Kernel-based Methods for Measuring Biodiversity and Predicting Quality of Service

Dries GEEBELEN

Jury:
Prof. Dr. ir. Carlo Vandecasteele, chair
Prof. Dr. ir. J. Vandewalle, promotor
Prof. Dr. ir. J.A.K. Suykens, co-promotor
Prof. Dr. L. De Meester
Prof. Dr. ir. B. De Moor
Dr. S. Michiels
Prof. Dr. ir. M. Verleysen
(Université catholique de Louvain)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering

December 2015

© Katholieke Universiteit Leuven – Faculty of Engineering Science
Kasteelpark Arenberg 10 postbus 2440, B-3001 Heverlee (Leuven) (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotocopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm or any other means without written permission from the publisher.

Preface

After graduating as a master in computer sciences, my brother Kristof and me had the plan to make algorithms for financial trading. During the last year of my university education, I took the course neural networks which was taught by Professor Joos Vandewalle and, because he made a good impression on me, I decided to ask him whether he wanted to support our project. He accepted to support me as a PhD student and added, "At the middle of your PhD, you will probably have a tough time, and that is ok. You just have to hang on" and "It is ok to switch subject. I, myself, switched my subject after one year." Those sentences were more relevant to my PhD than I expected at that moment. At the start of my PhD, I wanted to prove that I was PhD worthy by publishing as soon as possible. I submitted my first journal paper in the first year, although it took two more years before it was accepted. Professor Johan Suykens helped me excessively during that time and therefore we decided to make him copromotor. After a year, I switched my subject to traffic data and I was expected to cooperate with Chris Tampère. Chris was always very nice and fair to me, but, unfortunately, this cooperation didn't lead to results. Partially due to the lack of communication and good will of the person in charge of the data and partially because of me. Hereby, I want to give my sincere apologies to Chris for not delivering the results that I promised. In the meantime, Kristof and me wrote a paper about QoS prediction of web services because my specialisation was signal processing and my brother's specialisation was web services. At the start of the third year, I didn't have a clear goal anymore and PhD life was becoming tough. I had several ideas, but none of them felt like something I really wanted to do. Near the end of the same year, I had an idea to improve fixed size least squares support vector machines, one of the algorithms developed by my copromotor Johan and other people within our research group. After excessive experiments, it seemed that my improvement was not that useful because the extra computational burden was too big. Because my time was running out, I desperately searched for other applications of the algorithm and luckily I found one in ecology. I started to update my knowledge about ecology and adapted the paper to that field. One year later, the paper became more than hundred pages long with approximately

fifty pages of the hardest proofs I have ever done. Moreover, I really liked working on the subject. It absorbed me completely and it made me feel useful in the sense that I am convinced that it improves the field of ecology significantly. Luckily, at the end of my PhD, I finally found something I could put my heart into.

First, I would like to thank my promotor Joos Vandewalle and copromotor Johan Suykens for not only supporting me with the scientific part of my PhD but also with the human part. Joos gave me the space and aid to discover what I really wanted to do. Also, he opened a lot of doors for me every time I needed external help. If science had a human shape, then its attitude would be close to that of Johan. I've met very few people as dedicated to science. Also, Johan is always ready to help people by reading their papers or giving advice, whether they are his PhD students or not. Second, I thank professor Luc De Meester for cooperating on the ecology paper. I had the privilege that my paper was corrected by a biology professor which increased the quality of the paper significantly. Furthermore, I want to thank all other members of my jury, Carlo Vandecasteele, Bart De Moor, Sam Michiels and Michel Verleysen for the time and effort spent reading my thesis. Finally, I would like to thank my family, colleagues and friends. I will not try to name all, but I will limit myself to mentioning the people closest to me. I thank my brother Kristof for providing me a house and for being both a fun and responsible big bother during my PhD. I would also like to thank my parents Jaak and Suzanne for showing interest in my PhD and for supporting me. I thank my colleagues Kim and Philippe for being more than just colleagues and for inventing star configurations. I thank Geert and Tom for the experimental food on Wednesdays and the good times during the several snowboard trips. Lastly, I would like to thank Gervasio for the friendship and for helping me out when I really needed it.

Dries Geebelen
Leuven, September 2015

Abstract

Kernel methods are established methods in machine learning algorithms that can be used for diverse purposes including regression, classification, clustering, principal components analysis, and canonical-correlation analysis. They owe their name to the use of kernel functions. The kernel trick allows kernel methods to operate in a feature space without computing the coordinates of the data in that space. This causes the computational complexity of the algorithm to become independent from the number of feature dimensions. In this dissertation we propose kernel methods that (1) quantify diversity, richness and derived properties in a similarity-based framework and (2) predict the Quality of Service (QoS) of workflow compositions. From an application point of view, these methods are not related.

Our first application is quantifying biodiversity in a similarity-based framework. There are two categories of diversity indices: those that depend on the species abundances and those that don't. The latter are referred to as richness indices. Several commonly-used indices in ecology assume that all species are completely different which is in contrast to reality, where some species clearly differ more than others. Therefore recently many diversity, richness and derived measures that incorporate similarity have been proposed. However, as we will show in this dissertation, most, if not all, of the existing similarity-based indices have suboptimal properties. For example, some richness indices change when we add a species identical to a species already present in the community. We propose indices that overcome these issues. Furthermore, the proposed indices have a clear meaning in a well-defined framework. For example, the proposed diversity, called the Optimal Lossy Compression (OLC) diversity, express the trade-off between the approximation error and the description length when compressing the representation of the species optimally.

In the second application we predict whether a composition of services is likely to comply with the service level agreements (SLAs). If the functional behavior of services within a service-based system are equivalent, then non-functional properties such as Quality of Service (QoS) attributes become the main criteria to determine which service is selected to execute a certain task.

Examples of QoS attributes are security, privacy, response time, throughput, reputation, reliability, availability and cost. We make the following three main contributions with respect to the prediction of these QoS values. First, we propose a simulation technique based on Petri nets that generates composite time series using monitored QoS data of its elementary services. This techniques implicitly takes into account time related information and mutual dependencies between participating services. Second, we propose a kernel-based quantile estimator with online adaptation of the constant offset to predict future QoS values. The online adaption guarantees that, under certain weak conditions, the number of times the predicted value is worse than the actual value converges to the quantile value specified in the SLA. Third, we introduce two performance indicators for comparing different QoS prediction algorithms. Our validation in the context of two case studies shows that the proposed algorithms outperform existing approaches by drastically reducing the violation frequency of the SLA while maximizing the usage of the candidate services.

Finally we propose a method to reduce the computational complexity of kernel-based classifiers during execution. This algorithm consist of three steps. First we apply the classifier. Second we make the classification problem separable in the feature space by changing or removing the labels of misclassified training data points. Finally we apply a hard-margin SVM which is typically much sparser than a soft-margin SVM. We also propose a smoothed version of this algorithm which ensures numerical stability and reduces the number of support vectors even more.

Contents

Abstract iii

Contents v

Abbreviations xiii

1 Introduction 1

1.1 Motivation and approach 1

1.2 Kernel methods 4

1.2.1 Which algorithms can be made kernel-based? 4

1.2.2 LS-SVM and kernel-based ridge regression 5

1.2.3 Other kernel-based regression algorithms 10

1.3 Quantifying biodiversity 12

1.3.1 Why is biodiversity important? 12

1.3.2 How is biodiversity measured? 13

1.3.3 Which indices do we propose? 14

1.3.4 How to define indices in our framework? 14

1.3.5 OLC diversity, similarity and uniqueness 16

1.3.6 OLC richness and derived indices 17

1.4 QoS of workflow compositions 18

1.4.1 Research challenges 18

1.4.2 Problems of existing methods 19
1.4.3 Contributions in this thesis 19
1.5 Reducing the run time complexity 20
1.5.1 Computational complexity of SVMs 20
1.5.2 Reducing the run time complexity 20
1.6 Overview 21
References 21

2 Similarity-based Diversity, Richness and Derived Indices 25
2.1 Introduction 26
2.2 Related work 31
2.2.1 Rao's index 32
2.2.2 Lineage-based phylogenetic diversity 32
2.2.3 Average similarity diversity 34
2.2.4 Diversity with respect to eigenvalues of matrices 34
2.2.5 Distance-based functional diversity 35
2.2.6 Faith's total branch length 36
2.2.7 Originality of species within a set 36
2.2.8 Equal splits and fair proportion 37
2.2.9 Phylogenetic diversity based on Brownian motion 37
2.2.10 Evenness based on terminal branch lengths 38
2.3 Similarity-based framework 38
2.3.1 Restricted framework 39
2.3.2 General framework 39
2.3.3 Notations 42
2.4 Optimal lossy compression and entropy 43
2.4.1 Compression algorithm 45
2.4.2 OLC entropy 46

2.5 OLC diversity 49
2.5.1 Definition of the OLC diversity 49
2.5.2 Properties of the OLC diversity 50
2.6 OLC richness 51
2.6.1 Definition of the OLC richness 52
2.6.2 Properties of the OLC richness 53
2.6.3 Mutual richness 55
2.7 OLC Contribution vector 56
2.7.1 Definition of the OLC contribution vector 56
2.7.2 Properties of the OLC contribution vector 57
2.8 OLC average contribution and OLC evenness 58
2.9 Decomposition in alpha and beta diversity 60
2.10 Comparison with related work on simulated data 63
2.10.1 Used similarity 63
2.10.2 Does skewness matter? 63
2.10.3 Comparison of phylogenetic diversity indices 67
2.10.4 Comparison of phylogenetic richness and contribution indices 68
2.10.5 Comparison of phylogenetic evenness measures 70
2.10.6 Problems of distance-based functional diversity measures 73
2.11 Applications to real-world data 73
2.11.1 Butterfly population 73
2.11.2 Prosimian primates 75
2.11.3 Tree thinning 76
2.12 Conclusion and Future Work 76
2.13 Acknowledgements 79
References 79

Appendices **84**

2.A Running example 84
2.B Comments on other indices 91
2.B.1 Strong replication principle and the modularity property . 91
2.B.2 Chiu and Chao's distance-based replication principle . . . 91
2.C Extending an index to the general framework 92
2.C.1 When is an index meaningful in the general framework? 93
2.C.2 Expressing an index as a function of the similarity matrix 94
2.C.3 Using a similarity function that is not meaningful 94
2.D OLC uniqueness vector 95
2.E Our notation applied to alpha and beta 96
2.F Approximation error, explanatory power and entropy 97
2.F.1 Minimum approximation error property 97
2.F.2 Explanatory power 99
2.F.3 Derivation of the OLC entropy 101
2.G Reformulating the quadratic and min OLC diversity 102
2.H Properties of the OLC diversity 103
2.H.1 Modularity 103
2.H.2 Strong replication principle 104
2.H.3 Twin property 105
2.H.4 Upperbound 105
2.I Richness optimization problem 107
2.I.1 Quasi-concavity of the OLC diversity 108
2.J Properties of the richness 110
2.J.1 Monotonically increasing 112
2.J.2 Non-changing 112
2.J.3 Strictly monotonically increasing 113
2.J.4 Additivity 119
2.J.5 Subadditivity 120

2.J.6 Strict subadditivity 122
2.J.7 Interpretability 124
2.K Properties of the contribution vector 124
2.K.1 Partitioning 124
2.K.2 Consistent with richness of subsets 124
2.K.3 Consistent with richness of orthogonal subsets 127
2.K.4 Twin property 128
2.L Alpha, beta and gamma 128
2.M Phylogenetic positive semidefinite kernel 131
2.N Nucleotide probabilities in phylogenetic trees 132
2.N.1 Three equally distant species 132
2.N.2 Skewed and unskewed tree 134

3 Quality of Service Prediction of Workflow Compositions 137
3.1 Introduction 138
3.2 Motivation 143
3.2.1 Running Example 143
3.2.2 Quantile vs. Average SLO Compliance Estimation . . . 144
3.2.3 Time Series Prediction 144
3.2.4 Mutual Dependencies 145
3.3 Related Work 145
3.4 QoS Considerations 149
3.4.1 QoS for Elementary Service 149
3.4.2 QoS for Composite Services 151
3.5 QoS Aggregation 152
3.5.1 Petri Net Graph 152
3.5.2 Petri Net Execution Semantics 154
3.5.3 Petri Net Execution Time System 158

3.5.4 From Non-Deterministic to Deterministic 160
3.5.5 Example Simulated QoS Calculation 161
3.5.6 Discussion 165
3.6 QoS Prediction 166
3.6.1 Performance Indicators 166
3.6.2 Kernel-Based Quantile Estimation 171
3.6.3 Online Adaptation of an Unconditional Quantile Estimator 173
3.6.4 Kernel-Based Quantile Estimation with Online Adaptation of the Constant Offset 174
3.7 Experimental Evaluation 175
3.7.1 Prediction For Individual Services 175
3.7.2 Prediction For Composite Services 178
3.8 Conclusion 183
3.9 Future Work 184
3.10 Acknowledgements 184
References 185

Appendices **188**

3.A Reformulation of the optimization problem 188
3.B Obtaining data of real web services 189

4 Reducing the Run Time Complexity of SVM Classifiers **191**

4.1 Introduction 192
4.2 Support Vector Machine Classifier 193
4.3 SCA 194
4.3.1 Approximating the decision Boundary 194
4.3.2 SCA 195
4.3.3 SSCA 197
4.3.4 NSCA 199

4.4 Other pruning methods 200
4.4.1 Cross-Training 200
4.4.2 Greedy iterative methods 200
4.4.3 FS LS-SVM 200
4.5 Experimental Results 201
4.6 Conclusion 201
4.7 Acknowledgement 205
References 205

5 Conclusion **207**
5.1 Summary 207
5.2 Limitations and future work 208
5.2.1 Measuring biodiversity 208
5.2.2 Predicting QoS 212
5.2.3 Reducing the run time complexity of SVM Classifiers . 212
References 213

List of publications **215**

Abbreviations

ASD	Average Similarity Diversity
AVaR	Average Value-at-Risk
BBMC	Bootstrap Based Monte Carlo
BPEL	Business Process Execution Language
BPMN	Business Process Modeling Notation
CDF	Cumulative Distribution Function
ES	Equal Splits
FAD	Functional Attribute Diversity
FD	Functional Diversity
FS LS-SVM	Fixed Size Least Squares Support Vector Machines
FP	Fair Proportion
FTBL	Faith's Total Branch Length
IBBT	Interdisciplinary Institute for Broadband Technology
ICT	Information and Communications Technology
ISNR	Improved Silverman's Normal Rule of thumb
IWT	Agentschap voor Innovatie door Wetenschap en Technologie
KL	Karhunen-Loève
KQ	Kernel-based Quantile estimator
KQOA	Kernel-based Quantile estimator with Online Adaptiation of the constant offset
LS-SVM	Least Squares Support Vector Machines
MEA	Millennium Ecosystem Assessment
MVC	Model-View-Controller
OASIS	Organization for the Advancement of Structured Information Standards
OLC	Optimal Lossy Compression
OLS	Ordinary Least Squares
OMG	Object Management Group
OSS	Originality of a Species within a Set
PAE	Phylogenetic Abundance-Evenness
PCA	Principal Component Analysis

PNET	Petri Net Execution Time system
PNG	Petri Net Graph
Pol	Polynomial
PSE	Phylogenetic Species Evenness
PSR	Phylogenetic Species Richness
QoS	Quality of Service
QP	Quadratic Programming
RBF	Radial Basis Function
RoR	Ruby On Rails
SaaS	Software as a Service
SCA	Separable Case Approximation
SLA	Service Level Agreement
SLO	Service Level Objective
SOAP	Simple Object Access Protocol
SOC	Service-Oriented Computing
SSCA	Smoothed Separable Case Approximation
SVM	Support Vector Machine
SW	Sliding Window estimator
SWQ	Sliding Window Quantile estimator
WS-BPEL	Web Services Business Process Execution Language

Chapter 1

Introduction

Kernel methods are algorithms that use the kernel trick which allows them to operate in huge or even infinite dimensional feature space without explicitly computing the coordinates of the data in that space. Instead, the inner products between the feature vectors, which can be calculated using the kernel function, are calculated. The kernel trick can thus reduce the computational complexity considerably. All algorithms presented in this dissertation are kernel methods that have in common that they use this trick. This dissertation can be divided into three parts. In the first part we measure biodiversity and related properties. In the second part we predict Quality of Service (QoS) attributes of workflow compositions. Finally, we propose a method to reduce the computational complexity of kernel-based classifiers at run time. The kernel function has the following meaning in these applications. In the biodiversity application, the kernel function is used to express the similarity between types (e.g., species). The kernel trick is used here to extend a limited framework - in which only one similarity function can be used - to a more broad framework - in which any meaningful similarity function can be used. In the two other applications, the kernel function represents the inner product of the feature vectors and the kernel trick is used to transform linear regression into non-linear regression.

1.1 Motivation and approach

The diversity of a group of individuals reflects how different the individuals are. Diversity can be quantified in different ways; whether or not one population is more diverse than another often depends on the used definition. For example, if population A consists of three horses and three donkeys and population B consists of five horses and one frog, then it is not clear which population is

the most divers. The abundances (i.e., how often each species occurs) are the most balanced in population A, while the two species are the most different in population B (i.e., a frog and a horse differ more than a donkey and a horse). In some cases, however, it is clear which population is the most diverse. For instance, if population C only contains horses and population D contains both horses and cows, then it is clear that population D is more diverse than population C. We can unite all the scenarios for which it is clear how the index should behave into a set of properties. If an index does not meet these properties, then this means that it results in wrong conclusions for some of these scenarios. An important problem of most existing similarity-based indices is that they do not meet the properties that they are expected to satisfy. For example, some existing richness indices can decrease when species are added. The main contribution of our similarity-based framework is that our indices do satisfy these properties. Another issue of competing similarity-based indices is that it is not always clear what similarity means in their framework. This makes it hard to choose the similarity function in practice. For example, if we have a set of binary features, then the similarity can be defined as the proportion of shared features as well as the square or the square root of this proportion. If it is not clear what similarity exactly means, then the user cannot know which similarity measure he should use. In our framework we relate diversity to optimal compression. We assign a representation vector to each type and the smaller the size of the compressed representation of a population, the less diverse it is. The similarity between two types is now defined as the inner product of their representation vectors. In other words, if a user knows which representation vectors he wants to compress, then the similarity follows from this choice. A third problem of existing work is that a lot of non-trivial assumptions are implicitly made. This creates a false sense of confidence because these assumptions are often hidden from the user (e.g., the algorithm implicitly chooses the similarity function on behalf of the user). Our framework overcomes this issue by making most non-trivial choices explicit. Finally, our indices have the advantage that they are linked in one framework. For example, it is clear how diversity, richness and uniqueness relate to each other. Moreover, the indices do not have conflicting values. For instance, if we add a species that has a non-zero uniqueness, then the richness will always strictly increase.

A workflow composition is a composite service that consist of a number of individual services that are executed in series, parallel or according to other composition patterns. In this dissertation we propose a method to predict future QoS attributes - such as response time and availability - of workflow compositions based on historical QoS values of the individual services. Our algorithm consists of two parts: aggregation and prediction. In the aggregation part, we aggregate information about the individual services into information about the composition. In the prediction part, we predict future values based on historical values. In practice the service provider and the customer typically want to know how likely it is that the contract will be met and therefore they

are interested in quantile values rather than mean values. For example, they want to be sure that the response of the composition will be below 20 seconds in 99% of the cases. Some of the existing work treats QoS values as fixed which does not correspond to reality where QoS values - such as response time - often change considerably over time. Others algorithms have the disadvantage that they predict mean values instead of quantile values. There exist algorithms that - similar to our approach - predict quantile values. The main difference between our algorithm and these methods is that we do aggregation before prediction instead of vice versa. More specifically, we aggregate the historical values of individual service into historical values of the composition and we then perform time series prediction on these aggregated values. In related work they first predict future quantile values of the individuals service and then aggregate these predictions. The latter approach has the disadvantage that the quantile values of the individual services are not sufficient to derive the quantile values of the entire composition. Therefore some algorithms calculate upperbounds of the quantile values instead. Other methods solve this issue by predicting entire probability distributions for the individual services. Predicting probability distributions is, however, much harder than predicting point estimates such as means or quantile values. The main disadvantage of our method compared to these methods is that when we change the composition slightly, the prediction algorithm has to be retrained completely. We cannot reuse intermediate results. Algorithms that do predictions first can reuse the predictions of the individual services that remain in the composition.

Finally, we propose an algorithm that reduces the run-time complexity of kernel-based classifiers. This reduction is achieved by first modifying the training set such that it becomes separable in the feature space and then applying the separable case Support Vector Machine (SVM). The main advantage of our algorithm is that it is easy to understand and implement. The trade-off between the accuracy and the reduction in run-time complexity is comparable to that of other state-of-the-art algorithms. A disadvantage of our algorithm is that it doesn't reduce the training complexity. For most applications reducing the training complexity is more important than reducing the run-time complexity. However, there exist applications where the run-time complexity is more important. For example, in handwritten digit recognition as used by optical machines to automatically read postcodes of letters. The lower the run-time complexity of the algorithm, the faster a letter can be sorted after reading its postcode, and the higher the throughput of the sorting machine.

1.2 Kernel methods

1.2.1 Which algorithms can be made kernel-based?

Here we give a very brief introduction to kernel based methods. For a more extensive summary, we refer the reader to [1]. Support Vector Machines [2, 3] (SVM) is the first method in the literature that uses the kernel trick. Typically SVM and related kernel-based optimization problems, such as Least Squares SVM (LS-SVM) [4], are solved in the dual space (i.e., first the dual problem is created and then this dual problem is solved). We want to note here that in some occasions they are also solved in the primal space (e.g., when the number of datapoints is much greater greater than the number of feature dimensions). Although this primal-dual technique is very useful, it hides in a sense why and how a method can be made kernel-based. Therefore we opt for a different strategy to explain kernel-based methods in this introduction. This strategy is partially inspired by the Generalized Representer Theorem [5] and its proof. The first step in making an algorithm kernel-based, is writing the algorithm as a function of the pairwise inner products of the input vectors, i.e., the input vectors can only occur in the algorithm as arguments of such inner products. An algorithm is uniquely defined by these inner products if and only if all sets of input vectors with the same pairwise inner products result in the same outcome. Suppose $\mathcal{X} = \{x_i \in \mathbb{R}^N\}_{i=1}^n$ and $\mathcal{Z} = \{z_i \in \mathbb{R}^{N'}\}_{i=1}^n$ are two sets of input vectors where $N \geq N'$ and $x_i^\intercal x_j = z_i^\intercal z_j$ for $i, j \in \{1, \ldots, n\}$. We can now conclude that

$$x_i^\intercal x_j = z_i^\intercal z_j, \;\; i, j \in \{1, \ldots, n\} \tag{1.1}$$

$$\Rightarrow X^\intercal X = Z^\intercal Z \tag{1.2}$$

$$\Rightarrow \Lambda = \Sigma_X^\intercal \Sigma_X = \Sigma_Z^\intercal \Sigma_z \tag{1.3}$$

$$\Rightarrow \Sigma_X' = \Sigma_Z \tag{1.4}$$

$$\Rightarrow \Sigma_X' V^\intercal = \Sigma_Z V^\intercal \tag{1.5}$$

$$\Rightarrow U_X'^\intercal X = U_Z^\intercal Z \tag{1.6}$$

$$\Rightarrow Z = U_{XZ}^\intercal X \tag{1.7}$$

$$\Rightarrow z_i = U_{XZ}^\intercal x_i, \;\; i, j \in \{1, \ldots, n\}, \tag{1.8}$$

where

- $X \in \mathbb{R}^{N \times n}$ and $Z \in \mathbb{R}^{N' \times n}$ are matrices whose columns equal the input vectors of the sets $\mathcal{X} = \{x_i \in \mathbb{R}^N\}_{i=1}^n$ and $\mathcal{Z} = \{z_i \in \mathbb{R}^{N'}\}_{i=1}^n$, respectively.

- the eigenvalue decomposition of $A = X^\intercal X = Z^\intercal Z$ equals $A = V \Lambda V^\intercal$ with $V, \Lambda \in \mathbb{R}^{n \times n}$.
- the singular value decomposition of X equals $X = U_X \Sigma_X V^\intercal$ with $U_X \in \mathbb{R}^{N \times N}$ and $\Sigma_X \in \mathbb{R}^{N \times n}$.
- the singular value decomposition of Z equals $Z = U_Z \Sigma_Z V^\intercal$ with $U_Z \in \mathbb{R}^{N' \times N'}$ and $\Sigma_Z \in \mathbb{R}^{N' \times n}$.
- $U'_X \in \mathbb{R}^{N \times N'}$ contains the N' leftmost columns of U_X.
- $\Sigma'_X \in \mathbb{R}^{N' \times n}$ contains the N' topmost rows of Σ_X.
- $U_{XZ} \in \mathbb{R}^{N \times N'}$ is defined as $U_{XZ} = U'_X U_Z^\intercal$. Note that U_{XZ} has orthonormal columns.

In words this means that if the pairwise inner products are the same in both sets, then the input vectors in one of the sets represent the input vectors in the other set with respect to a different orthonormal basis. The reverse also holds: changing the orthonormal basis preserves the pairwise inner products. We can now conclude that an algorithm can be made kernel-based if and only if changing the basis of the input vectors to another orthonormal basis does not change the outcome of the algorithm. Lasso (a linear regression algorithm that implicitly does input selection) and sparse PCA (an extension of PCA that additionally makes the principal components sparse in the input variables), for example, use the Manhattan norm in the input space. This norm changes when we change the coordinate system which cause the outcome of these algorithms to change as well. Therefore they cannot be made kernel-based. Sometimes the outcome of an algorithm consists of a vector that belongs to the input space (e.g., the weight vector of the regression model in ridge regression) and in most cases this vector changes when we change the basis in the sense that it also undergoes this basis transformation. We have to make the representation of this vector independent of the used basis (e.g., by expressing it as a linear combination of the input vectors) before we can make the algorithm kernel-based. Ordinary Least Squares (OLS), ridge regression and Principal Component Analysis (PCA) are three algorithms that can be made kernel-based in this way.

1.2.2 LS-SVM and kernel-based ridge regression

In ridge regression with additional bias term we solve the following optimization problem

$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \sum_{i=1}^{n} \left(y_i - x_i^\intercal w - b\right)^2 + C\|w\|_2^2, \tag{1.9}$$

where $\{x_i \in \mathbb{R}^p\}_{i=1}^n$ contains the input vectors, $\{y_i \in \mathbb{R}\}_{i=1}^n$ contains the corresponding labels, $\|w\|_2^2$ is the regularization term, $C \in \mathbb{R}_{\geq 0}$ is the regularization constant, and $\hat{w} \in \mathbb{R}^p$ and $\hat{b} \in \mathbb{R}$ are the linear coefficients and constant offset that solve the optimization problem, respectively. The predicted label of an unseen datapoint equals $\hat{y}(x) = x^\intercal \hat{w} + \hat{b}$. If $C = 0$, then the regularization term disappears and ridge regression becomes OLS. If we increase C, then both the smoothness of the learned function and the training error will increase. In other words, the larger C, the more smoothness is favored over a small training error. The importance of smoothness will be illustrated later in this section. The weight vector $\hat{w}$ changes when we change the basis of the input vectors and therefore the optimization problem cannot be made kernel-based in this form. We will show now that $\hat{w}$ is a linear combination of the input vectors ($\hat{w} = \sum_{j=1}^n \hat{\alpha}_j x_j$) and these linear coefficients do not change when we change the basis. We decompose the weight vector w as follows $w = w_p + w_\perp$, where w_p is the part of w that is projected onto the space spanned by the input vectors $\{x_i\}_{i=1}^n$, and $w_\perp$ is the part of w that is orthogonal to this space. The optimization problem becomes

$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \sum_{i=1}^n \left(y_i - x_i^\intercal w_p - b\right)^2 + C\left(\|w_p\|_2^2 + \|w_\perp\|_2^2\right). \tag{1.10}$$

Note that $x_i^\intercal w_\perp = 0$ for $i \in \{1, \ldots, n\}$ because $w_\perp$ is orthogonal to the input vectors. This means that the orthogonal component $w_\perp$ does not influence the training errors and therefore it must be the zero vector in order to minimize the regularization term. We thus derived that $\hat{w}$ is a linear combination of the input vectors. The optimization problem now equals

$$\min_{\alpha \in \mathbb{R}^n, b \in \mathbb{R}} \sum_{i=1}^n \left(y_i - \left(\sum_{j=1}^n \alpha_j x_i^\intercal x_j\right) - b\right)^2 + C\left(\sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j x_i^\intercal x_j\right), \tag{1.11}$$

where the predicted label becomes

$$\hat{y}(x) = \left(\sum_{j=1}^n \hat{\alpha}_j x^\intercal x_j\right) + \hat{b}. \tag{1.12}$$

We have formulated the ridge regression optimization problem as a function of inner products. In Figure 1.1 we apply OLS ($C = 0$) to toy data. Note that the true function is not linear and therefore we are unable to learn the true function using OLS or ridge regression.

In a second step we replace the inner product by another function. However, in order to make sure that we do not loose important properties (e.g., convexity in the case of ridge regression) this other function must be a positive semidefinite kernel $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ where the two arguments of the kernel are the two

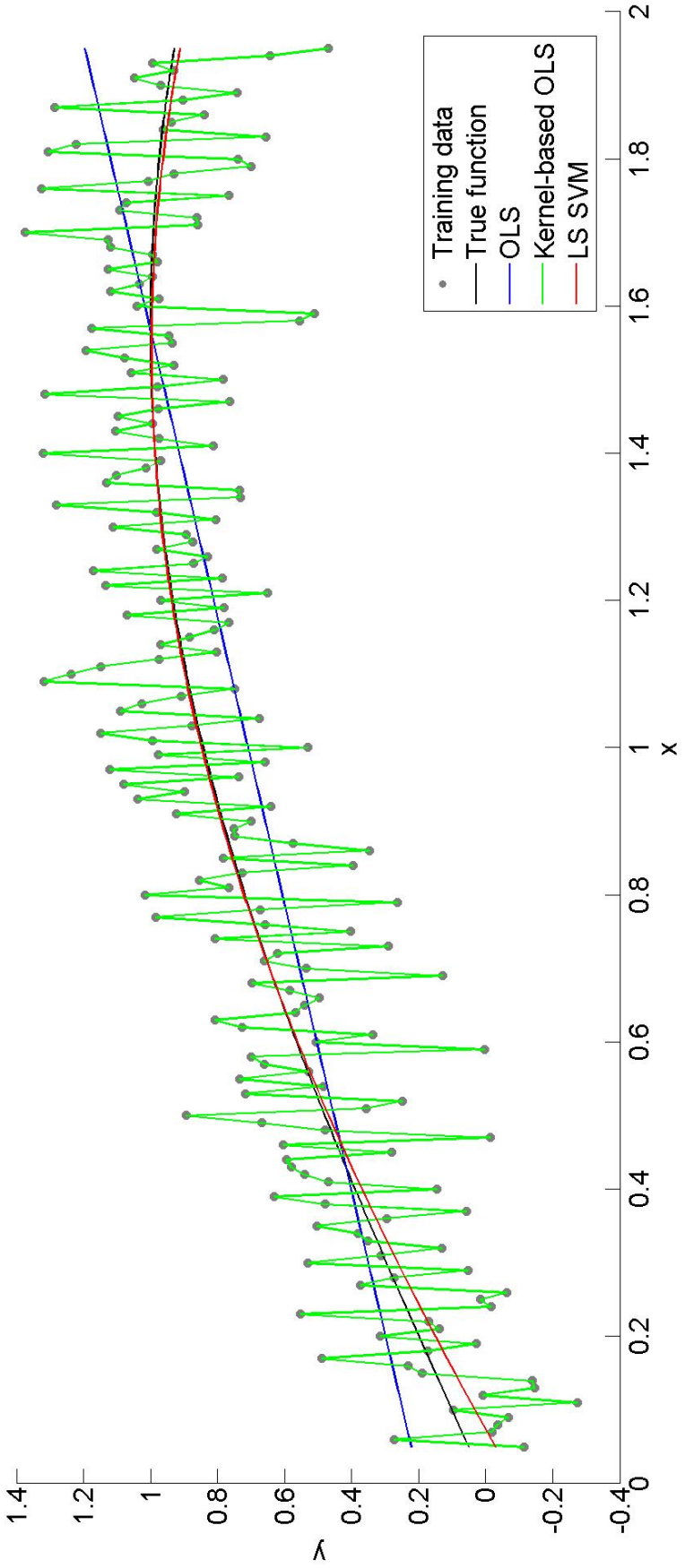


Figure 1.1: Visualisation of training data that contains additive Gaussian noise ($y_i = \sin(x_i) + n_i$, where n_i is Gaussian noise with mean 0 and variance 0.04), the true function ($y = \sin(x)$), the linear function learned by OLS ($\hat{y} = x\hat{w} + \hat{b}$, where $\hat{w} = 0.513$ and $\hat{b} = 0.196$), the non-smooth non-linear function learned by kernel-based OLS, and the smooth non-linear function learned by LS-SVM. In the case of kernel-based methods we use an RBF kernel. In the case of LS-SVM the hyperparameters of the algorithm (i.e., the bandwidth of the RBF kernel and the weight of the complexity term) are optimized using 10-fold crossvalidation.

input vectors. A positive semidefinite kernel has the following properties: it is symmetric in its arguments ($k(x, z) = k(z, x)$) and, as proven by Mercer [6], there exists a feature mapping $\varphi : \mathbb{R}^p \to \mathbb{R}^q$ such that the kernel can be expressed as

$$k(x, z) = \varphi(x)^T \varphi(z). \tag{1.13}$$

We call $\varphi(x)$ the feature vector corresponding to input vector x. The last property thus means that there exists a feature mapping such that the kernel becomes the inner product of the corresponding feature vectors. Replacing the inner product by a positive semidefinite kernel is therefore equivalent to first mapping the input vectors onto the corresponding feature vectors and then applying the original algorithm to these feature vectors. Because, in a sense, we only change the inputs of the algorithm and not the algorithm itself, the properties of the algorithm (e.g., convexity) still hold. A popular kernel function is the Radial Basis Function (RBF) kernel ($k(x_i, x_j) = \exp\left(-\|x_i - x_j\|_2^2/\sigma^2\right)$). This kernel reflects the distance between two vectors. It equals one when this distance equals zero and it converges to zero when the distance approaches infinity. The ridge regression optimization problem now transforms into

$$\min_{\alpha \in \mathbb{R}^n, b \in \mathbb{R}} \sum_{i=1}^{n} \left(y_i - \left(\sum_{j=1}^{n} \alpha_j k(x_i, x_j) \right) - b \right)^2 + C \left(\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \right). \tag{1.14}$$

and the corresponding regression model becomes

$$\hat{y}(x) = \left(\sum_{j=1}^{n} \hat{\alpha}_j k(x, x_j) \right) + \hat{b}. \tag{1.15}$$

The benefits of replacing the inner products by kernels are to some extent algorithm-specific. In the case of ridge regression, the kernel trick transforms a linear model into a non-linear blackbox predictor. Certain kernels (e.g., the RBF kernel) make this predictor non-parametric which means that we no longer need to make assumptions with respect to the structure of the model. When using such kernels any continuous function - and thus not only linear functions - can be represented as long as the function satisfies certain mild conditions (e.g., it is sufficiently smooth). By applying the kernel trick we thus transform a linear approximator into a universal approximator. An important consequence of this transformation is that due to the richness of the hypothesis space (i.e., almost any function can be learned), we need choose C strictly greater than zero in order to avoid overfitting. A smooth function that causes a small training error often generalizes better to unseen data than a non-smooth function that causes no training error. In Figure 1.1 we can see that the function learned by the kernel-based OLS algorithm goes through all the datapoints. However, because

of its non-smooth behavior it differs a lot from the true function and therefore it will generalize poorly to unseen data. Kernel-based ridge regression, on the other hand, is much closer to the true function because it makes a trade-off between the smoothness and the training error.

