidean distance or other distance functions. The two most used distance functions are the Euclidean distance $dist_e(p, s_i)$ and the power Euclidean distance $dist_{pe}(p, s_i)$ which are shown as below:

$$dist_e(p, s_i) = \|p - s_i\| = \sqrt{(x_p - x_{s_i})^2 + (y_p - y_{s_i})^2}, \tag{5.2}$$

$$dist_{pe}(p, s_i) = \|p - s_i\|^2 = (x_p - x_{s_i})^2 + (y_p - y_{s_i})^2. \tag{5.3}$$

The Voronoi diagram is defined as the collection of Voronoi cells, $\upsilon(S) = \{\upsilon(s_1), ..., \upsilon(s_n)\}$.

5.2.2 Weighted Voronoi Diagram

In the Voronoi diagram, the area of a cell is fixed and only depends on the positions of its associated and neighboring sites. Hence, in order to use the cells to depict additional information (e.g. data value), a mechanism to control the areas of cells is required. To achieve this, a positive real weight is associated with each site. When calculating the distance, the associated weights should also be taken into account.

The power Euclidean distance function produces a straight line between two sites (the solid black lines in Fig. 5.2), thus in this chapter the Voronoi plots are all based on the power Euclidean distance with the exception of our proposed orthogonal Voronoi plots that is based on a new distance function.

Formally, let $W = \{w_1, w_2, ..., w_n\}$ be a set of positive weights associated with the

set of sites S correspondingly. The weighted power Euclidean distance based on the original power Euclidean distance (Eq. 5.3) can be written as follows:

$$dist_{awpe}(p, s_i) = \|p - s_i\|^2 - w_i. \tag{5.4}$$

Increasing the weight value will increase the area of the cell. However, it is non-linear in general. Besides that, a too large weight value may lead to empty cells. We will discuss this overweight issue in the description of our algorithms.

5.2.3 Centroidal Voronoi Diagram

Centroidal Voronoi diagram is a special type of Voronoi diagram such that the site s_i is located at the center of each cell $v(s_i)$ [131]. This kind of diagram usually generates cells with good aspect ratio (i.e. the ratio of the sides of the oriented minimum bounding rectangle is close to one). This is desirable as a good aspect ratio ensures good readability in visualization. Let $\overline{v(s_i)}$ and $A(v(s_i))$ be the polygonal boundary and area of cell $v(s_i)$ respectively, the centroid $c_i = centroid(\overline{v(s_i)})$ can be calculated in linear time. For all the n sites, the computation of the centroid needs $O(n)$ time.

5.2.4 Voronoi Treemap

The Voronoi treemap is a recursive partitioning of a plane, same as the treemap. Starting from the root of a hierarchy, a weighted Voronoi diagram is generated in the region Ω with one cell for each child of the root. An iterative optimization process adaptively adjusts the value of weight and the position of the site, such that the areas of the cells meet the requirement. The final layout requires that the area of each cell should be in proportion to the associated value. In practice, if the area error is smaller than a threshold, then we will say the requirement is met and the iterative process has converged. Formally, let $value_{s_i}$ be the associated value of site s_i and $E_{threshold}$ be the threshold of the area error, then the convergent requirement can be expressed as:

$$\frac{\sum_{s_i \in S} \left| A\left(v\left(s_i\right)\right) - A\left(\Omega\right) * \frac{value_{s_i}}{value_S} \right|}{A\left(\Omega\right)} < E_{threshold}. \tag{5.5}$$

Once the iterative process has converged, the above-mentioned processes recurse to subdivide child region until all the leaves of the hierarchy are represented by cells with desired areas.

5.3 Orthogonal Voronoi Treemap

In this section, we introduce our orthogonal Voronoi treemap algorithm (OVT). We first provide an overview of our algorithm in Sect. 5.3.1, and then we discuss the three main steps of our algorithm: initialize site status (Sect. 5.3.2), update site status (Sect. 5.3.3), and compute the weighted orthogonal Voronoi diagram (Sect. 5.3.4). Finally, we give some implementation details in Sect. 5.3.5.

The overall structure of our algorithm similar to that of Nocaj and Branches [97]. However, ours is different in the following aspects:

- a new initialization strategy;
- a modified site update strategy;
- a novel distance function;
- a heuristic algorithm for diagram generation;

5.3.1 Overview

The proposed orthogonal Voronoi treemap follows the same rules as the traditional Voronoi treemap. Because of the recursive feature of treemaps, a single layer is enough. A pseudo-code to compute single layer orthogonal Voronoi treemap is summarized in Algorithm 5.1.

An initialization process is first conducted on the hierarchical data to generate the initial positions and weight for each site. This is achieved by the function *Initialization*(S, Ω, W) (Line 1). Then, an initial weighted centroidal orthogonal Voronoi diagram by the function *ComputeWOVDiagram*(S, Ω, W) is generated based on the initial status of sites (Line 3). After that, an iterative process updates the site status by the function *AdaptPositionsWeights*(S, $\upsilon(S)$, W) and redraws the diagram until

Algorithm 5.1 Compute Orthogonal Voronoi Treemap (single layer)

Input: S; Ω; W; i_{max}; $E_{threshold}$;
Output: $\upsilon(S)$;
ComputeOVTreemap(S, Ω, W)

1: *Initialization*(S, Ω, W) ;
2: $error = +\infty$;
3: $\upsilon(S)$ =*ComputeWOVDiagram*(S, Ω, W) ;
4: **for** $i = 1 : i_{max}$
5: $[S, W] = AdaptPositionsWeights(S, \upsilon(S), W)$;
6: $\upsilon(S)$ =*ComputeWOVDiagram*(S, Ω, W) ;
7: $error = \dfrac{\sum_{s_i \in S} \left| A\left(\upsilon\left(s_i\right)\right) - A\left(\Omega\right) * \dfrac{value_{s_i}}{value_S} \right|}{A\left(\Omega\right)}$;
8: **if** $error < E_{threshold}$, **then**
9: **return** $\upsilon(S)$;
10: **end**
11: **end**
12: **return** $\upsilon(S)$;

the convergence requirement is met. To handle hierarchical data with multiple layers, the function *ComputeOVTreemap(S, Ω, W)* will be called recursively.

5.3.2 Initialization

The initialization step refers to the determination of the initial position of each site and the initial value of weight which is used to control the area of cell. Instead of a random initial value, a reasonable initial site status can significantly improve the algorithm performance. In this section, we determine the layout initial status and the initial weight with the help of the treemap layout.

To start, the hierarchical dataset is visualized by the squarified treemap algorithm [82] since it has been proved that the squarified treemap has a good aspect ratio [132]. It should be noted other treemap algorithm also could be used here. Once the treemap is plotted, we generate a site at the center of each rectangle and calculate the relative position of this site respect to its parent's rectangle boundary. As shown in Fig. 5.1 (left), the value of R_x and R_y is then rescaled to the interval $[0, 1]$, with respect to the size of the outer rectangle. This relative position is set as the initial position of this site. When it is used in our algorithm, this relative position of that site is decoded according to its parent's new cell. This decoding is conducted by con-

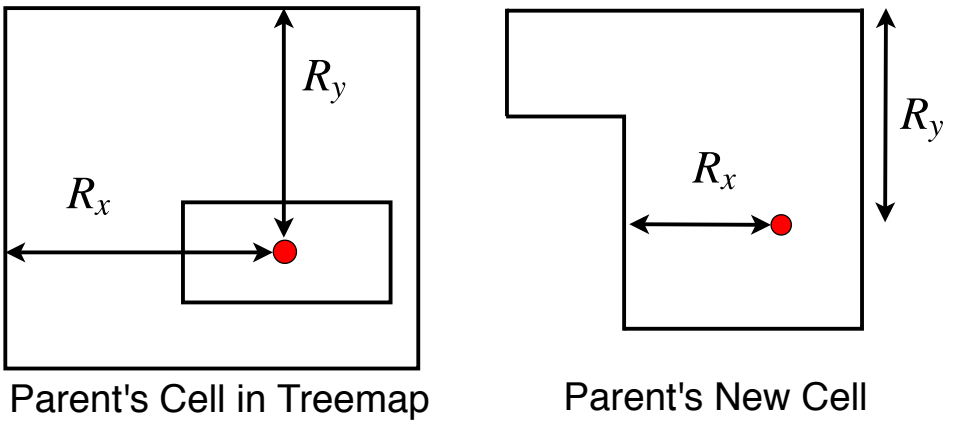


Figure 5.1. Transform the position of a site in rectangle cell of the treemap into the orthogonal rectangle cell as the initial position.

sidering R_y first and then R_x. In this way, the position of sites in the treemap will be transformed into the orthogonal rectangles of our plot as the initial position. For the initial weight value, we set it to the half of the area of the site's cell in the treemap. Our experiments show that with these initial setting the final layout is much better than that with random initial status.

5.3.3 Site Status Update

To update the weight value and the position of each site, we modify the strategy used by Hahn et al. [98] since our orthogonal segmentation is different. Instead of updating position and weight separately, we merge these two functionalities into one and make modifications based on our case. The pseudo code for the function *AdaptPositionsWeights(S, $v(S)$, W)* is depicted in Algorithm 5.2. The main difference between our new *AdaptPositionsWeights(S, $v(S)$, W)* and previous methods [97, 98] is that we do not need to check the maximum weight value allowed for each site during the adjustment. The reason is that when we generate the segmentation lines, we have taken the overweight case into consideration (Sect. 5.3.4.1) in order to avoid the generation of empty cells.

Algorithm 5.2 Adapt positions and weights.

Input: S; $v(S)$; W;
Output: S; W;
AdaptPositionsWeights(S, v(S), W)

$f_s = 0$;
foreach $s_i \in S$
    $c_i = centroid(v(s_i))$;
    $A_{current} = A(v(s_i))$;
    $A_{target} = A(\Omega) * \frac{value_{s_i}}{value_S}$;
    $f_{adapt} = \frac{A_{target}}{A_{current}}$;
    if $f_s \neq 0$ **and** $sgn(f_{adapt} - 1) \neq sgn(f_s - 1)$, **then**
        $f_{adapt} = min(1 + \rho, max(f_{adapt}, 1 - \rho))$;
    end
    $w_i = max(w_i * \sqrt{f_{adapt}},\ \epsilon)$;
    $f_s = f_{adapt}$;
    $s_i^* = s_i + (c_i - s_i) * (1 - 0.5\rho)$;
    if s_i^* within $v(s_i)$, **then**
        $s_i = s_i^*$;
    else
        $s_i = s_i$;
    end
end
return S, W;

5.3.4 Computation of the Orthogonal Voronoi Diagram

In this section, we describe the computation of the proposed orthogonal Voronoi diagram. Since the diagram is formed by segmentation lines between sites, the strategy to partition the space between two sites is important. According to Eq. 5.1, the space partitioning can be treated as a clustering of points based on the sites, if the whole plane is covered by uniformly distributed points. There are two main problems here. The first problem is how to calculate the distance to the sites, while the second is that for a certain point which site should it belong to. These two problems are straightforward when using the Euclidean distance in the Voronoi diagram. However, they are tough when trying to axis-alignedly subdivide the bounded region. To achieve this, we introduce a new distance function in Sect. 5.3.4.1 as well as how to find the reference site among multiple sites in Sect. 5.3.4.2. Once the partition between two sites is confirmed, how to allocate the partitioned spaces to form a diagram is considered. Here, a sweepline + skyline heuristic algorithm is introduced in Sect. 5.3.4.3.