We would like to explain why it is important that the feature mapping is not explicitly used in the formulation of the algorithm. The feature mapping corresponding to certain kernels can be huge or infinite dimensional. The simple kernel $k(x, z) = (x^\intercal z)^2$, for example, results in the following feature mapping

$$\varphi(x) = \text{vec}\left(xx^\intercal\right), \tag{1.16}$$

where $\text{vec}(A)$ stack the columns of matrix A on top of each other. The dimension of the feature space thus becomes the square of the dimension of the input space which means that an input dimension of 100 corresponds to a feature dimension of 10000. In the case of the RBF kernel it is even worse; it has been shown the corresponding feature space is infinite-dimensional. Although the kernel functions are fairly cheap to evaluate, the corresponding feature mappings often aren't which is the reason why we only want that the kernel function - and thus the inner product of the feature vectors - occurs in the algorithm.

Solving the kernel-based optimization problem corresponds to solving a linear system as we explain now. We first rewrite the optimization problem in matrix notation

$$\min_{\alpha\in\mathbb{R}^n, b\in\mathbb{R}} \left\| Y - \begin{bmatrix} 1_v & \Omega \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} \right\|_2^2 + C \begin{bmatrix} b \\ \alpha \end{bmatrix}^T \begin{bmatrix} 0 & 0_h \\ 0_v & \Omega \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix}, \tag{1.17}$$

where $\Omega \in \mathbb{R}^{p\times p}$ is the kernel matrix with $\Omega_{ij} = k(x_i, x_j)$, $1_v \in \mathbb{R}^{p\times 1}$ is a column vector containing only ones, $0_h \in \mathbb{R}^{1\times p}$ is a row vector containing only zeros, and $0_v \in \mathbb{R}^{p\times 1}$ is a column vector containing only zeros. To solve this optimization problem, we set the derivative equal to zero, which results in the following linear system

$$\begin{bmatrix} 1_h 1_v & 1_h\Omega \\ \Omega 1_v & \Omega^2 + C\Omega \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 1_h Y \\ \Omega Y \end{bmatrix}. \tag{1.18}$$

Note that the equation in the bottom block matrices equals

$$\Omega \left(\begin{bmatrix} 1_v & \Omega + CI \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} - Y \right) = 0, \tag{1.19}$$

where $I \in \mathbb{R}^{n\times n}$ is the identity matrix. In other words the factor at the righthand side of Ω has to belong to the null space of Ω. It can be shown that there always exists a solution for which the righthand term equals the zero vector when $C > 0$. Therefore we can solve the following linear system instead

$$\begin{bmatrix} 1_h 1_v & 1_h\Omega \\ 1_v & \Omega + CI \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 1_h Y \\ Y \end{bmatrix}. \tag{1.20}$$

The new linear system has only one solution when $C > 0$ while the original system can have multiple solutions. In the case that we look for a particular solution (e.g., Downs et al. [7] select the most sparse solution), then we are often interested in all solutions such that we can select the best one. A higher sparsity is an advantage because it corresponds to a lower computational complexity. However, in practice Ω is often near full rank and not much sparsity can be gained this way. If we subtract the top block matrices by 1_h times the bottom block matrices, we get

$$\begin{bmatrix} 0 & -C1_h \\ 1_v & \Omega + CI \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix}. \tag{1.21}$$

Finally, we divide the top block matrices by $-C$

$$\begin{bmatrix} 0 & 1_h \\ 1_v & \Omega + CI \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix}. \tag{1.22}$$

The resulting linear system is the same one that can be derived from the Lagrange dual formulation in [4] after stating (1.9) as a constrained optimization problem

$$\min_{w,b,e} \sum_{i=1}^{n} e_i^2 + Cw^T w \tag{1.23}$$

$$\text{s.t. } e_i = y_i - w^\intercal \varphi(x_i) - b, \ i = 1, \ldots, n, \tag{1.24}$$

which is known as LS-SVM regression. In this case $w = \sum_{i=1}^{n} \alpha_i \varphi(x_i)$ follows from the conditions for optimality.

1.2.3 Other kernel-based regression algorithms

We are often interested in other loss functions (e.g., we want a quantile estimator instead of a mean estimator) and other smoothness functions (e.g., we want the 1-norm to be small instead of the 2-norm such that the solution is sparse in the parameters). The question is which changes we can make such that the algorithm can still be made kernel-based. As explained at the start of this section, an algorithm can be made kernel based if the result of the algorithm is independent of the orthonormal basis in which the input vector are represented. Under the assumption that w undergoes the same basis change as the input vectors, we can conclude that:

- In the regularization term we cannot use other norms of w (e.g., the 1-norm) because this causes the optimal value of the optimization problem and the outcome of the algorithm to change when we change the basis. Only the 2-norm remains unchanged when changing the basis.

- Any loss function can be used. The training error does not depend on the basis in which w and the input vectors are represented and therefore any function of these training errors will also remain unchanged. For those unfamiliar with loss functions; a loss function is a function that determines how the training errors are penalized in the optimization problem.

We can thus use the following kernel-based optimization problems for regression

$$\min_{\alpha\in\mathbb{R}^n, b\in\mathbb{R}} \sum_{i=1}^{n} l(y - f(x_i)) + C\left(\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i\alpha_j k(x_i, x_j)\right), \tag{1.25}$$

where $f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) + b$, and l is the loss function. In this dissertation we are particularly interested in the loss functions

$$l_{\mathrm{r}}(y_i - f_{\mathrm{r}}(x_i)) = \begin{cases} 0, & y_i f_{\mathrm{r}}(x_i) > 1, \\ 1 - y_i f_{\mathrm{r}}(x_i), & \text{otherwise}, \end{cases} \tag{1.26}$$

$$l_\epsilon(y_i - f_\epsilon(x_i)) = \begin{cases} 0, & |y_i - f_\epsilon(x_i)| \leq \epsilon, \\ |y_i - f_\epsilon(x_i)| - \epsilon, & \text{otherwise}, \end{cases} \tag{1.27}$$

$$l_\tau(y_i - f_\tau(x_i)) = \begin{cases} \tau(y_i - f_\tau(x_i)), & y_i - f_\tau(x_i) \geq 0, \\ (\tau - 1)(y_i - f_\tau(x_i)), & y_i - f_\tau(x_i) < 0. \end{cases} \tag{1.28}$$

SVM classification is the kernel-based algorithm that minimizes the ramp loss function (l_{r}) and SVM regression is the one that minimizes the epsilon-insensitive loss function (l_ϵ). These loss functions cause $\hat{\alpha}$ to be sparse. However, this sparsity is often limited. The SVM classifier has a nice geometrical interpretation. It corresponds to finding two parallel hyperplanes which divide the feature space in three areas such that:

- The data is split as clean as possible. A clean split means that the area between the hyperplanes splits the feature vectors in two parts such that all feature vectors corresponding to class 1 are at one side of the area and all feature vectors belonging to class 2 are at the other side. This implies that in a clean split there are no feature vectors that belong to the area between the hyperplanes.
- The distance between these hyperplanes is as big as possible.

Increasing the regularization constant C results in a greater distance between the hyperplanes and a less clean split. Because the sparsity of SVM classifiers is often limited - and the run time complexity thus quite high - we propose in Chapter 4 a method that increases the sparsity much more. Kernel-based

quantile esimtation minimizes the τ-loss function (l_τ). Similar to the fact that minimizing the square loss corresponds to an estimation of the mean, minimizing the tau loss corresponds to estimating the τ-quantile value. In the case of QoS prediction (see Chapter 3), for example, a service provider wants to be relatively sure that the contract made with the customer will be met. Therefore he is interested in the quantile value (e.g., 99% quantile value) of the quality attribute (e.g., the response time) rather than the mean value. Although we focus on regression methods in the introduction, other algorithms can also be made kernel-based. In Chapter 2 we apply the kernel trick to algorithms that measure diversity and related properties.

1.3 Quantifying biodiversity

1.3.1 Why is biodiversity important?

Biodiversity is the degree of variation of life. There is mounting evidence that biodiversity increases the stability of ecosystem functions [8]. As stated by the Millennium Ecosystem Assessment (MEA) [9], ecosystems are important because they provide provisioning services such as food, wood and fresh water; regulating services that affect climate, floods, disease, wastes, and water quality; cultural services that provide recreational, aesthetic, and spiritual benefits; and supporting services that support the tree other services, such as soil formation, photosynthesis, and nutrient cycling. MEA also mentions how these services influence the well-being of humans: clean air, clean water, and access to food affects human health; access to natural resources and safety with respect natural and human disasters affects human security; material well-being, health, security and changes to local environments (especially for cultures that have a strong connection to these environments) affects social relations; and the presence of the aforementioned constituents of well-being affects the freedom of choice and action (i.e., the ability to control what happens to you and being able to achieve what you want). The decrease in biodiversity and the changes in ecosystems are mainly caused by habitat destruction (mining, logging, trawling and urban sprawl), climate change, invasive alien species, overexploitation, and pollution. A disturbing fact is that high-income countries use five times more ecological resources than low-income countries. However, because wealthy nations outsource their resource depletion to poor nations, the latter suffer most of the ecosystem losses. During the last 40 years, the populations of mammals, birds, reptiles, amphibians and fish suffered a decrease of 83% in Latin America while they increased by 10% in wealthy countries [10]. Minimizing the decrease in biodiversity is one of the key challenges of the twenty first century. The United Nations General Assembly declared 2011-2020 the United Nations Decade on Biodiversity (Resolution 65/161). In the course of this

decade, governments are encouraged to develop, implement and communicate the results of national strategies with respect to the reduction of biodiversity loss. Science can contribute significantly to this challenge because, as stated by the MAE, "Improved capability to predict the consequences of changes in drivers for biodiversity, ecosystem functioning, and ecosystem services, together with improved measures of biodiversity, would aid decision-making at all levels." However, whether or not we will succeed in making this planet sustainable will ultimately depend on the willingness of society to implement the necessary measures.

1.3.2 How is biodiversity measured?

Biodiversity is a multidimensional concept that embraces different levels of biological systems (i.e., molecular, organismic, population, species and ecosystem), different attributes of communities (e.g., the number of different organism types, the quantity in which each type occurs, how these attributes change in space and time) and different features of organisms (e.g. diversity can reflect taxonomic as well as functional differences). One of the most straightforward ways to measure biodiversity is to count the number of species. It is claimed that there are roughly 8.7 million different species on this planet - give or take 1.3 million [11]. The number of species is the most popular biodiversity index because of the following reasons. First, it is a measure that is easy to understand and therefore easy to communicate to decision makers. Second, gathering data is a difficult and expensive process, and this measure only requires an estimate of the number of species. There is no need for data that reflects the quantity in which each species occurs or how different each pair of species is. Third, the index does not require subjective decisions. When measuring functional diversity, we need to decide which functional traits we consider, what the difference between each pair of traits is, and how we combine this information into a single distance or similarity measure. Finally, although a huge amount of alternative indices have been proposed in the literature, most of them have suboptimal properties. Therefore it is doubtful whether some of them would result in a more meaningful value, even when quantifying the specific biodiversity dimension for which they are designed. Despite the fact that the number of species is a valuable index, other indices are needed to fully capture the different dimensions of biodiversity. Until recently, most of the popular diversity indices were equivalence-based which means that two organisms are either different or equivalent. In reality some organisms differ more than others which cannot be modelled by a simple equivalence relation. Therefore there is a need for a more continuous approach which is provided by similarity-based indices. The similarity-based indices that measure biodiversity can be divided into two categories: diversity and richness indices. A diversity index is an index that reflects how many types there are, how dissimilar the

types are and how even the rarity of the different types is. A richness index only reflects the number of types and their dissimilarity. Both diversity and richness indices are preferably expressed in the effective number of types, which is the exponential function of the number of bits or nats (the units used in entropy).

1.3.3 Which indices do we propose?

In this dissertation we propose a similarity-based framework in which we define methods that can be used to measure the diversity and richness of groups of entities. This framework is not designed for a specific level of biodiversity and can be applied to all of them. Moreover, it is not restricted to biodiversity; it can measure any kind of diversity (e.g., it can also quantify gender, age and ethnic diversity in workplaces). Often we are not merely interested in the diversity and richness itself but also in the underlying causes, and therefore we propose a number of derived indices. The contribution vector that we propose expresses how much each type contributes to the richness. In the context of nature conservation, the contribution of a species can be used as one of the indicators of its conservation priority. We express the diversity as a product of the number of types, the average contribution and the evenness. The average contribution equals the aforementioned contribution averaged over the types and it reflects how much diversity is lost due to the similarity between the types. A low average contribution means that the types are very similar. The evenness expresses the loss in diversity due to the types not being equally rare. A low evenness means that some types are rarer than others. The diversity of a pooled group consisting of a number of subgroups can be decomposed in an alpha and beta component, where the alpha and beta diversity express how much within and among-subgroup differences contribute to the diversity of the pooled group, respectively. A high alpha means that the subgroups have a high diversity and a high beta means that the compositions of the subgroups differ a lot. This dissertation mainly focusses on the mathematical aspects of these indices. Developing similarity functions for specific dimensions of biodiversity is part of the future work.

1.3.4 How to define indices in our framework?

If we want to define a concept such as the equality relation or the similarity function, then there often exists a specific kind of objects such that defining the concept logically with respect to these objects is straightforward. For example, in a set of elements that is partitioned in different cells where each cell is an equivalence class, it makes sense to define two elements as equal if and only if they belong to the same cell. In this restricted framework, we can then propose indices. The equivalence-based richness, for instance, is an index that we define

as the number of equivalence classes. In practice, however, it would be nice if we could generalize this framework which means that (1) we can define the concept (e.g., the equivalence relation) in a meaningful way with respect to any type of objects, and (2) any index (e.g., the richness) that we define in the restricted framework can also be used in this general framework. The generalization trick - a new trick that we propose in this paper which is inspired by the kernel trick - accomplishes this as follows. First, we assign an object of the restricted framework - which we call a label - to each object of the general framework such that applying the concept to the labels results in the same value as applying the concept to the original objects. In other words, we map the general framework on the restricted framework. In the case of the equivalence relation, this mean that we have to create a set of labels that is partitioned into cells such that the labels of equivalent objects are members of the same cell. Second, we apply the index to these labels. Suppose, for instance, that we have three objects, named a, b and c, with the equivalence relation $\{(a,a),(b,b),(c,c),(a,b),(b,a)\}$ (two objects are equal if and only if they form a pair in this set) and we would to like to measure the richness of this group of objects. First, we assign the labels of the set $S = \{\{l_1, l_2\}, \{l_3\}\}$ to these objects where l_1, l_2 and l_3 are the labels of a, b and c, respectively. Second, the richness of the three objects equals the number equivalence classes in the label set, which is 2. By using the generalization trick we managed to apply the richness in the general framework

One can notice that it is not always possible to apply the generalization trick. For example, if we have the following equivalence relation $\{(a,a),(b,b),(c,c),(a,b),(b,a),(b,c),(c,b)\}$, then it is not possible to create a partitioned label set such that, on the one hand, the labels of a and b, and b and c belong to the same cell and, on the other hand, the labels of a and c belong to different cells. However, we do not wish to allow the concept to be defined in any way. We state that the concept is defined meaningfully if and only if the generalization trick can be applied. It can be shown that an equivalence relation is meaningful according this definition if and only if it is reflexive, symmetric and transitive. We thus derived the generally accepted properties that an equivalence relation must meet.

In this thesis we are not interested in the equivalence relation, but rather in the similarity function. The restricted similarity-based framework that we propose is limited to vectors of unit norm with positive inner products. These vectors are called feature vectors. The similarity between two feature vectors is defined as their square inner product. The smaller the angle between the feature vectors, the more similar they thus are. This similarity equals 0 when the feature vectors are orthogonal and equals 1 when they are identical. All the indices that we propose in this thesis are defined in this restricted framework and the generalization trick then makes sure that these indices can be applied to any meaningful similarity function in the general framework. It can be shown that a similarity function is meaningful - and that the generalization trick can

thus be applied - if and only if (1) the similarity of an element with itself equals 1, (2) the similarity function is symmetric and (3) the square root of the similarity function is a positive semidefinite kernel. Note that the type of objects, the similarity function and the indices in the restricted framework can all be chosen differently which implies that our definition of these indices and the similarity function is not the only possible meaningful definition.

1.3.5 OLC diversity, similarity and uniqueness

The first major index that we propose is the OLC diversity. The OLC diversty satisfies a number of favorable properties such as the twin property, the strong replication principle and the modularity property. The twin property states that representing two identical types by one type, or vice versa, shouldn't change the diversity; more concretely, if we have two identical types, which we call twins, then removing one of these twins from the group and adding its relative abundance to the other twin should not change the diversity. If we have a pooled group consisting of n completely distinct equally-sized subgroups with the same diversity d, then the strong replication principle says that the diversity of the pooled group should be n times d. The modularity property states that the diversity of a pooled group consisting of completely distinct subgroups is entirely determined by the diversity and the size of the subgroups. The OLC diversity is related to optimal lossy compression in the sense that it reflects the trade-off between the approximation error and the description length when compressing the assigned feature vectors optimally. The OLC diversity is the key index in our framework because the choice of the similarity depends on this index, i.e., the similarity should be chosen as the square inner product of the features vector that we would like to compress when quantifying the OLC diversity.

A disadvantage of several other similarity-based diversity indices is that it is not clear what similarity means in their framework. Without a proper definition, similarity is a vague concept (e.g., the similarity in one framework can be defined as the square of the similarity in another framework) and choosing the similarity properly becomes an impossible task. Because uniqueness is a concept that matches our intuition better than similarity with respect to certain aspects (e.g., we want a type with unique features to be partially unique), we also propose the OLC uniqueness vector. The OLC uniqueness vector expresses the uniqueness of each type as a function of the pairwise similarities. When designing the similarity function, it is important to check whether the implied uniqueness behaves as desired. Another reason why we define the OLC uniqueness, is because it is used in the properties of some of the indices (e.g., the richness should strictly increase when a partially unique type is added).

1.3.6 OLC richness and derived indices

The OLC richness is a second important index that we propose. It is defined as the OLC diversity maximized in the relative abundances. We choose this definition for the OLC richness because of the following three reasons. First, the OLC richness should meet a number of important properties such as: (1) the richness of a pooled group consisting of completely distinct subgroups equals the sum of the richnesses of these subgroups, (2) the richness of a pooled group consisting of subgroups that are not completely distinct is strictly smaller than the sum of their richnesses, (3) adding a type to the group doesn't decrease the richness, and (4) adding a type with a non-zero uniqueness strictly increases the richness. Second, we want the diversity to be the richness times the evenness where the evenness is maximally one. This implies that the richness can never be smaller than the diversity. Finally, we want the richness to not only reflect the number of different features that occur, but also the combinations in which they occur. For instance, adding a species with feature set 'AD' to a set containing two species with feature sets 'AC' and 'BD' should increase the richness because the combination of feature 'A' and 'D' didn't occur yet.

We propose the following derived indices. The OLC contribution vector expresses how much each type contributes to the OLC richness. Ideally, this index meets the following properties: (1) the sum of the contributions of the types belonging to the set equals the richness of the set, (2) a type with a non-zero uniqueness has a non-zero contribution, (3) the sum of the contributions of the types belonging to a subset equals at most the richness of that subset, and (4) if we have pooled group consisting of two completely distinct subgroups, then the contribution of a subgroup equals its richness. In this dissertation we discuss when the OLC contribution vector satisfies which properties. The OLC average contribution equals the OLC richness divided by the number of types. The index is close to 0 when the types are very similar and equals 1 when the types are completely distinct. The OLC evenness is defined as the OLC diversity divided by the OLC richness. It approaches 0 when one type dominates the group and becomes 1 when all types are equally rare. If we have a pooled group consisting of different subgroups, then the OLC alpha diversity expresses the OLC diversity averaged over the subgroups (this average is not necessarily an arithmetic mean). The OLC beta diversity equals the OLC diversity of the pooled group divided by the OLC alpha diversity. Intuitively, if we randomly draw an entity from the pooled group, then alpha reflects the uncertainty with respect to its type if we know to which subgroup the entity belongs, and beta reflects the increase in uncertainty when we do not know its subgroup.

1.4 QoS of workflow compositions

1.4.1 Research challenges

A workflow composition is a set of elementary services that are executed according to a certain pattern. This composition pattern can be a sequence (multiple tasks are executed sequentially), a parallel execution (multiple tasks are executed concurrently), an exclusive choice (one of the tasks will be executed), a loop (a task that is repeatedly executed) or any combination of the aforementioned patterns. In a commercial setting, consumers expect the services that they use to deliver certain levels of quality. Popular quantitative QoS attributes are response time, throughput, reputation, reliability, availability, and cost. The QoS on which both the consumer and the provider agree is often formally stated in a service level agreement (SLA) where the provider needs to assure that this agreement is met. Different strategies can be used to perform QoS-aware selection. A prospective strategy is to select the services before the execution of the application starts. A reactive strategy is to change the composition at run time, when a QoS constraint is expected to be violated or when the QoS requirements have been changed. Monitoring (i.e., gathering historical data) and analysis (i.e., drawing the right conclusions from the data) are important aspects of both strategies. There are three key challenges with respect QoS-aware service composition:

- **QoS prediction**: the prediction of future QoS values. In most cases there is a considerably amount of uncertainty about what the future values of QoS attributes will be. QoS prediction methods can be used to reduce this uncertainty.
- **QoS aggregation**: converting QoS information of the individual services into QoS information of the composition. There are two possible approaches here. One approach is to first perform prediction on the individual services and then aggregate these predictions. Another approach is to first convert the historical QoS values of the individual service into historical QoS values of the composite service and then perform prediction on the composition.
- **QoS selection**: selecting the individual services that will be used in the composition. Applying QoS aggregation and QoS prediction to all possible combinations of individual services that meet the functional requirements is often computationally unfeasible. Therefore we need an algorithm that can select a good composition within a reasonable amount of time.

In this dissertation we focus on the first two challenges.

1.4.2 Problems of existing methods

In the literature several methods have been proposed to tackle the above challenges. These methods have the following drawbacks. With respect to QoS prediction, QoS attributes are sometimes treated as fixed values while in reality they change over time. Also, when they are treated as variable, the used prediction algorithms are often insufficiently adapted to the specific needs of the service provider. Some of these algorithms predict average QoS values instead quantile values. Others perform Monte Carlo simulations on historical data and then take the quantile value with respect to these simulations. Note that the quantile value of past data does not correspond to the quantile value of future data. For example, if the response time follows a sawtooth wave with a period of one hour, then a response time in the near future will almost constantly exceed the 99%-quantile value of the last ten seconds. Most of the existing algorithm therefore do not provide any assurance that the probability of an SLA violation is below a certain threshold. With respect to QoS aggregation, a lot of methods have been proposed that calculate the QoS values of a composite service based on the QoS values of its constituents. Hard composition rules such as addition, maximum or conjunction convert point estimates of the individual services into points estimates estimates of the composition. However, in order to calculate the quantile value of the composition, the entire probability distributions of the QoS values of individual services are needed. Therefore these rules are restricted to the estimation of upperbounds which makes them overly pessimistic. Soft composition rules, on the other hand, combine probability distributions of the individual services. This implies that probability distributions of the individual services need to be predicted, which is much more difficult than performing point estimation. Furthermore, they make the assumption that the distributions are independent which is often not the case. Dependence can occur because of several reasons: both services run on the same server; users choose the fastest service out of a group of services which causes the services to become equally fast; many non-automatic service compositions that require human interaction have a higher response time outside working hours and the response times are thus correlated; etc. In these scenarios, soft composition rules give distorted results.

1.4.3 Contributions in this thesis

In this dissertation we first aggregate the QoS values and then perform prediction. This approach has the advantage that (1) we do not need to aggregate probability distributions because there is no uncertainty with respect to past QoS values, and (2) the aggregation implicitly takes into account dependencies because these dependencies are already present in the past data (e.g., if two services are equally fast, then they will be equally fast in the historical data). By

means of Petri nets, we describe in detail how to perform this aggregation for different QoS attributes and combinations of composition patterns. For the QoS prediction task, we use a kernel-based quantile estimator with online adaptation of the constant offset. This estimator can have any relevant variable as input including time or recently monitored QoS values which means that it takes into account the fact that QoS attributes can change over time. Also, kernel-based methods can be chosen non-parametric and therefore no explicit assumptions with respect to the relation between inputs and outputs need to be made. The adaptation of the constant offset assures that under rather weak assumptions the number of times the predicted value is worse than the actual value converges to the quantile value specified in the SLA. To compare our method with other methods, we propose two performance indices. The first performance index checks how close the predicted QoS attribute is to the actual QoS attribute and the second performance index verifies if the actual quantile value is in accordance with the quantile value specified in the SLA.

1.5 Reducing the run time complexity

1.5.1 Computational complexity of SVMs

When applying prediction methods there are two important phases: the training phase and the evaluation phase. During the training phase we use labeled data to learn the design of the prediction algorithm. Learning this design often corresponds to determining certain parameters (e.g., linear coefficients in the case of linear regression). Once the algorithm is trained, we can start the evaluation phase which means that we use the algorithm to predict the labels of new data. The computational complexity of the training phase and the evaluation phase are called the training and run time complexity, respectively. The run time complexity of SVMs scales linearly with the number of support vectors and, in most cases, the number of support vectors scales linearly with the number of labeled datapoints used to train the algorithm.

1.5.2 Reducing the run time complexity

We propose a new method that decreases the run time complexity of Support Vector Machine (SVM) classifiers by reducing the number of support vectors. This reduction is achieved by an algorithm called Separable Case Approximation (SCA) that changes or removes the labels of misclassified training data points such that the classification problem becomes separable in the feature space. We then apply a hard-margin SVM which is typically much sparser than a soft-margin SVM. The norm of the weight vector achieved by SCA can, however,

become arbitrarily large. Therefore we also propose a smoothed version, called Smoothed Separable Case Approximation (SSCA), that additionally removes training datapoints that are close to the decision boundary which is equivalent to upper bounding the norm of weight vector of the approximating solution. This upper bounding ensures numerical stability, reduces the time to find the pruned solution, avoids overfitting during the approximation phase and reduces the number of support vectors even more.

1.6 Overview

This dissertation is organized as follows. Chapter 2 explains the diversity, richness and related indices that we propose. The QoS prediction algorithms are discussed in Chapter 3. In Chapter 4 we propose SCA and SSCA; two algorithms that reduce the run time complexity of SVM classifiers. Chapter 5 concludes this dissertation. This dissertation is a series of papers. Chapter 2 corresponds to a submitted paper, and Chapter 3 and Chapter 4 correspond to published papers.

References

[1] Schölkopf, B., and A. J. Smola. Learning with Kernels. MIT Press, 2002.

[2] Vapnik, V., Statistical Learning Theory. John Wiley & Sons, 1998.

[3] Cortes, C., and V. Vapnik, "Support vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, 1995.

[4] Suykens, J. A. K., T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, Least Squares Support Vector Machines. Singapore: World Scientific, 2002.

[5] Schölkopf, B., R. Herbrich, and A. J. Smola. (2001). "A Generalized Representer Theorem". Computational Learning Theory. Lecture Notes in Computer Science, no 2111, pp 416–426, 2001.

[6] Mercer, J. "Functions of positive and negative type, and their connection with the theory of integral equations," Philos. Trans. Royal Soc. London, vol. 209, nos. 441–458, pp. 415–446, 1909.

[7] Downs, T., K. E. Gates, and A. Masters, "Exact simplification of support vector solutions," J. Mach. Learn. Res., vol. 2, pp. 293–297, Dec. 2001.

[8] Cardinale, B. J., J. E. Duffy, A. Gonzalez, D. U. Hooper, C. Perrings, P. Venail, A. Narwani, G. M. Mace, D. Tilman, D. A. Wardle, A. P. Kinzig,

G. C. Daily, M. Loreau, J. B. Grace, A. Larigauderie, D. S. Srivastava, and S. Naeem. “Biodiversity loss and its impact on humanity.” Nature 486, no. 7401, pp. 59-67, 2012.

[9] Millennium Ecosystem Assessment. Ecosystems and human well-being, vol. 5: Washington DC: Island Press, 2005.

[10] World Wildlife Fund (WWF). Living Planet Report 2014: Species and spaces, people and places. World Wildlife Found Editor, 2014.

[11] Mora, C., D. P. Tittensor, S. Adl, A. G. B. Simpson, B. Worm. “How Many Species Are There on Earth and in the Ocean?”. PLoS biology 9, no. 8: e1001127, 2011.

Chapter 2

Similarity-based Diversity, Richness and Derived Indices

Extending true diversity, richness and derived indices to a similarity-based framework using optimal lossy compression

Dries Geebelen, Luc De Meester, Johan A.K. Suykens, Joos Vandewalle

Submitted to *Ecological Monographs*.

Author contributions: D.G. did the literature study, designed the indices, proved the theorems, implemented the methods in matlab, performed the experiments, and wrote the paper. L.D.M. corrected the text in detail and rewrote certain paragraphs in order to improve the biological part of the paper. The remaining authors provided feedback.

Abstract

Biodiversity is at the core of ecological and evolutionary research, both with respect to the mechanisms that maintain biodiversity as well as its consequences on population, community and ecosystem function. For many applications, there is a need for diversity indices that reflect not only the relative abundances of the different types, but also the similarities between these types, e.g., with regard to their traits or phylogenetic history. Most existing similarity-based diversity, richness and related indices have suboptimal properties. Here we propose indices that overcome these problems. First, we propose the Optimal Lossy Compression (OLC) diversity. This diversity index is related to lossy compression in the sense that it reflects the trade-off between the approximation error and the description length when compressing the representation of a group of entities optimally. The OLC diversity satisfies the strong replication principle, the modularity property and the twin property; three properties for which it is desired that they are met by diversity indices. Second, we propose a richness index, named the OLC richness, that is defined as the OLC diversity maximized in the relative abundances. The OLC richness has a number of favorable properties, such as (1) adding a type never decreases the richness, (2) the richness of the union of completely distinct groups equals the sum of their richnesses, and (3) when applied to phylogenetic trees, the richness takes into account the skewness of the tree. In order to gain insight into the underlying mechanisms that cause a certain diversity or richness, we also propose a number of derived indices: the OLC contribution vector expresses how much each type contributes toward the OLC richness; the OLC average contribution and the OLC evenness quantify the influence of the type similarities and the relative abundances on the OLC diversity, respectively; and the decomposition of the OLC diversity in alpha and beta gives insight into both within- and among-subgroup differences. We validate our indices on both simulated and real-world data.

2.1 Introduction

While the majority of biodiversity studies have largely focused on species, genotype or other entities of specific taxonomic levels, for many questions it is useful to take similarity between types into account. Phylogenetic diversity measures reflect the phylogenetic similarity between the species and are useful for gaining insight into how communities are structured and in prioritizing for diversity conservation [1, 2]. Functional diversity measures [3, 4, 2] reflect the similarity of functional traits in communities and ecosystems and are increasingly used to obtain insight into ecosystem functions, ecosystem services and the response to environmental gradients.

Although there is no consensus on how diversity indices should take into account type similarities, there nevertheless exists properties that meaningful similarity-based diversity measures are expected to satisfy, such as the twin property [5], the strong replication principle [6] and the modularity property [7]. The twin property says that representing two identical types by one type, or vice versa, should not change the diversity; more concretely, if we have a group that contains two identical types - which we call twins - then the diversity should not change when we remove one of these twins and add its relative abundance to the other twin. If we have a pooled group consisting of N equally large, completely distinct subgroups with the same diversity d, then the strong replication principle says that the diversity of the pooled group equals N times d. The modularity property states that the diversity of a pooled group consisting of completely distinct subgroups is fully determined by the size and the diversity of the subgroups. The twin property should not only be met by diversity indices. Any similarity-based index on which actual decisions are based is expected to meet this property because a decision should not depend on which of two equivalent representations is used. Nevertheless, indices that do not satisfy the twin property can still be interesting from a purely informative perspective (e.g., it is interesting to know how many types there are in a group of entities).

A lot of interesting and useful diversity, richness and related indices have been proposed in the literature. However, incorporating similarity in these indices is not at all trivial, which is reflected by the fact that most existing similarity-based indices have properties that can be considered suboptimal. Table 2.1, Table 2.2 and Table 2.3 list disadvantages of commonly used similarity-based diversity, richness and other indices, respectively. A more detailed discussion of these disadvantages can be found in Section 2.2. The main contribution of this paper is that we propose indices that do not have these issues. The diversity and richness indices that we propose are called the Optimal Lossy Compression (OLC) diversity and the OLC richness, respectively. In general, it is not trivial to empirically verify how good a certain diversity or richness index is. Often, the main guarantee for generating reasonable results, is provided by the properties that the index satisfies. Therefore this paper provides an extensive list of properties for which it is useful for the user to know whether they hold or not. Because we are often interested in the underlying causes of the measured diversity or richness, we also propose the following indices: the OLC contribution vector quantifies the contribution of each type towards the richness; the OLC average contribution and the OLC evenness express how much diversity is lost due to the similarities between the types and not having the abundances that maximize the diversity, respectively; and the decomposition of the OLC diversity in alpha and beta expresses how much both within- and among-subgroup differences contribute to the diversity. When designing the type similarities, it is important to verify whether the uniqueness implied by the similarity function behaves as desired (e.g., in most applications, it is desired that a type with a unique feature is partially unique). Therefore we propose the

Table 2.1: Limitations and suboptimal properties of commonly used similarity-based diversity indices. The strong replication principle, the modularity property and the twin property are denoted by SRP, MP and TP, respectively. A 'yes' means that the property always holds when used according to the instructions mentioned by the authors, and a 'no' means that it doesn't always hold.

Index	SRP	MP	TP	Limitations and suboptimal properties	Reference
Rao's index	yes[1,2]	yes[1,2]	yes	has only a quadratic version	Rao [8]
Lineage-based phylogenetic diversity	no[3,4]	no[4]	yes	can only be applied to phylogenetic trees and dendograms; makes wrong assumptions with respect to higher order feature similarities	Chao et al. [6]
Average Similarity Diversity (ASD)	yes	yes	yes	Shannon version is upperbounded by the inverse of the similarity of the two most dissimilar types	Leinster and Cobbold [7]
Pavoine and Izsák's diversity	no[4,5]	no[4,5]	yes	can decrease when the intraspecific component increases	Pavoine and Izsák [9]
Distance-based functional diversity	no[1,3]	no[1]	no	functional diversity of two species does not depend on the species abundances	Chiu and Chao [10]

[1] In the case of distance-based indices, we define completely distinct as a fixed distance.

[2] We first transform the index into a true diversity index before checking wether the property holds.

[3] We disagree with the authors, which claim that either the replication principle or the strong replication principle holds.

[4] In the case of ultrametric trees (i.e., trees where the path-lengths from the root to the tips are equal), the property holds. In the case of non-ultrametric trees, it doesn't.

[5] Property does not hold when using intraspecific components.

Table 2.2: Limitations and suboptimal properties of commonly used similarity-based richness indices.

Index	Limitations and suboptimal properties	Reference
Faith's Total Branch Length (FTBL)	does not take into account the skewness of the tree; makes wrong assumptions with respect to higher order feature similarities	Faith [11]; Faith et al. [12]
Phylogenetic Species Richness (PSR)	does not satisfy the twin property	Helmus et al. [13]
Functional Attribute Diversity (FAD)	does not satisfy the twin property	Walker at al. [14]
Functional Diversity (FD)	additional linking or hierarchical clustering needs to be performed	Petchey and Gaston [15]

OLC uniqueness vector, which expresses the uniqueness of each type within a set of types as a function of the pairwise similarities. A list of the units in which the indices are expressed, is shown in Table 2.4. The MATLAB implementation of the proposed indices can be downloaded from: LINK-COMES-HERE. The indices, properties, notations and algorithms proposed in this paper, are often illustrated with an example (see Appendix 2.A), and we recommend taking a look at these examples when more clarity is needed.

From a mathematical point of view, there already exist a number of indices that are equivalent or similar to the OLC diversity. The Von Neumann entropy [21], which is used to measure quantum information, is equivalent to the Shannon OLC entropy, where the OLC entropy is the logarithm of the OLC diversity. The quadratic Rényi entropy, as applied to Gaussian kernels in Fixed Size Least Square Support Vector Machines (FS LS-SVM) [22], is not only mathematically equivalent to the quadratic OLC entropy, it is also used for the same purpose: to quantify how different the datapoints are. Finally, the index of Pavoine and Izsák [9], which is the first such method used to measure ecological diversity, can be considered more general than the OLC diversity in the sense that it can be applied with respect to any pairwise function as long as this function meets certain conditions (i.e, it must be a positive semidefinite kernel), while the OLC diversity only allows similarity functions. However, our index is only a special case of the index of Pavoine and Izsák [9] when the similarity in our framework is the square of the similarity in their framework. Despite the existence of mathematically equivalent and similar indices, the OLC diversity is

Table 2.3: Limitations and suboptimal properties of commonly used similarity-based evenness (E), originality (O) and contribution (C) indices, where originality and contribution indices are defined as indices that express how unique each type is within the set and how much each type contributes to the richness, respectively. We also discuss a decomposition in alpha and beta diversity (AB).