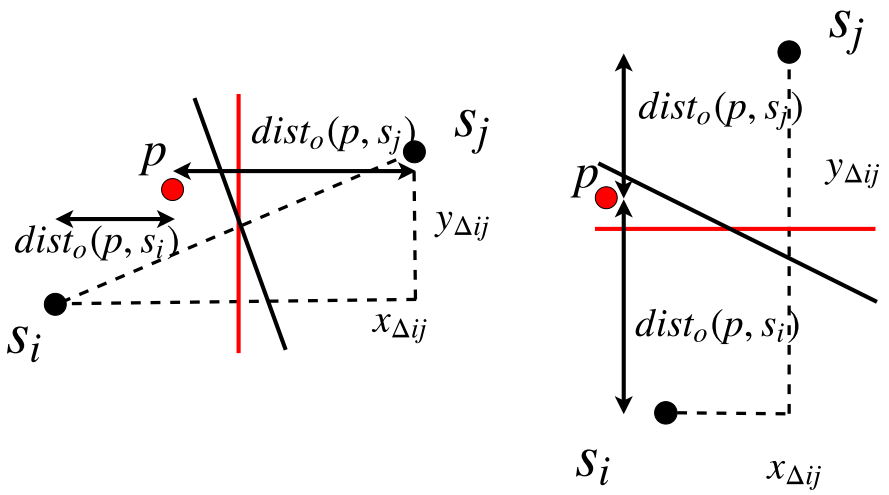


Figure 5.2. Two kinds of relative positions for sites s_i and s_j where $x_{\Delta ij} = |x_i - x_j|$ and $y_{\Delta ij} = |y_i - y_j|$. The black solid line is the segmentation based on Euclidean distance while the red solid line is an axis-aligned segmentation. The distance $dist_o(p, s_i)$ and $dist_o(p, s_j)$ in both cases are illustrated.

5.3.4.1 Distance Function

The new distance function considers the relative positions of two sites rather than only one site in previous distance function. Formally, when calculating the distance of point p and site s_i, we consider the relative positions of site pair s_i and s_j to decide which coordinate should be considered. Since we try to divide the space in axis-aligned manner, we consider two kinds of relative positions between site s_i and s_j based on their x-axis difference $x_{\Delta ij} = |x_i - x_j|$ and y-axis difference $y_{\Delta ij} = |y_i - y_j|$. Then for an arbitrary point p, the distance to site s_i in both cases are depicted in Fig. 5.2 and defined as:

$$dist_o(p, s_i) = \begin{cases} |x_p - x_{s_i}| & if\ x_{\Delta ij} > y_{\Delta ij}, \\ |y_p - y_{s_i}| & if\ x_{\Delta ij} < y_{\Delta ij}. \end{cases} \tag{5.6}$$

When a positive weight value is associated with each site, the distance function (Eq. 5.6) should be modified as:

$$dist_o(p, s_i) = \begin{cases} |x_p - x_{s_i}| - w_i & if\ x_{\Delta ij} > y_{\Delta ij}, \\ |y_p - y_{s_i}| - w_i & if\ x_{\Delta ij} < y_{\Delta ij}. \end{cases} \tag{5.7}$$

For the left-hand-side case in Fig. 5.2, a vertical line is needed to separate the site pair while a horizontal line is needed for the right-hand-side case in Fig. 5.2. Hence, the axis-aligned segmentation line L is defined as:

$$L_{s_i s_j} : \begin{cases} x = \frac{1}{2}\left(x_{s_i} + x_{s_j}\right) & if\ x_{\Delta ij} > y_{\Delta ij}, \\ y = \frac{1}{2}\left(y_{s_i} + y_{s_j}\right) & if\ x_{\Delta ij} < y_{\Delta ij}. \end{cases} \tag{5.8}$$

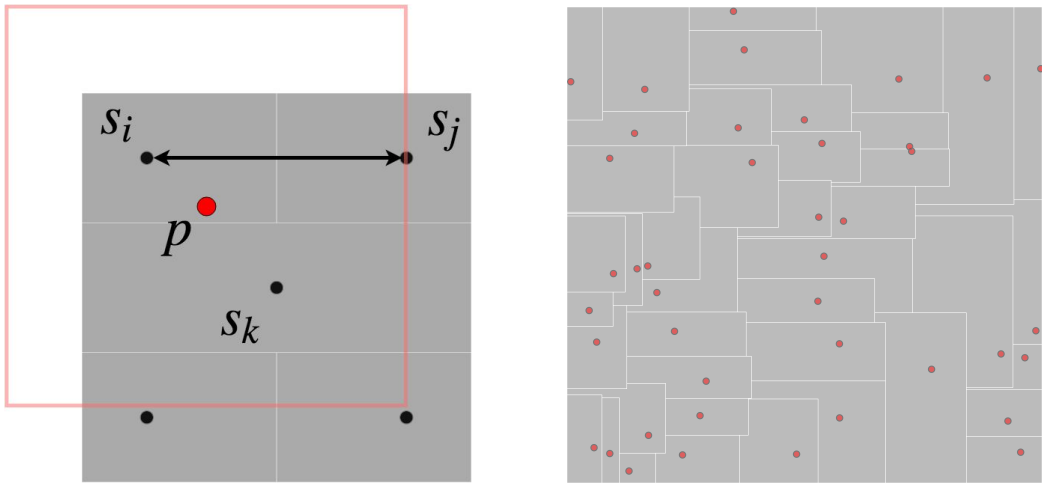
If two sites have large $y_{\Delta ij}$, then they will be separated by a horizontal line. Otherwise, these two sites will be separated by a vertical line. For the case that $x_{\Delta ij} = y_{\Delta ij}$, we break the tie by choosing a horizontal segmentation.

To guarantee that there is no cell with empty region, the segmentation lines should be located in between. This is based on an assumption that the sum of w_i and w_j are smaller than the distance between two sites. However, it is possible that this assumption fails. In this case, we partition the space according to the ratio of the weight values by the function *GenerateWLine(s_i, s_j, w_i, w_j)*. A pseudo-code for this function is depicted in Algorithm 5.3.

Algorithm 5.3 Generate Weighted Segmentation Line

Input: s_i; s_j; w_i; w_j;
Output: $L_{s_i s_j}$;
GenerateWLine(s_i, s_j, w_i, w_j)

1: **if** site s_i and s_j are horizontal neighbors, **then**
2: **if** $|x_{s_i} - x_{s_j}| - w_i - w_j >= 0$, **then** $L_{s_i s_j} = min(x_{s_i} + w_i, x_{s_j} + w_j) + \frac{1}{2}\left(|x_{s_i} - x_{s_j}| - w_i - w_j\right)$;
3: **else** $L_{s_i s_j} = (x_{s_i} < x_{s_j})\ ?\ x_{s_i} + \frac{w_i}{w_i + w_j} * |x_{s_i} - x_{s_j}| : x_{s_j} + \frac{w_j}{w_i + w_j} * |x_{s_i} - x_{s_j}|$;
4: **end**
5: **else**
6: **if** $|y_{s_i} - y_{s_j}| - w_i - w_j >= 0$, **then** $L_{s_i s_j} = min(y_{s_i} + w_i, y_{s_j} + w_j) + \frac{1}{2}\left(|y_{s_i} - y_{s_j}| - w_i - w_j\right)$;
7: **else** $L_{s_i s_j} = (y_{s_i} < y_{s_j})\ ?\ y_{s_i} + \frac{w_i}{w_i + w_j} * |y_{s_i} - y_{s_j}| : y_{s_j} + \frac{w_j}{w_i + w_j} * |y_{s_i} - y_{s_j}|$;
8: **end**
9: **end**
10: **return** $L_{s_i s_j}$;



(a) (b)

Figure 5.3. (a) One example for choosing the reference sites for certain point p in red. A square in pink centered at the point p is built, including at least one horizontal site pair and one vertical site pair. By first considering horizontal site pair s_i and s_j, and then vertical site pair s_i and s_k, point p has smallest distance to site s_i. (b) A more complex case is given in which the orthogonal Voronoi diagram is generated based on the positions of a series of random sites (in red).

5.3.4.2 Clustering

Once the distance function is confirmed, the second question is that for point p, which site should it belong to after calculating the distance. According to Eq. 5.7, point p may have multiple distance values to the same site when considering different site pairs. To eliminate confusion, we first define the concept of valid neighborhood relationship between sites. Then we describe a principle to handle the case with multiple site pairs from the perspective of point p.

For a site pair, if there are no other sites located in the rectangle region formed by the two sites, then these two sites are valid neighbors. In terms of the relative positions, these two sites are either horizontal neighbors (i.e. neighbors in the horizontal direction) or vertical neighbors (i.e. neighbors in the vertical direction). The valid neighbor sites can also be considered from the perspective of the final diagram. If the existence of site s_j contributes to the generation of the cell of site s_i, then site s_i and s_j are valid neighbors.

For the case when there are multiple sites around, we define a principle to do clustering. For easy understanding, we take the case in Fig. 5.3 (a) as an example. Firstly, we design the smallest square (in pink) centered at point p, including multiple

sites such that there are at least one pair of vertical valid neighbor sites (s_i and s_k) and one pair of horizontal valid neighbor sites (s_i and s_j). Secondly, the segmentation line between the horizontal neighbor sites determines which site the point p should belong to. Thirdly, suppose point p belongs to site s_i, then check the distance between point p and all of the vertical neighbors of site s_i to find the smallest one. If site s_i has the smallest or second smallest vertical distance to point p, then point p will be clustered accordingly. Otherwise, extend the square until such kind of case exists. A more complex case is illustrated in Fig. 5.3 (b).

However, this principle is complex and not automatic. Hence, a computer-based algorithm should be designed to automatically execute this principle. In the next section, a sweepline + skyline algorithm is described to carry out the partitioning based on the new distance function.

5.3.4.3 Sweepline + skyline algorithm

The proposed sweepline + skyline algorithm aims to automatically partition the canvas into orthogonal sub-regions based on the new distance function proposed in Sect. 5.3.4.1. Our idea is motivated by the sweep line algorithm for Voronoi diagram [129] and the skyline strategy used in cutting and packing problem [86]. The sweep line used in our algorithm is a vertical line moving from left to right. When the sweep line hits a new site, the relationships of this new site and all its left-hand-side site pairs are checked to generate vertical or horizontal segmentation lines. Meanwhile, a skyline is defined to record the current segmentation lines for all active sites. When the sweep line hits a new site and new segmentation lines are generated, the skyline will be updated correspondingly.