Index	Type	Limitations and suboptimal properties	Reference
Phylogenetic Species Evenness (PSE)	E	does not meet the twin property	Helmus et al. [13]
Phylogenetic Abundance-Evenness (PAE)	E	does not meet the twin property	Cadotte et al. [16]
Originality of a Species within a Set (OSS)	O	partially unique species do not always have a non-zero originality; does not express how original a species is in the absolute sense (e.g. an originality of 0.5 can mean that a species is completely distinct from the other species as well as that it is very similar to the other species)	Champely and Chessel [17], Pavoine and Izsák [9]
Equal Splits (ES)	C	small changes in phylogenetic species similarity can cause large changes in the contributions	Redding and Mooers [18]
Fair Proportion (FP)	C	the contribution of a type decreases when another type is duplicated[1]	Isaac et al. [19]
Phylogenetic lineage-based alpha and beta diversity	AB	alpha is not some kind of average of the diversities of the different subgroups; alpha can become smaller than one and beta can become greater than gamma	Chiu et al. [20]

[1] Duplication means that we add a type that is equivalent to the original type.

Table 2.4: The main indices that we propose and the units in which they are expressed.

Index	Units
Similarity	Proportion of the type that is not unique with respect to the other type
Uniqueness vector	Vector where each element expresses the proportion of the type that is unique with respect to the other types belonging to the set
OLC diversity	Effective number of types
OLC richness	Effective number of types
OLC contribution vector	Vector where each element is expressed in the effective number of types (i.e., the part of the richness that is assigned to each type)
OLC average contribution	Proportion, one minus this index equals the proportion of the richness that is lost due to the similarities between the types
OLC evenness	Proportion, one minus this index equals the proportion of the diversity that is lost due not having the optimal abundances
OLC alpha diversity	Effective number of types per subgroup
OLC beta diversity	Effective number of subgroups

unique because of the similarity-based framework in which it is defined and the meaning it has in this framework.

2.2 Related work

Due to the large number of diversity and related indices that have been proposed in the literature, we mainly focus on measures that can be applied to phylogenetic trees in the related work. Nevertheless, the framework that we propose is suited for many other applications.

2.2.1 Rao's index

Rao's index [8] is defined as

$$\mathrm{RI} = \sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mathbf{D}_{ij} \mathbf{p}_i \mathbf{p}_j, \tag{2.1}$$

where $\mathbf{D}_{ij}$ is the distance between type i and type j, and $\mathbf{p}$ is the abundance vector. It is a clear and intuitive index for which we didn't find any disadvantages. This index has the limitation, however, that it only covers the quadratic case and it can thus not be used to measure, for example, Shannon diversity. In order to meet the strong replication principle and the modularity property, we must first transform Rao's index into a true diversity index as follows:

$$\mathrm{RQD} = \left(\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mathbf{S}_{ij} \mathbf{p}_i \mathbf{p}_j \right)^{-1} = (1 - \mathrm{RI})^{-1}, \tag{2.2}$$

where RQD denotes Rao's quadratic diversity, and the similarity between type i and type j equals $\mathbf{S}_{ij} = 1 - \mathbf{D}_{ij}$. Note that before applying the transformation, the distances should be scaled such that a distance equal to one corresponds to completely distinct types.

2.2.2 Lineage-based phylogenetic diversity

The mean diversity of order q over T years is a phylogenetic diversity measure proposed by Chao et al. [6] that is defined as

$${}^{q}\bar{\mathrm{D}}(T) = \frac{1}{T} \left(\sum_{i=1}^{n_b} \mathbf{x}_i \left(\frac{\mathbf{a}_i}{T} \right)^q \right)^{\frac{1}{1-q}}, \tag{2.3}$$

where n_b is the number of branches in the tree, $\mathbf{x}_i$ is the length of branch i, and $\mathbf{a}_i$ equals the sum of the relative abundances of the species descended from branch i. They also developed the phylogenetic diversity of order q through T years:

$${}^{q}\mathrm{PD}(T) = T \ {}^{q}\bar{\mathrm{D}}(T) = \left(\sum_{i=1}^{n_b} \mathbf{x}_i \left(\frac{\mathbf{a}_i}{T} \right)^q \right)^{\frac{1}{1-q}}. \tag{2.4}$$

The latter index is a generalization of Faith's total branch length. Chiu et al. [20] proposed a decomposition in alpha and beta diversity of ${}^{q}\bar{\mathrm{D}}(T)$:

$${}^{q}\bar{\mathrm{D}}_{\gamma}(T) = \frac{1}{T} \left(\sum_{i=1}^{n_b} \mathbf{x}_i \left(\frac{\sum_{j=1}^{N} \mathbf{w}_j \mathbf{A}_{ij}}{T} \right)^q \right)^{\frac{1}{1-q}}, \tag{2.5}$$

$$ {}^{q}\bar{\mathrm{D}}_{\alpha}(T) = \frac{1}{TN}\left(\sum_{i=1}^{n_b}\mathbf{x}_i\sum_{j=1}^{N}\left(\frac{\mathbf{w}_j\mathbf{A}_{ij}}{T}\right)^{q}\right)^{\frac{1}{1-q}}, \tag{2.6} $$

$$ {}^{q}\bar{\mathrm{D}}_{\beta}(T) = \frac{{}^{q}\bar{\mathrm{D}}_{\gamma}(T)}{{}^{q}\bar{\mathrm{D}}_{\alpha}(T)}, \tag{2.7} $$

where N is the number of subgroups, $\mathbf{w}_j$ is the size of subgroup j, and $\mathbf{A}_{ij}$ equals the sum of the relative abundances of the species descended from branch i in subgroup j. These indices have the following limitations and suboptimal properties. They can only be applied to phylogenetic trees and dendograms. If it is not natural to represent a set of types by a dendogram, then it is preferable to use diversity measures that do not require the, often non-straightforward, conversion into a dendogram. Also, we claim that ${}^{q}\bar{\mathrm{D}}(T)$ does not satisfy the modularity property and the strong replication principle when applied to non-ultrametric trees (see Appendix 2.B.1 for a counterexample). According to us, the strong replication principle should hold in a phylogenetic setting when the position of the root is the same in all subgroups and not when, as argued by Chao et al. [6], the mean path-length is the same. The reasoning behind this claim is the following. The length of the branches shared by two species is used as a proxy for the shared features. This length should thus not depend on the subgroup to which both species belong and the position of the root should therefore be the same in all subgroups. It cannot be the case that two species are completely different when belonging to one subgroup and quite similar when belonging to another subgroup. A last problem of ${}^{q}\bar{\mathrm{D}}(T)$ is that it does not model higher order similarities (i.e., the number of features that three or more species have in common) correctly and, as we show in Section 2.10.2, their Shannon version depends on these similarities. For example, if we have an ultrametric phylogenetic tree that contains three phylogenetically equally distant species, then according to most substitution models - such as the JC69 model [23], the JTT model [24] and the LG model [25] - there exist features that belong to species 1 and species 2, but not to species 3. However, ${}^{q}\bar{\mathrm{D}}(T)$, which uses the shared evolutionary history as a proxy for the shared features, says that no such features exist because all the common ancestors of species 1 and species 2 are also ancestors of species 3. The proposed alpha and beta diversity have the problem that they do not satisfy the assumptions in Jost [26] because alpha is not some kind of average of the diversities of the different subgroups (see Section 2.10.3). For instance, if all subgroups have the same diversity d and different relative sizes, then alpha is smaller than d. Moreover, if the aforementioned subgroups contain only one type (and thus have a diversity equal to one), then alpha becomes smaller than one and beta becomes greater than gamma.

2.2.3 Average similarity diversity

The similarity-based diversity proposed by Leinster and Cobbold [7] equals

$$ {}^{q}\mathrm{D}^{Z}(\mathbf{p}) = \left(\sum_{i=1}^{n_t} \mathbf{p}_i \mathbf{z}_i^{q-1} \right)^{\frac{1}{1-q}}, \tag{2.8} $$

where $\mathbf{p} \in \mathbb{R}^{n_t}$ is the abundance vector, and $\mathbf{z}_i$ is the expected similarity between an individual of the i-th species and an individual chosen at random. The authors did not give their diversity measure a name. We will call it the Average Similarity Diversity (ASD) because, as explained by the authors, it can be expressed as follows:

$$ {}^{q}\mathrm{D}^{Z}(\mathbf{p}) = \left(\left(\sum_{i=1}^{n_t} \mathbf{p}_i \left(\sum_{j=1}^{n_t} \mathbf{p}_j \mathbf{S}_{ij} \right)^{q-1} \right)^{\frac{1}{q-1}} \right)^{-1}, \tag{2.9} $$

where $\mathbf{S} \in \mathbb{R}^{n_t \times n_t}$ is the similarity matrix with $\mathbf{S}_{ij}$ the similarity between the i-th and the j-th species. The inverse of ASD is thus some kind of average of the similarities between the entities. If the similarity between the two most dissimilar species equals s_m, then ASD is upperbounded by ${}^{q}\mathrm{D}^{Z}(\mathbf{p}) \leq 1/s_m$ for any q. This index has the issue that the aforementioned upperbound is not desired when measuring the Shannon diversity. For example, when the phylogenetic similarity between each pair of species equals 0.5 and there are one million equally abundant species, then the Shannon ASD is only 2.

2.2.4 Diversity with respect to eigenvalues of matrices

Pavoine and Izsák [9] defined the diversity with respect to a positive semidefinite matrix $\mathbf{C} \in \mathbb{R}^{n_t \times n_t}$ and an abundance vector $\mathbf{p} \in \mathbb{R}^{n_t}$ as follows:

$$ {}^{q}\mathrm{D}^{\lambda}(\mathbf{p}, \mathbf{C}) = {}^{q}\mathrm{D}\left(\frac{\boldsymbol{\lambda}}{\sum_{i=1}^{n_t} \boldsymbol{\lambda}_i} \right), \tag{2.10} $$

where the diagonal elements of $\mathbf{C}$ are the intraspecific components, the off-diagonal elements of $\mathbf{C}$ are the interspecific components, $\boldsymbol{\lambda} \in \mathbb{R}^{n_t}$ contains the eigenvalues of $\mathbf{P}^{\frac{1}{2}}\mathbf{C}\mathbf{P}^{\frac{1}{2}}$, $\mathbf{P} \in \mathbb{R}^{n_t \times n_t}$ is a diagonal matrix with $\mathbf{P}_{ii} = \mathbf{p}_i$, and ${}^{q}\mathrm{D}$ is the type-neutral true diversity (the type-neutral true diversity is the diversity in which all types are assumed to be completely distinct). This index is an interesting contribution in the sense that it is the first diversity index in ecology that calculates the diversity with respect to the eigenvalues of a positive semidefinite matrix. The index of Pavoine and Izsák [9] is mathematically more general than our index when the similarity in their framework is chosen as the

square root of the similarity in our framework. Therefore we want to elaborate on the advantages of the OLC diversity compared to this index. A diversity index is expected to bridge the entire distance between the type data and the diversity. Converting the type data in the correct pairwise similarities is as important as applying the right diversity index to these similarities. It must therefore be clear to the user how the chosen similarity function relates to the obtained diversity. The OLC diversity expresses how efficiently we can compress the real vector representations of the types. The similarity in our framework should then, as we will explain later in this paper, be chosen such that it equals the square inner product of these desired real vector representations. In the framework of Pavoine and Izsák [9], it is not entirely clear how the intraspecific and interspecific components relate to the obtained diversity, which makes it difficult to choose the similarities properly. For example, if we replace a certain similarity by its square or square root, then it is not clear what exactly changes in the framework of Pavoine and Izsák [9] and it is thus hard to decide which of these three similarities is the most appropriate. Also, if intraspecific components are used - as is the case for non-ultrametric trees - then the index of Pavoine and Izsák [9] no longer satisfies the strong replication principle and the modularity property (see Appendix 2.B.1). Furthermore, increasing an intraspecific component can decrease the diversity, which is undesired. Note that the incorporation of intraspecific components is what makes this index more general than our index.

2.2.5 Distance-based functional diversity

Chiu and Chao [10] proposed the following functional diversity measures:

$$ {}^{q}\mathrm{D}(Q) = \left(\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \frac{\mathbf{D}_{ij}}{Q} (\mathbf{p}_i \mathbf{p}_j)^q \right)^{\frac{1}{2(1-q)}}, \tag{2.11} $$

$$ {}^{q}\mathrm{MD}(Q) = {}^{q}\mathrm{D}(Q) \cdot Q, \tag{2.12} $$

$$ {}^{q}\mathrm{FD}(Q) = {}^{q}\mathrm{D}(Q) \cdot {}^{q}\mathrm{MD}(Q), \tag{2.13} $$

where n_t is the number of species, $\mathbf{D}_{ij}$ is the functional distance between species i and species j, and $Q = \sum_{i=1}^{S} \sum_{j=1}^{S} \mathbf{D}_{ij} \mathbf{p}_i \mathbf{p}_j$ is the expected distance between two randomly selected species. They gave the following interpretation to these diversity measures: ${}^{q}\mathrm{D}(Q)$ is the functional Hill number which expresses the effective number of equally abundant and functionally equally distinct species, ${}^{q}\mathrm{MD}(Q)$ is the mean functional diversity per species, and ${}^{q}\mathrm{FD}(Q)$ is the total functional diversity. The total functional diversity becomes Walker's FAD [14] for $q = 0$. These indices have the following problems. First, these indices do not meet the twin property (see Section 2.10.6). Second, we claim that ${}^{q}\mathrm{D}(Q)$

does not meet the strong replication principle and the modularity property. The distance between two types represents their dissimilarity and therefore completely distinct corresponds either to a fixed maximum distance or to a distance equal to infinity. One can check that in neither of these cases, the strong replication principle or the modularity property are satisfied by ${}^{q}\mathrm{D}(Q)$. Chiu and Chao [10] propose a replication principle that is different from the one that we propose. We disagree that it is desired that this replication principle holds. For example, if we have N equally large, nearly-equivalent subgroups with the same diversity d, then it is desired that the diversity of the pooled group is close to d. If these subgroups meet certain criteria (see Appendix 2.B.2), however, then the replication principle mentioned in Chiu and Chao [10] states that the diversity of the pooled group should be exactly N times d, which is not close to d. A last disadvantage is that the total functional diversity of two species does not depend on the species abundances (see Section 2.10.6). In other words, the functional diversity does not reflect the abundance evenness when there are two species.

2.2.6 Faith's total branch length

Faith's Total Branch Length (FTBL) [11, 12] is a phylogenetic richness measure that is defined as the sum of the lengths of all the branches that are members of the minimum spanning path. In recent versions, the root of the tree is also included, such that a tree containing one species has a total length different from zero. FTBL has the disadvantage that it does not take into account the skewness of the tree (see Section 2.10.2). This causes a population consisting of five species of reptiles and five species of mammals not to have a higher richness than a population consisting of eight species of reptiles and two species of mammals. Also, similar to the lineage-based phylogenetic diversity proposed by Chao et al. [6], FTBL makes wrong assumptions with respect to higher order feature similarities (see Section 2.10.2).

2.2.7 Originality of species within a set

Originality of Species within a Set (OSS) [27] quantifies the uniqueness of a species. It assigns a relative abundance to each species, such that the expected distance between two species randomly drawn from the set is maximized. This approach is similar to the approach of Champely and Chessel [17]. Pavoine and Izsák [9] modified this approach by maximizing the quadratic diversity instead of minimizing the distance. All of these originality indices have the disadvantage that they do not always assign a non-zero originality to species that are partially unique. Furthermore, these originality indices only have a meaning in the relative sense; the originality reflects whether a certain species

is more original than another species belonging to the group, but it does not reflect whether the species is very similar to or completely distinct from the other species. In the case of ultrametric phylogenetic trees, for example, the originality of each species in a group containing two species equals 0.5 regardless of how phylogenetically similar these species are.

2.2.8 Equal splits and fair proportion

Equal Splits (ES) [18] and Fair Proportion (FP) [19] assign a contribution to each species with respect to FTBL. Along the path from the root of an ultrametric tree to the leaves, the richness increases proportional to the traveled distance. At each node, ES equally divides the richness over all outgoing branches. The resulting richness at each leaf equals the ES contribution of the corresponding species. In the case of FP, the contribution of a species with respect to the richness equals the weighted sum of the branch lengths along the path from the root of an ultrametric tree to the leaf representing the species, where the weight corresponding to a branch equals the inverse of number of species that share that branch. In Section 2.10.4 we show that ES and FP have the following problems. ES has the issue that adding a branch with zero length can change the contributions. FP has the problem that the contribution of a type decreases when another type is duplicated. Therefore, in a group consisting of a greater flamingo, an African forest elephant and an African bush elephant, the contribution of the greater flamingo decreases considerably when the two African elephants are treated as separate nearly-identical species instead of as one species.

2.2.9 Phylogenetic diversity based on Brownian motion

The methods proposed by Helmus et al. [13] are based on Brownian motion. Let X_i denote the value of neutral trait X for species i, and suppose that as time progresses, X_i randomly increases or decreases incrementally. The value of X_i at any point in time will be normally distributed, and if time is proportional to the branch lengths, then the variance of X_i is proportional to the distance from the root of the tree to leaf i. Their proposed Phylogenetic Species Variability (PSV) assumes equal abundances and expresses the expected variance of this neutral trait divided by the expected variance in case the covariance matrix $\mathbf{C} \in \mathbb{R}^{n_t \times n_t}$ is diagonal. PSV thus becomes

$$\mathrm{PSV} = \frac{1 - \mathbf{u}^\intercal \mathbf{C} \mathbf{u}}{1 - \mathbf{u}^\intercal \mathbf{u}}, \tag{2.14}$$

where $\mathbf{u} \in \mathbb{R}^{n_t}$ is the uniform abundance vector, i.e., $\mathbf{u}_i = \frac{1}{n_t}$ for $i \in \{1, \ldots, n_t\}$. They also propose Phylogenetic Species Evenness (PSE), which is the same

measure but based on the real abundances instead of equal abundances:

$$\text{PSE} = \frac{1 - \mathbf{p}^\intercal \mathbf{C} \mathbf{p}}{1 - \mathbf{u}^\intercal \mathbf{u}}, \tag{2.15}$$

and Phylogenetic Species Richness (PSR), which equals

$$\text{PSR} = n_t \cdot \text{PSV}. \tag{2.16}$$

These indices have the problem that they do not meet the twin property (see Section 2.10.4 2.10.5).

2.2.10 Evenness based on terminal branch lengths

Cadotte et al. [16] proposed the following phylogenetic abundance-evenness measure:

$$\text{PAE} = \frac{\text{FTBL} + \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i (\mathbf{c}_i - 1)}{\text{FTBL} + (\bar{\mathbf{c}} - 1) \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i}, \tag{2.17}$$

where FTBL is Faith's total branch length, $\boldsymbol{\lambda}_i$ is the terminal branch length of species i, $\mathbf{c}_i$ is the number of individuals belonging to species i, and $\bar{\mathbf{c}}$ is the average number of individuals per species. Because we only work with the relative abundance vector $\mathbf{p}$ in this paper, we approximate PAE as follows:

$$\text{PAE'} = \frac{\sum_{i=1}^{n_t} \mathbf{p}_i \boldsymbol{\lambda}_i}{\sum_{i=1}^{n_t} \frac{1}{n_t} \boldsymbol{\lambda}_i}. \tag{2.18}$$

For the number of individuals going to infinity, PAE converges to PAE'. Both PAE and PAE' have the disadvantage that they do not meet the twin property (see Section 2.10.5).

2.3 Similarity-based framework

In this section we will propose a framework in which we define similarity and uniqueness. The similarity between two types reflects the similarity in their features, and the uniqueness of a type reflects the proportion of the type's features that are unique within a set of types. The reason we propose the concept of uniqueness in addition to the concept of similarity is twofold. First, uniqueness matches our intuition better than similarity for certain aspects. For instance, in most applications we want a type with a unique feature to be partially unique and a type with nothing but unique features to be completely unique. Second, the desired properties of certain indices depend on the uniqueness of a type. For example, a type that is partially unique within a set should strictly increase the

richness when added to the set. We will first define similarity and uniqueness in a restricted framework that limits the object of interest to real vectors of unit norm with nonnegative inner products - where unit norm means that the Euclidean norm equals one - because with respect to these specific objects, we have found a way to define both concepts such that they behave as desired. We then generalize the framework to any kind of objects and any kind of similarity function, as long as the similarity function is meaningful (i.e., the similarity must satisfy certain properties such as the symmetry property).

2.3.1 Restricted framework

In the restricted framework, the similarity between $\mathbf{f_i}$ and $\mathbf{f_j}$, two vectors of unit norm with a nonnegative inner product, is defined as their square inner product:

$$s(\mathbf{f_i}, \mathbf{f_j}) = (\mathbf{f_i}^\intercal \mathbf{f_j})^2. \tag{2.19}$$

The similarity equals zero when the vectors are orthogonal and equals one when the vectors are equal. Note that the similarity uniquely defines the inner product between the feature vectors as $\mathbf{f_i}^\intercal \mathbf{f_j} = \sqrt{s(\mathbf{f_i}, \mathbf{f_j})}$ because the inner products are nonnegative. The uniqueness of a feature vector within a set of feature vectors is defined as the square error when approximating the feature vector by a linear combination of the other feature vectors:

$$\mathrm{u}_i = \min_{\mathbf{a}} \left\| \mathbf{f_i} - \sum_{j=1}^{i-1} \mathbf{a}_j \mathbf{f_j} - \sum_{j=i+1}^{n_t} \mathbf{a}_{j-1} \mathbf{f_j} \right\|_2^2, \tag{2.20}$$

where n_t is the number of feature vectors belonging to the set, $\|\mathbf{v}\|_2^2$ is the square Euclidean norm of vector $\mathbf{v}$, and $\mathbf{a} \in \mathbb{R}^{n_t - 1}$ is a vector that contains the coefficients of the linear combination. The uniqueness equals zero when the feature vector is a linear combination of the remaining vectors, and it equals one when the vector is orthogonal to, and thus completely distinct from, the remaining feature vectors. If there are only two feature vectors in the set, then it can be shown that the uniqueness and the similarity have the following relation:

$$\mathrm{u}_1 = \mathrm{u}_2 = 1 - \mathbf{S}_{1,2}, \tag{2.21}$$

where $\mathbf{S}_{1,2}$ is the similarity between both feature vectors. As desired, the more similar the feature vectors are, the less unique they are.

2.3.2 General framework

We will now extend the framework such that a similarity-based index defined in the restricted framework can be applied to any meaningful similarity expressed

between any kind of objects. In the general framework, we have a set $\mathcal{D} = \{d_i \in D\}_{i=1}^{n_t}$ that contains data corresponding to each type, where n_t is the number of types. This data can be anything and it is therefore not restricted to vectors of real numbers. The similarity function $s : D \times D \to [0, 1]$ models the type similarity and $s(d_i, d_j)$ thus becomes the similarity between type i and type j. A concise representation of the similarity between each pair of types is the similarity matrix:

$$\mathbf{S} = \begin{bmatrix} s(d_1, d_1) & s(d_1, d_2) & \cdots & s(d_1, d_{n_t}) \\ s(d_2, d_1) & s(d_2, d_2) & \cdots & s(d_2, d_{n_t}) \\ \vdots & \vdots & \ddots & \vdots \\ s(d_{n_t}, d_1) & s(d_{n_t}, d_2) & \cdots & s(d_{n_t}, d_{n_t}). \end{bmatrix}. \tag{2.22}$$

We map the general framework onto the restricted framework by assigning a vector of unit norm - which we call the feature vector - to each type such that the similarity between each pair of types equals the similarity - and thus the square inner product - of the assigned feature vectors. An index that we define in the restricted framework can now also be used in the general framework as follows: we first map the general framework onto the restricted framework by assigning the feature vectors, and we then apply the index to these feature vectors. Example 3 of Appendix 2.A shows by example how the general framework works by first defining a similarity function with respect to the dietary habits of birds and then calculating the uniqueness of these dietary habits. Not every index defined in the restricted framework is meaningful in the general framework, and therefore we explain in Appendix 2.C which condition an index must meet so that using it in the general framework makes sense. All the indices that we propose in this paper meet this condition.

It should be noted that for certain choices of the similarity function, there does not exist a corresponding feature mapping. This is not a problem because we do not wish to allow any function to be a similarity function. Similar to the fact that an equality relation should be reflexive, symmetric and transitive, a similarity function should meet certain properties in order to be meaningful. We state that a similarity function is meaningful if and only if there exists at least one corresponding feature mapping or, equivalently, if and only if the similarity function meets the following properties:

- The similarity of an element with itself equals one. Mathematically, if d belongs to D, then $s(d, d) = 1$. This property is equal to the reflexivity property of the equivalence relation: an element is always equivalent to itself, where equivalence corresponds to a similarity of one.
- The similarity is symmetric. Mathematically, if both d and d' belong to D, then $s(d, d') = s(d', d)$. This property is an extension of the symmetry property of the equivalence relations.

- The corresponding kernel k, defined as $k(d, d') = \sqrt{s(d, d')}$, is positive semidefinite. The positive semidefiniteness of the corresponding kernel means that it is possible to assign a vector containing real numbers to each type such that the inner product of each pair of vectors equals the square root of the similarity between the corresponding types (see Mercer [28]). Mathematically, there has to exist a feature mapping φ that maps the raw data onto real vectors such that for all d and d' belonging to D the following holds:

$$k(d, d') = \sqrt{s(d, d')} = \varphi(d)^\intercal \varphi(d'). \tag{2.23}$$

  The vector $\mathbf{f} = \varphi(d)$ becomes the feature vector corresponding to the type with data d. This property is an extension of the transitivity property of the equivalence relation: if two types are equivalent if and only if their similarity equals 1, then this property implies the transitivity property of the corresponding equivalence relation.

Because the inner product is symmetric, the third property implies the second property. The first property is needed to ensure that the feature vectors mentioned in the third property are of unit norm. The following example illustrates the importance of the positive semidefiniteness of the corresponding kernel. Suppose that the type similarities equal

$$\mathbf{S} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \tag{2.24}$$

If type 1 is equal to type 2 and type 2 is equal to type 3, then type 1 must be equal to type 3. The similarity matrix, however, states that type 3 is completely different from type 1 because $\mathbf{S}_{3,1} = 0$. The similarity matrix cannot be the result of a meaningful similarity function because it is not possible to assign a feature vector to each type such that $f_1 = f_2$, $f_2 = f_3$ and f_3 is orthogonal to f_1. The similarity matrix is thus, as desired, not allowed in our framework. It can be shown that the following similarity matrix:

$$\mathbf{S}' = \begin{bmatrix} 1 & 0 & 0.9 \\ 0 & 1 & 0.9 \\ 0.9 & 0.9 & 1 \end{bmatrix} \tag{2.25}$$

is also not allowed, and it is thus not possible in our framework that two completely distinct types are 90% similar to the same third type. The maximal similarity that two completely distinct types can have with regard to a third type is 50%.

The uniqueness vector $\mathrm{u} \in \mathbb{R}^{n_t}$ corresponding to a set of types is defined as the vector for which element i expresses the uniqueness of type i within the set. To

Table 2.5: The ecological meaning of similarity and uniqueness, and the implications of these concepts with respect to the feature vectors.

Mathematical expression	Ecological meaning	Implications with respect to the feature vectors
$s(d_i, d_j) = 0$	Type i and type j are completely distinct	$\mathbf{f_i}$ is orthogonal to $\mathbf{f_j}$
$s(d_i, d_j) = 1$	Type i and type j are equivalent	$\mathbf{f_i} = \mathbf{f_j}$
$\mathrm{u}_i = 0$	Type i is not unique within the set of types	$\mathbf{f_i}$ can be expressed as a linear combination of the other feature vectors
$\mathrm{u}_i > 0$	Type i is to some extent unique within the set of types	$\mathbf{f_i}$ cannot be expressed as a linear combination of the other feature vectors
$\mathrm{u}_i = 1$	Type i is completely distinct from the other types	$\mathbf{f_i}$ is orthogonal to the other feature vectors

calculate the uniqueness of each type we can, as mentioned before, map the general framework on the restricted by assigning the feature vectors, and then calculate the uniqueness of these feature vectors by applying Equation (2.20). Alternatively, the uniqueness vector can directly be computed as follows:

$$\mathrm{u} = \odot^{-1}\mathrm{diag}\left(\left(\odot^{\frac{1}{2}}\mathbf{S}\right)^{-1}\right), \tag{2.26}$$

where $\mathrm{diag}(\mathbf{A})$ is the vector containing the main diagonal elements of $\mathbf{A}$, and $\odot^p\mathbf{A}$ is the element-wise power of $\mathbf{A}$, i.e., $(\odot^p\mathbf{A})_{ij} = \mathbf{A}_{ij}^p$. Appendix 2.D proves that Equation (2.26) holds. We can only evaluate Equation (2.26) when $\odot^{\frac{1}{2}}\mathbf{S}$ is invertible and therefore we also explain in Appendix 2.D how the uniqueness vector can be calculated directly when $\odot^{\frac{1}{2}}\mathbf{S}$ is not invertible. A summary of the concepts of similarity and uniqueness is given in Table 2.5.

2.3.3 Notations

We represent a group of types by a type model T. The model T is the singleton $T = (\mathcal{D})$, where $\mathcal{D} = \left\{d_i\right\}_{i=1}^{n_t}$ is a set of data and d_i is the data corresponding to

type i. This data (e.g., phylogenetic or functional data) is needed to determine the similarity between each pair of types when computing the diversity or other indices. A group of entities additionally depends on the relative abundances and therefore an entity model is represented by the double $E = (T, g)$, where $T = (\mathcal{D})$ models the types and g is a function such that $g(d)$ expresses the relative abundance of the type with data d. This means that $g(d_i)$ equals $\mathbf{p}_i$, the relative abundance of type i, and $g(d)$ equals 0 when $d \notin \mathcal{D}$. If we want to refer to the diversity of a group of entities represented by model E, we can now use the simple notation ${}^q\mathrm{D}(E)$. Because we will often take the union of a number of groups, we define in Table 2.6 two mathematical operations, namely addition of type models and weighted addition of entity models, that denote this union. Example 4 and 5 of Appendix 2.A illustrate these notations. The kernel matrix and the weighted kernel matrix are defined as

$$\mathbf{\Omega} = \odot^{\frac{1}{2}}\mathbf{S}, \;\; \mathbf{\Omega}(\mathbf{p}) = \odot^{\frac{1}{2}}\mathbf{S}(\mathbf{p}), \tag{2.27}$$

respectively, where $\mathbf{p} \in \mathbb{R}^{n_t}$ is the abundance vector, $\mathbf{S}(\mathbf{p}) \in \mathbb{R}^{n_t \times n_t}$ is the weighted similarity matrix defined as $\mathbf{S}(\mathbf{p}) = \mathbf{PSP}$, $\mathbf{P} \in \mathbb{R}^{n_t \times n_t}$ is a diagonal matrix with $\mathbf{P}_{ii} = \mathbf{p}_i$ for $i \in \{1, \ldots, n_t\}$, and $\odot^{\frac{1}{2}}(\mathbf{A})$ is the element-wise square root of $\mathbf{A}$, i.e., $(\odot^{\frac{1}{2}}(\mathbf{A}))_{ij} = \sqrt{\mathbf{A}_{ij}}$. We define these matrices for two reasons. First, in the definition of nearly every index in this paper, there occurs the element-wise square root of either the similarity matrix or the weighted similarity matrix, and creating a separate symbol for these matrices simplifies the formulas. The OLC uniqueness vector (see Equation (2.26)), for example, now becomes

$$\mathrm{u} = \odot^{-1}\mathrm{diag}(\mathbf{\Omega}^{-1}). \tag{2.28}$$

Second, our framework is inspired by the literature on kernel-based methods (see Schölkopf and Smola [29]), and this literature makes use of the kernel matrix instead of the similarity matrix. By defining the kernel matrix in our framework, the position of our indices among these kernel-based methods becomes more clear. Because we are of the opinion that the importance of notation is underestimated in the literature regarding diversity and related indices, we show in Appendix 2.E how our notation improves clarity and conciseness when expressing the decomposition in alpha and beta of the equivalence-based diversity.

2.4 Optimal lossy compression and entropy

Compression algorithms reduce the space needed to store data by converting it into a new format. In information theory, there are two types of compression: lossy and lossless compression. With lossless compression (e.g., zipping a file), all information is recovered when converting the data back to the original format. Lossy compression (e.g., saving an image as a low quality JPEG file, or reducing

Table 2.6: Two operations that denote the union of groups of types and entities.

Operation	Addition of type models
Notation	$T = \sum_{i=1}^{n_w} T_i$, where $T_i = (\mathcal{D}_i)$ for $i \in \{1, \ldots, n_w\}$
Definition	$T = (\mathcal{D})$, where $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_{n_w}$. The sum of type models is thus a type model that contains all the types that belong to at least one of the summed models.
Ecological Meaning	T models the types of the pooled group whose subgroups are represented by the type models $\{T_i\}_{i=1}^{n_w}$.
Operation	Weighted Addition of entity models
Notation	$E = \sum_{i=1}^{n_w} \mathbf{w}_i E_i$, where $E_i = (T_i, g_i)$ for $i \in \{1, \ldots, n_w\}$
Definition	$E = (T, g)$, where $T = \sum_{i=1}^{n_w} T_i$ and $g(d) = \sum_{i=1}^{n_w} \mathbf{w}_i g_i(d)$. The weighted sum of entity models is thus an entity model, and the relative abundance of a type in this entity model equals the weighted sum of the type's relative abundances in the summed models.
Ecological Meaning	E models the types and the relative abundances of the pooled group whose subgroups are represented by the entity models $\{E_i\}_{i=1}^{n_w}$. The weight vector $\mathbf{w} \in \mathbb{R}^{n_t}$ contains the relative sizes of the subgroups, i.e., $\mathbf{w}_i$ is the relative size of subgroup i.

the dimensionality of data by projecting it on a few principal components), on the other hand, is irreversible because it results in a loss of information. The type-neutral Shannon entropy is related to lossless compression in the sense that it expresses a lowerbound on the expected number of bits that are needed to describe the type of a randomly drawn entity. For instance, if we have a group that contains two animals of different species, then we need one bit to describe the type. The fact that the two species may be similar (e.g., phylogenetically related) doesn't reduce this number of bits. In the case of lossy compression, however, similarity starts playing a role. If we always say that the type is species 1, which corresponds to a description length of zero bits, then we are right in case the individual indeed belongs to species 1 and we are almost right in case the individual belongs to a related species. The more similar the

two species are, the smaller the error. The entropy proposed in this section expresses the trade-off between the description length and the approximation error when compressing the type representation optimally. Because this includes lossy compression, the similarity between the types plays a role as desired.

2.4.1 Compression algorithm

We will now explain how to compress a set of feature vectors optimally. We want to preserve the similarity between the types as much as possible. Changing the basis of the feature vectors to a new orthonormal basis does not change the similarities, while translating the origin does. Therefore, similar to Principal Component Analysis (PCA), the compression algorithm will express the feature vectors with respect to a new orthonormal basis, and, opposed to PCA, the algorithm will not mean center the feature vectors. Before we explain the compression algorithm in more detail, we need to define the description length and the approximation error, where the approximation error is a measure for the loss in information. The description length of a feature set representation is the number of coordinates that the feature vectors have with respect to the corresponding basis. The approximation error of the compressed feature set $\mathcal{F}' = \{\mathbf{f'}_i \in \mathbb{R}^{n_f}\}_{i=1}^{n_t}$ with respect to the uncompressed feature set $\mathcal{F} = \{\mathbf{f_i} \in \mathbb{R}^{n_f}\}_{i=1}^{n_t}$ is defined as

$$S = \sum_{i=1}^{n_t} \mathbf{p}_i \|\mathbf{f_i} - \mathbf{f'}_i\|_2^2, \tag{2.29}$$

where $\mathbf{f'}_i$ is the compressed representation of the type i with respect to the original basis, and $\mathbf{p}_i$ is the relative abundance of type i. The approximation error thus equals the expected square distance between the uncompressed and the compressed feature vectors of a randomly drawn entity. Note that if this square distance in small, then the resulting similarities will also be close to each other.