Figure 5.4 illustrates an example for this process. As shown in Fig. 5.4 (a), for a given rectangular canvas with $width = 1000$ and $height = 680$, six sites are positioned based on their coordinates (we follow the image coordinate system where the y-axis is pointing downwards). The size of the canvas and the positions of sites are the input values of our algorithm. The first step (Fig. 5.4 (b)) is to create a vertical sweepline and a vertical skyline. The sweepline is initially located on the left-most site s_A while

Algorithm 5.4 Compute Orthogonal Voronoi Diagram

Input: S; Ω; W;
Output: $\upsilon(S)$;
ComputeWOVDiagram (S, Ω, W)

Sort S in the order of x_{s_i} ascending ;
Initialize $L_{sweepline}$;
Initialize $L_{skyline}$;
Initialize $\upsilon(S) = [\ ]$;
for $i = 1 : n$
    for $j = 1 : i - 1$
        if site s_j is closed, **then**
            continue;
        end
        if site s_j is not the valid neighbor of site s_i, **then**
            continue;
        end
        if site s_i and s_j are vertical neighbors, **then**
            $L_{s_i s_j}$ = *GenerateLine*(s_i,s_j, w_i, w_j) ;
            Update $L_{skyline}$ with $L_{s_i s_j}$;
        else
            $L_{s_i s_j}$ = *GenerateLine*(s_i,s_j, w_i, w_j) ;
            Mark site s_j as closed;
            Generate the bounding polygon $\upsilon(s_j)$ for site s_j;
            $\upsilon(S).push(\upsilon(s_j))$;
            Update $L_{skyline}$ with $L_{s_i s_j}$;
        end
    end
    Update $L_{sweepline}$;
end
for $i = 1 : n$
    if site s_i is closed, **then** continue;
    else
        Generate the bounding polygon $\upsilon(s_i)$ for site s_i;
        $\upsilon(S).push(\upsilon(s))$;
    end
end
return $\upsilon(S)$;

the skyline is on the left-hand-side of the canvas. The length of both lines equal to the height of the canvas. The second step (Fig. 5.4 (c)) is to sweep the sweepline from left to right to hit the next site s_B. Since $y_{\Delta AB} = 272$ is larger than $x_{\Delta AB} = 96$, a horizontal line $L_{AB} : y = 296$ is built between site s_A and s_B. The skyline is then updated by adding a horizontal line segment. After checking all the left-hand-side site pairs of site s_B, the sweepline moves to the next site s_C (Fig. 5.4 (d)). Since site s_A is not closed and is the horizontal neighbor of site s_C, then a vertical line $L_{AC} : x = 208$ is built. Once a vertical line is generated, the left site s_A of this horizontal neighbor sites should be closed and the bounding polygon of site s_A is formed based on the current skyline and the new vertical segment (as well as the canvas boundary). After that, the skyline is updated. If site s_B is not closed and is the vertical neighbor of site s_C, then a horizontal line $L_{BC} : y = 256$ is built and the skyline is updated again (Fig. 5.4 (e)). This process is repeated until the sweepline reaches the last site s_F (Fig. 5.4 (f)). The residual sites (s_E and s_F) will then be closed and the bounding polygon for each site is formed by the current skyline and sweepline together. A pseudo-code for the whole process is depicted in Algorithm 5.4.

5.3.5 Implementation

The implementation of our proposed algorithms is in JavaScript. Our code mainly depends on the D3.js package [133] and previous implementation on the Voronoi treemap [134][2]. The color schema for some of the plots in this chapter is generated by the Tree Color [3], such as Fig. 5.12 (b, c). The source code of our implementation can be found in Github.

5.4 Performance

In the following, we evaluate the performance of our proposed algorithm from multiple aspects. First, we check the average number of site pairs and valid neighbors for each site since they have a strong relationship with the algorithm complexity in Sect. 5.4.1.

[2]This JavaScript implementation is based on the work by Noca and Brandes [97] in which the program is run in Java.

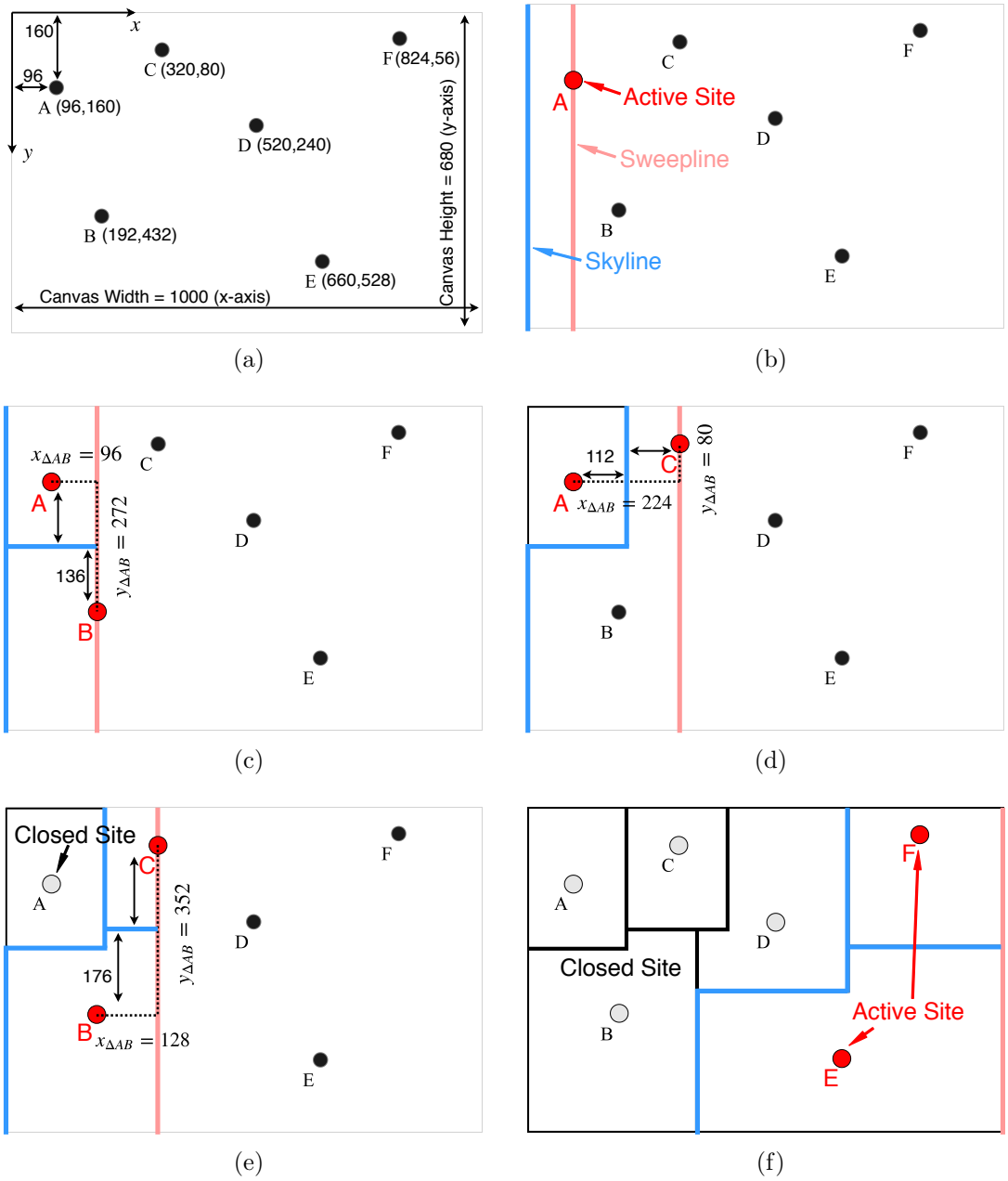


Figure 5.4. Overview of the sweepline + skyline algorithm. (a) Initially, six sites are positioned inside the canvas based on their initial positions. (b) The skyline (in blue) and the sweepline (in pink) are initialized and the first site s_A (left-most) is activated. (c) Sweep the sweepline to meet the second site s_B and determine the relationship of s_A and s_B. Then a horizontal segmentation line L_{AB} is added to the skyline. (d) Sweep the sweepline to site s_C and determine the relationship of s_A and s_C. Since s_A and s_C are horizontal neighbors, a vertical segmentation line is generated. In this case, site s_A is closed. A bounding polygon formed by the skyline and the canvas boundary and the vertical segmentation line is built for s_A. Then the skyline is updated and site s_A is marked as closed status. (e) Sites s_B and s_C are then considered and a horizontal segmentation line is added to the skyline. (f) The process is continued until the last site s_F is considered. All the active sites will be closed and the corresponding bounding polygon will be built based on the current skyline and the canvas boundary.

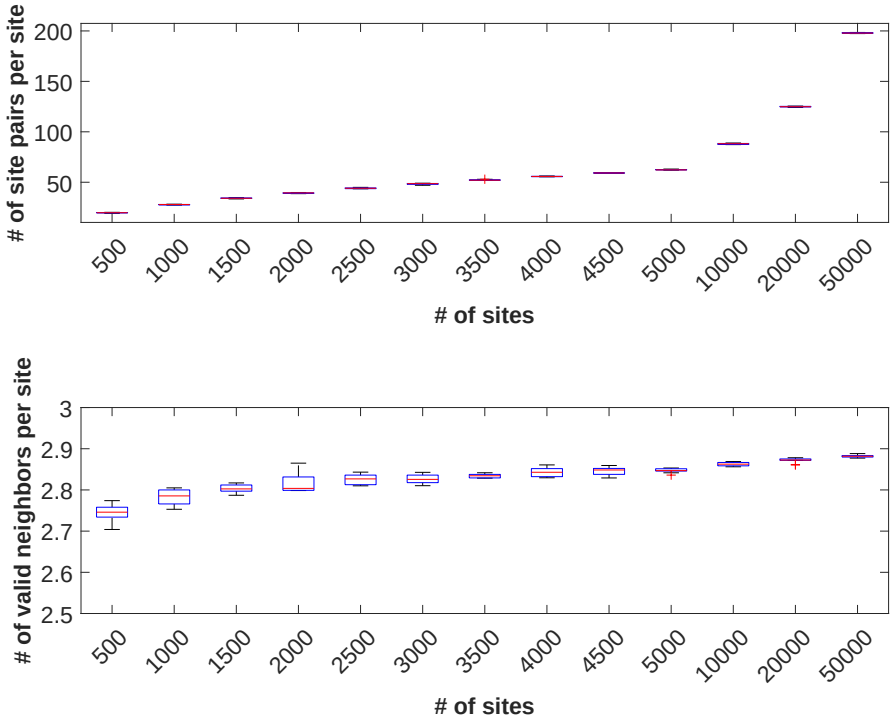


Figure 5.5. The average number of site pairs (top) and valid neighbors (bottom) per site. Noted that the x-axis is not linear.

Second, we compare our algorithm with the Voronoi treemap in terms of running time during a single iteration in Sect. 5.4.2. Then, the converging rate and the aspect ratio are analyzed in Sect. 5.4.3 and Sect. 5.4.4, respectively.

5.4.1 Site Pairs and Valid Neighbor

The number of neighbors for each site contributes to the computation complexity of our proposed algorithms. As depicted in Algorithm 5.4, all the non-closed sites will be checked to update the status by generating new segmentation lines and updating the skyline. This will lead to an $O(n^2)$ computation complexity. However, in practice, much fewer times are needed for the diagram to update its status. To show this, we generate a series of random datasets with a different number of sites (from 500 to 50000). For each dataset, the positions of sites are initialized randomly in an empty square canvas (900×900). We run our proposed algorithm on each dataset ten times to have a fair result since our tests are based on random data. During the experiments, two variables are recorded: the total number of site pairs checked and

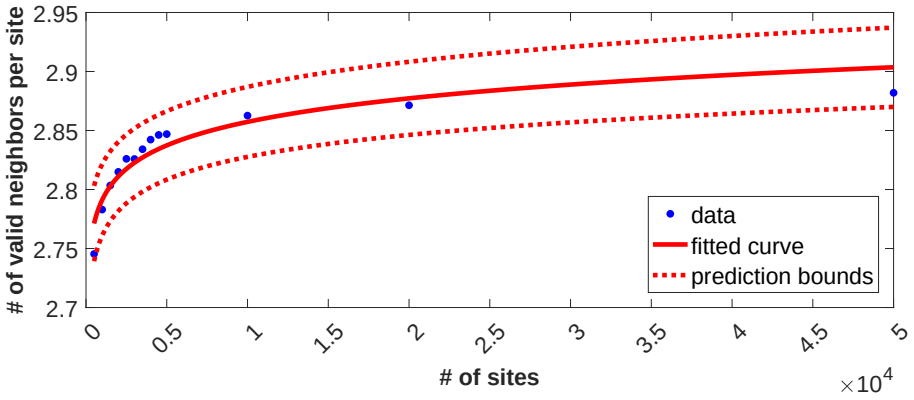


Figure 5.6. The fitting line and the prediction for the average number of valid neighbors per site.

the total number of valid neighbors which change the diagram status. In the end, the average number of site pairs and the average number of valid neighbors per site are illustrated in Fig. 5.5 as box plots (noted that the x-axis is not linear).