Given a set of feature vectors $\mathcal{F}$, an abundance vector $\mathbf{p}$ and a positive integer N, our compression algorithm calculates the N-truncated Karhunen-Loève (KL) representation, which is the best approximation with description length N, as follows:

- First, it calculates the second moment matrix $\boldsymbol{\rho}$ which is defined as

$$\boldsymbol{\rho} = \sum_{i=1}^{n_t} \mathbf{p}_i \mathbf{f_i} \mathbf{f_i}^\intercal = [\sqrt{\mathbf{p}_1}\mathbf{f_1}, \ldots, \sqrt{\mathbf{p}_{n_t}}\mathbf{f_{n_t}}][\sqrt{\mathbf{p}_1}\mathbf{f_1}, \ldots, \sqrt{\mathbf{p}_{n_t}}\mathbf{f_{n_t}}]^\intercal = \mathbf{F}\mathbf{P}\mathbf{F}^\intercal, \tag{2.30}$$

  where $\mathbf{F} = [\mathbf{f_1}, \ldots, \mathbf{f_{n_t}}]$, and $\mathbf{P} \in \mathbb{R}^{n_t \times n_t}$ is a diagonal matrix with $\mathbf{P}_{ii} = \mathbf{p}_i$. The algorithm does not subtract the mean feature vector and therefore

the second moment matrix is not equal to the covariance matrix of the feature vectors.

- Second, it diagonalizes the second moment matrix

$$\boldsymbol{\rho} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\intercal}, \tag{2.31}$$

where $\mathbf{U}$ is an orthogonal matrix, and $\boldsymbol{\Lambda}$ is a diagonal matrix with diagonal elements in nonincreasing order.

- Third, it changes the basis to $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^{n_t}$ where $\mathbf{u}_i$ is the i-th column of $\mathbf{U}$. The feature set $\mathcal{F}_{\text{KL}} = \{\mathbf{f}_{\mathbf{KL},\mathbf{i}} = \mathbf{U}^{\intercal}\mathbf{f}_{\mathbf{i}}\}_{i=1}^{n_t}$ contains the representation of the feature vectors with respect to the basis $\mathcal{U}$. We call this representation the KL representation because our algorithm is similar to PCA and PCA is the empirical version of the KL expansion [30, 31].

- Lastly, it truncates the KL representation. The N-truncated KL representation $\mathcal{F}_{\text{KL}}^{(N)}$ is the representation for which (1) the feature vectors of the KL representation are truncated such that only the first N coordinates remain and (2) all but the first N basis vectors in $\mathcal{U}$ are removed.

In Appendix 2.F.1 we prove that of all representations with description length N, the N-truncated KL representation is the one that minimizes the approximation error. Appendix 2.F.2 shows that the approximation error of the N-truncated KL representation equals $1 - \sum_{i=1}^{N} \boldsymbol{\lambda}_i$, where $\boldsymbol{\lambda}_1 \geq \cdots \geq \boldsymbol{\lambda}_{n_t}$ are the eigenvalues of the weighted kernel matrix $\boldsymbol{\Omega}(\mathbf{p})$. Example 7 in Appendix 2.A illustrates this algorithm.

2.4.2 OLC entropy

Ideally, an entropy that reflects the trade-off between the approximation error and the description length has the following properties:

- If the types are completely distinct - and the similarity matrix thus equals the identity matrix - then the entropy equals the equivalence-based entropy.

- If for any N, two groups of entities cause the same approximation error when compressing their representation to description length N, then both groups have the same entropy. In other words, the entropy is fully determined by these approximation errors. For example, if the N-truncated approximation error of both groups equals 0.5, 0.2 and 0 for N equal to 1, 2 and 3, respectively, then both groups have the same entropy.

- If for any N, group 1 doesn't cause a smaller approximation error than group 2 when compressing the representation to description length N, then the entropy of group 1 is higher than or equal to the entropy of group 2. For example, if the N-truncated approximation error of group 1 equals 0.5, 0.2 and 0 for N equal to 1, 2 and 3, respectively, and the ones of group 2 equal 0.4, 0.1 and 0, then the entropy of group 1 equals at least the entropy of group 2.

In Appendix 2.F.3 we prove that the OLC entropy, which we define below, is the only entropy index that satisfies all of the above properties. For $q \geq 0$, the OLC entropy ${}^{q}\mathrm{H}_{\mathrm{OLC}}$ of entity model E when using similarity function s, is defined as

$$ {}^{q}\mathrm{H}_{\mathrm{OLC}}(E, s) = {}^{q}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = \frac{1}{1-q} \ln\left(\mathrm{tr}(\mathbf{\Omega}(\mathbf{p})^{q})\right), \tag{2.32} $$

where $\mathrm{tr}(\mathbf{A})$ denotes the trace of matrix $\mathbf{A}$, and $\mathbf{\Omega}$ and $\mathbf{p}$ are the kernel matrix and the abundance vector corresponding to the group of entities, respectively. This entropy can be expressed as follows with respect to the eigenvalues $\boldsymbol{\lambda}_1 \geq \cdots \geq \boldsymbol{\lambda}_{n_t}$ of $\mathbf{\Omega}(\mathbf{p})$:

$$ {}^{q}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = \frac{1}{1-q} \ln\left(\sum_{i=1}^{n_t} \boldsymbol{\lambda}_i^{q}\right). \tag{2.33} $$

For q equal to 0, 1, and ∞, the OLC entropy is defined as

$$ {}^{0}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = \lim_{q \to 0^{+}} {}^{q}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = \ln(\mathrm{rank}(\mathbf{\Omega}(\mathbf{p}))) = \ln\left(\sum_{i=1}^{n_t} \boldsymbol{\lambda}_i^{0}\right), \tag{2.34} $$

$$ {}^{1}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = \lim_{q \to 1} {}^{q}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = -\mathrm{tr}(\mathbf{\Omega}(\mathbf{p}) \ln(\mathbf{\Omega}(\mathbf{p}))) = -\sum_{i=1}^{n_t} \boldsymbol{\lambda}_i \ln(\boldsymbol{\lambda}_i), \tag{2.35} $$

$$ {}^{\infty}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = \lim_{q \to \infty} {}^{q}\mathrm{H}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) = -\ln(\boldsymbol{\lambda}_1). \tag{2.36} $$

Intuitively, the OLC entropy can be interpreted as follows. Each coordinate of the KL representation represents a part of the feature vectors. We define the explanatory power of a coordinate as how much the approximation error increases when we remove this coordinate from the representation. It can be shown that, in the case of the KL representation, this increase in approximation error is independent of which other coordinates have already been removed. Moreover, each eigenvalue of the weighted kernel matrix equals the explanatory power of one of these coordinates. The OLC entropy is the type-neutral entropy applied to the eigenvalues of the weighted kernel matrix, and therefore it reflects

both how much coordinates are needed to express the feature vectors losslessly and how evenly the total explanatory power, which equals 1, is distributed over these coordinates. If a few coordinates explain a large portion of the feature vectors, then the entropy is low. If we need a lot of coordinates to explain even a relatively small portion of the feature vectors, then the entropy is high. This intuitive explanation is mathematically supported in Appendix 2.F.2. In Example 8 of Appendix 2.A we calculate the OLC entropy of the dietary habits of birds.

Methods that calculate the entropy of the eigenvalues of a positive semidefinite matrix have been proposed extensively in the past. Three of these methods inspired the proposed OLC entropy: the representation entropy of the KL expansion, the Von Neumann entropy and FS LS-SVM. A link between the approximation error due to truncation and the representation entropy, which is the entropy applied to the diagonal elements of the covariance matrix, was made by Watanabe et al. [32]. He showed that the KL expansion is optimal in the sense that (1) the N-truncated KL expansion minimizes the mean squared error over all representations with description length N, and (2) the KL expansion minimizes the representation entropy. Basically, the OLC entropy is the representation entropy of the KL expansion extended to our similarity-based framework, where the main difference is that we do not mean center our data. Note that the minimum representation entropy property does not explicitly state that the representation entropy of KL expansion expresses the trade-off between the approximation error and the description length when compressing the representation optimally, but nevertheless this property is related. As explained by Fayngold and Fayngold [33], the Von Neumann entropy [21] is the extension of classical entropy concepts to the field of quantum mechanics, and it represents the departure from a pure quantum state. It is zero in case of a pure quantum state and it is maximal in case of a perfectly mixed quantum state, which corresponds to all states being orthogonal to each other and having equal probabilities. The mathematical equivalence of the Von Neumann entropy and the Shannon OLC entropy, where the density matrix in the quantum mechanics framework corresponds to the second moment matrix in our framework, implies that all properties that are proven to hold for the Von Neumann entropy, also hold for the Shannon OLC entropy. FS LS-SVM [22] uses a basis set selection algorithm that maximizes the quadratic Rényi entropy of a Gaussian kernel matrix. This quadratic Rényi entropy is mathematically equivalent to the quadratic OLC entropy and furthermore, a Gaussian kernel corresponds to a meaningful similarity function in our framework. In contrast to the OLC entropy, however, this entropy is derived in the context of kernel density estimation and it thus has a different meaning. The fact that we combine kernels and entropy in order to quantify how different datapoints are, is inspired by this algorithm.

2.5 OLC diversity

Many authors in ecology acknowledge there is a need for consensus on a clear, standard terminology related to type diversity [34, 35]. According to Jost [36], the effective number of types provides a unified and intuitive interpretation of diversity. The effective number of types is therefore also called the true diversity. The OLC diversity is the extension of true diversity to our similarity-based framework. In addition to the type-neutral diversity, the OLC diversity also reflects how similar the types are.

2.5.1 Definition of the OLC diversity

The OLC diversity is the exponential function of the OLC entropy

$$ {}^{q}\mathrm{D}_{\mathrm{OLC}}(E,s) = \exp\left({}^{q}\mathrm{H}_{\mathrm{OLC}}(E,s)\right). \tag{2.37} $$

If all types are completely distinct, then the OLC diversity equals the equivalence-based true diversity. The OLC diversity is thus an extension of the equivalence-based true diversity to types that are not completely distinct. The interpretation of ${}^{q}\mathrm{D}_{\mathrm{OLC}}(E,s)$ for the different values of q is the following:

- The Hartley OLC diversity, ${}^{0}\mathrm{D}_{\mathrm{OLC}}(E,s)$, equals the size of the largest subset of types in which each type is to some extent unique. If all types are to some extent unique, then it equals the number of types.
- The Shannon OLC diversity, ${}^{1}\mathrm{D}_{\mathrm{OLC}}(E,s)$, is the OLC diversity that extends the Shannon equivalence-based diversity. It can be argued that the Shannon OLC diversity balances the influence of types with rare and common features. When $q < 1$, the influence of rare features is exaggerated. When $q > 1$, the influence of common features is exaggerated.
- The collision or quadratic OLC diversity, ${}^{2}\mathrm{D}_{\mathrm{OLC}}(E,s)$, reflects the average similarity between entities. More precisely, the inverse of the collision diversity equals the expected similarity between two randomly selected entities; $\left({}^{2}\mathrm{D}_{\mathrm{OLC}}(E,s)\right)^{-1} = \sum_{i=1}^{n_t}\sum_{j=1}^{n_t}\mathbf{p}_i\mathbf{p}_j\mathbf{S}_{ij}$ (see Appendix 2.G for a proof). The inverse of the collision diversity thus corresponds to Rao's index [8] and is an extension of the Simpson index [37].
- The min OLC diversity, ${}^{\infty}\mathrm{D}_{\mathrm{OLC}}(E,s)$, reflects how dominant the dominant feature direction is. More precisely, the inverse of the min OLC diversity equals the expected similarity between the dominant feature direction and an entity drawn at random, where the dominant feature direction is the direction that maximizes this expected similarity; $\left({}^{\infty}\mathrm{D}_{\mathrm{OLC}}(E,s)\right)^{-1} = \max_{\mathbf{f}}\sum_{i=1}^{n_t}\mathbf{p}_i s(\mathbf{f},\mathbf{f_i})$ (see Appendix 2.G for a proof). It can be shown

Table 2.7: Summary of the properties that the OLC diversity satisfies.

Interpretability: the OLC diversity equals the number of types when the similarity between each pair of types equals zero and the relative abundances of the types are equal.

Modularity: the diversity of a pooled group consisting of completely distinct subgroups is entirely determined by the diversity and the size of the subgroups.

Strong replication principle : the diversity of a pooled group that consists of n_w completely distinct subgroups with the same diversity d, equals n_w times d.

Twin property : if we add a type identical to an existing type and we distribute the relative abundance of the type over both itself and its twin, then the OLC diversity doesn't change.

Upperbound: for $q > 1$, the OLC diversity is upperbounded by $s_{\mathrm{m}}^{-\frac{q}{2(q-1)}}$ where s_{m} is the similarity between the two most dissimilar types. For $0 \leq q \leq 1$, no finite upperbound with respect to s_{m} exists. Note that this upperbound holds for any number of types, and thus also for the number of types going to infinity.

that the dominant feature vector is the eigenvector corresponding to the largest eigenvalue of the second moment matrix $\boldsymbol{\rho}$. The inverse of the min OLC diversity is an extension of the Berger-Parker index [38].

In Example 9 of Appendix 2.A we calculate the OLC diversity of the dietary habits of birds.

2.5.2 Properties of the OLC diversity

The properties of the OLC diversity are summarized in Table 2.7. For the proof of these properties we refer to Appendix 2.H. The interpretability property is important because it allows the OLC diversity to be interpreted as follows: an OLC diversity equal to x means that a set containing x complete distinct types where each type has relative abundance $1/x$ has the same diversity. It thus implies that the diversity is expressed in the effective number of types. The diversity of a pooled group consisting of completely distinct subgroups equals

$$ {}^{q}\mathrm{D}_{\mathrm{OLC}}\left(\sum_{i=1}^{n_w}\mathbf{w}_i E_i, s\right) = \left(\sum_{i=1}^{n_w}\mathbf{w}_i^q \left({}^{q}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right)\right)^{1-q}\right)^{\frac{1}{1-q}}, \tag{2.38} $$

where $\mathcal{E} = \{E_i\}_{i=1}^{n_w}$ is a set containing the models of the subgroups, and $\mathbf{w}$ is the weight vector with $\mathbf{w}_i$ the relative size of subgroup i in the pooled group. For q equal to 0, 1 and ∞, this diversity becomes

$$ {}^{0}\mathrm{D}_{\mathrm{OLC}}\left(\sum_{i=1}^{n_w}\mathbf{w}_i E_i, s\right) = \sum_{i=1}^{n_w} {}^{0}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right), \tag{2.39} $$

$$ {}^{1}\mathrm{D}_{\mathrm{OLC}}\left(\sum_{i=1}^{n_w}\mathbf{w}_i E_i, s\right) = \exp\left(\sum_{i=1}^{n_w} -\mathbf{w}_i \ln\left(\mathbf{w}_i\right)\right) \exp\left(\sum_{i=1}^{n_w}\mathbf{w}_i \ln\left({}^{1}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right)\right)\right) \tag{2.40} $$

$$ = {}^{1}\mathrm{D}(\mathbf{w})\, M_0\big({}^{1}\mathrm{D}\big(E_1, s\big), \ldots, {}^{1}\mathrm{D}\big(E_{n_w}, s\big)\big), \tag{2.41} $$

$$ {}^{\infty}\mathrm{D}_{\mathrm{OLC}}\left(\sum_{i=1}^{n_w}\mathbf{w}_i E_i, s\right) = \min_{i=1}^{n_w}\left(\frac{{}^{\infty}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right)}{\mathbf{w}_i}\right), \tag{2.42} $$

where ${}^{1}\mathrm{D}$ is the type-neutral Shannon diversity, and $M_0\big({}^{1}\mathrm{D}\big(E_1\big), \ldots, {}^{1}\mathrm{D}\big(E_{n_w}\big)\big)$ is the weighted geometric mean of the diversities with weights $\mathbf{w}_1, \ldots, \mathbf{w}_{n_w}$. As stated by the modularity property, the diversity of a pooled group consisting of completely distinct subgroups is thus entirely determined by the diversity and the size of the subgroups. The twin property says that treating one type as two identical types, or vice versa, shouldn't change the diversity. The upperbound property states that, for q greater than 1, the diversity will not exceed $s_{\mathrm{m}}^{-\frac{q}{2(q-1)}}$, no matter how many types there are. For q smaller than or equal to 1, on the other hand, the diversity will diverge to infinity for the number of types going to infinity. Example 10 in Appendix 2.A illustrates this upperbound.

2.6 OLC richness

In a similarity-based framework, the richness should reflect both the number of different types there are in a group of entities and how similar these types are. It should not depend on the relative abundances. There are different ways in which the type-neutral richness can be extended to a similarity-based framework. A popular way is to assign features to each type, and the richness then reflects the number of unique features there are in the set of types. We are not entirely satisfied with such a definition, as the following example shows. Suppose that we have two types with nucleotide sequences $\mathrm{S}_1 = \mathrm{AA}$ and $\mathrm{S}_2 = \mathrm{CC}$, where A and C denote adenine and cytosine, respectively. At each position in the

sequence, we choose each unique nucleotide to be a unique feature. The effective number of unique features in this set equals 2 ($= 0.5 \cdot 2 + 0.5 \cdot 2$), where 0.5 is the relative weight of each position. If we add two types with nucleotide sequences $S_3 = AC$ and $S_4 = CA$ to this set, then the effective number of features remains 2 despite the fact that the added types have a unique DNA code. In other words, not only the number of unique features but also the combinations in which they occur should influence the richness in this case. Note that if we had chosen the features differently, e.g., the first feature reflects the dietary habits of a species and the second feature reflects its breeding behavior, then often we still want a unique combination of features to strictly increase the richness, even when the individual features are not unique.

2.6.1 Definition of the OLC richness

For $q \geq 0$, we define the OLC richness as

$$
{}^{q}\mathrm{R}_{\mathrm{OLC}}(T, s) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathbf{\Omega}) = \max_{\mathbf{p}} {}^{q}\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega}, \mathbf{p}) \tag{2.43}
$$

$$
\text{subject to } \sum_{i=1}^{n_t} \mathbf{p}_i = 1, \mathbf{p} \geq 0,
$$

where $\mathbf{\Omega}$ is the kernel matrix corresponding to type model T and similarity function s. We choose the OLC richness to be the OLC diversity maximized in the relative abundances because of three reasons: we want the diversity to be the richness times the evenness, where the evenness is maximally one; the richness should satisfy a number of important properties (e.g., adding a type should not decrease the richness) and this definition satisfies most of these properties; and we want the richness to depend on the combinations of features in addition to the number of unique features. If we apply the OLC richness to the nucleotide sequence example mentioned at the beginning of this section, where the similarity between two types is chosen as the square of the proportion of positions at which the nucleotides are the same, then adding types 3 and 4 to the set containing types 1 and 2 increases the richness from 2 to 2.83. Pavoine and Izsák [9] maximize a quadratic diversity more general than the quadratic OLC diversity in order to calculate the originality of each species. However, they only use the abundance vector that maximizes the diversity and not the value of the maximized diversity itself. More information about the kind of optimization problem we are dealing with and how to solve it can be found in Appendix 2.I. Example 11 in Appendix 2.A illustrates this definition.

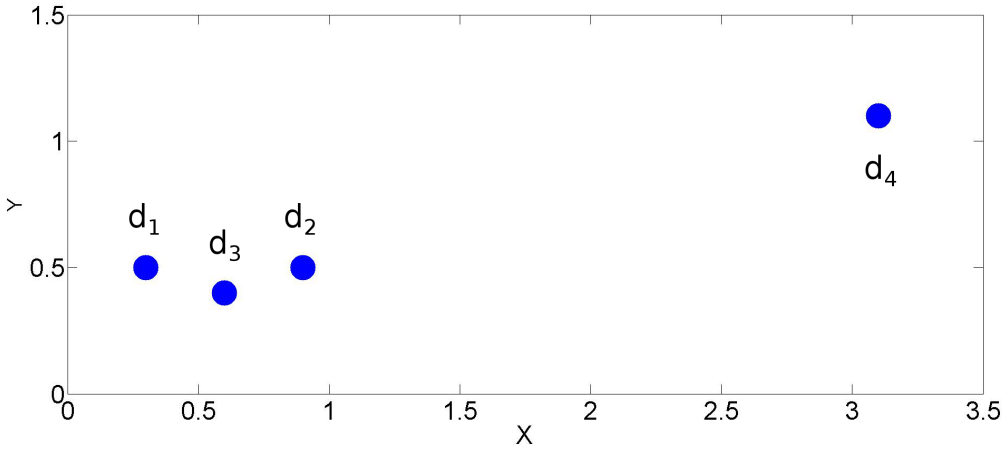


Figure 2.1: Visualisation of the datapoints $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$ and $\mathbf{d}_4$ defined in Example 1. The X-coordinate equals the first element of the vector $\mathbf{d}_i$ and the Y-coordinate equals the second element.

2.6.2 Properties of the OLC richness

Table 2.8 summarizes the properties that the OLC richness satisfies for different orders. All these properties are proven in Appendix 2.J. The first property is important for interpretability in the sense that a richness equal to x means that a type set containing x completely distinct types has the same richness. The second, third and fourth properties discuss how the richness changes when new types are added, and the fifth, sixth and seventh properties explain how the richness of the union of sets relates to the sum of their richnesses. Finally, the last property describes an upperbound of the OLC richness that is independent of the number of types, and this upperbound thus also holds for the number of types going to infinity. The Shannon OLC richness satisfies all of the first eight properties in Table 2.8, which makes it an interesting index for many applications. Note that the non-changing property and the additivity property are the richness variants of the twin property and the modularity property, respectively. Furthermore, if all the subgroups have the same richness, then the additivity property corresponds to the strong replication principle.

Example 1 (OLC richness when using a distance-based similarity). Suppose that we have four types corresponding to the following four datapoints:

$$\mathbf{d}_1 = \begin{bmatrix} 0.3 \\ 0.5 \end{bmatrix}, \mathbf{d}_2 = \begin{bmatrix} 0.9 \\ 0.5 \end{bmatrix}, \mathbf{d}_3 = \begin{bmatrix} 0.6 \\ 0.4 \end{bmatrix}, \mathbf{d}_4 = \begin{bmatrix} 3.1 \\ 1.1 \end{bmatrix}. \tag{2.44}$$

Figure 2.1 visualises these datapoints. We choose to model the similarities between the types as follows: $s(\mathbf{d}, \mathbf{d}') = \exp\left(-4\|\mathbf{d} - \mathbf{d}'\|_2^2\right)$. The similarity function corresponds to the following Gaussian kernel: $k(\mathbf{d}, \mathbf{d}') = \sqrt{s(\mathbf{d}, \mathbf{d}')} = \exp\left(-2\|\mathbf{d} - \mathbf{d}'\|_2^2\right)$. The greater the distance between the datapoints, the less similar the corresponding types are. The similarity function is meaningful

Table 2.8: Summary of the properties that ${}^{q}\mathrm{R}_{\mathrm{OLC}}$ satisfies for the different values of q. In the case of the top eight properties, a 'yes' means that the property holds and a 'no' means that it does not hold. In the case of the upperbound property, the upperbound is shown in the column of the corresponding order.

	$q = 0$	$q \in]0, 1]$	$q \in]1, \infty]$
Interpretability: the OLC richness equals the number of types when all types are completely distinct.	yes	yes	yes
Monotonically increasing: adding a type does not decrease the OLC richness.	yes	yes	yes
Strictly monotonically increasing: adding a type that is to some extent unique with respect to the type set strictly increases the OLC richness.	yes	yes	no
Non-changing: adding a type that already occurs in the type set, does not change the OLC richness.	yes	yes	yes
Subadditivity: the OLC richness of the union of type sets is not greater than the sum of the OLC richnesses of these type sets.	yes	yes	yes
Strict subadditivity: the OLC richness of the union of type sets that are not completely distinct is strictly smaller than the sum of their OLC richnesses.	no	yes	no
Additivity: the OLC richness of the union of completely distinct type sets equals the sum of their OLC richnesses.	yes	yes	yes
Upperbound: the OLC richness has the following upperbound with respect to s_{m}, the similarity between the two most dissimilar types:	∞	∞	$s_{\mathrm{m}}^{-\frac{q}{2(q-1)}}$

Table 2.9: We calculate the OLC richness of several sets of type vectors. The datapoints corresponding to the entities are visualised in Figure 2.1. The following similarity function is used: $s(\mathbf{d}_i, \mathbf{d}_j) = \exp\left(-4\|\mathbf{d}_i - \mathbf{d}_j\|_2^2\right)$. The set of datapoints for which we calculate the OLC richness is displayed in the leftmost column.

	${}^0\mathrm{R}_{\mathrm{OLC}}$	${}^1\mathrm{R}_{\mathrm{OLC}}$	${}^2\mathrm{R}_{\mathrm{OLC}}$	${}^\infty\mathrm{R}_{\mathrm{OLC}}$
$\mathcal{D}^{(1)} = \{\mathbf{d}_1, \mathbf{d}_2\}$	2	1.767	1.617	1.345
$\mathcal{D}^{(2)} = \{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3\}$	3	1.801	1.617	1.345
$\mathcal{D}^{(3)} = \{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3, \mathbf{d}_4\}$	4	2.801	2.617	2.345
$\mathcal{D}^{(4)} = \{\mathbf{d}_3, \mathbf{d}_4\}$	2	2.000	2.000	2.000
$\mathcal{D}^{(5)} = \{\mathbf{d}_1, \mathbf{d}_3\}$	2	1.355	1.197	1.100

because $s(\mathbf{d}, \mathbf{d}) = 1$, $s(\mathbf{d}, \mathbf{d}') = s(\mathbf{d}', \mathbf{d})$, and the Gaussian kernel is positive semidefinite. The similarity matrix and uniqueness vector corresponding to the types become

$$\mathbf{S} = \begin{bmatrix} 1.00 & 0.24 & 0.67 & 0.00 \\ 0.24 & 1.00 & 0.67 & 0.00 \\ 0.67 & 0.67 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 \end{bmatrix}, \quad \mathrm{u} = \begin{bmatrix} 0.23 \\ 0.23 \\ 0.01 \\ 1.00 \end{bmatrix}, \tag{2.45}$$

respectively. Table 2.9 shows the OLC richness of different subgroups. The Hartley OLC richness equals the number of types because all types are to some extent unique. The Shannon OLC richness always strictly increases when types are added, because the added types are partially unique. The collision and the min OLC richness, on the other hand, remain the same when type 3 - which has a non-zero uniqueness - is added to the group containing types 1 and 2. Type 4 is as good as completely different from the other types, and the richness thus increases by approximately one when this type is added. ■

2.6.3 Mutual richness

We define the mutual richness of T_1 and T_2 when using similarity function s as

$${}^q\mathrm{M}_{\mathrm{R,OLC}}(T_1, T_2, s) = {}^q\mathrm{R}_{\mathrm{OLC}}(T_1, s) + {}^q\mathrm{R}_{\mathrm{OLC}}(T_2, s) - {}^q\mathrm{R}_{\mathrm{OLC}}(T_1 + T_2, s). \tag{2.46}$$

Intuitively, when adding the richness of T_1 and T_2, the mutual richness is counted twice. Therefore, subtracting the richness of the union results in the mutual richness. The mutual richness is always positive because the richness is

subadditive. The main reason that we mention the mutual richness, is because it is used in the properties of the contribution vector.

2.7 OLC Contribution vector

A contribution vector expresses how much each type contributes to the observed richness. Ideally, the contribution vector has the following interpretation. We distribute the total richness over the types that cause the richness. If a certain part of the richness is caused by only one type (e.g., because the type has unique features), then this part is allocated entirely to that type. If a certain part of the richness is shared by multiple types (e.g., because the types have features in common), then this part is divided over these types. The more types share a certain part, the more types there are among which the richness will be distributed, and the less each type gets on average. The contribution of a type equals the sum of parts that are allocated to that type. If the richness caused by a type is mostly unique in the sense that it doesn't share the richness with other types, then the contribution is close to its maximum value, which equals one. If the richness caused by a type is shared by many other types, then the contribution can be close to its minimum value, which equals zero. In Section 2.7.2 we propose properties that correspond to this interpretation.

2.7.1 Definition of the OLC contribution vector

The OLC contribution of a type is defined as the OLC richness of the set of types multiplied by the relative abundance of the type when the OLC diversity is maximized. The reasoning behind this definition is the following: the higher the relative abundance of a type when maximizing the diversity, the more influence this type has on the diversity in this optimum, and the more it contributes to the richness. For example, if the abundance of a type is zero in the optimum, then the diversity in the optimum does not depend on this type, and therefore its contribution can be set equal to zero. A related approach to measure the originality of a species was done by Champely and Chessel [17] and later by Pavoine et al. [27] and Pavoine and Izsák [9]. These methods only investigate the quadratic version, however, and they do not multiply the relative abundances that maximize the diversity by the richness. For $q > 0$, the OLC contribution vector corresponding to a set of types is defined as the vector for which element i is the contribution of type i. The OLC contribution vector thus equals the following scalar-vector product:

$$\mathbf{{}^{q}c}_{\mathrm{OLC}}(T,s) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(T,s)\,\hat{\mathbf{p}} = \left(\max_{\mathbf{p}} {}^{q}\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega},\mathbf{p})\right)\left(\arg\max_{\mathbf{p}} {}^{q}\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega},\mathbf{p})\right), \tag{2.47}$$

where $\hat{\mathbf{p}} \in \mathbb{R}^{n_t}$ is the abundance vector that maximizes the OLC diversity. For $q = 0$, the OLC contribution vector is defined as ${}^{\mathbf{0}}\mathbf{c}_{\mathrm{OLC}}(T, s) = \lim_{q \to 0_+} {}^{\mathbf{q}}\mathbf{c}_{\mathrm{OLC}}(T, s)$. To calculate the contribution vector, we need to know the OLC richness and the abundance vector $\hat{\mathbf{p}}$ that maximizes the diversity. Both can be found by solving Optimization Problem (2.43). Example 12 in Appendix 2.A illustrates this definition.

2.7.2 Properties of the OLC contribution vector

Table 2.11 summarizes the properties that ${}^{\mathbf{q}}\mathbf{c}_{\mathrm{OLC}}$ satisfies. In Appendix 2.K we prove that these properties hold. The richness of the set is distributed over the types, and the part that is allocated to a certain type is referred to as the contribution of that type. Summing the type contributions should thus, as stated by the first property, result in the richness again. Ideally, a partially unique type strictly increases the richness, and therefore, according to the second property, it should have a non-zero contribution. The richness of the union of two sets can be divided into three parts: (1) the mutual richness, which equals ${}^{q}\mathrm{M}_R(T_1, T_2, s)$; (2) the richness unique to T_1, which equals ${}^{q}\mathrm{R}(T_1, s) - {}^{q}\mathrm{M}_R(T_1, T_2, s)$; and (3) the richness unique to T_2, which equals ${}^{q}\mathrm{R}(T_2, s) - {}^{q}\mathrm{M}_R(T_1, T_2, s)$. We don't specify how the mutual richness should be distributed over the two sets, but the contribution of T_1 should equal at least the richness unique to T_1 and maximally the richness unique to T_1 plus the mutual richness. Note that ${}^{q}\mathrm{R}(T_1, s) - {}^{q}\mathrm{M}_R(T_1, T_2, s) = {}^{q}\mathrm{R}(T_1 + T_2, s) - {}^{q}\mathrm{R}(T_2, s)$. In order to comply with the aforementioned reasoning, a contribution vector must meet the third property. Because we did not manage to prove that the third property holds for all orders, we also propose a weaker version. If a subset and its complementary subset are orthogonal, then, according to third property, the upper- and lowerbound of the contribution of that subset are both equal to its richness. Therefore the contribution of such a subset must equal its richness as stated by the fourth property. Finally, the twin property implies that representing two identical types by one type, or vice versa, shouldn't change the contribution of the other types in the set. If the third property would hold for the Shannon OLC contribution vector, then the Shannon OLC contribution vector would satisfy all the properties. The quadratic OLC contribution vector meets the third property, but it does not satisfy the second property.

Example 2 (Contribution vector when using a distance-based similarity). Table 2.10 shows the contribution of the types described in Example 1 towards the OLC richness. The Shannon OLC contribution of each type is nonzero. The quadratic and min OLC contribution of type 3 is zero despite the fact that the type is partially unique. In this example, the contribution of a subset is always smaller than the richness of that subset where the OLC richness of the subsets is shown in Table 2.9. The subset $\mathcal{D}^{(2)} = \{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3\}$ is approximately completely distinct from its complementary subset $\{\mathbf{d}_4\}$ and

the sum of the contributions of the data vectors belonging to $\mathcal{D}^{(2)}$ are therefore approximately equal to the richness of $\mathcal{D}^{(2)}$. We want to emphasize that different similarities give different results. If we use the following similarity function instead:

$$s(\mathbf{d}, \mathbf{d}') = \exp\left(-4\|\mathbf{d} - \mathbf{d}'\|_2\right), \tag{2.48}$$

then the contribution vector becomes ${}^{\mathbf{1}}\mathbf{c}_{\mathrm{OLC}} = [0.64, 0.63, 0.32, 0.99]^\intercal$ and the Shannon OLC richness equals 2.58 . The contributions of types 1, 2 and 4 are lower and the contribution of type 3 is higher. For those familiar with kernels, this similarity function corresponds to a Laplacian kernel. The similarity function is meaningful because $s(\mathbf{d}, \mathbf{d}) = 1$, $s(\mathbf{d}, \mathbf{d}') = s(\mathbf{d}', \mathbf{d})$ and a Laplacian kernel is positive semidefinite. ■

2.8 OLC average contribution and OLC evenness

In this section we will show how the OLC diversity can be decomposed into three components: the number of types, the OLC average contribution and the OLC evenness. The OLC average contribution is defined as the average contribution of the types with respect to the OLC richness

$${}^{q}\bar{\mathrm{c}}_{\mathrm{OLC}}(T, s) = \frac{1}{n_t}\sum_{i=1}^{n_t} {}^{\mathbf{q}}\mathbf{c}_{\mathrm{OLC},i} = \frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}(T, s)}{n_t}. \tag{2.49}$$

The OLC average contribution also equals the richness divided by the number of types and thus expresses how much of the richness is lost due to the similarities between the types. The OLC average contribution does not meet the twin

Table 2.10: We calculate the OLC contribution vector of a set of types. The datapoints corresponding to the types are visualised in Figure 2.1. The following similarity function is used: $s(\mathbf{d}, \mathbf{d}') = \exp\left(-4\|\mathbf{d} - \mathbf{d}'\|_2^2\right)$. The datavector of the type to which the contribution corresponds is shown in the left column.

	${}^{\mathbf{0}}\mathbf{c}_{\mathrm{OLC}}$	${}^{\mathbf{1}}\mathbf{c}_{\mathrm{OLC}}$	${}^{\mathbf{2}}\mathbf{c}_{\mathrm{OLC}}$	${}^{\infty}\mathbf{c}_{\mathrm{OLC}}$
$\mathbf{d}_1$	1	0.797	0.809	0.673
$\mathbf{d}_2$	1	0.797	0.809	0.673
$\mathbf{d}_3$	1	0.207	0.000	0.000
$\mathbf{d}_4$	1	1.000	1.000	1.000

Table 2.11: Summary of the properties ${}^{\mathbf{q}}\mathbf{c}_{\mathrm{OLC}}$ satisfies for the different values of q. A 'yes/no' is used if we did not manage to prove that the property nor manage to find a counterexample. A subset is complementary to another subset if it contains all the elements of the set that are not members of the other subset.

	q = $[0, 1]$	q = 2	q = $]1, 2[\cup]2, \infty]$
Partitioning: the sum of the contributions of the types belonging to the set equals the richness of the set.	yes	yes	yes
Non-zero contribution: a type with a non-zero uniqueness has a non-zero contribution.	yes	no	no
Consistent with richness of subsets: the sum of the contributions of the types belonging to a subset equals at most the richness of that subset, and is greater than or equal to the total richness minus the richness of the complementary subset.	yes/no	yes	yes/no
Consistent with richness of orthogonal subsets: the sum of the contributions of the types belonging to a subset equals the richness of that subset if the subset is completely distinct from its complementary subset.	yes	yes	yes
Twin property: If we represent a type by two identical types then the contributions of the types that differ from this type do not change.	yes	yes	yes

property and should therefore only be used for purely informative purposes. We define the OLC evenness as follows:

$$ {}^{q}\mathrm{E}_{\mathrm{OLC}}(E,s) = \frac{{}^{q}\mathrm{D}_{\mathrm{OLC}}(E,s)}{{}^{q}\mathrm{R}_{\mathrm{OLC}}(E,s)}. \tag{2.50} $$

The OLC evenness describes how much of the diversity is lost due to not having the abundances that maximize the diversity. For $q = 2$, this becomes an extension of the Simpson measure of evenness [39], which equals the Simpson index multiplied by the number of types. For $q = 1$, it is an extension of SHE-analysis [40] that decomposes diversity in a richness and evenness component. We can now decompose the OLC diversity as follows:

$$ {}^{q}\mathrm{D}_{\mathrm{OLC}}(E,s) = n_t \cdot {}^{q}\bar{\mathrm{c}}_{\mathrm{OLC}}(T,s) \cdot {}^{q}\mathrm{E}_{\mathrm{OLC}}(E,s). \tag{2.51} $$

Note that both the OLC average contribution and the OLC evenness range from zero to one. Example 13 in Appendix 2.A illustrates this decomposition.