The average number of site pairs per site refers to how many pairs of sites need to be checked for each site. If they are not valid neighbors, then the diagram status will not be changed. Since the check process is much less time expensive than updating the diagram, a small amount of computation time will be taken. It should be noted that the average number of site pairs per site is not linearly increased along the number of sites. We formulate the relationship between the number of sites and the average number of site pairs per site as follows:

$$\#\ of\ sitepairs\ per\ site = 0.8563 * 2^{0.7256 * log(\#\ of\ \ sites)} + 0.2494. \tag{5.9}$$

To evaluate the goodness of fit, we calculate the goodness-of-fit statistics including the sum of squares due to error (SSE), R-square, Adjusted R-square, root mean squared error (RMSE) [135]. In this case, the mean values in the boxplot are used for fitting and we get a good fitting with SSE=0.0664, R-square=1.0, Adjusted R-square=1.0, and RMSE=0.0815.

Although the number of site pairs per site is large, there are only limited valid neighbors which will lead to the change of the diagram status. According to Fig. 5.5

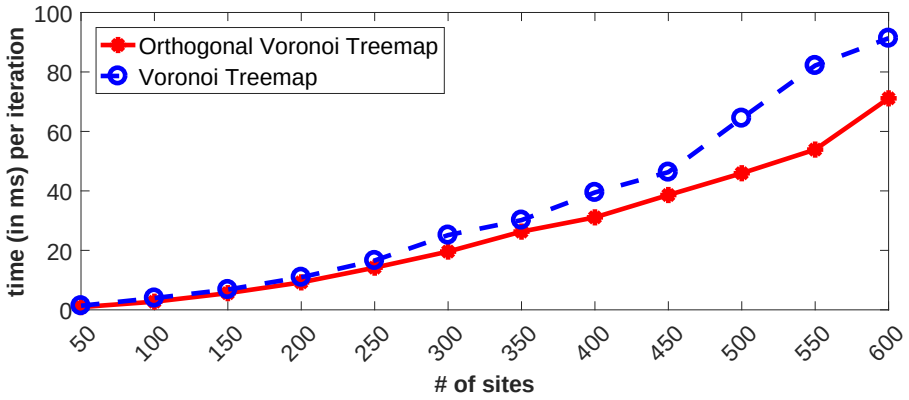


Figure 5.7. The running time (in ms) for a single iteration.

(bottom), we formulate the relationship by fitting, which can be expressed as:

$$\#\ of\ valid\ neighbors\ per\ site = 0.02875 * log(\#\ of\ sites) + 2.592. \tag{5.10}$$

The goodness-of-fit statistics in this case is SSE=0.0018, R-square=0.8929, Adjusted R-square=0.8832, and RMSE=0.0127 which indicate that the fitting function is suitable. We plot the fit and the prediction in Fig. 5.6.

Formally, we can conclude that the computation complexity for a single iteration is $O(n \cdot \log(n))$ and the overall complexity is $O(k \cdot n \cdot \log(n))$ in Algorithm 5.4. Since the complexity for updating sites status (Algorithm 5.2) is linear, we can show that the proposed orthogonal Voronoi treemap has $O(k \cdot n \cdot \log(n))$ complexity for a single layer hierarchical data.

5.4.2 Single Iteration Comparison

Table 5.1. The running time (in ms) for a single iteration.

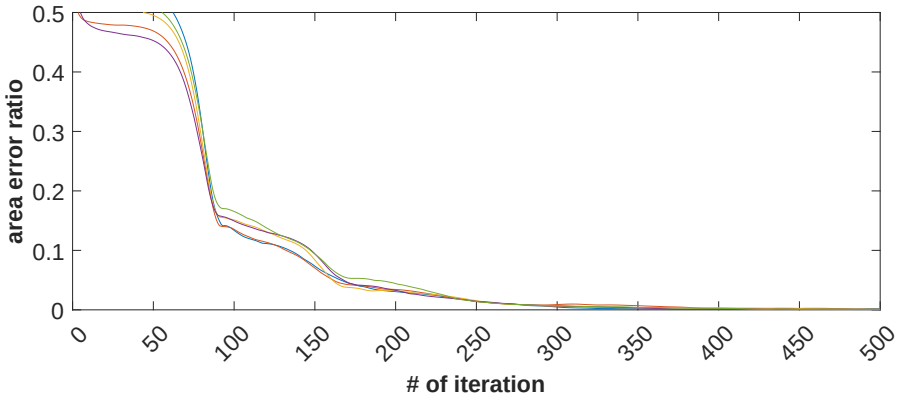
	50	100	150	200	250	300	350	400	450	500	550	600
VT [97]	0.22	0.3	0.43	0.61	0.79	1.05	1.19	1.43	1.82	1.94	2.2	2.52
VT	1.38	3.96	6.83	10.95	16.59	25.13	30.17	39.48	46.30	64.45	82.17	91.34
OVT	0.93	2.74	5.61	9.24	14.19	19.60	26.31	31.10	38.61	45.94	53.90	71.13

In addition to the computation complexity analysis above, we directly compare the running time of our algorithm during a single iteration to that of the Voronoi treemap. Although several running time are provided in the literature [94, 95, 97], we compare our algorithm with the approach of Nocaj and Brandes [97] since it is the

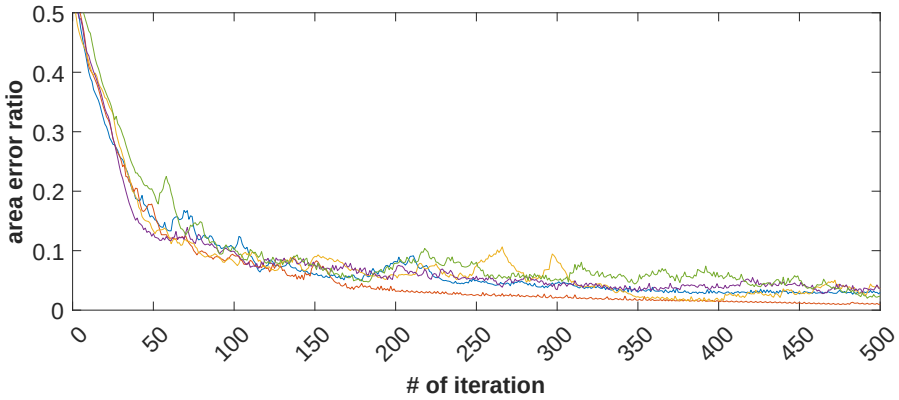
fastest to date. However, their algorithm is implemented in a different programming language. To have a fair comparison, we utilize the JavaScript implementation of their algorithm instead. We run the JavaScript Voronoi treemap package [134] and our algorithm on the same computer (PC, Window 10, Intel Xeon CPU, 3.6 GHz, 16 GB memory). The experiment is also conducted on a series of random datasets with a different number of sites (from 50 to 600) which are associated with random values (approximately 30 % of sites with random values in $[0, 10]$ and the rest of the sites with random values in $[0, 1]$). For each dataset, both algorithms run 1000 iterations (we comment out the converge constraint in Algorithm 5.1 such that the algorithm will keep running until the maximum iteration number is reached). The average running time (in ms) is recorded in Table 5.1 and plotted in Fig. 5.7. Meanwhile, we list the concrete running time by Nocaj and Brandes [97] in Java as a reference.

Our algorithm needs less time per iteration than the Voronoi treemap as shown in Fig. 5.7 when both are run in the same programming language. The reason for this is that we modified the update sites status process in Sect. 5.3.3. In one iteration of the Voronoi treemap, it updates the sites' positions and weights, draws a Voronoi treemap, updates weights and then again draws a Voronoi treemap. In our modification, we merge the update of positions and weights into one step and handle the overweight case in the generation of segmentation line (Algorithm 5.3). Hence in one iteration of our algorithm, we only need to draw the orthogonal Voronoi treemap once rather than twice in the Voronoi treemap.

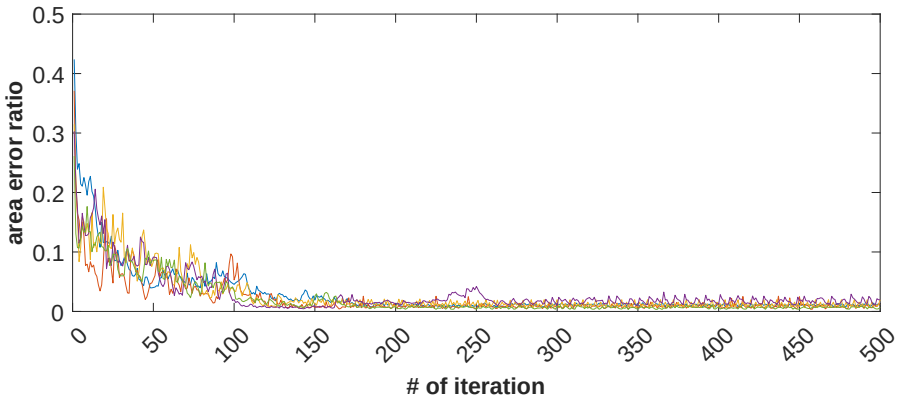
Compared with the original running time provided by Nocaj and Brandes [97], both our algorithm and the Voronoi treemap JavaScript package [134] need much more time. Since no hardware-accelerated code is used in all cases, we believe that this difference is due to the capabilities of different programming languages. It would be one interesting future work to increase the efficiency by implementing the algorithm in Java in the backend and then displaying the results in JavaScript in the frontend.



(a) Voronoi Treemap



(b) Orthogonal Voronoi Treemap with random initial status



(c) Orthogonal Voronoi Treemap with designed initial status

Figure 5.8. The converge rate of the Voronoi treemap, our orthogonal Voronoi treemap with random initial status and with our proposed initial status. Within the 500 iterations, we record the area error of each iteration to indicate the converge rate.

5.4.3 Converge Rate

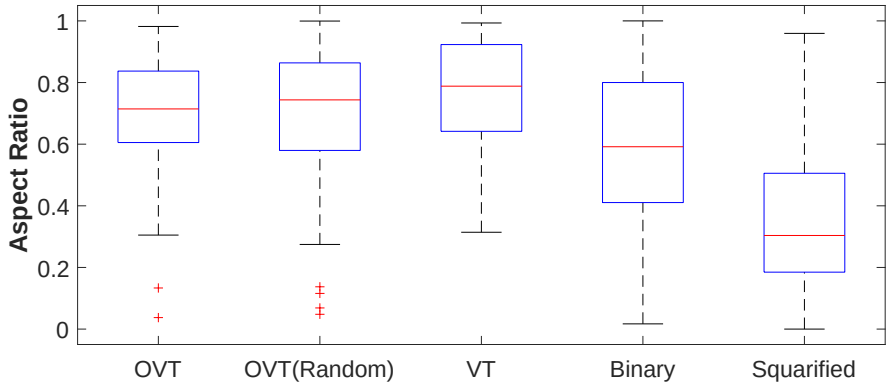
In this section, we test the converging speed of the algorithms by considering the changes of the area error ratio along the iteration number increases, since the area of each element in the plot should match its associated value. Starting from initial positions and initial weights, the status of sites is updated iteratively as discussed

in Sect. 5.3.3. To test the converging speed, we generate a single-layer hierarchical dataset with 200 random positioned sites which are associated with random values (approximately 30 % of sites with random values in $[0, 10]$ and the rest of the sites with random values in $[0, 1]$). Both the Voronoi treemap and our algorithms are run 5 times. Moreover, we compare our initialization strategy with a random initial status for our algorithms. We calculate and record the area error ratios of both algorithms in each iteration. The results are plotted in Fig. 5.8.