2.9 Decomposition in alpha and beta diversity

Alpha, beta, and gamma diversity measures are basic tools in ecology and conservation biology. First we will cite the five axioms that a meaningful decomposition in alpha and beta should satisfy, as stated by Jost [26]:

1. Alpha and beta should be free to vary independently; a high value of the alpha component should not, by itself, force the beta component to be high (or low), and vice versa.
2. A given number should denote the same amount of diversity or uncertainty, whether it comes from the alpha component, the beta component, or the gamma component, so that a diversity index could be meaningfully partitioned into within-community and among-community components.
3. Alpha is some type of average of the diversity indices of the communities or samples that make up the region.
4. Gamma must be completely determined by alpha and beta.
5. Alpha can never be greater than gamma.

Although the above axioms make sense, we would nevertheless like to make some minor modifications. First, Equation A.7 in Appendix A of Jost [26] holds for any diversities of the subgroups, while Equation A.6 only holds when the diversities of the subgroups are the same. This means that Equation A.7 makes the implicit assumption that beta does not depend on the subgroup

diversities when the subgroups are completely distinct. It can be argued that this assumption is implied by the first axiom, but this implication is not at all trivial. Therefore we would prefer to explicitly state this assumption. Second, instead of saying that the diversity should denote the same amount of diversity for all components, it is, in our opinion, more to the point if we immediately state which kind of diversity is expected for each component (i.e., the effective number of types per subgroup for alpha, the effective number of subgroups for beta, and the effective number of types for gamma). Finally, the first axiom is somewhat vague because it is unclear which features of the pooled group are fixed and which can vary when determining the range. The modified axioms become:

1. The alpha diversity should be expressed in the effective number of types per subgroup, the beta diversity in the effective number of subgroups, and the gamma diversity in the effective number of types. Because the effective number of types equals the effective number of types per subgroup multiplied by the effective number of subgroups, this implies that gamma must be alpha multiplied by beta.

2. The alpha diversity only depends on the relative sizes and the diversities of the subgroups; it does not depend on differences between subgroups, e.g., on how dissimilar types belonging to different subgroups are. Moreover, the alpha diversity is some type of average of the diversities of the subgroups.

3. When the relative sizes of the subgroups are fixed, then (1) the beta diversity is maximized when the subgroups are completely distinct, in which case it becomes the type-neutral diversity applied to the relative sizes of the subgroups, and (2) the beta diversity is minimized when the subgroups are identical, in which case it becomes one. In other words, the range of the beta diversity depends on the number of subgroups and their relative sizes, and the exact value that the beta diversity has within this range depends on how different the subgroups are.

4. The alpha diversity can never be greater than the gamma diversity.

The modifications in the axioms do not change the resulting decomposition. In Appendix 2.L, we prove that there exists a meaningful decomposition of gamma diversity in alpha and beta diversity that satisfies the above axioms if and only if (1) the subgroups are equally weighted or (2) the order q equals 0 or 1. This means that only the Hartley and Shannon OLC diversity can be meaningful decomposed for unequally weighted subgroups. Note that in the type-neutral framework, the decomposition in alpha and beta diversity doesn't satisfy the axioms either when the communities are not equally weighted and when the order q is different from both 0 and 1. Our framework therefore does not further restrict the scenarios in which the decomposition can be applied meaningfully.

For equally weighted subgroups, i.e., $\mathbf{w}_i = \frac{1}{n_w}$ for $i \in \{1, \ldots, n_w\}$, the alpha diversity of a set of entity models $\mathcal{E} = \{E_i\}_{i=1}^{n_w}$ equals

$$
\begin{aligned}
{}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) &= M_{1-q}\left({}^{q}\mathrm{D}_{\mathrm{OLC}}\left(E_1, s\right), \ldots, {}^{q}\mathrm{D}_{\mathrm{OLC}}\left(E_{n_w}, s\right)\right) \\
&= \left(\sum_{i=1}^{n_w} \frac{1}{n_w} \left({}^{q}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right)\right)^{1-q}\right)^{\frac{1}{1-q}}, \qquad (2.52)
\end{aligned}
$$

where M_{1-q} is the generalised mean of order $1-q$. For q equal to 0 or 1, the alpha diversity for both equally weighted and unequally weighted subgroups becomes

$$
\begin{aligned}
{}^{0}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) &= M_1^{(u)}\left({}^{0}\mathrm{D}_{\mathrm{OLC}}\left(E_1, s\right), \ldots, {}^{0}\mathrm{D}_{\mathrm{OLC}}\left(E_{n_w}, s\right)\right) \\
&= \sum_{i=1}^{n_w} \frac{1}{n_w} {}^{0}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right), \qquad (2.53)
\end{aligned}
$$

$$
\begin{aligned}
{}^{1}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) &= M_0\left({}^{1}\mathrm{D}_{\mathrm{OLC}}\left(E_1, s\right), \ldots, {}^{1}\mathrm{D}_{\mathrm{OLC}}\left(E_{n_w}, s\right)\right) \\
&= \exp\left(\sum_{i=1}^{n_w} \mathbf{w}_i \ln\left({}^{1}\mathrm{D}_{\mathrm{OLC}}\left(E_i, s\right)\right)\right), \qquad (2.54)
\end{aligned}
$$

where $M_1^{(u)}$ is the unweighted arithmetic mean, and M_0 is the weighted geometric mean with weights $\mathbf{w}_1, \ldots, \mathbf{w}_{n_w}$. In all cases, the gamma and beta diversity are defined as

$$
{}^{q}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) = {}^{q}\mathrm{D}_{\mathrm{OLC}}\left(\sum_{i=1}^{n_w} \mathbf{w}_i E_i, s\right), \qquad (2.55)
$$

$$
{}^{q}\mathrm{D}^{\beta}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) = \frac{{}^{q}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s)}{{}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s)}. \qquad (2.56)
$$

If we replace the OLC diversity by the type-neutral diversity in the above formulas, then we get the decomposition of the type-neutral diversity, as proposed by Jost [26]. The high level reasoning behind both decompositions is thus the same: the gamma diversity is the diversity of the pooled group, the alpha diversity is either the weighted or the unweighted generalised mean of order $1-q$ of the subgroup diversities, and the beta diversity equals the gamma diversity divided by the alpha diversity. The only difference is the way in which the diversity of a group of entities is measured. Example 14 in Appendix 2.A illustrates this decomposition.

2.10 Comparison with related work on simulated data

In our simulated experiments, we focus on ultrametric phylogenetic trees. Nevertheless our methods can be used in many more applications. The OLC diversity and the OLC richness are, for example, also suited to quantify functional diversity and functional richness, respectively. For simplicity, we scale the trees such that they have unit length.

2.10.1 Used similarity

To apply our method on ultrametric phylogenetic trees of unit length, we first have to choose a similarity function to quantify phylogenetic similarity. We choose the similarity equal to the sum of lengths of the branches from which both species are descendants. The similarity function thus becomes

$$s(d_i, d_j) = \sum_{k \in (B_i \cap B_j)} \mathbf{x}_k, \tag{2.57}$$

where $\mathbf{x}_k$ is the length of branch k, and B_i and B_j are sets containing the indices of the branches from which species i and species j are descendants, respectively. This similarity function is meaningful in our framework: it satisfies $s(d_i, d_i) = 1$ because the tree has unit length; it is symmetric because $s(d_i, d_j) = s(d_j, d_i)$; and the corresponding kernel is positive semidefinite as proven in Appendix 2.M. Note that we can extend this similarity function to trees that aren't ultrametric as follows:

$$s'(d_i, d_j) = \frac{\sum_{k \in (B_i \cap B_j)} \mathbf{x}_k}{\sqrt{\sum_{l \in B_i} \mathbf{x}_l \sum_{m \in B_j} \mathbf{x}_m}}. \tag{2.58}$$

The latter similarity also satisfies $s(d_i, d_i) = 1$ and $s(d_i, d_j) = s(d_j, d_i)$. In Appendix 2.M we prove the positive semidefiniteness of the corresponding kernel.

2.10.2 Does skewness matter?

In the simulated experiments we will investigate the effect of the skewness of phylogenetic trees on existing phylogenetic diversity and richness indices. Here we discuss whether or not it is desired that these diversity and richness indices depend on the skewness. We claim that the richness should reflect the skewness for the following reason. Suppose that we have three species, called species A, species B and species C, and the phylogenetic distance between each pair of

species equals 1. To quantify this phylogenetic distance, we used a maximum likelihood estimator that assumes the JC69 model [23]. In Appendix 2.N we derive that, according to this JC69 model, the expected proportion of sites in the nucleotide sequences that are, on the one hand, equivalent for species B and species C and, on the other hand, different for species A and species C, equals 18.6%. Species C is thus expected to share some features with species B which it doesn't share with species A. Adding species C to the set that contains only species A should thus increase the richness more than adding species C to the set containing both species A and species B. Now suppose that instead, we have three species belonging to two phylogenetically unrelated classes - one of the species belongs to the first class and the remaining two species belong to the second class - and the phylogenetic distance between species belonging to the same class always equals the same value d. If we add an additional species belonging to the first class, then, similar to the aforementioned case, the richness should increase more than when we add an additional species belonging to the second class. In other words, the case in which the tree becomes the least skewed, results in the highest richness. Note that we use the JC69 model to simplify the calculations in the appendix. More complex models, such as the JTT model [24] and the LG model [25], result in the same conclusions. The reasoning can also be explained intuitively and independent of the used model; if all species are equally related and we have a group of thousand species, then it is not unlikely that a new species has some features which it doesn't share with one of these species but nevertheless shares with another one. The higher the number of species in the set, the more features and combinations of features of the added species are expected to already occur, and the less the richness should therefore increase when we add this species.

In the case of diversity, things become more complex. We first propose an index similar to the lineage-based index proposed by Chao et al. [6] that is defined with respect to nucleotide sequences instead of phylogenetic trees. We call this index the nucleotide diversity. For each position in the sequences, we calculate the type-neutral diversity of the nucleotides at that position, and we then take an average similar to the one used in Chao et al. [6] to combine these diversities. The nucleotide diversity equals

$$ {}^{q}\mathrm{D}_{\mathrm{Nu}} = \left(\sum_{i=1}^{n_S} \frac{1}{n_S} \left(f_{G,i}^{q} + f_{A,i}^{q} + f_{T,i}^{q} + f_{C,i}^{q} \right) \right)^{\frac{1}{1-q}}, \tag{2.59} $$

where n_S is the length of each sequence, and $f_{X,i}$ is the frequency of nucleotide X at position i. We can extend this diversity to phylogenetic trees by replacing the actual frequencies at the positions by the, according to the used model (e.g., JC69 model), expected frequencies at the positions:

$$ {}^{q}\mathrm{D}_{\mathrm{Nu}} = \left(\sum_{f_g, f_a, f_t, f_c} p(f_g, f_a, f_t, f_c) \left(f_g^q + f_a^q + f_t^q + f_c^q \right) \right)^{\frac{1}{1-q}}, \tag{2.60} $$

where $p(fg, fa, ft, fc)$ is the probability that the frequencies of guanine, adenine, thymine, and cytosine at a certain position equal f_g, f_a, f_t and f_c, respectively. Note the sum of the frequencies must equal one for this probability to be nonzero. In Figure 2.2 we apply the nucleotide diversity to two trees, one of which is more skewed than the other. For $q = \infty$, this index is not very useful in the sense that it always equals one when applied to phylogenetic trees. For $q = 2$, we can rewrite the nucleotide diversity as follows:

$$ {}^{2}\mathrm{D}_{\mathrm{Nu}} = \frac{1}{\sum_i^{n_t} \sum_j^{n_t} \mathbf{p}_i \mathbf{p}_j \mathbf{S}_{ij}}, \tag{2.61} $$

where $\mathbf{S}_{ij}$ is the proportion of nucleotides of species i that are shared with species j, or vice versa. The quadratic nucleotide diversity thus only depends on the nucleotides that are shared pairwise; higher order similarities (e.g., how many nucleotides species A, species B and species C have in common) do not matter. Because both trees in Figure 2.2 are equivalent from the perspective of pairwise similarities, they also have the same quadratic diversity. For the order q equal to 0 or 1, higher order similarities do matter. Suppose, for example, that we have the following two sets of nucleotide sequences, which are from a pairwise perspective identical (two different species share one nucleotide in both sets):

$$ \mathcal{S} = \{\mathrm{S}_1 = \mathrm{CGG}, \mathrm{S}_2 = \mathrm{GCG}, \mathrm{S}_3 = \mathrm{GGC}\}, \tag{2.62} $$

$$ \mathcal{S}' = \{\mathrm{S}'_1 = \mathrm{GCC}, \mathrm{S}'_2 = \mathrm{GAA}, \mathrm{S}'_3 = \mathrm{GTT}\}, \tag{2.63} $$

then, when all species have equal relative abundances, the nucleotide diversities equal

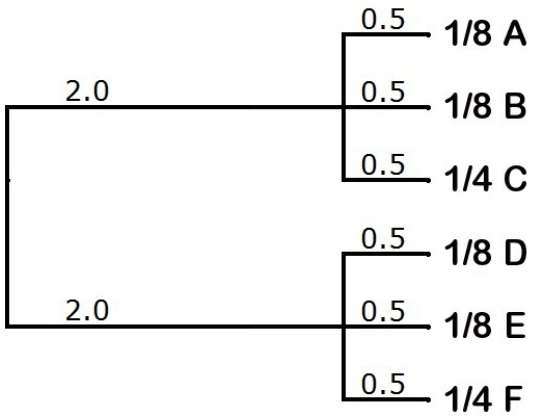
$$ {}^{0}\mathrm{D}_{\mathrm{Nu}}(\mathcal{S}) = \mathrm{R}_{\mathrm{Nu}}(\mathcal{S}) = \frac{1}{3}(2 + 2 + 2) = 2, \tag{2.64} $$

$$ {}^{1}\mathrm{D}_{\mathrm{Nu}}(\mathcal{S}) = M_0(1.89, 1.89, 1.89) = 1.89, \tag{2.65} $$

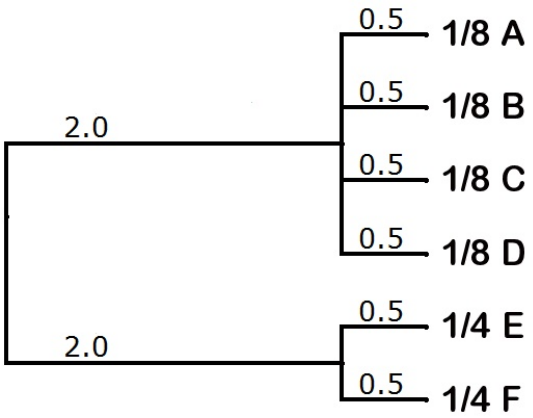
$$ {}^{0}\mathrm{D}_{\mathrm{Nu}}(\mathcal{S}') = \mathrm{R}_{\mathrm{Nu}}(\mathcal{S}') = \frac{1}{3}(1 + 3 + 3) = 2.33, \tag{2.66} $$

$$ {}^{1}\mathrm{D}_{\mathrm{Nu}}(\mathcal{S}') = M_0(1, 3, 3) = 2.08, \tag{2.67} $$

where M_0 is the geometric mean. The set $\mathcal{S}'$ has both more common and more rare nucleotides. The Richness and Shannon diversity give, compared to the quadratic diversity, relatively more weight to rare nucleotides and therefore assign a higher value to $\mathcal{S}'$. For the same reason, the trees in Figure 2.2 have a different nucleotide richness and nucleotide Shannon diversity. FTBL and the lineage-based diversity indices proposed by Chao et al. [6] use the branches in the phylogenetic tree that are shared by two or more species as a proxy for their shared features, and then calculate the diversity or richness with respect



(a) Least skewed tree, denoted by t_u. Its nucleotide diversities equal: ${}^0\mathrm{D}_{\mathrm{Nu}}(t_u) = \mathrm{R}_{\mathrm{Nu}}(t_u) = 2.92$, ${}^1\mathrm{D}_{\mathrm{Nu}}(t_u) = 2.50$, ${}^2\mathrm{D}_{\mathrm{Nu}}(t_u) = 2.21$, ${}^\infty\mathrm{D}_{\mathrm{Nu}}(t_u) = 1$.



(b) Most skewed tree, denoted by t_s. Its nucleotide diversities equal: ${}^0\mathrm{D}_{\mathrm{Nu}}(t_s) = \mathrm{R}_{\mathrm{Nu}}(t_s) = 2.90$, ${}^1\mathrm{D}_{\mathrm{Nu}}(t_s) = 2.49$, ${}^2\mathrm{D}_{\mathrm{Nu}}(t_s) = 2.21$, ${}^\infty\mathrm{D}_{\mathrm{Nu}}(t_s) = 1$.

Figure 2.2: Two trees that have a different skewness, but for which all other parameters, such as the relative size of the animal classes and the relative abundances of the species, are the same. Each branch is accompanied by its length. At the right of the tips of the tree are the relative abundances and the names of the species. We assume that the JC69 model is used to estimate the trees.

to these features. They do not model the third and higher order similarities correctly, and this is the reason they do not take into account the skewness of the tree. For example, in the tree shown in Figure 2.2a, they wrongly assume that all features shared by species A and species C are also shared by species B and species C. Moreover, even when applied correctly, we are of the opinion that the nucleotide diversity or a similar index should not be used because they only reflect how often each feature occurs and not in which combinations they occur. If we have two equally abundant species with DNA sequences $\mathrm{S}_1 = \mathrm{AA}$ and $\mathrm{S}_2 = \mathrm{CC}$, for instance, then the nucleotide diversity equals 2. If we add individuals belonging to the species with sequences $\mathrm{S}_3 = \mathrm{AC}$ and $\mathrm{S}_4 = \mathrm{CA}$ to this group such that all four species become equally abundant, then the nucleotide diversity remains 2 for all orders - which is not desired. We did not find a scientific explanation of why also taking into account the feature combinations wouldn't cancel or reverse the influence of the skewness. Personally, we are of the opinion that how evenly the species are distributed over the classes should influence the Shannon diversity. In the more extreme case, we prefer to have 500 species belonging to each class over 998 species belonging to one class and 2 species belonging to the other class.

2.10.3 Comparison of phylogenetic diversity indices

We will now apply different diversity indices to the trees visualised in Figure 2.3. Table 2.12 shows these diversities. We start with discussing the **min-diversity**. The inverse of ${}^{\infty}\mathrm{D}_{\mathrm{OLC}}(E,s)$, the min OLC diversity, equals the similarity between the dominant feature direction and an individual chosen at random. The inverse of ${}^{\infty}\mathrm{D}^{Z}(\mathbf{p})$, the min ASD, equals the expected similarity between the dominant species and an individual chosen at random. If the similarity in the OLC framework equals the similarity in the AS framework (which is the case in our experiments), then both measures are equivalent except for one important difference: the dominant feature direction in the AS framework is restricted to feature directions that correspond to species, while the dominant feature direction in OLC framework can be any normalized feature vector belonging to the feature space. If we take tree 1, for example, then species D is the dominant species and the expected similarity between an individual chosen at random and the dominant species equals 0.4 (= 1/2.5). The dominant feature direction, on the other hand, is the normalised average of the feature vectors of species A, B and C. Its expected similarity with an entity selected at random, equals 0.45 (= 1/2.21). The extra freedom in our framework will cause a higher value for the dominance of the dominant feature direction and thus a lower min-diversity, as we can verify in Table 2.12. The inverse of ${}^{\infty}\bar{\mathrm{D}}(T)$ equals the maximal value of a_i, the abundance descended from branch i, for all branches belonging to the tree

$$ {}^{\infty}\bar{\mathrm{D}}(T) = \frac{1}{\max_{i=1}^{n_b} a_i}. \tag{2.68} $$

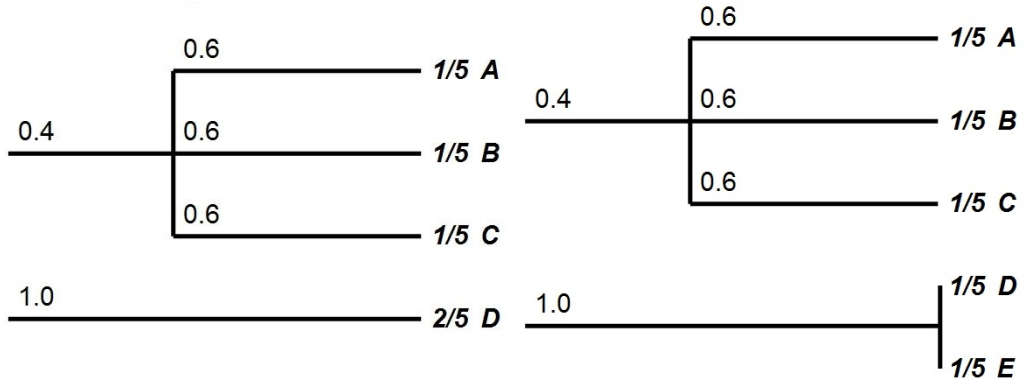
The index ${}^{\infty}\bar{\mathrm{D}}(T)$ doesn't seem that useful because it is even independent of the length of the branches. The **collision diversity** indices ${}^{2}\mathrm{D}_{\mathrm{OLC}}(E,s)$, ${}^{2}\bar{\mathrm{D}}(T)$ and ${}^{2}\mathrm{D}^{Z}(\mathbf{p})$ are equivalent. All three of them equal the expected total length of the branches from which two randomly drawn entities are descendants. Note that if we had chosen a different similarity for either the OLC diversity or ASD, then the indices would become different. For all three indices, the collision diversity reflects pairwise differences. To compare the **Shannon diversities**, we created the two trees shown in Figure 2.3c and Figure 2.3d. In Figure 2.3c, there descend five species from both branch 2 and branch 3. In Figure 2.3d, on the other hand, there only descend two species from branch 3 while eight species descend from branch 2. ${}^{1}\mathrm{D}_{\mathrm{OLC}}(E,s)$ gives a slightly greater diversity to tree 3 while ${}^{1}\bar{\mathrm{D}}(T)$ and ${}^{1}\mathrm{D}^{Z}(\mathbf{p})$ give the same diversity to both. Only the Shannon OLC diversity thus takes into account the skewness of the tree. As mentioned in Section 2.10.2, we are of the opinion that the skewness should be reflected by the Shannon diversity. We do not have a waterproof reasoning to support this claim, however. The **Hartley diversities** have the following meaning: ${}^{0}\mathrm{D}_{\mathrm{OLC}}(E,s)$ expresses the size of the largest subset in which each type is to some extent unique; the diversity ${}^{0}\bar{\mathrm{D}}(T)$ equals FTBL; and the inverse

of ${}^0\mathrm{D}^Z(\mathbf{p})$ is the harmonic mean of $(Zp)_i$. Because all three Hartley diversity indices measure a different kind of diversity, comparison makes little sense here. Note that we do not consider the Hartley OLC diversity as the richness variant of our criteria. We will compare FTBL with the OLC richness in Section 2.10.4. Finally, we discuss the **alpha and beta diversities**. In the case of tree 1, all subgroups contain only one species, which implies that their diversity equals one, and the alpha diversity should therefore also be one. In the case of tree 3, all subgroups have the same composition as the pooled group, and the beta diversity should thus be one. The decomposition proposed by Chiu et al. [20] satisfies neither of these conditions. Moreover, this decomposition can gives values for the alpha diversity which are smaller than one and values for the beta diversity that are greater than the gamma diversity. This undesired behavior of the decomposition is a result of the fact that it does not satisfy the third axiom mentioned by Jost [26], which states that alpha is some type of average of the diversities of the different subgroups. Instead, they propose an axiom that states that the beta diversity of N completely distinct assemblages should equal N, regardless of the weights and the diversities of individual assemblages. We do not understand why the beta diversity should be independent of the weights. The decomposition in alpha and beta of the OLC diversity, on the other hand, behaves as desired because it satisfies all the axioms mentioned by Jost [26].

2.10.4 Comparison of phylogenetic richness and contribution indices

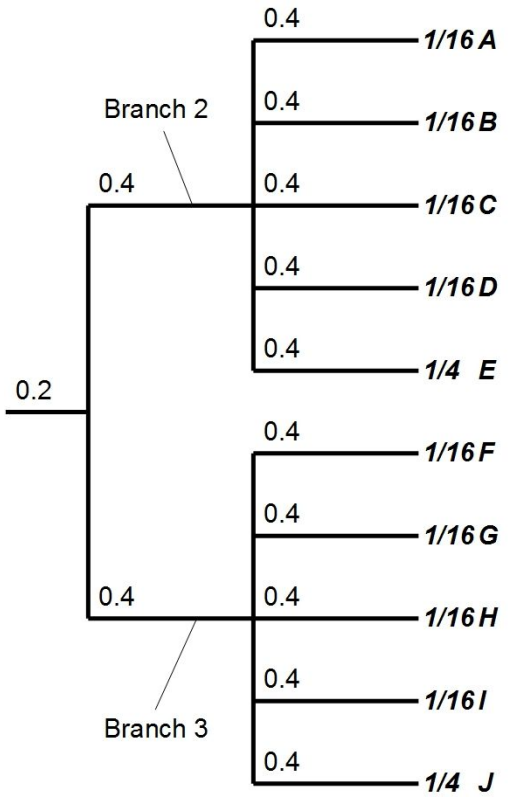
In Table 2.13 we show the results of applying richness and contribution indices to the trees visualised in Figure 2.4. For a richness index it is desired that tree 1 and tree 2 have the same richness, and that tree 3 and tree 4 have the same richness. Tree 1 and tree 2 should also have a higher richness than tree 3 and tree 4 because they are less skewed (see Section 2.10.2). The Shannon OLC richness is the only index in Table 2.13 that results in the desired ranking. FTBL doesn't give tree 1 and tree 2 a higher richness than tree 3 and tree 4. PSR doesn't give tree 3 and tree 4 the same richness and therefore does not meet the twin property.

It is desired that the contribution vectors of tree 1 and tree 2 are the same. Also, according to the twin property, species A, B, C, D and E in tree 3 should have the same contributions as species A, B, C, D and E in tree 4, respectively. In Table 2.13, only the Shannon OLC contribution vector has these desired properties. FP doesn't give the same contribution to species A, B, C, D and E in tree 3 and tree 4. ES doesn't cause the contribution vectors of tree 1 and tree 2 to be equal. The OLC average contribution, which quantifies the relative difference between the OLC richness and the number of species, changes after

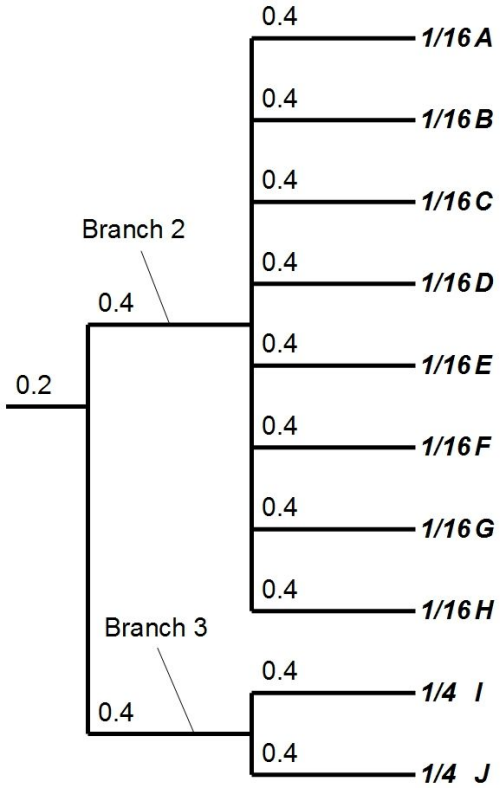


(a) Tree 1

(b) Tree 2



(c) Tree 3



(d) Tree 4

Figure 2.3: Simulated ultrametric phylogenetic trees are used to compare diversity and evenness indices. Tree 2 equals tree 1 where species D is duplicated and its abundance is distributed over the duplicates. Tree 2 thus represents the same group of entities as tree 1.

Table 2.12: The diversities corresponding to the phylogenetic trees shown in Figure 2.3. For the alpha and beta component of tree 1, we choose four subgroups, where each subgroup contains the individuals belonging to one species (each subgroup thus contains only one species). For the alpha and beta component of tree 3, we choose two subgroups that have the same relative abundances as the pooled group, and the relative sizes of the subgroups equal 0.8 and 0.2.

	gamma				alpha	beta	alpha	beta
	Tree 1	Tree 2	Tree 3	Tree 4	Tree 1		Tree 3	
${}^0\mathrm{D}_{\mathrm{OLC}}$	4	4	10	10	1	4	10	1
${}^1\mathrm{D}_{\mathrm{OLC}}$	3.03	3.03	3.30	3.27	1	3.03	3.30	1
${}^2\mathrm{D}_{\mathrm{OLC}}$	2.66	2.66	2.16	2.16	\	\	\	\
${}^\infty\mathrm{D}_{\mathrm{OLC}}$	2.21	2.21	1.55	1.55	\	\	\	\
${}^0\bar{\mathrm{D}}(T)$	3.2	3.2	5	5	1	3.2	5	1
${}^1\bar{\mathrm{D}}(T)$	2.91	2.91	3.03	3.03	0.95	3.07	2.50	1.21
${}^2\bar{\mathrm{D}}(T)$	2.66	2.66	2.16	2.16	0.89	2.99	1.59	1.36
${}^\infty\bar{\mathrm{D}}(T)$	1.6	1.6	1	1	0.63	2.54	0.63	1.60
${}^0\mathrm{D}^Z$	2.67	2.67	2.18	2.18	\	\	\	\
${}^1\mathrm{D}^Z$	2.66	2.66	2.17	2.17	\	\	\	\
${}^2\mathrm{D}^Z$	2.66	2.66	2.16	2.16	\	\	\	\
${}^\infty\mathrm{D}^Z$	2.5	2.5	2	2	\	\	\	\

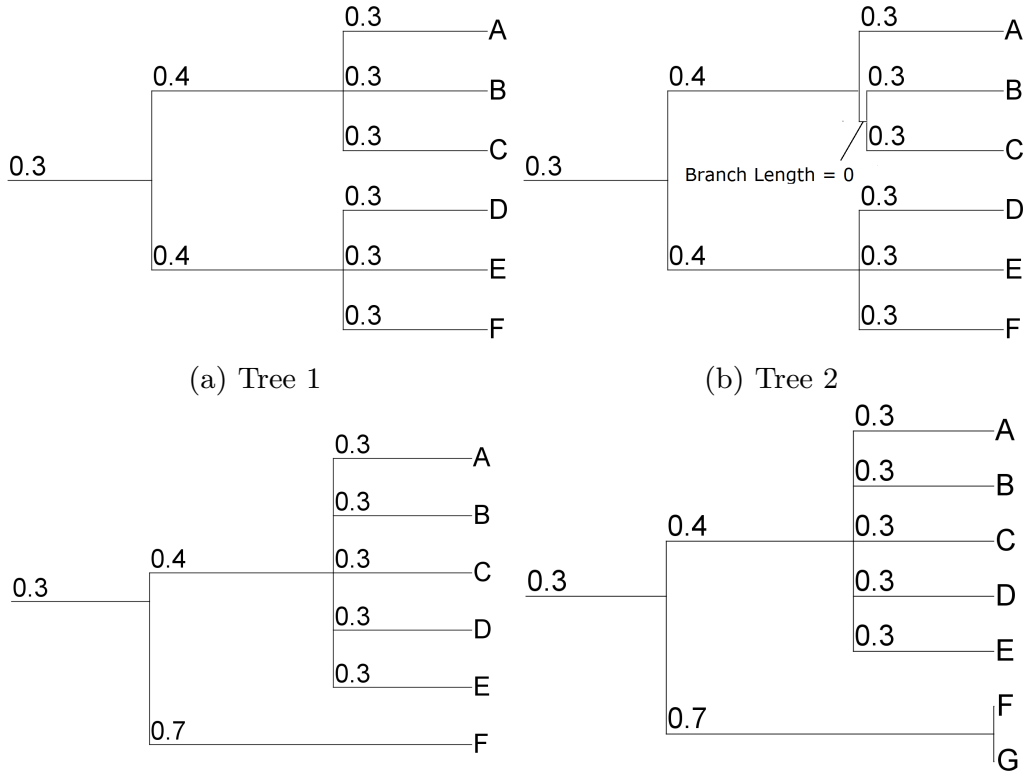
duplication because the number of species changes. The latter index should thus only be used to explain the difference between the number of species and the richness. One should be careful when basing actual decisions on this index.

2.10.5 Comparison of phylogenetic evenness measures

Table 2.14 applies a number of evenness measures to the trees visualised in Figure 2.3. PAE' and PSE have the issue that they assign a different evenness to tree 1 and tree 2, and they therefore do not meet the twin property. The OLC evenness of tree 4 is higher than the OLC evenness of tree 3 for the following reason: to maximize the OLC diversity of tree 4, species I and species J should have higher abundances than the remaining species, while in the case of tree 3, all species should have equal abundances. This is why the abundances of tree 4 cause the OLC diversity to be close to the OLC richness, while in tree 3 the difference between the OLC diversity and the OLC richness is greater. The smaller the relative difference between the OLC diversity and the OLC richness, the greater the OLC evenness, and the OLC evenness of tree 4 is thus higher.

Table 2.13: Different richness and contribution indices applied to the phylogenetic trees shown in Figure 2.4. In case the result is a vector, the first element corresponds to species A, the second to species B, etc.

	Tree 1	Tree 2	Tree 3	Tree 4
n_t	6	6	6	7
${}^1\mathrm{R}_{\mathrm{OLC}}$	2.54	2.54	2.35	2.35
${}^1\bar{\mathrm{c}}_{\mathrm{OLC}}$	0.42	0.42	0.39	0.34
${}^1\mathbf{c}_{\mathrm{OLC}}$	$\begin{bmatrix}0.42\\0.42\\0.42\\0.42\\0.42\\0.42\end{bmatrix}$	$\begin{bmatrix}0.42\\0.42\\0.42\\0.42\\0.42\\0.42\end{bmatrix}$	$\begin{bmatrix}0.32\\0.32\\0.32\\0.32\\0.32\\0.76\end{bmatrix}$	$\begin{bmatrix}0.32\\0.32\\0.32\\0.32\\0.32\\0.38\\0.38\end{bmatrix}$
PSR	3.24	3.24	2.60	3.33
FTBL	2.9	2.9	2.9	2.9
FP	$\begin{bmatrix}0.48\\0.48\\0.48\\0.48\\0.48\\0.48\end{bmatrix}$	$\begin{bmatrix}0.48\\0.48\\0.48\\0.48\\0.48\\0.48\end{bmatrix}$	$\begin{bmatrix}0.43\\0.43\\0.43\\0.43\\0.43\\0.75\end{bmatrix}$	$\begin{bmatrix}0.42\\0.42\\0.42\\0.42\\0.42\\0.39\\0.39\end{bmatrix}$
ES	$\begin{bmatrix}0.48\\0.48\\0.48\\0.48\\0.48\\0.48\end{bmatrix}$	$\begin{bmatrix}0.55\\0.45\\0.45\\0.48\\0.48\\0.48\end{bmatrix}$	$\begin{bmatrix}0.41\\0.41\\0.41\\0.41\\0.41\\0.85\end{bmatrix}$	$\begin{bmatrix}0.41\\0.41\\0.41\\0.41\\0.41\\0.43\\0.43\end{bmatrix}$



(a) Tree 1 (b) Tree 2

(c) Tree 3 (d) Tree 4

Figure 2.4: Simulated ultrametric phylogenetic trees that are used to compare different richness and contribution indices. Adding a branch with zero length to tree 1 results in tree 2. Tree 4 equals tree 3 except that one species is represented twice (species G equals species F).