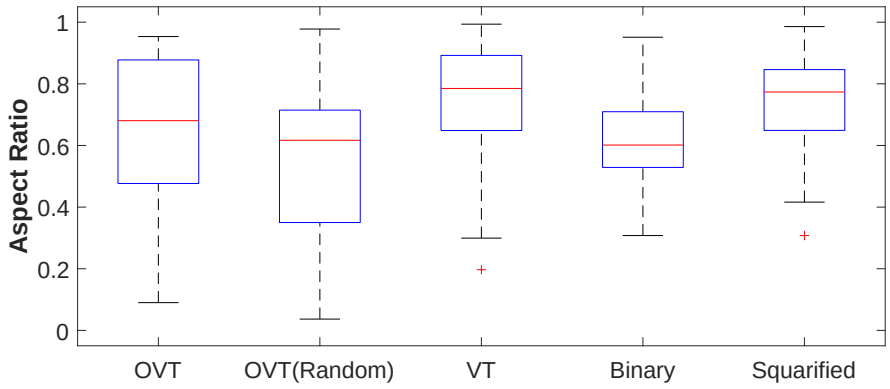
As illustrated, our algorithm with the proposed initial status performs better than that with random initial status and the Voronoi treemap. Compared with random initial status, the proposed initial status guarantees that the algorithm starts from a lower area error. Moreover, the area error ratio can reach a tough constraint (such as 0.01 area error ratio) within 200 iterations. For our algorithm with random initial status, even after 500 iterations this tough constraint may not be satisfied. Compared with the Voronoi treemap with the random initial position and small positive weight, our algorithm generally converges faster. It is believed that the reason for this result is our effective initialization strategy. Since our orthogonal Voronoi treemap is similar to the treemap, using treemap to set initial status significantly contributes to the fast convergence in our algorithm. However, although our algorithm can converge under a tough constraint, it should be noted that our algorithm with both initial statuses has larger fluctuation than the Voronoi treemap. We consider that this is due to the non-consistent distance function used to partition the space (sometimes horizontal, sometimes vertical). While for the Voronoi treemap, the partitioning is straightforward.

5.4.4 Aspect Ratio

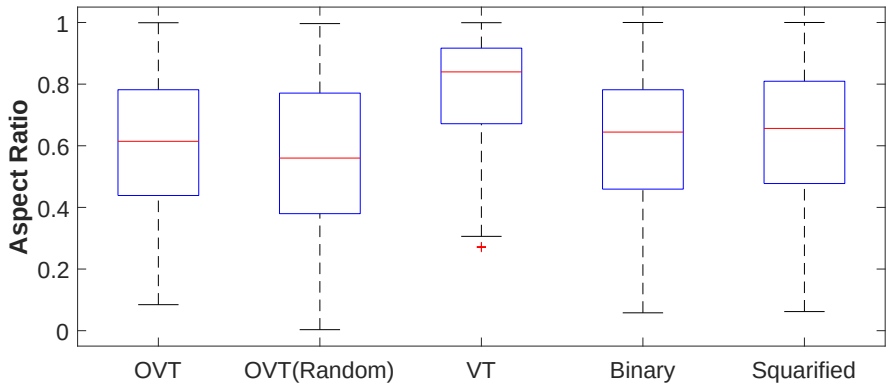
We discuss the aspect ratio of our algorithm and compare with the Voronoi treemap and different treemap layouts in this section. Our experiments are conducted on three datasets, including a random dataset (single layer with 100 sites), the global GDP dataset(two layers with 42 leave nodes), and the Flare class hierarchy (four layers with 220 leave nodes). For our algorithm and the Voronoi treemap, we set the maximum



(a) Test on random dataset



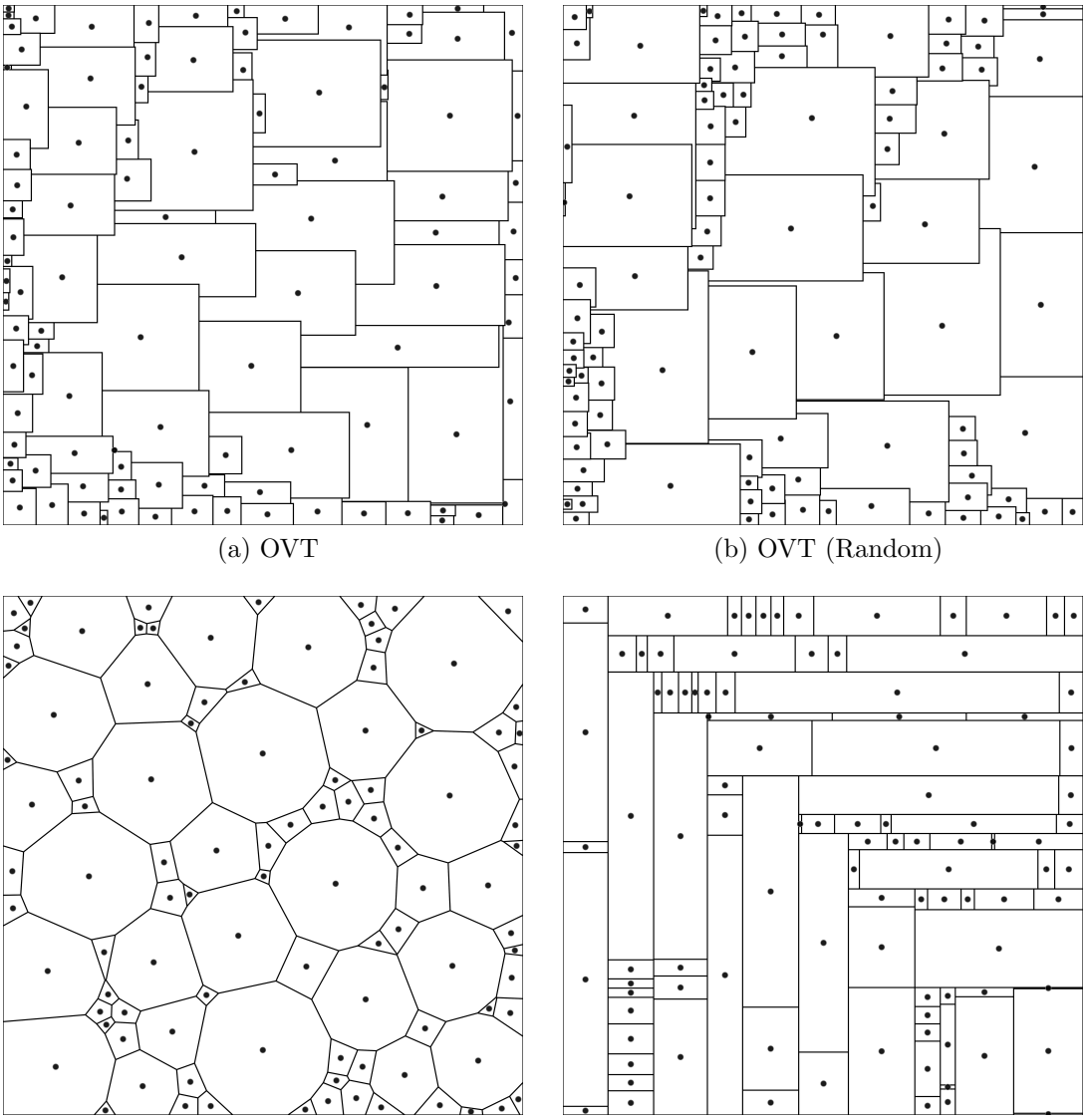
(b) Test on the globalGDP dataset



(c) Test on the Flare dataset

Figure 5.9. The boxplots of aspect ratio. In these tests, our orthogonal Voronoi treemap (OVT) has a comparable mean aspect ratio with the treemaps (binary treemap and squarified treemap). While the Voronoi treemap has the best aspect ratio in all tests.

iteration number to 500 and the area error threshold for convergence to 0.01. For our algorithm and the treemaps, the aspect ratio of the axis-aligned minimum bounding box is calculated while for the Voronoi treemap, the aspect ratio of the oriented minimum bounding box is calculated. The result is illustrated as boxplots shown in



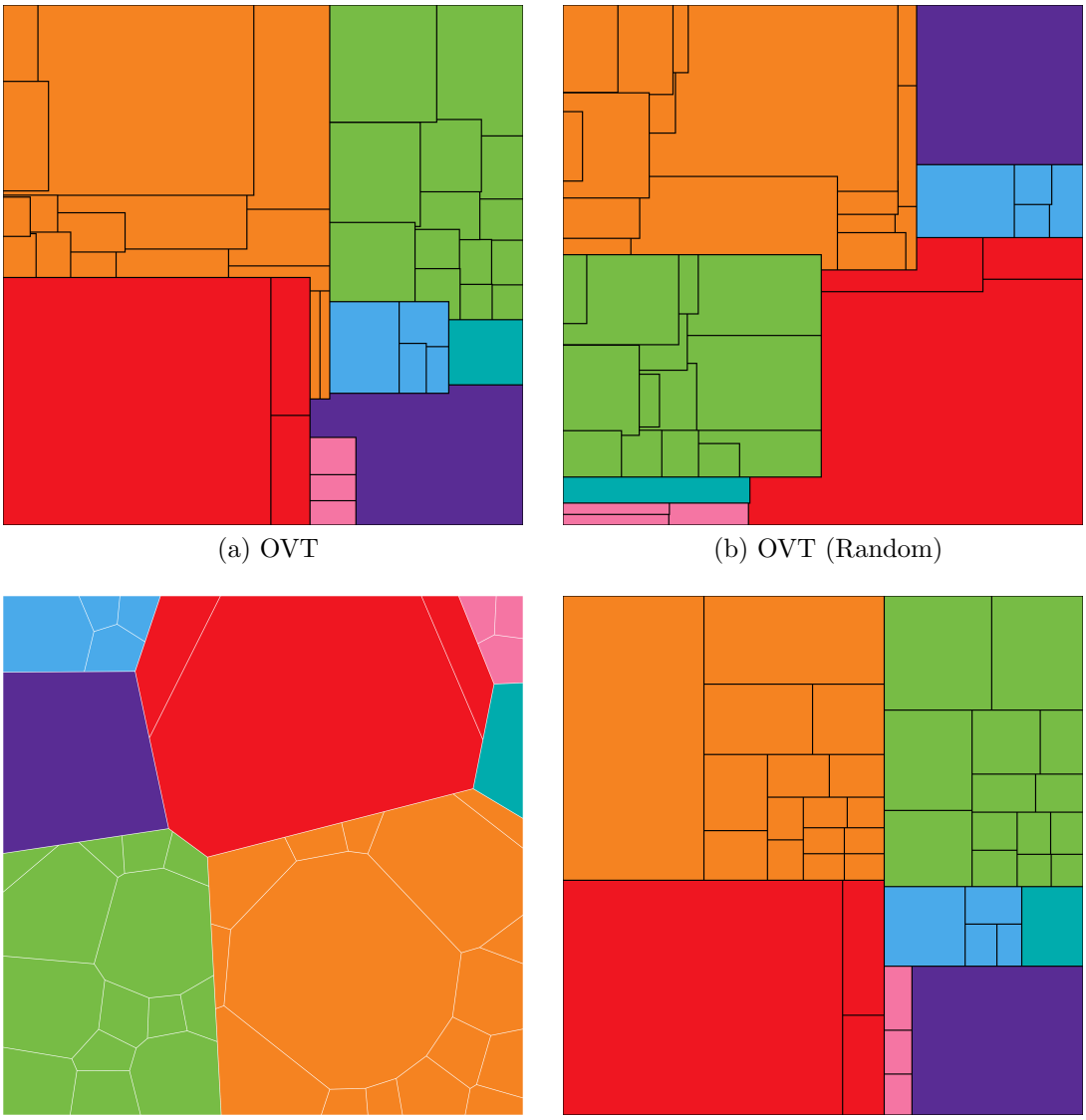
(a) OVT (b) OVT (Random)

(c) Voronoi Treemap (d) Squarified

Figure 5.10. The final layouts on the random dataset.

Fig. 5.9. Meanwhile, we display the final layouts on the random dataset in Fig. 5.10 and the final layouts on the globalGDP dataset in Fig. 5.11. The final layout of our algorithm on the Flare data is shown in Fig. 5.12.

When considering the mean aspect ratio (the red line) in Fig. 5.9, our algorithm is better than treemap layouts on the random dataset and has similar results on two real datasets, although the Voronoi treemap has the best aspect ratio in three cases. When comparing the different initial status of our algorithm, we can find that with the proposed initialization strategy our algorithm has a better aspect ratio (the closer to one the better) and small distribution range.



(a) OVT (b) OVT (Random)

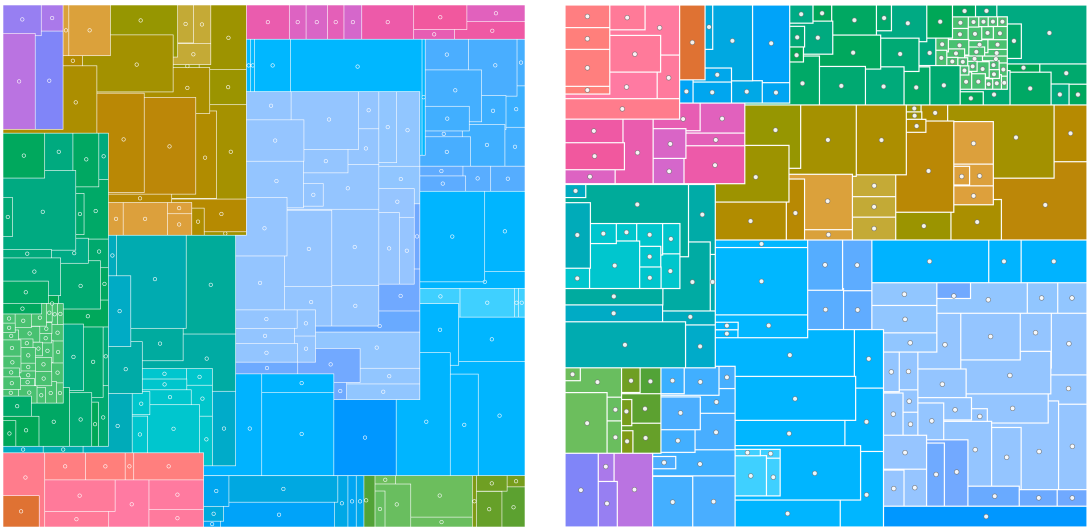
(c) Voronoi Treemap (d) Squarified

Figure 5.11. The final layouts on the globalGDP dataset.

5.5 Discussion

Our algorithm has a comparable performance with the state-of-the-art Voronoi treemap algorithms in terms of computation complexity, computation time, converge speed, and aspect ratio. Moreover, according to the description of our algorithm, the proposed orthogonal Voronoi treemap is flexible to the changes of data value similar to the Voronoi treemap. Meanwhile, it utilizes nested orthogonal rectangles to present each cell and produces a rectangle-like layout.

There are also a few drawbacks in our algorithm. Firstly, as mentioned in Sect. 5.4.3, our algorithm has large fluctuation during the iteration. This shows that the area error in our algorithm is not monotonously decreasing. Fortunately, with our proposed



(a) Orthogonal Voronoi Treemap (Random) (b) Orthogonal Voronoi Treemap

Figure 5.12. (a) The orthogonal Voronoi treemap visualizes the Flare class hierarchy (colored by TreeColors [3]) with random initial status. (b) The orthogonal Voronoi treemap uses the squarified treemap as initial status.

initialization strategy, the area error can be decreased to an acceptably small value. The second drawback is that the position of some sites in our algorithm may move in a large range. This leads to a non-stable layout. In the site status update procedure, we have no strategy used to preserve the relative positions of sites.

It would be an interesting future work for our algorithm to preserve the relative positions of sites during iteration in order to visualize dynamic hierarchical data for which a stable layout is essential [95, 98, 132]. Another future work is to utilize the treemap to visualize high-dimensional hierarchical dataset. Currently, the parameter associated to each site is usually only one dimension. When it comes to high-dimensional data visualization [107], how to visualize the values or other information while indicating the hierarchical structure would be an interesting direction.

5.6 Summary

In this chapter, we described a novel algorithm for the Voronoi treemap with nested orthogonal rectangles. To the best of our knowledge, this is the first time the sweep line strategy is used to generate the Voronoi treemap. We validated that the proposed algorithm has an $O(n \cdot \log(n))$ complexity which is the same as the state-of-the-art

Voronoi treemap. Moreover, by modifying the update procedure, it is validated that our algorithm requires less computation time than the Voronoi treemap in the same programming language. Owing to the proposed initialization strategy, our algorithm converges faster than the Voronoi treemap and has comparable aspect ratio against the treemap. With a tidy layout by nested orthogonal rectangles and the adjustment capability by site status modification, the proposed orthogonal Voronoi treemap plays the role of the bridge in connecting the treemap and the Voronoi treemap.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this thesis, we aim to visualize engine-associated datasets such that domain experts can get better insights. To achieve this, we have designed a validated specific visualization application and several novel generic visualization methods.

The major contributions of this thesis are summarized as follow:

- In Chapter 3, we have designed EngineQV, a single-page web-based integrated visualization system that integrates engine-associated datasets, allowing domain experts to explore the external factors that affect the aircraft engine performance. The designed visualization system satisfies the task requirements captured in advance. By associating multiple geo-temporal engine-associated datasets, the system is capable of providing various use cases. These use cases contribute to the verification of known external factors affecting the performance of aircraft engine. They help the domain experts to get new observation, such as the strange behaviors for flights around the south-east Asia. According to feedback of the domain experts from Rolls-Royce, the functionalities of the EngineQV satisfy their requirements.

- In Chapter 4, we have proposed novel view exploration operations for high-dimensional data visualization, including the focus+context radial distortion strategy in the RadViz and the geometry algorithm to generate the boundaries of

the star coordinate plot. With the proposed radial distortion strategy, the view can be presented based on the user requirements and the clumping effect in the RadViz is alleviated. Based on the experimental results, our distortion method overcomes previous distortion methods when handling the clumping effect, in terms of the view quality according to the attribute variance and the attribute ratio. The geometry algorithm for the star coordinate plot boundaries fill a gap in the research field since no related work is done before. The boundaries provide an overview of the plot and contribute to better understanding. Both works are modifications based on existing visualization methods by providing novel operations. They contribute to the visualization of the engine simulation data generated by the NASA EngineSim [123].

- In Chapter 5, we have proposed novel visualization method for implicit hierarchical visualization by using nested orthogonal rectangles. The proposed orthogonal Voronoi treemap aims to keep a good balance between shape simplicity and adjustment ability. To achieve this, we first define a new distance function and neighborhood relationship, then design a sweepline + skyline heuristic algorithm for plot generation, and finally propose an initialization strategy to increase the capability. Based on the experimental result, the proposed orthogonal Voronoi treemap has $O(n \cdot log(n))$ complexity which is the same as the state-of-the-art methods. However, it has better performance than the traditional Voronoi treemap in terms of computation time and converge rate. Meanwhile, the proposed orthogonal Voronoi treemap has better performance than the traditional treemap in terms of aspect ratio.

6.2 Future Work

Generally, this thesis has proposed specific visualization application for solving real-world problem and generic visualization methods for discriminating visualization. There are several promising directions for extending our current research as future work.

6.2.1 Advanced Specific Visualization

The future work of the proposed EngineQV visualization system can be conducted in two aspects: an advanced system with more datasets, and a generic system for other engine-related problems.

The EngineQV system can be improved with additional domain knowledge of the problem, since it is designed based on current understanding. As time goes by, new observation and hypothesis may be obtained by exploring with the EngineQV. In this case, additional datasets, visualization forms, and data exploration operations could be added into the system, to further increase the capability.

On the other hand, the visualization system for new real-world engine-related problems can benefit from the EngineQV. In our system, the designed architecture model and querying model can also be utilized by combining with other datasets and visualization forms. Hence, the researcher or designer can focus more on the data model and the visual design, rather than the system architecture. This will speed up the whole design procedure.

6.2.2 Novel Generic Visualization Methods

For the high-dimensional data visualization methods such as RadViz and the SC, the boundaries have not been studied deeply. Although we have proposed the geometry algorithms to calculate the boundaries in our work, the physical meaning of the boundaries are not fully discussed. The bounding polygon can not only provide a better view, but also provide additional information. It is possible to give special meaning to the boundary edge or utilize the distance between the origin and the boundary. This is an interesting research direction to visualize continuous datasets, such as time series data and to find the status changes.

When it comes to the proposed orthogonal Voronoi treemap for the hierarchical visualization, it would be an interesting future work for our algorithm to preserve the relative positions of sites during iteration in order to visualize dynamic hierarchical data for which a stable layout is essential [95, 98, 132]. Another future work is to utilize the treemap to visualize high-dimensional hierarchical dataset. Currently, the

parameter associated to each site is usually only one dimension. When it comes to high-dimensional data visualization [107], how to visualize the values or other information while indicating the hierarchical structure would be an interesting direction. The third future work is to utilize the orthogonal Voronoi treemap to transfer between the treemap and the Voronoi treemap, since the orthogonal Voronoi treemap is a balance between them and has their characters.

Besides the high-dimensional data and hierarchical data, other data types such as time series data, exist in the engine-associated datasets. Various relevant visualization methods for these data types have been proposed in the literature. It is an interesting research direction to make modification on these existing methods and propose novel ideas.

List of Publications

The research papers that were published and submitted during the course of my PhD study are listed as follows.

Journal Papers:

1. **Yan-Chao Wang**, Qian Zhang, Feng Lin, Chi Keong Goh, Hock Soon Seah, **PolarViz: a discriminating visualization and visual analytics tool for high-dimensional data**, *The Visual Computer*, Springer, 2018.
2. **Yan-Chao Wang**, Feng Lin, and Hock-Soon Seah. **Orthogonal Voronoi Diagram and Treemap**, *arXiv preprint*, arXiv:1904.02348, 2019.

Conference Papers:

3. **Yan-Chao Wang**, Qian Zhang, Feng Lin, Chi-Keong Goh, Xuan Wang, and Hock-Soon Seah. **Histogram equalization and specification for high-dimensional data visualization using radviz**. In the *Computer Graphics International Conference* (CGI), ACM, 2017.
4. **Yan-Chao Wang**, Feng Lin, and Hock-Soon Seah. **Evaluation of Star Coordinate Boundaries**. In the *Computer Graphics International Conference* (CGI), ACM, 2018.
5. Qian Zhang, Chung Soo Ahn, Jigang Liu, **Yan Chao Wang**, Feng Lin, Hock Soon Seah, **Interactive geo-temporal data visualization for aircraft engine**, In the *International Workshop on Advanced Image Technology* (IWAIT), SPIE, 2019.
6. **Yan-Chao Wang**, Qian Zhang, Feng Lin, and Hock-Soon Seah. **EngineQV: Investigating External Cause of Engine Failures Based**

on Geo-temporal Association. In the *IEEE Pacific Visualization Symposium* (PacificVis), IEEE, 2019.

Bibliography

[1] M. Rubio-Sánchez, L. Raya, F. Diaz, and A. Sanchez, "A comparative study between radviz and star coordinates," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 619–628, 2016.

[2] K. Daniels, G. Grinstein, A. Russell, and M. Glidden, "Properties of normalized radial visualizations," *Information Visualization*, vol. 11, no. 4, pp. 273–300, 2012.