Table 2.14: The OLC evenness, PAE' and PSE applied to the trees shown in Figure 2.3.

	Tree 1	Tree 2	Tree 3	Tree 4
${}^{1}E_{OLC}$	0.988	0.988	0.911	0.960
${}^{2}E_{OLC}$	0.997	0.997	0.951	0.988
${}^{\infty}E_{OLC}$	0.950	0.950	0.979	0.995
PAE'	1.09	1	1	1
PSE	0.83	0.78	0.60	0.60

2.10.6 Problems of distance-based functional diversity measures

Finally, we show that the distance-based functional diversity measures proposed by Chiu and Chao [10] do not meet the twin property. Suppose that we have species with the following relative abundances and phylogenetic and functional species distances:

$$\mathbf{p} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \ \mathbf{D} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{2.69}$$

then ${}^{q}\mathrm{D}(Q) = 2$, ${}^{q}\mathrm{MD}(Q) = 1$, and ${}^{q}\mathrm{FD}(Q) = 2$ for $0 \leq q \leq \infty$. If we now duplicate the first species such that we get the following distance matrix and abundance vector

$$\mathbf{p}' = \begin{bmatrix} 0.25 \\ 0.25 \\ 0.5 \end{bmatrix}, \ \mathbf{D}' = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}, \tag{2.70}$$

then the indices change into ${}^{q}\mathrm{D}(Q') = 2\sqrt{2}$, ${}^{q}\mathrm{MD}(Q') = \sqrt{2}$, and ${}^{q}\mathrm{FD}(Q') = 4$ for $0 \leq q \leq \infty$. This means that treating one species as two identical species can change these indices, which shouldn't be the case. Also, if the relative abundances and functional species distances are equal to

$$\mathbf{p} = \begin{bmatrix} p_1 \\ 1 - p_1 \end{bmatrix}, \ \mathbf{D} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{2.71}$$

then ${}^{q}\mathrm{FD}(Q') = 2$ for $0 \leq q \leq \infty$ regardless of the value of p_1. In other words, the relative abundances do not matter in the case of two species, while they should matter for $q > 0$.

2.11 Applications to real-world data

2.11.1 Butterfly population

Devries et al. [41] surveyed fruit-feeding butterfly populations in the canopy and understory at a site in the Ecuadorian rainforest. Table 2.15 shows the abundances of the six most abundant species in each subfamily as listed in this paper. We set the similarity between two butterflies equal to

- 0 when they belong to different subfamilies
- 1/3 when they belong belong to the same subfamily but are of different genera

Table 2.15: Abundances of the 6 most abundant species in each subfamily of the fruit-feeding nymphalid butterflies at a site in the Ecuadorian rainforest. Subfamily abbreviations: (C)= Charaxinae, (N) = Nymphalinae, (M) = Morphinae, (B) = Brassolinae, and (S) = Satyrinae. Because of the low relative abundance of Morphinae, only one species of this subfamily was included.

		Understory	Canopy	Total
Prepona laertes	(C)	15	0	15
Archaeoprepona demophon	(C)	14	37	51
Zaretis itys	(C)	25	11	36
Memphis arachne	(C)	89	23	112
Memphis oVa	(C)	21	3	24
Memphis xenocles	(C)	32	8	40
Catonephele acontius	(N)	22	89	111
Nessaea obrina	(N)	0	209	209
Nessaea hewitsoni	(N)	1	153	154
Smyrna blomfildia	(N)	197	28	225
Historis odius	(N)	106	8	114
Colobura dirce	(N)	238	438	676
Morpho achilles	(M)	0	98	98
Caligo illioneus	(B)	0	33	33
Caligo idiomenius	(B)	0	40	40
Catoblepia xanthus	(B)	0	42	42
Catoblepia berecynthia	(B)	0	44	44
Opsiphanes cassina	(B)	21	14	35
Opsiphanes invirae	(B)	27	5	32
Hermeuptychia hermes	(S)	12	141	153
Cissia proba	(S)	0	161	161
Cissa myncea	(S)	0	151	151
Cissia penelope	(S)	8	1612	1620
Pareuptychia occirhoe	(S)	1	419	420
Ypthimoides erigone	(S)	0	295	295
Total abundances		829	4062	4891

- 2/3 when they are congeneric but they don't belong to the same species
- 1 if they belong to the same species.

The OLC diversities and their decomposition into the number of types, the OLC average contribution and the OLC evenness are shown in Table 2.16. The decomposition in alpha and beta of the Shannon OLC diversity becomes: ${}^1\mathrm{D}^{\alpha}_{\mathrm{OLC}} = 5.50$, ${}^1\mathrm{D}^{\beta}_{\mathrm{OLC}} = 1.27$ and ${}^1\mathrm{D}^{\gamma}_{\mathrm{OLC}} = 6.96$. The weights used in this decomposition are equal to the relative sizes: 0.17 for the understory and 0.83 for the canopy. The canopy has a higher collision OLC richness than the understory, but because the abundances in the canopy are less evenly distributed, its collision OLC diversity is nevertheless lower. The canopy has a lower min OLC diversity, which means that it is dominated to a higher extent by a set of closely related species. This can be explained by the fact that the Cissia penelope accounts for 40% of the total abundances in the canopy.

2.11.2 Prosimian primates

We apply the Shannon OLC richness, the Shannon OLC contribution vector and the OLC uniqueness vector to the phylogeny of prosimian primates. The

Table 2.16: The OLC richnesses and OLC diversities of the butterfly population described in Table 2.15. The Shannon and the collision OLC diversity are decomposed in the number of species, the OLC average contribution and the OLC evenness.

	Understory	Canopy	Total
Number of species	16	24	25
${}^1\mathrm{R}_{\mathrm{OLC}}$	9.60	12.7	13.1
${}^2\mathrm{R}_{\mathrm{OLC}}$	7.22	9.17	9.34
${}^{\infty}\mathrm{R}_{\mathrm{OLC}}$	5.49	6.89	6.95
${}^1\mathrm{D}_{\mathrm{OLC}}$	5.32	5.54	6.96
${}^2\mathrm{D}_{\mathrm{OLC}}$	3.38	3.06	3.87
${}^{\infty}\mathrm{D}_{\mathrm{OLC}}$	2.03	1.86	2.23
${}^1\bar{\mathrm{c}}_{\mathrm{OLC}}$	0.60	0.53	0.53
${}^2\bar{\mathrm{c}}_{\mathrm{OLC}}$	0.45	0.38	0.37
${}^1\mathrm{E}_{\mathrm{OLC}}$	0.55	0.43	0.53
${}^2\mathrm{E}_{\mathrm{OLC}}$	0.47	0.33	0.41

tree shown in Figure 2.5 is from Vos [42]. It was also used in the context of deriving conservation priorities [43]. The figure shows that, as desired, both the contribution and the uniqueness of a species strongly depend on the extent to which the species is closely related to other species. The Daubentonia madagascariensis, for example, has both the highest contribution and the highest uniqueness, and it can be argued that it is the species that differs the most from other species. The species of the genus Lepilemur and Eulemur, on the other hand, have the lowest contribution and the lowest uniqueness, and they can be considered the most closely related to other species. Note that the sum of the contributions equals the richness.

2.11.3 Tree thinning

Shimatani [44] collected data from the overstory tree species in the Fred Russ experimental forest in Michigan. Similar to Chao et al. [6], we only consider the abundance data of block 4 for the thinned site and the un-thinned site. The phylogenetic tree and the abundances are visualized in Figure 2.6. A four-level (species, genus, family, subclass) taxonomic measure based on the Simpson index was used by Shimatani [44]. The taxonomic index states that thinning increases the diversity in contrast to the collision species-neutral true diversity, which states the opposite. Chao et al. [6] applied a lineage-based taxonomic and phylogenetic diversity index to the same data and concluded that the taxonomic diversity indeed gives different conclusions than the species-neutral true diversity, but the phylogenetic diversity ranks both sites similar to the species-neutral true diversity. We also apply our phylogenetic OLC diversity to this dataset. The OLC diversity and its decomposition are shown in Table 2.17. We get similar results as the lineage-based phylogenetic diversity; for both $q = 1$ and $q = 2$, the un-thinned site has higher Shannon and collision OLC diversities. Although our model ranks both sites similar to the type-neutral true diversity, the cause of this ranking differs. The higher Shannon and collision OLC diversities of the un-thinned site are caused by a higher richness, while the higher species-neutral true diversity of the un-thinned site is caused by a higher evenness.

2.12 Conclusion and Future Work

In this paper we proposed OLC diversity and OLC richness, an extension of true diversity and richness to a similarity-based framework, respectively. The OLC diversity is related to lossy compression in the sense that it reflects the trade-off between the approximation error and the description length when compressing the representation of a group of entities optimally. The OLC richness is the OLC diversity maximized in the relative abundances. To expose the underlying

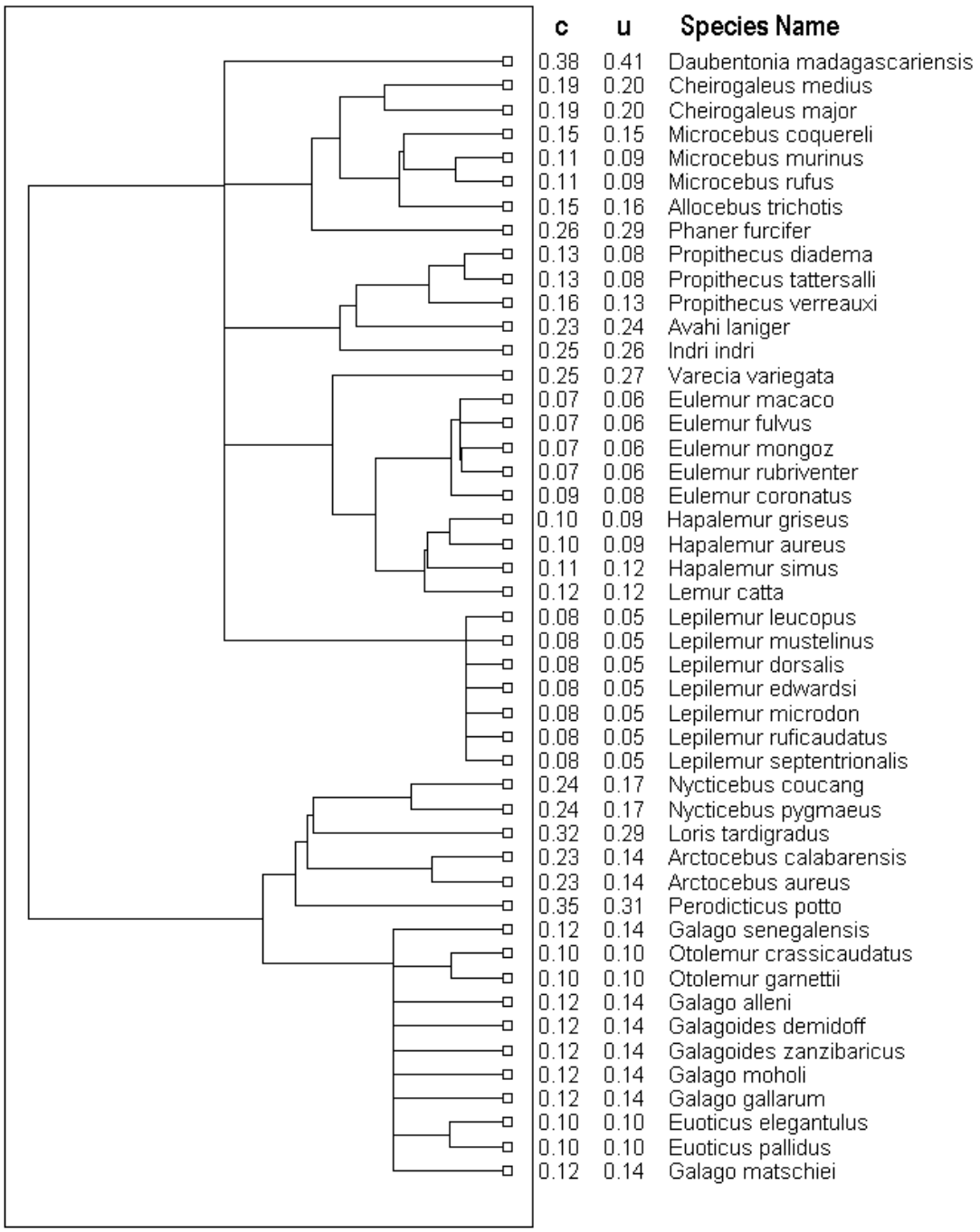
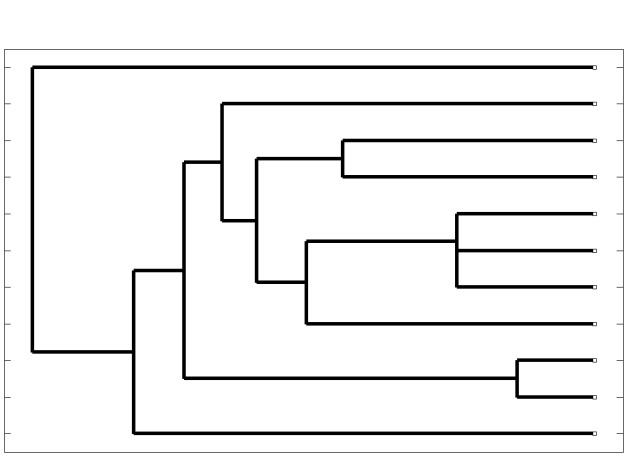


Figure 2.5: Phylogenetic tree of prosimian primates. The Shannon OLC richness of the tree equals 6.89. At the right of each leaf there is the Shannon OLC contribution (c), the OLC uniqueness of each type with respect to the other types (u), and the name of the species.



T	UT	Species Name
0.000	0.034	Sassafras albidum
0.076	0.041	PoPulus grandidentata
0.008	0.021	Ostrya virginiana
0.303	0.014	Quercus rubra
0.015	0.021	Ulmus americana
0.015	0.172	Ulmus rubra
0.008	0.000	Celtis occidentalis
0.379	0.345	Prunus serotina
0.136	0.124	Acer rubrum
0.045	0.228	Acer saccharum
0.015	0.000	Fraxinus american

Figure 2.6: Phylogenetic tree at the thinned and un-thinned site. At the right of each leaf there is the relative abundance at the thinned site (T), the relative abundance at the un-thinned site (UT) and the name of the species.

Table 2.17: The OLC diversity and its decomposition for the thinned site (T) and un-thinned site (UT). We also show the discrete true diversity, denoted by ${}^{q}\mathrm{D}$, for comparison.

	T (q=1)	UT (q=1)	T (q=2)	UT (q=2)
Number of species	10	9	10	9
${}^{q}\mathrm{D}$	4.97	5.66	3.81	4.55
${}^{q}\mathrm{R}_{\mathrm{OLC}}$	4.15	4.47	2.63	3.24
${}^{q}\mathrm{D}_{\mathrm{OLC}}$	2.83	3.02	1.94	2.05
${}^{q}\bar{\mathrm{c}}_{\mathrm{OLC}}$	0.42	0.50	0.26	0.36
${}^{q}\mathrm{E}_{\mathrm{OLC}}$	0.68	0.68	0.74	0.63

mechanisms that cause the diversity or richness, the following derived indices were proposed: (1) the OLC contribution vector quantifies the contribution of each type towards the OLC richness, (2) the OLC average contribution expresses how much of the richness is lost due to the similarity between the types, (3) the OLC evenness expresses how much of the diversity is lost due to not having the abundances that maximize the diversity, and (4) the decomposition in alpha and beta provides insight in the differences between subgroups. We either prove or show by experiment that the proposed indices satisfy a number of important properties and furthermore, that they overcome several problems that existing and commonly used indices have. In this paper we explained the general framework such that in future work we can focus on designing similarity

functions for specific applications. A problem of the phylogenetic similarity function used in this paper is the fact that it depends on the position of the root; it doesn't make sense that the actual similarity between species would depend on this position. Therefore, in a future paper, we are planning to propose a more advanced similarity function that is independent of this position.

2.13 Acknowledgements

Research supported by Research Council KUL: GOA/10/09 MaNet, PFV/10/002 (OPTEC), several PhD, postdoc & fellow grants; Flemish Government: IOF/KP/SCORES4CHEM; FWO: PhD, postdoc grants, projects: G.0588.09 (Brain-machine), G.0377.09 (Mechatronics MPC), G.0377.12 (Structured systems); IWT: PhD Grants, projects: SBO LeCoPro, SBO Climaqs, SBO POM, EUROSTARS SMART; iMinds 2013; Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012-2017); EU: FP7-EMBOCON (ICT-248940), FP7-SADCO (MC ITN-264735), ERC ST HIGHWIND (259 166), ERC AdG A-DATADRIVE-B (290923); COST: Action ICO806: IntelliCIS.

References

[1] Forest, F., R. Grenyer, M. Rouget, T. J. Davies, R. M. Cowling, D. P. Faith, A. Balmford, J. C. Manning, S. Proches¸ M. van der Bank, G. Reeves, T. A. J. Hedderson, and V. Savolainen. 2007. Preserving the evolutionary potential of floras in biodiversity hotspots. Nature 445.7129: 757–760.

[2] Cadotte, M. W., C.H. Albert and S.C. Walker, 2013. The ecology of differences: assessing community assembly with trait and evolutionary distances. Ecology Letters 16: 1234–1244.

[3] Ackerly, D.D. and W.K. Cornwell, 2007. A trait-based approach to community assembly: partitioning of species trait values into within- and among-community components. Ecology Letters 10: 135–145.

[4] Schleuter, D., M. Daufresne, F. Massol, and C. Argillier. 2010. A user's guide to functional diversity indices. Ecological Monographs 80.3: 469–484.

[5] Weitzman, M. L. 1992. On diversity. The Quarterly Journal of Economics 57: 363–405

[6] Chao, A., C.-H. Chiu, and L. Jost. 2010. Phylogenetic diversity measures based on Hill numbers. Philosophical Transactions of the Royal Society B: Biological Sciences 365.1558: 3599–3609.

[7] Leinster, T., C. A. Cobbold. 2012. Measuring diversity: the importance of species similarity. Ecology 93.3: 477–489.

[8] Rao, C. R. 1982. Diversity and dissimilarity coefficients: a unified approach. Theoretical Population Biology 21: 24–43.

[9] Pavoine, S., and J. Izsák. 2014. New biodiversity measure that includes consistent interspecific and intraspecific components. Methods in Ecology and Evolution 5: 165–172.

[10] Chiu, C. H., and A. Chao. 2014. Distance-based functional diversity measures and their decomposition: a framework based on Hill numbers. PloS one, 9(7): e100014.

[11] Faith, D. P. 2004. Conservation evaluation and phylogenetic diversity. Biological Conservation 61.1: 1–10.

[12] Faith, D. P., C.A.M. Reid, and J. Hunter. 2004. Integrating phylogenetic diversity, complementarity, and endemism for conservation assessment. Conservation Biology 18.1: 255–261.

[13] Helmus, M. R., T.J. Bland, C.K. Williams, and A.R. Ives. 2007. Phylogenetic measures of biodiversity. The American Naturalist 169.3: E68–E83.

[14] Walker, B., A. Kinzig, and J. Langridge. 1999. Original articles: plant attribute diversity, resilience, and ecosystem function: the nature and significance of dominant and minor species. Ecosystems, 2(2): 95–113.

[15] Petchey, O. L., and K. J. Gaston. 2002. Functional diversity (FD), species richness, and community composition. Ecology Letters 5: 402–411.

[16] Cadotte M. W., T. J. Davies, J. Regetz, S. W. Kembel, E. Cleland, and T. H. Oakley. 2010. Phylogenetic diversity metrics for ecological communities: integrating species richness, abundance and evolutionary history. Ecology Letters 13.1: 96–105.

[17] Champely, S., and D. Chessel. 2002. Measuring biological diversity using Euclidean metrics. Environmental and Ecological Statistics 9.2: 167–177.

[18] Redding, D. W., and A. Ø. Mooers. 2006. Incorporating evolutionary measures into conservation prioritization. Conservation Biology 20: 1670–1678.

[19] Isaac, N. J. B., S.T. Turvey, B. Collen, C. Waterman, and J.E. Baillie. 2007. Mammals on the EDGE: conservation priorities based on threat and phylogeny. PLoS One 2.3: e296.

[20] Chiu, C.-H., L. Jost, and A. Chao. 2013. Phylogenetic beta diversity, similarity, and differentiation measures based on Hill numbers. Ecological Monographs. In Press.

[21] Von Neumann, J. 1996. Mathematical foundations of quantum mechanics. Princeton university press 2.

[22] Suykens, J.A.K., T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. 2002. Least Squares Support Vector Machines. World Scientific, Singapore.

[23] Jukes, T. H., and C. R. Cantor. 1969. Evolution of Protein Molecules (21–132). Academic Press, New York.

[24] Jones, D. T., W. R. Taylor, and J. M. Thornton. 1992. The rapid generation of mutation data matrices from protein sequences. Computer applications in the biosciences: CABIOS 8.3: 275–282.

[25] Le, S.Q., and O. Gascuel. 2008. An improved general amino acid replacement matrix. Molecular biology and evolution 25.7: 1307–1320.

[26] Jost, L. 2007. Partitioning diversity into independent alpha and beta components. Ecology 88.10: 2427–2439.

[27] Pavoine, S., S. Ollier, and A. B. Dufour. 2005. Is the originality of a species measurable? Ecology Letters 8.6: 579–586.

[28] Mercer, J. 1909. Functions of positive and negative type and their connection with the theory of integral equations, Philosophical Transactions of the Royal Society A: 415–446.

[29] Schölkopf, B., and A. Smola. 2002. Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press.

[30] Karhunen L. 1946. Uber Lineare Methoden in der Wahrscheinlichkeitsrechnung. Annales Academia Scientiarum Fennicae, Series AI: Mathematica-Physica 37: 1–79.

[31] Loeve M. 1978. Probability theory, vol. II. Graduate texts in mathematics 46: 0–387.

[32] Watanabe, S., P.F. Lambert, C.A. Kulikowski, J.L. Buxton, and R. Walker. 1967. Evaluation and selection of variables in pattern recognition. Computer and Information Sciences 2: 91–122.

[33] Fayngold, M., and V. Fayngold. 2013. Quantum Mechanics and Quantum Information. Wiley.

[34] Tuomisto, H. 2010. A consistent terminology for quantifying species diversity? Yes, it does exist. Oecologia 164.4: 853–860.

[35] Moreno, C. E., and P. Rodríguez. 2010. A consistent terminology for quantifying species diversity? Oecologia 163.2: 279–282.

[36] Jost, L. 2006. Entropy and diversity. Oikos 113.2: 363–375.

[37] Simpson, E. H. 1949. Measurement of diversity. Nature 163: 688.

[38] Berger, W. H., and F. L. Parker. 1970. Diversity of planktonic Foramenifera in deep sea sediments. Science 168: 1345–1347.

[39] Smith, B., and J. B. Wilson. 1996. A consumer's guide to evenness indices. Oikos: 70–82.

[40] Buzas, M. A., and L.-A. C. Hayek. 1996. Biodiversity resolution: an integrated approach. Biodiversity Letters: 40–43.

[41] Devries, P. J., D. Murray, and R. Lande. 1997. Species diversity in vertical, horizontal, and temporal dimensions of a fruit-feeding butterfly community in an Ecuadorian rainforest. Biological journal of the Linnean Society 62.3: 343–364.

[42] Vos, R. A. 2006. Inferring large phylogenies: the big tree problem. Diss. Biological Sciences Department-Simon Fraser University.

[43] Steel, M., A. Mimoto, and A. Ø. Mooers. 2007. Hedging our bets: the expected contribution of species to future phylogenetic diversity. Evolutionary bioinformatics online 3: 237–244.

[44] Shimatani, K. 2001. On the measurement of species diversity incorporating species differences. Oikos 93.1: 135–147.

[45] Cortes, C., and V. Vapnik. 1995. Support-vector networks. Machine learning 20.3: 273–297.

[46] Schölkopf, B., A. Smola, and K.-R. Müller. 1997. Kernel principal component analysis. Artificial Neural Networks - ICANN'97 (583–588). Springer Berlin Heidelberg.

[47] Alzate, C., and J.A.K. Suykens. 2010. Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA. Pattern Analysis and Machine Intelligence, IEEE Transactions on 32.2: 335–347.

[48] Eckart, C., and G. Young. 1936. The approximation of one matrix by another of lower rank. Psychometrika 1.3: 211–218.

[49] Grant, M., and S. Boyd. 2008. Graph implementations for nonsmooth convex programs, Recent Advances in Learning and Control (a tribute to M. Vidyasagar): 95–110.

[50] Grant, M., and S. Boyd. 2012. CVX: Matlab software for disciplined convex programming, version 2.0 beta. http://cvxr.com/cvx.

[51] Schur, I. 1923. Über eine Klasse von Mittelbildungen mit Anwendungen die Determinanten-Theorie. Sitzungsber. Berlin. Math. Gesellschaft 22: 9–20.

[52] Horn, A. 1954. Doubly stochastic matrices and the diagonal of a rotation matrix. American Journal of Mathematics 76.3: 620–630.

[53] Hadamard, J. 1908. Mémoire sur le probléme d'analyse relatif à l'équilibre des plaques élastiques encastrées. Mémoires présentés par divers savants à l'Académie des Sciences 33.

Appendix

2.A Running example

Example 3 (Dietary habits of the birds on an island). Suppose that we want to determine a similarity matrix and abundance vector that reflect the dietary habits of the birds on an island. A quarter of the birds belong to species A and the remaining three-quarters belong to species B. Birds of species A feed for 40% on small seeds and for 60% on medium-sized seeds, and birds of species B feed for 80% on medium-sized seeds and for 20% on large seeds. We choose to model the similarity between the dietary habits of two species as follows

$$s(d_Y, d_Z) = (\sqrt{a_Y a_Z} + \sqrt{b_Y b_Z} + \sqrt{c_Y c_Z})^2, \tag{2.72}$$

where $d_X = \{a_X, b_X, c_X\}$ is the data corresponding to species X, and a_X, b_X and c_X represent the proportion of the nutrition of species X that are seeds of small, medium and large size, respectively. The similarity matrix and abundance vector become

$$\mathbf{S} = \begin{bmatrix} 1 & 0.48 \\ 0.48 & 1 \end{bmatrix}, \quad \mathbf{p} = \begin{bmatrix} 0.25 \\ 0.75 \end{bmatrix}. \tag{2.73}$$

If we assign the following feature vectors to the types

$$\mathbf{f_X} = \varphi(d_X) = \begin{bmatrix} \sqrt{a_X} \\ \sqrt{b_X} \\ \sqrt{c_X} \end{bmatrix}, \tag{2.74}$$

then

$$(f_Y^\intercal f_Z)^2 = (\sqrt{a_Y a_Z} + \sqrt{b_Y b_Z} + \sqrt{c_Y c_Z})^2 = s(d_Y, d_Z), \tag{2.75}$$

and the similarity between each pair of types thus equals the square inner product, and thus the similarity, of the assigned feature vectors. The feature vectors corresponding to the birds of species A and species B equal

$$\mathbf{f_A} = \begin{bmatrix} \sqrt{0.4} \\ \sqrt{0.6} \\ 0 \end{bmatrix}, \quad \mathbf{f_B} = \begin{bmatrix} 0 \\ \sqrt{0.8} \\ \sqrt{0.2} \end{bmatrix}, \tag{2.76}$$

respectively. The uniqueness of species A can now be calculated using Expression (2.20)

$$\mathrm{u}_A = \min_a \|\mathbf{f_A} - a\mathbf{f_B}\|_2^2 = \left\|\mathbf{f_A} - \sqrt{0.48}\mathbf{f_B}\right\|_2^2 = 0.52, \tag{2.77}$$

where $\hat{a} = \sqrt{0.48}$ optimizes the objective function in a. The uniqueness of species B also equals 0.52. Alternatively, we can calculate the uniqueness of both species using Expression (2.26)

$$\mathrm{u} = \odot^{-1}\mathrm{diag}\left(\left(\odot^{\frac{1}{2}}\mathbf{S}\right)^{-1}\right) = \begin{bmatrix}0.52\\0.52\end{bmatrix}. \tag{2.78}$$

Finally, we would like to note that the chosen similarity function is meaningful because $s(d,d) = 1$, $s(d,d') = s(d',d)$ and the corresponding kernel is positive semidefinite (Equation (2.74) expresses the corresponding feature mapping). ■

Example 4 (Model of the dietary habits of the birds on the island). The type model representing the dietary habits of the birds described in Example 3 becomes $T = (\mathcal{D})$ where

$$\mathcal{D} = \left\{d_A = \{0.4, 0.6, 0\}, d_B = \{0, 0.8, 0.2\}\right\}. \tag{2.79}$$

The entity model equals $E = (T, g)$ where

$$g(d) = \begin{cases}0.25 & \text{if } d = d_A,\\ 0.75 & \text{if } d = d_B,\\ 0 & \text{otherwise .}\end{cases} \tag{2.80}$$

■

Example 5 (Union of the dietary habits of the birds on two islands). Suppose that close to the island on which the birds described in Example 3 live, there is a smaller island on which birds of species B and species C live. Half of the birds on the new island belong to species B and the remaining half belong to species C. Birds of species C solely eat small seeds. On the new island there live four times less birds. The type model representing the second island becomes $T' = (\mathcal{D}')$ where

$$\mathcal{D}' = \{d_B = \{0, 0.8, 0.2\}, d_C = \{0, 0, 1\}\}, \tag{2.81}$$

and the corresponding entity model equals $E' = \{T', g'\}$ where

$$g'(d) = \begin{cases}0.5 & \text{if } d = d_B,\\ 0.5 & \text{if } d = d_C,\\ 0 & \text{otherwise .}\end{cases} \tag{2.82}$$

The type model $T'' = T + T'$ represents both islands where

$$T'' = (\mathcal{D}'')\,, \tag{2.83}$$

$$\mathcal{D}'' = \mathcal{D} \cup \mathcal{D}' = \{d_A = \{0.4, 0.6, 0\}, d_B = \{0, 0.8, 0.2\}, d_C = \{0, 0, 1\}\}\,. \tag{2.84}$$

The entity model corresponding to both islands equals $E'' = 0.8E + 0.2E'$ where

$$E'' = (T'', g'')\,, \tag{2.85}$$

$$g''(d) = 0.8g(d) + 0.2g'(d) = \begin{cases} 0.8 \cdot 0.25 + 0.2 \cdot 0 = 0.2 & \text{if } d = d_A, \\ 0.8 \cdot 0.75 + 0.2 \cdot 0.5 = 0.7 & \text{if } d = d_B, \\ 0.8 \cdot 0 + 0.2 \cdot 0.5 = 0.1 & \text{if } d = d_C, \\ 0 & \text{otherwise}\,. \end{cases} \tag{2.86}$$

■

Example 6 (Kernel and weighted kernel matrix corresponding to the dietary habits). The kernel and the weighted kernel corresponding to the dietary habits of the birds described in Example 3 become

$$\mathbf{\Omega} = \odot^{\frac{1}{2}} \begin{bmatrix} 1 & 0.48 \\ 0.48 & 1 \end{bmatrix} = \begin{bmatrix} 1 & \sqrt{0.48} \\ \sqrt{0.48} & 1 \end{bmatrix}, \tag{2.87}$$

$$\mathbf{\Omega}(\mathbf{p}) = \odot^{\frac{1}{2}} \left(\begin{bmatrix} 0.25 & 0 \\ 0 & 0.75 \end{bmatrix} \begin{bmatrix} 1 & 0.48 \\ 0.48 & 1 \end{bmatrix} \begin{bmatrix} 0.25 & 0 \\ 0 & 0.75 \end{bmatrix} \right) = \begin{bmatrix} 0.25 & 0.3 \\ 0.3 & 0.75 \end{bmatrix}. \tag{2.88}$$

■

Example 7 (Compression of the dietary habits of the birds on the island). Suppose that we want to approximate the feature vectors corresponding to the dietary habits of the birds in Example 3 by a representation that has a description length of 1. Firstly, we calculate the second moment matrix

$$\begin{aligned} \boldsymbol{\rho} &= \mathbf{p}_1 \mathbf{f_1} \mathbf{f_1}^\intercal + \mathbf{p}_2 \mathbf{f_2} \mathbf{f_2}^\intercal \\ &= 0.25 \cdot \begin{bmatrix} \sqrt{0.4} \\ \sqrt{0.6} \\ 0 \end{bmatrix} \begin{bmatrix} \sqrt{0.4} & \sqrt{0.6} & 0 \end{bmatrix} + 0.75 \cdot \begin{bmatrix} 0 \\ \sqrt{0.8} \\ \sqrt{0.2} \end{bmatrix} \begin{bmatrix} 0 & \sqrt{0.8} & \sqrt{0.2} \end{bmatrix} \\ &= \begin{bmatrix} 0.1 & 0.12 & 0 \\ 0.12 & 0.75 & 0.3 \\ 0 & 0.3 & 0.15 \end{bmatrix}. \end{aligned} \tag{2.89}$$

Secondly, we diagonalise the second moment matrix

$$\boldsymbol{\rho} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{\intercal} \tag{2.90}$$

$$= \begin{bmatrix} 0.14 & 0.87 & 0.48 \\ 0.92 & 0.07 & -0.39 \\ 0.37 & -0.50 & 0.79 \end{bmatrix} \begin{bmatrix} 0.89 & 0 & 0 \\ 0 & 0.11 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0.14 & 0.87 & 0.48 \\ 0.92 & 0.07 & -0.39 \\ 0.37 & -0.50 & 0.79 \end{bmatrix}^{\intercal}. \tag{2.91}$$

Thridly, the basis set becomes

$$\mathcal{U} = \left\{ \begin{bmatrix} 0.14 \\ 0.92 \\ 0.37 \end{bmatrix}, \begin{bmatrix} 0.87 \\ 0.07 \\ -0.50 \end{bmatrix}, \begin{bmatrix} 0.48 \\ -0.40 \\ 0.79 \end{bmatrix} \right\}, \tag{2.92}$$

and the feature set equals

$$\mathcal{F}_{\text{KL}} = \{\mathbf{f}_{\text{KL},\mathbf{A}}, \mathbf{f}_{\text{KL},\mathbf{B}}\} = \left\{ \mathbf{U}^{\intercal}\mathbf{f}_{\mathbf{A}} = \begin{bmatrix} 0.80 \\ 0.60 \\ 0 \end{bmatrix}, \mathbf{U}^{\intercal}\mathbf{f}_{\mathbf{B}} = \begin{bmatrix} 0.99 \\ -0.16 \\ 0 \end{bmatrix} \right\}. \tag{2.93}$$

Lastly, we compute the basis set and feature set of the 1-truncated KL representation

$$\mathcal{U}^{(1)} = \left\{ \begin{bmatrix} 0.14 \\ 0.92 \\ 0.37 \end{bmatrix} \right\}, \quad \mathcal{F}^{(1)}_{\text{KL}} = \left\{ \mathbf{f}^{(\mathbf{1})}_{\text{KL},\mathbf{A}}, \mathbf{f}^{(\mathbf{1})}_{\text{KL},\mathbf{B}} \right\} = \left\{ \left[0.80\right], \left[0.99\right] \right\}. \tag{2.94}$$

The vectors $\mathbf{f}^{(\mathbf{1})}_{\text{KL},\mathbf{A}}$ and $\mathbf{f}^{(\mathbf{1})}_{\text{KL},\mathbf{B}}$ are the compressed representation of the features vectors. In the original coordinate system, this compressed representation equals

$$\mathcal{F}^{(1)} = \left\{ \mathbf{f}^{(1)}_{A}, \mathbf{f}^{(1)}_{B} \right\} = \left\{ \mathbf{U}^{(1)}\mathbf{f}^{(\mathbf{1})}_{\text{KL},\mathbf{A}} = \begin{bmatrix} 0.11 \\ 0.73 \\ 0.30 \end{bmatrix}, \mathbf{U}^{(1)}\mathbf{f}^{(\mathbf{1})}_{\text{KL},\mathbf{B}} = \begin{bmatrix} 0.14 \\ 0.90 \\ 0.37 \end{bmatrix} \right\}. \tag{2.95}$$

The approximation error of the 1-truncated KL model equals

$$0.25\|\mathbf{f}_{\mathbf{A}} - \mathbf{f}^{(\mathbf{1})}_{\mathbf{A}}\|_2^2 + 0.75\|\mathbf{f}_{\mathbf{B}} - \mathbf{f}^{(\mathbf{1})}_{\mathbf{B}}\|_2^2 = 1 - \boldsymbol{\lambda}_1 = 0.11, \tag{2.96}$$

where $\boldsymbol{\lambda}_1 = 0.89$ is the greatest eigenvalue of both the second moment matrix $\boldsymbol{\rho}$ and the weighted kernel matrix $\boldsymbol{\Omega}(\mathbf{p})$. ■

Example 8 (OLC entropy of the dietary habits of the birds on the island). Suppose that we want to calculate the Shannon OLC entropy of the dietary habits of the birds described in Example 3. The corresponding weighted kernel matrix is calculated in Example 6. The explanatory powers of the first and the second coordinate of the KL representation are the eigenvalues of the

weighted kernel matrix in nonincreasing order, and thus equal 0.89 and 0.11, respectively. The OLC Shannon entropy now becomes

$$^{1}\mathrm{H}_{\mathrm{OLC}}(\boldsymbol{\Omega}, \mathbf{p}) = -0.89 \ln(0.89) - 0.11 \ln(0.11) = 0.35. \tag{2.97}$$

■

Example 9 (OLC diversity of the dietary habits of the birds on the island). Suppose that we want to calculate the OLC diversity of model E in example 4. We calculated the weighted kernel matrix of the model in Example 6. Its eigenvalues are 0.89 and 0.11. The OLC diversity for q equal to 0, 1, 2 and ∞ becomes

$$^{0}\mathrm{D}_{\mathrm{OLC}}(E, s) = 0.89^{0} + 0.11^{0} = 2, \tag{2.98}$$

$$^{1}\mathrm{D}_{\mathrm{OLC}}(E, s) = \exp(-0.89 \ln(0.89) - 0.11 \ln(0.11)) = 1.41, \tag{2.99}$$

$$^{2}\mathrm{D}_{\mathrm{OLC}}(E, s) = \frac{1}{0.89^{2} + 0.11^{2}} = \frac{1}{0.80} = 1.24, \tag{2.100}$$

$$^{\infty}\mathrm{D}_{\mathrm{OLC}}(E, s) = \frac{1}{0.89} = 1.12. \tag{2.101}$$

The Hartley OLC diversity equals 2 because the dietary habits of both bird species are unique (see Example 3). The expected similarity between the dietary habits of two randomly drawn birds equals

$$\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mathbf{p}_i \mathbf{p}_j \mathbf{S}_{ij} = 0.25^{2} + 2 \cdot 0.25 \cdot 0.75 \cdot 0.48 + 0.75^{2} = 0.80, \tag{2.102}$$

which is the inverse of the collision OLC diversity. The feature vector corresponding to the dietary habit that dominates the group of birds is the eigenvector corresponding to the largest eigenvalue of $\boldsymbol{\rho}$

$$\mathbf{f_d} = \begin{bmatrix} 0.14 \\ 0.92 \\ 0.37 \end{bmatrix}. \tag{2.103}$$

This feature vector corresponds to 2% feeding on small seeds, 84% feeding on medium sized seeds and 14% feeding on large seeds because $\mathbf{f_d} = \varphi(\{0.02, 0.84, 0.14\})$. It has an expected similarity with an entity drawn at random equal to

$$\sum_{i=1}^{n_t} \mathbf{p}_i s(\mathbf{f_d}, \mathbf{f_i}) = \sum_{i=1}^{n_t} \mathbf{p}_i (\mathbf{f_d}^{\intercal} \mathbf{f_i})^{2} = 0.25 \cdot 0.64 + 0.75 \cdot 98 = 0.89, \tag{2.104}$$

which is the inverse of the min OLC diversity. ■

Example 10 (Upperbound of the OLC diversity). If we have a model E containing 100 equally abundant types and the similarity between each pair of types equals 0.09, then the OLC diversity for q equal to 0, 1, 2 and ∞ becomes

$$ {}^{0}\mathrm{D}_{\mathrm{OLC}}(E,s) = 100, \;\; {}^{1}\mathrm{D}_{\mathrm{OLC}}(E,s) = 44.8, \tag{2.105} $$

$$ {}^{2}\mathrm{D}_{\mathrm{OLC}}(E,s) = 10.1, \;\; {}^{\infty}\mathrm{D}_{\mathrm{OLC}}(E,s) = 3.26. \tag{2.106} $$

Table 2.7 tells us that the collision OLC diversity is upperbounded by $1/0.09 = 11.1$ and the min OLC diversity is upperbounded by $1/\sqrt{0.09} = 3.33$. The Hartley and the Shannon diversity do not have a finite upperbound with regard to this similarity. If we increase the number of equally abundant types to 1000 and the similarity between each pair of types remains 0.3, then the OLC diversity for q equal to 0, 1, 2 and ∞ becomes

$$ {}^{0}\mathrm{D}_{\mathrm{OLC}}(E',s) = 1000, \;\; {}^{1}\mathrm{D}_{\mathrm{OLC}}(E',s) = 231, \tag{2.107} $$

$$ {}^{2}\mathrm{D}_{\mathrm{OLC}}(E',s) = 11.0, \;\; {}^{\infty}\mathrm{D}_{\mathrm{OLC}}(E',s) = 3.33. \tag{2.108} $$

The min and collision OLC diversity converge to their upperbound while the Shannon and Hartley OLC diversity continue to increase significantly. ■

Example 11 (OLC richness of the dietary habits of the birds on the island). The quadratic OLC richness of model T in example 4 is the optimum of the following optimization problem

$$ {}^{2}\mathrm{R}_{\mathrm{OLC}}(T,s) = \left({}^{2}O\right)^{-1} \text{ where } {}^{2}O = \min_{\mathbf{p}} \mathbf{p}_1^2 + 2 \cdot 0.48 \mathbf{p}_1 \mathbf{p}_2 + \mathbf{p}_2^2 \tag{2.109} $$

$$ \text{subject to } \sum_{i=1}^{n_t} \mathbf{p}_i = 1, \mathbf{p} \geq 0. $$

The optimum is reached in $\hat{\mathbf{p}} = [0.5, 0.5]^{\intercal}$. The richness now becomes

$$ {}^{2}\mathrm{R}_{\mathrm{OLC}}(T,s) = {}^{2}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\Omega}(\hat{\mathbf{p}})) = \frac{1}{0.5^2 + 2 \cdot 0.48 \cdot 0.5^2 + 0.5^2} = 1.35. \tag{2.110} $$

■

Example 12 (OLC contribution vector of the dietary habits of the birds on the island). We have calculated in Example 11 that the quadratic OLC richness of model T in example 4 equals 1.35 and that the abundance vector that maximizes the richness objective function equals $\hat{\mathbf{p}} = [0.5, 0.5]^{\intercal}$. The quadratic OLC contribution vector thus becomes

$$ {}^{\mathbf{2}}\mathbf{c}_{\mathrm{OLC}}(T,s) = {}^{2}\mathrm{R}_{\mathrm{OLC}}(T,s)\, \hat{\mathbf{p}} = 1.35 \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 0.68 \\ 0.68 \end{bmatrix}. \tag{2.111} $$

■

Example 13 (Decomposition of the OLC diversity in the number of types, the evenness and the average contribution). Suppose that we want to get more insight in the Shannon OLC diversity of the feeding habits of the birds discussed in Example 3. The Shannon OLC diversity, the Shannon OLC richness and the number of types equal 1.41, 1.54 and 2, respectively. The Shannon OLC average contribution becomes

$$ {}^1\bar{\mathrm{c}}_{\mathrm{OLC}}(E,s) = \frac{{}^1\mathrm{R}_{\mathrm{OLC}}(E,s)}{n_t} = \frac{1.54}{2} = 0.77. \tag{2.112} $$

The richness is thus reduced by 23% due to the similarity between the types. The Shannon OLC evenness equals

$$ {}^1\mathrm{E}_{\mathrm{OLC}f}(E,s) = \frac{{}^1\mathrm{D}_{\mathrm{OLC}}(E,s)}{{}^1\mathrm{R}_{\mathrm{OLC}}(E,s)} = \frac{1.35}{1.52} = 0.92. \tag{2.113} $$

The diversity is thus reduced by 8% because the birds do not have the abundances that maximize the diversity. We can now decompose ${}^q\mathrm{D}_{\mathrm{OLC}}$ as follows

$$ {}^1\mathrm{D}_{\mathrm{OLC}}(E,s) = n_t \cdot {}^1\bar{\mathrm{c}}_{\mathrm{OLC}}(E,s) \cdot {}^1\mathrm{E}_{\mathrm{OLC}}(E,s) = 2 \cdot 0.77 \cdot 0.92 = 1.41. \tag{2.114} $$

■

Example 14 (Alpha and beta decomposition of the OLC diversity of the dietary habitats of birds). In Example 5, we described the dietary habits of the birds that live on the two islands. Suppose that we want to decompose the Shannon OLC diversity of these islands into its alpha and beta component. The set $\mathcal{E} = \{E, E'\}$ contains both models and the weight vector $\mathbf{w} = [0.8, 0.2]^\intercal$ reflects the size of these models. First we calculate the Shannon OLC diversity of the pooled group and both subgroups

$$ {}^1\mathrm{D}_{\mathrm{OLC}}(0.8E + 0.2E', s) = 1.74, \quad {}^1\mathrm{D}_{\mathrm{OLC}}(E,s) = 1.41, \quad {}^1\mathrm{D}_{\mathrm{OLC}}(E',s) = 1.80. \tag{2.115} $$

The alpha, beta and gamma diversity now become

$$ {}^1\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E},\mathbf{w},s) = \exp\left(0.8\ln\left({}^1\mathrm{D}_{\mathrm{OLC}}(E,s)\right) + 0.2\ln\left({}^1\mathrm{D}_{\mathrm{OLC}}(E',s)\right)\right) = 1.48, \tag{2.116} $$

$$ {}^1\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E},\mathbf{w},s) = {}^1\mathrm{D}_{\mathrm{OLC}}(0.8E + 0.2E', s) = 1.74, \tag{2.117} $$

$$ {}^1\mathrm{D}^{\beta}_{\mathrm{OLC}}(\mathcal{E},\mathbf{w},s) = \frac{{}^1\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E},\mathbf{w},s)}{{}^1\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E},\mathbf{w},s)} = 1.18. \tag{2.118} $$

■

2.B Comments on other indices

2.B.1 Strong replication principle and the modularity property

Here we will explain why the diversity indices of Pavoine and Izsák [9] and Chao et al. [6] do not meet the strong replication principle and modularity property when applied according to the instructions mentioned in their paper. Suppose that we want to determine the phylogenetic diversity of a group of animals where half of the animals belong to species A and the remaining half belong to species B. The phylogenetic tree of both species is shown in Figure 2.B.1. Note that the tree isn't ultrametric. We have the following two subgroups: subgroup 1 contains all animals belonging to species A, and subgroup 2 contains all animals belonging to species B. The diversity of the pooled group, when adding an intraspecific component due to a high evolution rate as proposed by Pavoine and Izsák [9], becomes

$$ {}^{q}\mathrm{D}^{\boldsymbol{\lambda}} = \left(\left(\frac{2}{3} \right)^{q} + \left(\frac{1}{3} \right)^{q} \right)^{\frac{1}{1-q}}. \tag{2.119} $$

According to the index of Chao et al. [6], the diversity of the pooled group equals

$$ {}^{q}\bar{\mathrm{D}}(T) = \frac{2}{3} \left(\left(\frac{1}{2} \right)^{q} + 2 \left(\frac{1}{4} \right)^{q} \right)^{\frac{1}{1-q}}. \tag{2.120} $$

Both subgroups have a diversity equal to 1, are completely distinct and have a relative size with respect to the pooled group equal to 0.5. Therefore, according to the strong replication principle, the diversity of the pooled group should be 2. However, in the case of both indices this does not hold when $q > 0$. If the phylogenetic tree of the two distinct species was ultrametric, then the diversity and the relative size of the subgroups remain the same. The diversity of the pooled group nevertheless changes to 2 and thus differs from the diversity in the non-ultrametric case. Because the diversities and the relative sizes of completely different subgroups do not uniquely define the diversity of the pooled group, the modularity property doesn't hold either.

2.B.2 Chiu and Chao's distance-based replication principle

We explain here why the distance-based replication principle proposed by [10] should not hold. Suppose that we have four species with the following functional data

$$ d_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \; d_2 = \begin{bmatrix} 9.99 \\ 0.01 \end{bmatrix}, \; d_3 = \begin{bmatrix} 0 \\ 0.01 \end{bmatrix}, \; d_4 = \begin{bmatrix} 9.99 \\ 0 \end{bmatrix}, \tag{2.121} $$

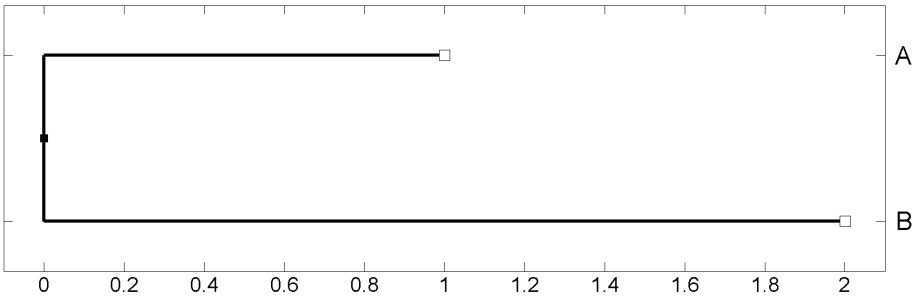


Figure 2.B.1: Phylogenetic tree of Species A and species B.

and we chose the functional distance equal to the Manhattan distance between the datapoints:

$$D = \begin{bmatrix} 0 & 10 & 0.01 & 9.99 \\ 10 & 0 & 9.99 & 0.01 \\ 0.01 & 9.99 & 0 & 10 \\ 9.99 & 0.01 & 10 & 0 \end{bmatrix}, \quad p = \begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}. \tag{2.122}$$

We divide this pooled group in two subgroups. The first subgroup contains the individuals belonging to species 1 and species 2, and the second subgroup contains the individuals species 3 and species 4. The functional distance matrix shows that species 1 is nearly equivalent to species 3, and species 2 is nearly equivalent to species 4. In other words, both subgroups are nearly equivalent. Note that the subgroups do not share any species, and the functional Hill number of order q for any pair of assemblages are the same:

$${}^{q}\mathrm{D}(Q_{11}) = {}^{q}\mathrm{D}(Q_{12}) = {}^{q}\mathrm{D}(Q_{21}) = {}^{q}\mathrm{D}(Q_{22}) = (4 \cdot (0.25)^{q})^{\frac{1}{2(1-q)}}. \tag{2.123}$$

The conditions of the replication principle proposed by [10] are thus met, which means that - according to this replication principle - the functional Hill number of the pooled group should be two times the diversity of the subgroups. We disagree with this claim and are of the opinion that the diversity of the pooled group should be close to d because both subgroups are nearly-equivalent and therefore the beta diversity should be close to one.

2.C Extending an index to the general framework

In order to use the indices proposed in this paper properly, one should just make sure that the used similarity function is meaningful. However, for those who want to propose new indices in this framework or just want to understand better how the framework works, we explain here (1) which condition an index must meet such that it can be used in the general framework, (2) how to avoid the need to calculate a corresponding feature mapping when evaluating indices

in the general framework, and (3) what happens if we use a similarity matrix that isn't meaningful.

2.C.1 When is an index meaningful in the general framework?

Multiple feature mappings can correspond to the same similarity function and therefore the value of an index might depend on which specific feature mapping is chosen. To handle this issue, we state that only indices that take the same value for all the corresponding feature mappings - and thus do not depend on the specific choice - are meaningful in the general framework. This might seem as a severe restriction with respect to the allowed indices, but it isn't. All the feature mappings corresponding to the same similarity function, result - by definition - in the same similarities. A similarity-based index that except for properties unrelated to the values of the feature vectors, such as the relative abundances, only reflects the similarities between the assigned feature vectors, is thus meaningful in the general framework. On the other hand, an index that does not merely reflect the similarities and therefore depends in some other way on the values of feature vectors, is not meaningful in the general framework. In other words, an index is meaningful in the general framework if and only if it can be written as a function of the similarity matrix and properties unrelated to the values of the feature vectors.

Example 15 (Index that is not meaningful in the general framework). In Example (3) we defined a similarity function s with respect to the dietary habits of birds. We also mentioned φ, one of the feature mappings that corresponds to this similarity function. Another feature mapping that results in this similarity function is

$$\mathbf{f}'_{\mathbf{X}} = \varphi' d_X = \begin{bmatrix} 0.8\sqrt{a_X} + 0.6\sqrt{b_X} \\ 0.8\sqrt{b_X} - 0.6\sqrt{a_X} \\ \sqrt{c_X} \end{bmatrix}, \tag{2.124}$$

where $d_X = \{a_X, b_X, c_X\}$ is the data corresponding to type X, and a_X, b_X and c_X represent the feeding on small, medium-sized and large seeds, respectively. One can verify that $s(d_Y, d_Z) = \left(\mathbf{f}'^{\intercal}_{\mathbf{Y}} \mathbf{f}'_{\mathbf{Z}}\right)^2$. The alternative feature vectors corresponding to the birds of species A and species B become

$$\mathbf{f}'_{\mathbf{A}} = \begin{bmatrix} 0.97 \\ 0.24 \\ 0 \end{bmatrix}, \quad \mathbf{f}'_{\mathbf{B}} = \begin{bmatrix} 0.54 \\ 0.72 \\ 0.45 \end{bmatrix}. \tag{2.125}$$

The Manhattan distance between two vectors is defined as

$$d_1(\mathbf{v}, \mathbf{v}') = \sum_{i=1}^{n_v} \|v_i - v'_i\|. \tag{2.126}$$

We can calculate the Manhattan distance between f_A and f_B, and f'_A and f'_B:

$$d_1(\mathbf{f}_A, \mathbf{f}_B) = \sqrt{0.4} + (\sqrt{0.8} - \sqrt{0.6}) + \sqrt{0.2} = 1.20 \tag{2.127}$$

$$d_1(\mathbf{f'}_A, \mathbf{f'}_B) = (0.97 - 54) + (0.72 - 0.24) + 0.45 = 1.36. \tag{2.128}$$

Feature vectors with the same similarities can thus have a different Manhattan distance and this distance is thus not a meaningful index in the general framework. ■

2.C.2 Expressing an index as a function of the similarity matrix

Writing an index directly as a function of the similarity matrix is what we call the similarity trick in this paper. The similarity trick has two important advantages: (1) as we showed above, it implicitly proves that the index is meaningful in the general framework; and (2) the feature mapping corresponding to certain similarity functions can be huge or even infinite dimensional, and not having to calculate the feature vectors can therefore reduce the computational complexity considerably (e.g., Equation (2.26) does not require a corresponding feature mapping to be found). In machine learning, there is a nearly equivalent technique which is called the kernel trick. The kernel trick implies that we express an index directly as a function of the kernel matrix (the kernel matrix contains the inner products of the feature vectors) and this trick is applied because of the same reason as the similarity trick, i.e., it reduces the computational complexity. The kernel trick is used by all kinds of algorithms including support vector machines [45], kernel principal component analysis [46] and kernel spectral clustering [47]. Basically, the similarity trick is the kernel trick adapted to our framework.

2.C.3 Using a similarity function that is not meaningful

Although the indices proposed in this paper do not require the feature vectors to be calculated, their meaning is nevertheless in the feature space. The uniqueness of a type, for example, is the square error when approximating the assigned feature vector by the feature vectors assigned to the other types belonging to the set. If there does not exist a corresponding feature mapping, which means that the similarity matrix isn't meaningful, then the meaning of the index no longer holds. To show what happens when we use a similarity function isn't meaningful, we apply Equation (2.26) to the similarity matrix defined in

Equation (2.25)

$$\mathrm{u}' = \odot^{-1}\mathrm{diag}\left(\left(\odot^{\frac{1}{2}}\mathbf{S}\right)^{-1}\right) = \begin{bmatrix} -8 \\ -8 \\ -0.8 \end{bmatrix}. \tag{2.129}$$

The formula implicitly uses the following reasoning in order to calculate the uniqueness of type 3. Type 1 is completely distinct from type 2, and therefore the similarity between type 3 and type 1, which equals 0.9, does not overlap with the similarity between type 3 and type 2, which also equals 0.9. The proportion of type 3 that is not unique with respect either type 1 or type 2 now becomes the sum of these similarities, which equals 180%. The uniqueness of a type equals 100% minus the proportion of the type that is not unique, and therefore equals -80%. The result doesn't make any sense because a proportion should be at least zero. The implicit reasoning made by the formula would be valid if the similarity matrix was meaningful. For example, if the similarity between both type 1 and type 3, and type 2 and type 3 would be 0.4 instead, then the uniqueness of type 3 equals 20% $(1 - 2 \cdot 0.4)$, which makes sense. Applying the uniqueness to a similarity matrix that isn't meaningful thus results in a uniqueness that isn't meaningful, and should therefore never be done.

2.D OLC uniqueness vector

Theorem 1. *If* $\boldsymbol{\Omega} = \odot^{\frac{1}{2}}\mathbf{S}$ *is of full rank, then the OLC uniqueness vector equals*

$$\mathrm{u} = \odot^{-1} \mathit{diag}(\boldsymbol{\Omega}^{-1}), \tag{2.130}$$

where $\mathit{diag}(\mathbf{A})$ *is the vector containing the main diagonal elements of* $\mathbf{A}$*, and* $\odot^{-1}(\mathbf{A})$ *is the element-wise inverse of* $\mathbf{A}$*, i.e.,* $\left(\odot^{-1}(\mathbf{A})\right)_{ij} = \mathbf{A}_{ij}^{-1}$.

Proof. We will proof that u_1 equals the OLC uniqueness of type 1. The mean squared error when approximating $\mathbf{f_1}$, the feature vector corresponding to type 1, by a linear combination of $\mathcal{F}_r = \{\mathbf{f_i}\}_{i=2}^{n_t}$, the feature vectors corresponding to the remaining types, equals

$$e_1^2 = 1 - \mathbf{f_1}^\intercal \mathbf{F_r} (\mathbf{F_r}^\intercal \mathbf{F_r})^{-1} \mathbf{F_r}^\intercal \mathbf{f_1}, \tag{2.131}$$

where $\mathbf{F_r} = [\mathbf{f_2}, \ldots, \mathbf{f_{n_t}}]$. The kernel matrix equals the following block matrix

$$\boldsymbol{\Omega} = \begin{bmatrix} 1 & \mathbf{f_1}^\intercal \mathbf{F_r} \\ \mathbf{F_r}^\intercal \mathbf{f_1} & \mathbf{F_r}^\intercal \mathbf{F_r} \end{bmatrix}, \tag{2.132}$$

and its blockwise inversion becomes

$$\boldsymbol{\Omega}^{-1} = \begin{bmatrix} \frac{1}{1-\mathbf{f_1}^\intercal \mathbf{F_r}(\mathbf{F_r}^\intercal \mathbf{F_r})^{-1}\mathbf{F_r}^\intercal \mathbf{f_1}} & -\frac{\mathbf{f_1}^\intercal \mathbf{F_r}(\mathbf{F_r}^\intercal \mathbf{F_r})^{-1}}{1-\mathbf{f_1}^\intercal \mathbf{F_r}(\mathbf{F_r}^\intercal \mathbf{F_r})^{-1}\mathbf{F_r}^\intercal \mathbf{f_1}} \\ -\frac{(\mathbf{F_r}^\intercal \mathbf{F_r})^{-1}\mathbf{F_r}^\intercal \mathbf{f_1}}{1-\mathbf{f_1}^\intercal \mathbf{F_r}(\mathbf{F_r}^\intercal \mathbf{F_r})^{-1}\mathbf{F_r}^\intercal \mathbf{f_1}} & (\mathbf{F_r}^\intercal \mathbf{F_r} - \mathbf{F_r}^\intercal \mathbf{f_1}\mathbf{f_1}^\intercal \mathbf{F_r})^{-1}. \end{bmatrix} \tag{2.133}$$

We can conclude that $\mathrm{u}_1 = \left(\left(\boldsymbol{\Omega}^{-1}\right)_{1,1}\right)^{-1} = e_1^2$. The proof for the remaining elements of the contribution vector, i.e., u_i with $i \in \{2, \ldots, n_t\}$, is similar. ■

In the case that $\boldsymbol{\Omega}$ is not of full rank, we first set the uniqueness to zero of the types corresponding to columns (or rows) of $\boldsymbol{\Omega}$ that are linearly dependent on other columns (or rows). We then iteratively remove a linearly dependent column and corresponding row until the reduced kernel matrix $\boldsymbol{\Omega}'$ is of full rank. We can now determine the uniqueness of the remaining types by applying Theorem 1 to $\boldsymbol{\Omega}'$.

2.E Our notation applied to alpha and beta

Suppose that we have a pooled group consisting of n_w subgroups, where the subgroups are represented by the models in the set $\mathcal{E} = \{E_i\}_{i=1}^{n_w}$, and $\mathbf{w} \in \mathbb{R}^{n_w}$ is a vector such that $\mathbf{w}_i$ is the relative size of subgroup i. We can now reformulate the Shannon version of the alpha, beta and gamma diversity proposed by Jost [26] as follows:

$${}^1\mathrm{D}^\gamma(\mathcal{E}, \mathbf{w}) = {}^1\mathrm{D}\left(\sum_{i=1}^{n_w} \mathbf{w}_i E_i\right), \tag{2.134}$$

$${}^1\mathrm{D}^\alpha(\mathcal{E}, \mathbf{w}) = M_0\big({}^1\mathrm{D}\big(E_1\big), \ldots, {}^1\mathrm{D}\big(E_{n_w}\big)\big) = \exp\left(\sum_{i=1}^{n_w} \mathbf{w}_i \ln\big({}^1\mathrm{D}\big(E_i\big)\big)\right), \tag{2.135}$$

$${}^1\mathrm{D}^\beta(\mathcal{E}, \mathbf{w}) = \frac{{}^1\mathrm{D}^\gamma(\mathcal{E}, \mathbf{w})}{{}^1\mathrm{D}^\alpha(\mathcal{E}, \mathbf{w})}, \tag{2.136}$$

where ${}^1\mathrm{D}$ is the type-neutral Shannon diversity, and $M_0\big({}^1\mathrm{D}\big(E_1\big), \ldots, {}^1\mathrm{D}\big(E_{n_w}\big)\big)$ is the weighted geometric mean of the diversities with weights $\mathbf{w}_1, \ldots, \mathbf{w}_{n_w}$. The notation relieves us from the details (i.e., how the diversity of a group of entities is calculated and how to take the union of groups of entities) such that the higher level reasoning becomes more clear (i.e., the alpha diversity is the weighted geometric mean of the diversities of the subgroups and the gamma diversity is the diversity of the pooled group, where each subgroup has relative size $\mathbf{w}_i$).

2.F Approximation error, explanatory power and entropy

2.F.1 Minimum approximation error property

Theorem 2. *Of all representations with description length N, the approximation error with respect to feature set $\mathcal{F} = \{\mathbf{f_i}\}_{i=1}^{n_t}$ and abundance vector $\mathbf{p}$ is minimized by the representation $\mathcal{F}_{KL}^{(N)} = \left\{\mathbf{f}_{KL,\mathbf{i}}^{(\mathbf{N})}\right\}_{i=1}^{n_t}$ where*

- $\mathbf{P} \in \mathbb{R}^{n_t \times n_t}$ *is a diagonal matrix with* $\mathbf{P}_{ii} = \mathbf{p}_i$,
- $\boldsymbol{\rho} = \mathbf{F}\mathbf{P}\mathbf{F}^\intercal$ *is the second moment matrix with* $\mathbf{F} = [\mathbf{f_1}, \ldots, \mathbf{f_{n_t}}]$,
- *the second moment matrix has the following diagonalization* $\boldsymbol{\rho} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\intercal$ *with* $\boldsymbol{\Lambda}$ *a diagonal matrix whose diagonal elements are in descending order and* $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_{n_f}]$ *an orthogonal matrix,*
- $\mathbf{U}^{(\mathbf{N})} = [\mathbf{u}_1, \ldots, \mathbf{u}_N]$ *is the matrix whose columns become the new basis vectors,*
- $\mathbf{F}_{KL}^{(\mathbf{N})} = [\mathbf{f}_{KL,\mathbf{1}}^{(\mathbf{N})}, \ldots, \mathbf{f}_{KL,\mathbf{n_t}}^{(\mathbf{N})}] = \mathbf{U}^{(\mathbf{N})\intercal}\mathbf{F}$ *is the matrix whose columns are the representation of the feature vectors with respect to these basis vectors.*

Proof. For simplicity, we assume that only the types with non-zero abundances belong to the feature set and, therefore, that $\mathbf{P}$ has full rank. Nevertheless the theorem also holds when $\mathbf{P}$ is not of full rank. To minimize the approximation error, we have to solve the following optimization problem

$$\min_{\substack{\mathbf{U}' \in \mathbb{R}^{n_f \times N} \\ \mathbf{F}'_{\mathbf{U}} \in \mathbb{R}^{N \times n_t}}} \sum_{i=1}^{n_t} \mathbf{p}_i \|\mathbf{f_i} - \mathbf{f'}_i\|_2^2 \tag{2.137}$$

$$\text{subject to } \mathbf{F}' = [\mathbf{f'}_1, \ldots, \mathbf{f'}_{n_t}] = \mathbf{U}'\mathbf{F}'_{\mathbf{U}},$$

where the columns of $\mathbf{U}'$ and $\mathbf{F}'_{\mathbf{U}}$ are the basis vectors and coordinates of the new representation, respectively. The theorem now holds if this optimization problem reaches its optimum in $\mathbf{U}' = \mathbf{U}^{(\mathbf{N})}$ and $\mathbf{F}'_{\mathbf{U}} = \mathbf{F}_{\mathrm{KL}}^{(\mathbf{N})}$. We can rewrite the optimization problem as follows

$$\min_{\substack{\mathbf{U}' \in \mathbb{R}^{n_f \times N} \\ \mathbf{F}'_{\mathbf{U}} \in \mathbb{R}^{N \times n_t}}} \|\mathbf{F}\mathbf{P}^{\frac{1}{2}} - \mathbf{F}'\mathbf{P}^{\frac{1}{2}}\|_F^2 \tag{2.138}$$

$$\text{subject to } \mathbf{F}' = \mathbf{U}'\mathbf{F}'_{\mathbf{U}},$$

where $\|\mathbf{A}\|_F$ is the Frobenius norm of matrix $\mathbf{A}$. For any $\mathbf{F}'$ of rank N or smaller, there exists a $\mathbf{U}' \in \mathbb{R}^{n_f \times N}$ and $\mathbf{F}'_{\mathbf{U}} \in \mathbb{R}^{N \times n_t}$ such that $\mathbf{F}' = \mathbf{U}'\mathbf{F}'_{\mathbf{U}}$, and vice versa. The rank of $\mathbf{F}'$ equals the rank of $\mathbf{F}'\mathbf{P}^{\frac{1}{2}}$ when $\mathbf{P}$ has full rank. Therefore Optimization Problem (2.138) is equivalent to

$$\min_{\mathbf{D}'} \|\mathbf{D} - \mathbf{D}'\|_F^2 \tag{2.139}$$

$$\text{subject to } \operatorname{rank}(\mathbf{D}') \leq N, \quad \mathbf{D} = \mathbf{F}\mathbf{P}^{\frac{1}{2}}.$$

Optimization Problem (2.139) is a low-rank approximation problem. According to the matrix approximation lemma [48], the N-truncated singular value decomposition (SVD) of $\mathbf{D}$ is the best low-rank approximation of $\mathbf{D}$. This N-truncated SVD is denoted by $\hat{\mathbf{D}}' = \mathbf{U_N}\mathbf{\Sigma_N}\mathbf{V_N}^\intercal$ where

- The SVD becomes of $\mathbf{D}$ becomes $\mathbf{D} = \mathbf{U}^{(\mathbf{D})}\mathbf{\Sigma}^{(\mathbf{D})}\mathbf{V}^{(\mathbf{D})\intercal}$ with $\mathbf{U}^{(\mathbf{D})} = \left[\mathbf{u}_\mathbf{1}^{(\mathbf{D})}, \ldots, \mathbf{u}_{\mathbf{n_f}}^{(\mathbf{D})}\right]$ an orthogonal matrix, $\mathbf{V}^{(\mathbf{D})} = \left[\mathbf{v}_\mathbf{1}^{(\mathbf{D})}, \ldots, \mathbf{v}_{\mathbf{n_t}}^{(\mathbf{D})}\right]$ an orthogonal matrix, and $\mathbf{\Sigma}^{(\mathbf{D})} \in \mathbb{R}^{n_f \times n_t}$ a diagonal matrix containing the singular values in descending order,
- $\mathbf{\Sigma_N} \in \mathbb{R}^{N \times N}$ is a diagonal matrix with the diagonal elements equal to the N largest singular values of D in descending order,
- $\mathbf{U_N} = \left[\mathbf{u}_\mathbf{1}^{(\mathbf{D})}, \ldots, \mathbf{u}_\mathbf{N}^{(\mathbf{D})}\right]$ contains the first N columns of $\mathbf{U}^{(\mathbf{D})}$,
- $\mathbf{V_N} = \left[\mathbf{v}_\mathbf{1}^{(\mathbf{D})}, \ldots, \mathbf{v}_\mathbf{N}^{(\mathbf{D})}\right]$ contains the first N columns of $\mathbf{V}^{(\mathbf{D})}$.