[3] M. Tennekes and E. de Jonge, "Tree colors: color schemes for tree-structured data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2072–2081, 2014.

[4] M. Sandberg, M. Kokkolaras, J.-O. Aidanpää, O. Isaksson, and T. Larsson, "A master-model approach to whole jet engine analysis and design optimization," in *World Congress on Structural and Multidisciplinary Optimization: 01/06/2009-05/06/2009*, 2009.

[5] G.-D. Sun, Y.-C. Wu, R.-H. Liang, and S.-X. Liu, "A survey of visual analytics techniques and applications: State-of-the-art research and future challenges," *Journal of Computer Science and Technology*, vol. 28, no. 5, pp. 852–867, 2013.

[6] M. Sedlmair, M. Meyer, and T. Munzner, "Design study methodology: Reflections from the trenches and the stacks," *IEEE Transactions on Visualization & Computer Graphics*, no. 12, pp. 2431–2440, 2012.

[7] S. Liu, D. Maljovec, B. Wang, P.-T. Bremer, and V. Pascucci, "Visualizing high-dimensional data: Advances in the past decade," *IEEE transactions on visualization and computer graphics*, vol. 23, no. 3, pp. 1249–1268, 2017.

[8] E. Bertini, A. Tatu, and D. Keim, "Quality metrics in high-dimensional data visualization: An overview and systematization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2203–2212, 2011.

[9] A. H. Robinson, "The thematic maps of charles joseph minard," 1967.

[10] "Flightradar24 live air traffic." https://www.flightradar24.com/.

[11] S. Ayhan, B. Fruin, F. Yang, and M. O. Ball, "Normstad flight analysis: visualizing air traffic patterns over the united states," in *Proceedings of the 7th ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pp. 1–10, ACM, 2014.

[12] C. Li, X. Liu, Q. Ma, and J. Shen, "Four-dimensional descent trajectory optimization for aircraft in turbulence," in *Information and Automation (ICIA), 2016 IEEE International Conference on*, pp. 755–759, IEEE, 2016.

[13] Z. Wang, M. Lu, X. Yuan, J. Zhang, and H. Van De Wetering, "Visual traffic jam analysis based on trajectory data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2159–2168, 2013.

[14] H. Liu, Y. Gao, L. Lu, S. Liu, H. Qu, and L. M. Ni, "Visual analysis of route diversity," in *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pp. 171–180, IEEE, 2011.

[15] X. Huang, Y. Zhao, C. Ma, J. Yang, X. Ye, and C. Zhang, "Trajgraph: A graph-based visual analytics approach to studying urban network centralities using taxi trajectory data," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 160–169, 2016.

[16] R. Scheepens, N. Willems, H. Van de Wetering, G. Andrienko, N. Andrienko, and J. J. Van Wijk, "Composite density maps for multivariate trajectories," *IEEE Transactions on Visualization & Computer Graphics*, no. 12, pp. 2518–2527, 2011.

[17] N. Adrienko and G. Adrienko, "Spatial generalization and aggregation of massive movement data," *IEEE Transactions on visualization and computer graphics*, vol. 17, no. 2, pp. 205–219, 2011.

[18] J. Zhao, S. M. Drucker, D. Fisher, and D. Brinkman, "Timeslice: Interactive faceted browsing of timeline data," in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 433–436, ACM, 2012.

[19] J. Fulda, M. Brehmel, and T. Munzner, "Timelinecurator: Interactive authoring of visual timelines from unstructured text," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 300–309, 2016.

[20] M. Graham and J. Kennedy, "A survey of multiple tree visualisation," *Information Visualization*, vol. 9, no. 4, pp. 235–252, 2010.

[21] M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts, "Visual comparison for information visualization," *Information Visualization*, vol. 10, no. 4, pp. 289–309, 2011.

[22] Y. Tu and H.-W. Shen, "Visualizing changes of hierarchical data using treemaps," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1286–1293, 2007.

[23] D. Holten and J. J. Van Wijk, "Visual comparison of hierarchically organized data," in *Computer Graphics Forum*, vol. 27, pp. 759–766, Wiley Online Library, 2008.

[24] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[25] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[26] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.

[27] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[28] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[29] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[30] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[31] A. Inselberg, *Parallel coordinates*. Springer, 2009.

[32] A. Pryke, S. Mostaghim, and A. Nazemi, "Heatmap visualization of population based multi objective algorithms," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 361–375, Springer, 2007.

[33] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, and E. Stanley, "Dna visual and analytic data mining," in *Visualization'97., Proceedings*, pp. 437–441, IEEE, 1997.

[34] E. Kandogan, "Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions," in *Proceedings of the IEEE Information Visualization Symposium*, vol. 650, p. 22, 2000.

[35] K. Chen and L. Liu, "ivibrate: Interactive visualization-based framework for clustering large datasets," *ACM Transactions on Information Systems (TOIS)*, vol. 24, no. 2, pp. 245–294, 2006.

[36] J. Sharko, G. Grinstein, and K. A. Marx, "Vectorized radviz and its application to multiple cluster datasets," *IEEE transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008.

[37] G. Albuquerque, M. Eisemann, D. J. Lehmann, H. Theisel, and M. Magnor, "Improving the visual analysis of high-dimensional datasets using quality mea-

sures," in *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, pp. 19–26, IEEE, 2010.

[38] A. Russell, K. Daniels, and G. Grinstein, "Voronoi diagram based dimensional anchor assessment for radial visualizations," in *Information Visualisation (IV), 2012 16th International Conference on*, pp. 229–233, IEEE, 2012.

[39] A. Russell, A. Kamayou, R. Marceau, K. Daniels, and G. Grinstein, "Point sensitivity for radial visualization under dimensional anchor motion," 2014.

[40] A. O. Artero and M. C. F. de Oliveira, "Viz3d: Effective exploratory visualization of large multidimensional data sets," in *Computer Graphics and Image Processing, 2004. Proceedings. 17th Brazilian Symposium on*, pp. 340–347, IEEE, 2004.

[41] L. Nováková and O. Štepanková, "Radviz and identification of clusters in multidimensional data," in *Information Visualisation, 2009 13th International Conference*, pp. 104–109, IEEE, 2009.

[42] A. Ibrahim, S. Rahnamayan, M. V. Martin, and K. Deb, "3d-radvis: visualization of pareto front in many-objective optimization," in *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pp. 736–745, IEEE, 2016.

[43] J. Heer and B. Shneiderman, "Interactive dynamics for visual analysis," *Queue*, vol. 10, no. 2, p. 30, 2012.

[44] J. F. Packer, M. Hasan, and F. F. Samavati, "Illustrative multilevel focus+ context visualization along snaking paths," *The Visual Computer*, vol. 33, no. 10, pp. 1291–1306, 2017.

[45] Y. K. Leung and M. D. Apperley, "A review and taxonomy of distortion-oriented presentation techniques," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 1, no. 2, pp. 126–160, 1994.

[46] N. Kadmon and E. Shlomi, "A polyfocal projection for statistical surfaces," *The Cartographic Journal*, vol. 15, no. 1, pp. 36–41, 1978.

[47] R. Spence and M. Apperley, "Data base navigation: an office environment for the professional," *Behaviour & Information Technology*, vol. 1, no. 1, pp. 43–54, 1982.

[48] J. D. Mackinlay, G. G. Robertson, and S. K. Card, "The perspective wall: Detail and context smoothly integrated," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 173–176, ACM, 1991.

[49] M. Sarkar and M. H. Brown, "Graphical fisheye views of graphs," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 83–91, ACM, 1992.

[50] C. Hughes, M. Glavin, E. Jones, and P. Denny, "Review of geometric distortion compensation in fish-eye cameras," *IET Conference Proceedings*, pp. 162–167(5), January 2008.

[51] E. Kandogan, "Visualizing multi-dimensional clusters, trends, and outliers using star coordinates," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 107–116, ACM, 2001.

[52] D. J. Lehmann and H. Theisel, "Orthographic star coordinates," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2615–2624, 2013.

[53] A. O. Artero, M. C. F. de Oliveira, and H. Levkowitz, "Enhanced high dimensional data visualization through dimension reduction and attribute arrangement," in *Information Visualization, 2006. IV 2006. Tenth International Conference on*, pp. 707–712, IEEE, 2006.

[54] J. S. Shaik and M. Yeasin, "Visualization of high dimensional data using an automated 3d star co-ordinate system," in *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pp. 1339–1346, IEEE, 2006.

[55] T. Van Long and L. Linsen, "Visualizing high density clusters in multidimensional data using optimized star coordinates," *Computational Statistics*, vol. 26, no. 4, p. 655, 2011.

[56] G. G. Zanabria, L. G. Nonato, and E. Gomez-Nieto, "istar (i*): An interactive star coordinates approach for high-dimensional data exploration," *Computers & Graphics*, vol. 60, pp. 107–118, 2016.

[57] M. Rubio-Sánchez and A. Sanchez, "Axis calibration for improving data attribute estimation in star coordinates plots," *IEEE transactions on visualization and computer graphics*, vol. 20, no. 12, pp. 2013–2022, 2014.

[58] Y. Koren and L. Carmel, "Visualization of labeled data using linear transformations.," in *INFOVIS*, Citeseer, 2003.

[59] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207–244, 2009.

[60] S. JacobGoldberger and R. GeoffHinton, "Neighbourhood components analysis," *NIPS04*, 2004.

[61] D. J. Lehmann and H. Theisel, "General projective maps for multidimensional data projection," in *Computer Graphics Forum*, vol. 35, pp. 443–453, Wiley Online Library, 2016.

[62] M. Soldati, M. Doulis, and A. Csillaghy, "Sphereviz-data exploration in a virtual reality environment," in *Information Visualization, 2007. IV'07. 11th International Conference*, pp. 680–683, IEEE, 2007.

[63] H.-J. Schulz, "Treevis.net: A tree visualization reference," *IEEE Computer Graphics and Applications*, no. 6, pp. 11–15, 2011.

[64] Q. V. Nguyen and M. L. Huang, "Enccon: an approach to constructing interactive visualization of large hierarchical data," *Information Visualization*, vol. 4, no. 1, pp. 1–21, 2005.

[65] E. M. Reingold and J. S. Tilford, "Tidier drawings of trees," *IEEE Transactions on Software Engineering*, vol. SE-7, no. 2, pp. 223–228, 1981.

[66] J. Q. W. II, "A node-positioning algorithm for general trees," *Software Practice and Experience*, vol. 20, no. 7, pp. 685–705, 2010.

[67] C. Buchheim, J. Michael, and S. Leipert, "Improving walkers algorithm to run in linear time," in *International Symposium on Graph Drawing*, Springer, 2002.

[68] G. G. Robertson, J. D. Mackinlay, and S. K. Card, "Cone trees: animated 3d visualizations of hierarchical information," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 189–194, ACM, 1991.

[69] T. Munzner, "H3: Laying out large directed graphs in 3d hyperbolic space," in *Information Visualization, 1997. Proceedings., IEEE Symposium on*, pp. 2–10, IEEE, 1997.

[70] M. Jia, L. Li, E. Boggess, E. S. Wurtele, and J. A. Dickerson, "Visualizing multivariate hierarchic data using enhanced radial space-filling layout," in *International Symposium on Visual Computing*, pp. 350–360, Springer, 2010.

[71] J. Lamping, R. Rao, and P. Pirolli, "A focus+ context technique based on hyperbolic geometry for visualizing large hierarchies," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 401–408, ACM Press/Addison-Wesley Publishing Co., 1995.

[72] J. Lamping and R. Rao, "The hyperbolic browser: A focus+ context technique for visualizing large hierarchies," *Journal of Visual Languages & Computing*, vol. 7, no. 1, pp. 33–55, 1996.

[73] J. Stasko and E. Zhang, "Focus+ context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations," in *Information Visualization, 2000. InfoVis 2000. IEEE Symposium on*, pp. 57–65, IEEE, 2000.

[74] H.-C. Lam and I. D. Dinov, "Hyperbolic wheel: A novel hyperbolic space graph viewer for hierarchical information content," *International Scholarly Research Notices*, vol. 2012, 2012.

[75] D. K. Urribarri, S. M. Castro, and S. Martig, "Gyrolayout: A hyperbolic level-of-detail tree layout," *J. UCS*, vol. 19, no. 1, pp. 132–156, 2013.

[76] J. B. Kruskal and J. M. Landwehr, "Icicle plots: Better displays for hierarchical clustering," *The American Statistician*, vol. 37, no. 2, pp. 162–168, 1983.

[77] Y. Chen, X. Zhang, Y. Feng, J. Liang, and H. Chen, "Sunburst with ordered nodes based on hierarchical clustering: a visual analyzing method for associated hierarchical pesticide residue data," *Journal of Visualization*, vol. 18, no. 2, pp. 237–254, 2015.

[78] F. Beck, M. Burch, T. Munz, L. Di Silvestro, and D. Weiskopf, "Generalized pythagoras trees for visualizing hierarchies," in *Information Visualization Theory and Applications (IVAPP), 2014 International Conference on*, pp. 17–28, IEEE, 2014.

[79] H.-J. Schulz, S. Hadlak, and H. Schumann, "The design space of implicit hierarchy visualization: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 4, pp. 393–411, 2011.

[80] B. Shneiderman, "Tree visualization with tree-maps: 2-d space-filling approach," *ACM Transactions on Graphics*, vol. 11, no. 1, pp. 92–99, 1992.

[81] G. Wang, T. Nakanishi, and A. Fukuda, "2-d layout for tree visualization: a survey," in *Proc. MATEC Web of Conferences*, vol. 56, EDP Sciences, 2016.

[82] M. Bruls, K. Huizing, and J. J. Van Wijk, "Squarified treemaps," in *Data Visualization 2000*, pp. 33–42, Springer, 2000.

[83] B. Shneiderman and M. Wattenberg, "Ordered treemap layouts," in *Proc. INFOVIS*, pp. 73–78, IEEE, 2001.

[84] B. B. Bederson, B. Shneiderman, and M. Wattenberg, "Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies," *ACM Transactions on Graphics*, vol. 21, no. 4, pp. 833–854, 2002.

[85] J. Wood and J. Dykes, "Spatially ordered treemaps," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008.

[86] E. K. Burke, G. Kendall, and G. Whitwell, "A new placement heuristic for the orthogonal stock-cutting problem," *Operations Research*, vol. 52, no. 4, pp. 655–671, 2004.

[87] Y. Wang and L. Chen, "Two-dimensional residual-space-maximized packing," *Expert Systems with Applications*, vol. 42, no. 7, pp. 3297–3305, 2015.

[88] T. Itoh, Y. Yamaguchi, Y. Ikehata, and Y. Kajinaga, "Hierarchical data visualization using a fast rectangle-packing algorithm," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 3, pp. 302–313, 2004.

[89] A. Kobayashi, K. Misue, and J. Tanaka, "Edge equalized treemaps," in *Proc. IV*, pp. 7–12, IEEE, 2012.

[90] M. Wattenberg, "A note on space-filling visualizations and space-filling curves," in *Proc. INFOVIS*, pp. 181–186, IEEE, 2005.

[91] S. Tak and A. Cockburn, "Enhanced spatial stability with hilbert and moore treemaps," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 1, pp. 141–148, 2013.

[92] J. Liang, Q. V. Nguyen, S. Simoff, and M. L. Huang, "Angular treemaps-a new technique for visualizing and emphasizing hierarchical structures," in *Proc. IV*, pp. 74–80, IEEE, 2012.

[93] D. Auber, C. Huet, A. Lambert, B. Renoust, A. Sallaberry, and A. Saulnier, "Gospermap: Using a gosper curve for laying out hierarchical data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 11, pp. 1820–1832, 2013.

[94] M. Balzer and O. Deussen, "Voronoi treemaps," in *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, pp. 49–56, IEEE, 2005.

[95] A. Sud, D. Fisher, and H.-P. Lee, "Fast dynamic voronoi treemaps," in *Proc. ISVD*, pp. 85–94, IEEE, 2010.

[96] D. Gotz, "Dynamic voronoi treemaps: A visualization technique for time-varying hierarchical data," *Phys. Rev. A*, vol. 30, no. 2, pp. 150–156, 2011.

[97] A. Nocaj and U. Brandes, "Computing voronoi treemaps: Faster, simpler, and resolution-independent," in *Computer Graphics Forum*, vol. 31, pp. 855–864, Wiley Online Library, 2012.

[98] S. Hahn, J. Trümper, D. Moritz, and J. Döllner, "Visualization of varying hierarchies by stable layout of voronoi treemaps," in *Proc. IVAPP*, pp. 50–58, IEEE, 2014.

[99] F. S. Duarte, F. Sikansi, F. M. Fatore, S. G. Fadel, and F. V. Paulovich, "Nmap: A novel neighborhood preservation space-filling algorithm," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2063–2071, 2014.

[100] W. Wang, H. Wang, G. Dai, and H. Wang, "Visualization of large hierarchical data by circle packing," in *Proc. CHI*, pp. 517–520, ACM, 2006.

[101] J. Görtler, C. Schulz, D. Weiskopf, and O. Deussen, "Bubble treemaps for uncertainty visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 719–728, 2018.

[102] C.-M. Chen, S. Dutta, X. Liu, G. Heinlein, H.-W. Shen, and J.-P. Chen, "Visualization and analysis of rotating stall for transonic jet engine simulation," *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 847–856, 2016.

[103] T. Kobayashi and D. L. Simon, "Integration of on-line and off-line diagnostic algorithms for aircraft engine health management," in *ASME Turbo Expo 2007: Power for Land, Sea, and Air*, pp. 649–659, American Society of Mechanical Engineers, 2007.

[104] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Prognostics and Health Management, 2008. PHM 2008. International Conference on*, pp. 1–9, IEEE, 2008.

[105] C. Zhang, P. Lim, A. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2306–2318, 2017.

[106] Y. C. Wang, F. Lin, and H. S. Seah, "Evaluation of star coordinate boundaries," in *Proc. CGI*, pp. 87–96, ACM, 2018.

[107] Y. C. Wang, Q. Zhang, F. Lin, C. K. Goh, and H. S. Seah, "Polarviz: a discriminating visualization and visual analytics tool for high-dimensional data," *The Visual Computer*, pp. 1–16, 2018.

[108] Y. C. Wang, Q. Zhang, F. Lin, and H. S. Seah, "Engineqv: Investigating external cause of engine failures based on geo-temporal association," in *Proc. IEEE Pacific Visualization Symposium*, IEEE, 2019.

[109] Q. Zhang, C. S. Ahn, J. Liu, Y. C. Wang, F. Lin, and H. S. Seah, "Interactive geo-temporal data visualization for aircraft engine," in *International Workshop on Advanced Image Technology (IWAIT) 2019*, vol. 11049, p. 110491P, International Society for Optics and Photonics, 2019.

[110] J. S. Yi, Y. ah Kang, J. T. Stasko, J. A. Jacko, *et al.*, "Toward a deeper understanding of the role of interaction in information visualization," *IEEE Transactions on Visualization & Computer Graphics*, no. 6, 2007.

[111] "ICAO airport code." https://en.wikipedia.org/wiki/ICAO_airport_code.

[112] "Particulates." https://en.wikipedia.org/w/index.php?title=PM10.

[113] S. Liu, W. Cui, Y. Wu, and M. Liu, "A survey on information visualization: recent advances and challenges," *The Visual Computer*, vol. 30, no. 12, pp. 1373–1393, 2014.

[114] Y.-C. Wang, Q. Zhang, F. Lin, C.-K. Goh, X. Wang, and H.-S. Seah, "Histogram equalization and specification for high-dimensional data visualization using radviz," in *Proceedings of the Computer Graphics International Conference*, p. 15, ACM, 2017.

[115] H. Theisel and M. Kreuseler, "An enhanced spring model for information visualization," in *Computer Graphics Forum*, vol. 17, pp. 335–344, Wiley Online Library, 1998.

[116] P. Hoffman, G. Grinstein, and D. Pinkney, "Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations," in *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, pp. 9–16, ACM, 1999.

[117] E. Bertini, L. Dell'Aquila, and G. Santucci, "Springview: Cooperation of radviz and parallel coordinates for view optimization and clutter reduction," in *Coordinated and Multiple Views in Exploratory Visualization, 2005.(CMV 2005). Proceedings. Third International Conference on*, pp. 22–29, IEEE, 2005.

[118] L. Nováková and O. Štěpánková, "Visualization of trends using radviz," *Journal of Intelligent Information Systems*, vol. 37, no. 3, p. 355, 2011.

[119] J. H. P. Ono, F. Sikansi, D. C. Corrêa, F. V. Paulovich, A. Paiva, and L. G. Nonato, "Concentric radviz: visual exploration of multi-task classification," in *Graphics, Patterns and Images (SIBGRAPI), 2015 28th SIBGRAPI Conference on*, pp. 165–172, IEEE, 2015.

[120] A. Basu and S. Licardie, "Alternative models for fish-eye lenses," *Pattern recognition letters*, vol. 16, no. 4, pp. 433–441, 1995.

[121] A. K. Jain, *Fundamentals of digital image processing*. Prentice-Hall, Inc., 1989.

[122] F. Zhou, W. Huang, J. Li, Y. Huang, Y. Shi, and Y. Zhao, "Extending dimensions in radviz based on mean shift," in *Visualization Symposium (PacificVis), 2015 IEEE Pacific*, pp. 111–115, IEEE, 2015.

[123] NASA, "Enginesim version 1.8a."

[124] M. De Berg, M. Van Kreveld, M. Overmars, and O. C. Schwarzkopf, "Computational geometry," in *Computational geometry*, pp. 1–17, Springer, 2000.

[125] G. T. Toussaint, "Solving geometric problems with the rotating calipers," in *Proc. IEEE Melecon*, vol. 83, p. A10, 1983.

[126] M. Burch, N. Konevtsova, J. Heinrich, M. Hoeferlin, and D. Weiskopf, "Evaluation of traditional, orthogonal, and radial tree diagrams by an eye tracking study," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2440–2448, 2011.

[127] S. Kieffer, T. Dwyer, K. Marriott, and M. Wybrow, "Hola: Human-like orthogonal network layout," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 349–358, 2016.

[128] M. de Berg, B. Speckmann, and V. van der Weele, "Treemaps with bounded aspect ratio," *Computational Geometry*, vol. 47, no. 6, pp. 683–693, 2014.

[129] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, vol. 2, no. 1-4, p. 153, 1987.

[130] Y.-C. Wang, F. Lin, and H.-S. Seah, "Orthogonal voronoi diagram and treemap," *arXiv preprint arXiv:1904.02348*, 2019.

[131] Q. Du, V. Faber, and M. Gunzburger, "Centroidal voronoi tessellations: Applications and algorithms," *SIAM review*, vol. 41, no. 4, pp. 637–676, 1999.

[132] M. Sondag, B. Speckmann, and K. Verbeek, "Stable treemaps via local moves," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 729–738, 2018.

[133] "D3.js." https://d3js.org/.

[134] "Voronoi treemap package in javascript." https://github.com/Kcnarf.

[135] "Mathworks: Evaluating goodness of fit." https://ww2.mathworks.cn/help/curvefit/evaluating-goodness-of-fit.html.