We now have to show that $\hat{\mathbf{D}}' = \mathbf{U_N}\mathbf{\Sigma_N}\mathbf{V_N}^\intercal$ equals $\mathbf{U}^{(\mathbf{N})}\mathbf{F}_{\mathrm{KL}}^{(\mathbf{N})}\mathbf{P}^{\frac{1}{2}}$. The eigenvectors of $\boldsymbol{\rho}$, which equal the columns of $\mathbf{U}$, are the left singular vectors of $\mathbf{F}\mathbf{P}^{\frac{1}{2}}$ because $\boldsymbol{\rho} = \mathbf{F}\mathbf{P}^{\frac{1}{2}}\left(\mathbf{F}\mathbf{P}^{\frac{1}{2}}\right)^\intercal$, and therefore $\mathbf{U_N} = \mathbf{U}^{(\mathbf{N})}$. We can now infer that

$$\mathbf{U_N}\mathbf{\Sigma_N}\mathbf{V_N}^\intercal = \mathbf{U_N}\left[\mathbf{\Sigma_N}, \mathbf{0_{N \times (n_t - N)}}\right]\mathbf{V}^{(\mathbf{D})\intercal} \tag{2.140}$$

$$= \mathbf{U_N}\left[\mathbf{I_{N \times N}}, \mathbf{0_{N \times (n_t - N)}}\right]\mathbf{\Sigma}^{(\mathbf{D})}\mathbf{V}^{(\mathbf{D})\intercal} \tag{2.141}$$

$$= \mathbf{U_N}\mathbf{U_N}^\intercal\mathbf{U}^{(\mathbf{D})}\mathbf{\Sigma}^{(\mathbf{D})}\mathbf{V}^{(\mathbf{D})\intercal} \tag{2.142}$$

$$= \mathbf{U_N}\mathbf{U_N}^\intercal\mathbf{F}\mathbf{P}^{\frac{1}{2}} \tag{2.143}$$

$$= \mathbf{U}^{(\mathbf{N})}\mathbf{F}_{\mathrm{KL}}^{(\mathbf{N})}\mathbf{P}^{\frac{1}{2}}, \tag{2.144}$$

where $\mathbf{I_{n \times n}} \in \mathbb{R}^{n \times n}$ is the identity matrix and $\mathbf{0_{m \times n}} \in \mathbb{R}^{m \times n}$ is a matrix containing only zeros. ■

2.F.2 Explanatory power

Here we will explain the explantory power of basis vectors. Suppose that we want to approximate a feature vector $\mathbf{f}$ by a linear combination of the orthonormal basis vectors belonging to the set $\mathcal{U} = \{\mathbf{u_i}\}_{i=1}^{N}$ such that the square error is minimized. We denote the matrix whose columns are the vectors belonging to $\mathcal{U}$ by $\mathbf{U}$. The approximating feature vector becomes $\hat{\mathbf{f}} = \mathbf{U}\mathbf{U}^\intercal\mathbf{f}$ because $\mathbf{U}\mathbf{U}^\intercal$ is a projection matrix that projects onto the column space of $\mathbf{U}$. The approximation error equals $\|\mathbf{f} - \hat{\mathbf{f}}\|_2^2 = 1 - \sum_{i=1}^{N} \left(\mathbf{u_i}^\intercal\mathbf{f}\right)^2$. If the set $\mathcal{U}$ is empty, then the approximation error equals 1, which is its maximal attainable value. The explanatory power of $\mathcal{U}$ is defined as how much the approximation error decreases if we approximate the feature vector by a linear combination of the basis vectors in $\mathcal{U}$ instead of by a linear combination of the basis vectors in the empty set. The explanatory power thus equals

$$R^2(\mathbf{f},\mathcal{U}) = \|\mathbf{f}\|_2^2 - \|\mathbf{f} - \hat{\mathbf{f}}\|_2^2 = \sum_{i=1}^{N} \left(\mathbf{u_i}^\intercal\mathbf{f}\right)^2. \tag{2.145}$$

The explanatory power of $\mathcal{U}$ with respect to a randomly drawn entity has the following expected value

$$R^2(\mathcal{F},\mathcal{U},\mathbf{p}) = \sum_{j=1}^{n_t} \mathbf{p}_j \sum_{i=1}^{N} \left(\mathbf{u_i}^\intercal\mathbf{f_j}\right)^2 = \sum_{i=1}^{N}\sum_{j=1}^{n_t} \mathbf{p}_j \left(\mathbf{u_i}^\intercal\mathbf{f_j}\right)^2 = \sum_{i=1}^{N} R^2(\mathcal{F},\mathbf{u}_i,\mathbf{p}). \tag{2.146}$$

This formula shows that the explanatory power $R^2(\mathcal{F},\mathcal{U},\mathbf{p})$ can be partitioned in the explanatory powers of each basis vector where $R^2(\mathcal{F},\mathbf{u_i},\mathbf{p})$ is the explanatory power of basis vector $\mathbf{u_i}$. Note that the basis vectors need to be orthogonal for the partitioning property to hold. In the case of the KL representation, the basis vectors (which are the normalised eigenvectors of the second moment matrix) are orthogonal and the partioning property thus holds.

Example 16 (Explanatory power of the coordinates of the KL representation of the dietary habits). Suppose that we want to know to which extent the feeding habits of the birds on the island described in Example 3 can be explained by each coordinate of the KL representation. We calculated the KL representation in Example 7. The explanatory power of the first coordinate, and thus of basis vector $\mathbf{u_1} = [0.14, 0.92, 0.37]^\intercal$, equals

$$R^2(\mathcal{F},\mathbf{u_1},p) = \mathbf{p}_A(\mathbf{u_1}^\intercal\mathbf{f_A})^2 + \mathbf{p}_B(\mathbf{u_1}^\intercal\mathbf{f_B})^2 = 0.89. \tag{2.147}$$

Similarly we can calculate the explanatory power of the other two coordinates

$$R^2(\mathcal{F},\mathbf{u}_2,p) = 0.11, \quad R^2(\mathcal{F},\mathbf{u}_3,p) = 0, \tag{2.148}$$

where $\mathbf{u_2} = [0.87, 0.07, -0.50]^\intercal$ and $\mathbf{u_3} = [0.48, -0.39, 0.79]^\intercal$. ■

Theorem 3. *The approximation error when removing all but the first N coordinates of the KL representation equals $1-\sum_{i=1}^{N}\boldsymbol{\lambda}_i$, where $\boldsymbol{\lambda}_1 \geq \boldsymbol{\lambda}_2 \geq \cdots \geq \boldsymbol{\lambda}_{n_t}$ are the eigenvalues of the second moment matrix $\boldsymbol{\rho}$.*

Proof. If $\boldsymbol{\rho} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\intercal$ is the diagonalization of the second moment matrix $\boldsymbol{\rho} = \mathbf{F}\mathbf{P}\mathbf{F}^\intercal$, then $\boldsymbol{\Lambda}$ has off-diagonal elements equal to zero and its diagonal elements equal

$$\boldsymbol{\Lambda} = \mathbf{U}^\intercal\mathbf{F}\mathbf{P}\mathbf{F}^\intercal\mathbf{U} = \left(\mathbf{U}^\intercal\mathbf{F}\mathbf{P}\right)\left(\mathbf{F}^\intercal\mathbf{U}\right) \tag{2.149}$$

$$= \begin{bmatrix} \mathbf{p}_1\mathbf{u_1}^\intercal\mathbf{f_1} & \mathbf{p}_2\mathbf{u_1}^\intercal\mathbf{f_2} & \dots & \mathbf{p}_{n_f}\mathbf{u_1}^\intercal\mathbf{f_{n_f}} \\ \mathbf{p}_1\mathbf{u_2}^\intercal\mathbf{f_1} & \mathbf{p}_2\mathbf{u_2}^\intercal\mathbf{f_2} & \dots & \mathbf{p}_{n_f}\mathbf{u_2}^\intercal\mathbf{f_{n_f}} \\ \vdots & \vdots & & \vdots \\ \mathbf{p}_1\mathbf{u_{n_f}}^\intercal\mathbf{f_1} & \mathbf{p}_2\mathbf{u_{n_f}}^\intercal\mathbf{f_2} & \dots & \mathbf{p}_{n_f}\mathbf{u_{n_f}}^\intercal\mathbf{f_{n_f}} \end{bmatrix} \tag{2.150}$$

$$\begin{bmatrix} \mathbf{u_1}^\intercal\mathbf{f_1} & \mathbf{u_2}^\intercal\mathbf{f_1} & \dots & \mathbf{u_{n_f}}^\intercal\mathbf{f_1} \\ \mathbf{u_1}^\intercal\mathbf{f_2} & \mathbf{u_2}^\intercal\mathbf{f_2} & \dots & \mathbf{u_{n_f}}^\intercal\mathbf{f_2} \\ \vdots & \vdots & & \vdots \\ \mathbf{u_1}^\intercal\mathbf{f_{n_f}} & \mathbf{u_2}^\intercal\mathbf{f_{n_f}} & \dots & \mathbf{u_{n_f}}^\intercal\mathbf{f_{n_f}} \end{bmatrix} \tag{2.151}$$

$$= \begin{bmatrix} \sum_{i=1}^{n_t}\mathbf{p}_i\left(\mathbf{u_1}^\intercal\mathbf{f_i}\right)^2 & 0 & \dots & 0 \\ 0 & \sum_{i=1}^{n_t}\mathbf{p}_i\left(\mathbf{u_2}^\intercal\mathbf{f_i}\right)^2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sum_{i=1}^{n_t}\mathbf{p}_i\left(\mathbf{u_{n_f}}^\intercal\mathbf{f_i}\right)^2 \end{bmatrix} \tag{2.152}$$

$$= \begin{bmatrix} R^2(\mathcal{F},\mathbf{u_1},\mathbf{p}) & 0 & \dots & 0 \\ 0 & R^2(\mathcal{F},\mathbf{u_2},\mathbf{p}) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R^2(\mathcal{F},\mathbf{u_{n_f}},\mathbf{p}) \end{bmatrix}, \tag{2.153}$$

where $\mathbf{u_i}$ is the i-th column of U. The eigenvalues of the second moment matrix thus equal the explanatory powers of the coordinates of the KL representation. The approximation error when removing all but the first N coordinates now becomes

$$1-\sum_{i=1}^{N}R^2(\mathcal{F},\mathbf{u_i},\mathbf{p}) = 1-\sum_{i=1}^{N}\boldsymbol{\lambda}_i. \qquad \blacksquare$$

One can verify in Example 16 that the explanatory power of each coordinate of the KL representation equals an eigenvalue of the second moment matrix. The feature vectors corresponding to certain kernels can be huge or infinite dimensional, which causes the calculation of the feature vectors to become computationally very expensive or even unfeasible. We will now apply the kernel

trick such that the approximation error of the N-truncated KL representation can be computed without explicit calculation of the feature vectors. The second moment matrix $\boldsymbol{\rho}$ and the weighted kernel matrix $\boldsymbol{\Omega}(\mathbf{p})$ have the same non-zero eigenvalues because $\boldsymbol{\Omega}(\mathbf{p}) = \mathbf{A}^\intercal\mathbf{A} = \mathbf{V}\left(\boldsymbol{\Sigma}^\intercal\boldsymbol{\Sigma}\right)\mathbf{V}^\intercal$ and $\boldsymbol{\rho} = \mathbf{A}\mathbf{A}^\intercal = \mathbf{U}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\intercal\right)\mathbf{U}^\intercal$ where $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\intercal$ is the singular value decomposition of $\mathbf{A} = \mathbf{F}\mathbf{P}^{\frac{1}{2}}$. This means that we can replace the eigenvalues of second moment matrix in Theorem 3 by the eigenvalues of the weighted kernel matrix. The approximation error of the N-truncated KL representation now becomes

$$1 - \sum_{i=1}^{N} \boldsymbol{\lambda}_i, \tag{2.154}$$

where $\boldsymbol{\lambda}_1 \geq \cdots \geq \boldsymbol{\lambda}_{n_t}$ are the eigenvalues of $\boldsymbol{\Omega}(\mathbf{p})$. Equation (2.154) does not require the feature vectors to be calculated and, furthermore, does not depend on the chosen feature mapping.

2.F.3 Derivation of the OLC entropy

We will show that the OLC entropy is the only entropy that has the following properties:

- If the types are completely distinct, then the entropy equals the type-neutral entropy.
- If for any N, two groups of entities cause the same approximation error when compressing their representation to description length N, then both groups have the same entropy. In other words, the entropy is fully determined by these approximation errors.
- If for any N, group 1 doesn't causes a smaller approximation error than group 2 when compressing their representation to description length N, then group 1 has a higher or equal entropy.

Because the approximation error is fully determined by $\boldsymbol{\Lambda} \in \mathbb{R}^{n_t}$, the vector that contains the eigenvalues of $\boldsymbol{\Omega}(\mathbf{p})$, the OLC entropy is, according to the second property, also fully determined by $\boldsymbol{\Lambda}$. We can now express the OLC entropy as

$${}^{q}\mathrm{H}_{\mathrm{OLC}}(\boldsymbol{\Omega}, \mathbf{p}) = {}^{q}\mathrm{H}_{\mathrm{OLC}}(\boldsymbol{\lambda}). \tag{2.155}$$

If the types are completely distinct, then the eigenvalues of $\boldsymbol{\Omega}(\mathbf{p})$ equal the relative abundances. Therefore, according to the first property, the following holds

$${}^{q}\mathrm{H}_{\mathrm{OLC}}(\boldsymbol{\lambda}) = {}^{q}\mathrm{H}(\boldsymbol{\lambda}), \tag{2.156}$$

where ${}^{q}\mathrm{H}$ denotes the type-neutral entropy. We now still have to verify whether the third property is satisfied. Suppose that we have the following approximation error inequalities for $N \in \{0, \ldots, n\}$

$$1 - \sum_{i=1}^{N} \boldsymbol{\lambda}_i \geq 1 - \sum_{i=1}^{N} \boldsymbol{\lambda}'_i, \tag{2.157}$$

where n is the maximum of the description lengths of the KL representations corresponding to group 1 and group 2, and $\{\boldsymbol{\lambda}_i\}_{i=1}^{n}$ and $\{\boldsymbol{\lambda}'_i\}_{i=1}^{n}$ are sets of positive real numbers such that $1 - \sum_{i=1}^{N} \boldsymbol{\lambda}_i$ and $1 - \sum_{i=1}^{N} \boldsymbol{\lambda}'_i$ are the approximation error of the N-truncated KL representation corresponding to group 1 and group 2 for $N \in \{0, \ldots, n\}$, respectively. In case N exceeds the number of coordinates of the KL representation, then we state that the N-truncated KL representation has an approximation error of zero despite the fact the we expand the representation instead of truncating it. Because $\sum_{i=1}^{n} \boldsymbol{\lambda}_i = \sum_{i=1}^{n} \boldsymbol{\lambda}'_i = 1$, these approximation error inequalities imply that $\boldsymbol{\lambda}$ is majorized by $\boldsymbol{\lambda}'$. The OLC entropy is Schur-concave in the explanatory powers, and the OLC entropy corresponding to $\boldsymbol{\lambda}$ is thus greater than or equal to the OLC entropy corresponding to $\boldsymbol{\lambda}'$. The third property therefore also holds.

2.G Reformulating the quadratic and min OLC diversity

Theorem 4. *For $q = 2$, the OLC diversity can be expressed as*

$${}^{2}\mathrm{D}_{OLC}(\boldsymbol{\Omega}, \mathbf{p}) = \left(\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mathbf{p}_i \mathbf{p}_j \mathbf{S}_{ij} \right)^{-1}. \tag{2.158}$$

Proof. By applying the definition of the OLC diversity we get

$${}^{2}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\Omega}(\mathbf{p})) \tag{2.159}$$

$$= \mathrm{tr} \left(\begin{bmatrix} \mathbf{p}_1 \mathbf{f_1}^{\intercal} \mathbf{f_1} & \sqrt{\mathbf{p}_1 \mathbf{p}_2} \mathbf{f_1}^{\intercal} \mathbf{f_2} & \cdots & \sqrt{\mathbf{p}_1 \mathbf{p}_{n_t}} \mathbf{f_1}^{\intercal} \mathbf{f_{n_t}} \\ \sqrt{\mathbf{p}_2 \mathbf{p}_1} \mathbf{f_2}^{\intercal} \mathbf{f_1} & \mathbf{p}_2 \mathbf{f_2}^{\intercal} \mathbf{f_2} & \cdots & \sqrt{\mathbf{p}_2 \mathbf{p}_{n_t}} \mathbf{f_2}^{\intercal} \mathbf{f_{n_t}} \\ \vdots & \vdots & & \vdots \\ \sqrt{\mathbf{p}_{n_t} \mathbf{p}_1} \mathbf{f_{n_t}}^{\intercal} \mathbf{f_1} & \sqrt{\mathbf{p}_{n_t} \mathbf{p}_2} \mathbf{f_{n_t}}^{\intercal} \mathbf{f_2} & \cdots & \mathbf{p}_{n_t} \mathbf{f_{n_t}}^{\intercal} \mathbf{f_{n_t}} \end{bmatrix}^2 \right)^{-1} \tag{2.160}$$

$$= \left(\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mathbf{p}_i \mathbf{p}_j (\mathbf{f_i}^{\intercal} \mathbf{f_j})^2 \right)^{-1} = \left(\sum_{i=1}^{n_t} \sum_{j=1}^{n_t} \mathbf{p}_i \mathbf{p}_j \mathbf{S}_{ij} \right)^{-1}, \tag{2.161}$$

which proves the theorem. ■

Theorem 5. *For $q = \infty$, the OLC diversity can be expressed as*

$$^{\infty}\mathrm{D}_{OLC}(\mathbf{\Omega}, \mathbf{p}) = \left(\max_{f} \sum_{i=1}^{n_t} \mathbf{p}_i s(\mathbf{f}, \mathbf{f_i}) \right)^{-1}. \tag{2.162}$$

Proof. By applying the definition of the OLC diveristy we get

$$^{\infty}\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega}(\mathbf{p})) = \frac{1}{\boldsymbol{\lambda}_1(\mathbf{\Omega}(\mathbf{p}))} = \frac{1}{\boldsymbol{\lambda}_1(\boldsymbol{\rho})} = \frac{1}{\max_{\mathbf{f}} \mathbf{f}^{\intercal}\boldsymbol{\rho}\mathbf{f}} = \frac{1}{\max_{\mathbf{f}} \mathbf{f}^{\intercal} \left(\sum_{i=1}^{n_t} \mathbf{p}_i \mathbf{f_i} \mathbf{f_i}^{\intercal}\right) \mathbf{f}} \tag{2.163}$$

$$= \left(\max_{\mathbf{f}} \sum_{i=1}^{n_t} \mathbf{p}_i (\mathbf{f}^{\intercal}\mathbf{f_i})^2 \right)^{-1} = \left(\max_{\mathbf{f}} \sum_{i=1}^{n_t} \mathbf{p}_i s(\mathbf{f}, \mathbf{f_i}) \right)^{-1}, \tag{2.164}$$

which proves the theorem. Note that $\mathbf{\Omega}(\mathbf{p})$ and $\boldsymbol{\rho}$ have the same non-zero eigenvalues. ■

2.H Properties of the OLC diversity

Here we prove that the OLC diversity satisfies the modularity property, the strong replication principle and the twin property. We also derive an upperbound with respect to the similarity of the two most dissimilar species.

2.H.1 Modularity

Theorem 6. *If the entity models belonging to the set $\mathcal{E} = \{E_i\}_{i=1}^{n_w}$ are completely distinct (meaning they have pairwise orthogonal feature sets), then*

$$^{q}\mathrm{D}_{OLC}\left(\sum_{i=1}^{n_w} \mathbf{w}_i E_i, s \right) = \left(\sum_{i=1}^{n_w} \mathbf{w}_i^q \left(^{q}\mathrm{D}_{OLC}\left(E_i, s\right)\right)^{1-q} \right)^{\frac{1}{1-q}}. \tag{2.165}$$

Proof. Suppose that $\mathbf{\Omega}^{(1)}\left(\mathbf{p^{(1)}}\right), \ldots, \mathbf{\Omega}^{(n_\mathbf{w})}\left(\mathbf{p}^{(n_\mathbf{w})}\right)$ and $\mathbf{\Omega}(\mathbf{p})$ are the weighted kernel matrices of $E_1, \ldots, E_{n_\mathbf{w}}$ and $E = \sum_{i=1}^{n_w} \mathbf{w}_i E_i$, respectively, then

$$\mathbf{\Omega}(\mathbf{p}) = \begin{bmatrix} \mathbf{w}_1 \mathbf{\Omega}^{(1)}\left(\mathbf{p^{(1)}}\right) & 0 & \cdots & 0 \\ 0 & \mathbf{w}_2 \mathbf{\Omega}^{(2)}\left(\mathbf{p^{(2)}}\right) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{w}_{n_w} \mathbf{\Omega}^{(n_\mathbf{w})}\left(\mathbf{p}^{(n_\mathbf{w})}\right) \end{bmatrix} \tag{2.166}$$

because the feature vectors belonging to completely distinct models have an inner product equal to zero. The eigenvalues of a block diagonal matrix are the union of the eigenvalues of the main diagonal blocks. Therefore, the vector containing the eigenvalues of $\boldsymbol{\Omega}(\mathbf{p})$ becomes

$$\boldsymbol{\lambda} = [\mathbf{w}_1\boldsymbol{\lambda}^{(\mathbf{1})}; \ldots; \mathbf{w}_{n_w}\boldsymbol{\lambda}^{(\boldsymbol{n_w})}], \tag{2.167}$$

where $\boldsymbol{\lambda}^{(\boldsymbol{i})}$ contains the eigenvalues of $\boldsymbol{\Omega}^{(i)}\left(\mathbf{p}^{(i)}\right)$ for $i \in \{1, \ldots, n_w\}$. The OLC diversity is the type-neutral diversity applied to the eigenvalues of the weighted kernel matrix. The modularity property of the OLC diversity can thus be reformulated as

$${}^{q}\mathrm{D}(\boldsymbol{\lambda}) = \left(\sum_{i=1}^{n_w} \mathbf{w}_i^q \left({}^{q}\mathrm{D}\left(\boldsymbol{\lambda}^{(\boldsymbol{i})}\right)\right)^{1-q}\right)^{\frac{1}{1-q}}, \tag{2.168}$$

where ${}^{q}\mathrm{D}(\boldsymbol{\lambda}^{(\boldsymbol{i})}) = {}^{q}\mathrm{D}_{\mathrm{OLC}}(E_i, s)$ for $i \in \{1, \ldots, n_w\}$, and ${}^{q}\mathrm{D}(\boldsymbol{\lambda}) = {}^{q}\mathrm{D}_{\mathrm{OLC}}(E, s)$. Equation (2.168) corresponds to the modularity property of the type-neutral diversity where the relative abundances are replaced by the eigenvalues of weighted kernel matrices. Because the modularity property holds in the type-neutral case, it thus also holds in the OLC framework. ■

2.H.2 Strong replication principle

Theorem 7. *If the entity models belonging to the set $\mathcal{E} = \{E_i\}_{i=1}^{n_w}$ are completely distinct and have the same OLC diversity which equals d, then*

$${}^{q}\mathrm{D}_{OLC}\left(\sum_{i=1}^{n_w} \frac{1}{n_w} E_i, s\right) = n_w d. \tag{2.169}$$

Proof. Applying Theorem 6 with equal weights and equal diversities results in

$$\left({}^{q}\mathrm{D}_{\mathrm{OLC}}(E, s)\right)^{1-q} = \sum_{i=1}^{n_w} \frac{1}{n_w^q} d^{1-q} \tag{2.170}$$

$$= d^{1-q} \frac{1}{n_w^{q-1}} \sum_{i=1}^{n_w} \frac{1}{n_w} \tag{2.171}$$

$$= (n_w d)^{1-q}. \tag{2.172}$$

Removing the power $1-q$ at both sides of the equation proves the theorem. ■

2.H.3 Twin property

Theorem 8. *If type i is equal to type j, then removing type j and adding its relative abundance to type i, doesn't change the OLC diversity.*

Proof. If $\mathbf{p}$ is the abundance vector before removing the duplicate and $\mathbf{q}$ is the abundance vector after removing the duplicate, then

$$\mathbf{q}_k = \begin{cases} \mathbf{p}_k, & k \in \{1, \ldots, n_t\}, k \neq i, k \neq j, \\ \mathbf{p}_i + \mathbf{p}_j & k = i, \\ 0 & k = j. \end{cases} \tag{2.173}$$

The OLC diversity is the the type-neutral diversity applied to the eigenvalues of the second moment matrix. The second moment matrix corresponding to abundance vector $\mathbf{q}$ becomes

$$\boldsymbol{\rho}' = \boldsymbol{\rho} - \mathbf{p}_i \mathbf{f_i} \mathbf{f_i}^\mathsf{T} - \mathbf{p}_j \mathbf{f_j} \mathbf{f_j}^\mathsf{T} + (\mathbf{p}_i + \mathbf{p}_j) \mathbf{f_i} \mathbf{f_i}^\mathsf{T} = \boldsymbol{\rho}, \tag{2.174}$$

where $\boldsymbol{\rho}$ is the second moment matrix corresponding to abundance vector $\mathbf{p}$. Because the second matrix remains unchanged after removing the duplicate, the OLC diversity doesn't change either. ■

2.H.4 Upperbound

Lemma 1. *The OLC diversity has the following upperbound*

$${}^q\mathrm{D}_{OLC}(E, s) \leq \left((n_t - 1) \left(\frac{1 - \sqrt{s_m}}{n_t} \right)^q + \left(\sqrt{s_m} + \frac{1 - \sqrt{s_m}}{n_t} \right)^q \right)^{\frac{1}{1-q}}, \tag{2.175}$$

where s_m is the similarity between the two most dissimilar types, and n_t is the number of types. For q equal to 0, 1 and ∞ this becomes

$${}^0\mathrm{D}_{OLC}(E, s) \leq n_t, \tag{2.176}$$

$${}^1\mathrm{D}_{OLC}(E, s) \leq \exp\Bigg(- (n_t - 1) \frac{1 - \sqrt{s_m}}{n_t} \ln \left(\frac{1 - \sqrt{s_m}}{n_t} \right) - \left(\sqrt{s_m} + \frac{1 - \sqrt{s_m}}{n_t} \right) \ln \left(\sqrt{s_m} + \frac{1 - \sqrt{s_m}}{n_t} \right) \Bigg), \tag{2.177}$$

$${}^\infty\mathrm{D}_{OLC}(E, s) \leq \frac{1}{\sqrt{s_m} + \frac{1 - \sqrt{s_m}}{n_t}}. \tag{2.178}$$

Proof. The largest eigenvalue of the weighted kernel matrix $\mathbf{\Omega}(\mathbf{p})$ is denoted by $\boldsymbol{\lambda}_1\left(\mathbf{\Omega}(\mathbf{p})\right)$, and this eigenvalue equals at least $\sqrt{s_m} + \frac{1-\sqrt{s_m}}{n_t}$ because

$$\boldsymbol{\lambda}_1\left(\mathbf{\Omega}(\mathbf{p})\right) \geq \left(\odot^{\frac{1}{2}}\mathbf{p}\right)^{\intercal} \mathbf{\Omega}(\mathbf{p}) \left(\odot^{\frac{1}{2}}\mathbf{p}\right) \tag{2.179}$$

$$= \mathbf{p}^{\intercal}\mathbf{\Omega}\mathbf{p} \tag{2.180}$$

$$\geq \mathbf{p}^{\intercal}\mathbf{\Omega}^{(s_m)}\mathbf{p} \tag{2.181}$$

$$\geq \mathbf{v}^{\intercal}\mathbf{\Omega}^{(s_m)}\mathbf{v} \tag{2.182}$$

$$= \sqrt{s_m} + \frac{1-\sqrt{s_m}}{n_t}, \tag{2.183}$$

where $\mathbf{v} = \left[\frac{1}{n_t}; \ldots; \frac{1}{n_t}\right]$, and $\mathbf{\Omega}^{(s_m)} \in \mathbb{R}^{n_t \times n_t}$ has diagonal elements equal to 1 and off-diagonal elements equal to $\sqrt{s_m}$. Equation (2.179) holds because the largest eigenvalue of $\mathbf{\Omega}(\mathbf{p})$ equals the optimum of the following optimization problem

$$\max_{\mathbf{u}\in\mathbb{R}^{n_t}} \mathbf{u}^{\intercal}\mathbf{\Omega}(\mathbf{p})\mathbf{u} \quad \text{subject to } \|\mathbf{u}\|_2 = 1. \tag{2.184}$$

The largest eigenvalue is thus greater than or equal to $\left(\odot^{\frac{1}{2}}\mathbf{p}\right)^{\intercal} \mathbf{\Omega}(\mathbf{p}) \left(\odot^{\frac{1}{2}}\mathbf{p}\right)$ because $\| \odot^{\frac{1}{2}} \mathbf{p}\|_2 = 1$. Equation (2.181) holds because $\mathbf{\Omega}_{ij} \geq \mathbf{\Omega}_{ij}^{(s)}$ and $\mathbf{p}_i\mathbf{p}_j \geq 0$ for $i, j \in [1, \ldots, n_t]$. Equation (2.182) holds because $\mathbf{v}$ minimizes the following optimization problem in $\mathbf{p}$

$$\min_{\mathbf{p}\in\mathbb{R}^{n_t}_{\geq 0}} \mathbf{p}^{\intercal}\mathbf{\Omega}^{(s)}\mathbf{p} = \sqrt{s_m} + (1-\sqrt{s_m})\mathbf{p}^{\intercal}\mathbf{p} \quad \text{subject to } \sum_{i=1}^{n_t} \mathbf{p}_i = 1, \mathbf{p} \geq 0. \tag{2.185}$$

In order to maximize the diversity, the remaining eigenvalues should be equal because

$$\max_{\boldsymbol{\lambda}_2,\ldots,\boldsymbol{\lambda}_{n_t}} \left(\boldsymbol{\lambda}_1^q + \sum_{i=2}^{n_t} \boldsymbol{\lambda}_i^q\right)^{\frac{1}{1-q}} \text{ subject to } \boldsymbol{\lambda}_2, \ldots, \boldsymbol{\lambda}_{n_t} \in \mathbb{R}_{\geq 0}, \quad \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i = 1 \tag{2.186}$$

is maximized when $\boldsymbol{\lambda}_i = \frac{1-\boldsymbol{\lambda}_1}{n_t-1}$ for $i \in \{2, \ldots, n_t\}$. By minimizing the largest eigenvalue and equalizing the remaining eigenvalues, we can conclude that diversity is maximized if the eigenvalues of the weighted kernel matrix equal

$$\boldsymbol{\lambda} = \left[\sqrt{s_m} + \frac{1-\sqrt{s_m}}{n_t}; \frac{1-\sqrt{s_m}}{n_t}; \ldots; \frac{1-\sqrt{s_m}}{n_t}\right]. \tag{2.187}$$

The weighted kernel matrix has these eigenvalues when the similarity between each pair of types equals s_m and the abundance vector $\mathbf{p}$ is uniform. In this case the diversity equals the upperbound. ■

Theorem 9. *If the similarity between each pair of types is positive, then for $q > 1$ the OLC diversity has the following upperbound*

$$ {}^{q}\mathrm{D}_{OLC}(E, s) \leq s_m^{-\frac{q}{2(q-1)}}, \tag{2.188} $$

where s_m is the similarity between the two most dissimilar types. For $0 \leq q \leq 1$, no finite upperbound exists.

Proof. Taking the limit for n_t going to infinity in Lemma 1 proves the theorem. ■

2.I Richness optimization problem

We will investigate which kind of optimization problem we are dealing with when calculating the OLC richness. As proven in Appendix 2.I.1, ${}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))$ is concave in $\mathbf{p}$ for $q = 0$, strictly concave for $0 < q \leq 1$, strictly quasiconcave for $1 < q < \infty$ and quasiconcave for $q = \infty$. Because ${}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))$ is a quasiconcave function with respect to $\mathbf{p}$, every local optimum is also a global optimum. For $q = 0$, the solution of the optimization problem equals the rank of the kernel matrix. The Hartley richness, ${}^{0}\mathrm{R}_{\mathrm{OLC}}$, thus equals the size of the largest subset of linearly independent feature vectors. To calculate the Shannon OLC richness, we apply gradient ascent. Due to the quasiconcavity property, gradient ascent will not converge to a local maximum that isn't a global maximum. For $q = 2$, the OLC richness is the inverse of the optimum of a quadratic programming (QP) problem

$$ {}^{2}\mathrm{R}_{\mathrm{OLC}}(\boldsymbol{\Omega}) = \left({}^{2}O\right)^{-1} \text{ where } {}^{2}O = \min_{\mathbf{p}} \mathbf{p}^{\intercal}(\odot^{2}\boldsymbol{\Omega})\mathbf{p} \tag{2.189} $$

$$ \text{subject to } \sum_{i=1}^{n_t} \mathbf{p}_i = 1, \;\; \mathbf{p} \geq 0. $$

The notation $\odot^{2}\boldsymbol{\Omega}$ expresses the element-wise square of matrix $\boldsymbol{\Omega}$, i.e., $(\odot^{2}\boldsymbol{\Omega})_{ij} = \boldsymbol{\Omega}_{ij}^{2}$. A lot of research has been done with regard to solving QP problems. In general, QP problems can be solved relatively fast. We use CVX [49, 50], a package for specifying and solving convex programs, to solve this optimization problem. For $q = \infty$, the OLC richness equals the inverse of the optimum of the optimization problem in which the largest eigenvalue of the weighted kernel matrix is minimized

$$ {}^{\infty}\mathrm{R}_{\mathrm{OLC}}(\boldsymbol{\Omega}) = \left({}^{\infty}O\right)^{-1} \text{ where } {}^{\infty}O = \min_{\mathbf{p}} \boldsymbol{\lambda}_1(\boldsymbol{\Omega}(\mathbf{p})) \tag{2.190} $$

$$\text{subject to } \sum_{i=1}^{n_t} \mathbf{p}_i = 1, \;\; \mathbf{p} \geq 0.$$

The notation $\boldsymbol{\lambda}_1(\mathbf{A})$ expresses the largest eigenvalue of the matrix $\mathbf{A}$. To solve this optimization problem, we use CVX as well.

2.I.1 Quasi-concavity of the OLC diversity

Lemma 2. *Of all positive semidefinite symmetric matrices that have the following diagonal elements* $\mathbf{d} = \{\mathbf{d}_1, \ldots, \mathbf{d}_{n_t}\}$*, the diagonal matrix maximizes* $f(\boldsymbol{\lambda}(\mathbf{A}))$ *in* $\mathbf{A}$ *where* $\boldsymbol{\lambda}(\mathbf{A})$ *is the vector containing the eigenvalues of* $\mathbf{A}$*, and* f *is quasiconcave function that is symmetric in these eigenvalues.*

Proof. According to the Shur-Horn theorem [51, 52], $\mathbf{d}$ is in the convex hull of the $n_t!$ vectors that are permutations of the coordinates of $\boldsymbol{\lambda}(\mathbf{A})$. Therefore the quasiconcavity implies that $f(\mathbf{d}) \geq f(\boldsymbol{\lambda}(\mathbf{A}))$, and $f(\boldsymbol{\lambda}(\mathbf{A}))$ is thus maximized when it equals $f(\mathbf{d})$ which is the case for the diagonal matrix. ■

Lemma 3. *If* $\boldsymbol{\lambda}(\mathbf{A})$ *is the vector containing the eigenvalues of* $\mathbf{A}$ *and* $f(\boldsymbol{\lambda}(\mathbf{A}))$ *is a strictly quasiconcave function that is symmetric in these eigenvalues, then the positive semidefinite symmetric matrix with diagonal elements* $\mathbf{d} = \{\mathbf{d}_1, \ldots, \mathbf{d}_{n_t}\}$ *that maximizes* $f(\boldsymbol{\lambda}(\mathbf{A}))$ *in* $\mathbf{A}$ *does not have non-zero off-diagonal elements.*

Proof. The only positive semidefinite symmetric matrix with eigenvalues equal to the diagonal elements is the diagonal matrix. Therefore, for all other positive semidefinite symmetric matrices $\mathbf{A}'$ the diagonal $\mathbf{d}$ is a linear combination of the $n_t!$ vectors formed by permuting the coordinates of $\boldsymbol{\lambda}(\mathbf{A}')$ where the linear combination is not of the form $\mathbf{e}^{(\mathbf{i})}$, where $\mathbf{e}^{(\mathbf{i})} \in \mathbb{R}^{n_t!}$ is a vector that contains only zeros, except for the i-th element, which equals one. The strict quasiconcavity thus implies that $f(\mathbf{d}) > f(\boldsymbol{\lambda}(\mathbf{A}'))$. ■

Lemma 4. *In the domain* $\mathbf{p} \in \mathbb{R}^{n_t}_{\geq 0}$*, the function* $g_q(\mathbf{p})$*, defined as*

$$g_q(\mathbf{p}) = \begin{cases} \sum_{i=1}^{n} -\mathbf{p}_i \ln(\mathbf{p}_i) & q = 1, \\ \sum_{i=1}^{n} \mathbf{p}_i^q & q \in [0, 1[\, \cup \,]1, \infty[, \end{cases} \tag{2.191}$$

is concave for $q = 0$*, strictly concave for* $0 < q \leq 1$ *and strictly convex for* $1 < q < \infty$*.*

Proof. The sum of concave functions is concave, the sum of strictly concave functions is strictly concave and the sum of strictly convex functions is strictly convex. ■

Lemma 5. *In the domain of symmetric positive semidefinite matrices, the function*

$$f_q(\boldsymbol{\rho}) = \begin{cases} \sum_{i=1}^{n_t} -\operatorname{tr}(\boldsymbol{\rho}\ln(\boldsymbol{\rho})) & q = 1, \\ \sum_{i=1}^{n_t} \operatorname{tr}(\boldsymbol{\rho}^q) & q \in [0,1[\ \cup\ ]1,\infty[\end{cases} \tag{2.192}$$

is concave for $q = 0$*, strictly concave for* $0 < q \leq 1$ *and strictly convex for* $1 < q < \infty$*.*

Proof. We will prove strict concavity for $0 < q \leq 1$. Strict convexity for $1 < q < \infty$ and concavity for $q = 0$ can be proven in a similar way. Note that $f_q(\boldsymbol{\rho}) = g_q(\boldsymbol{\lambda}(\boldsymbol{\rho}))$ where $\boldsymbol{\lambda}(\boldsymbol{\rho})$ is the vector containing the eigenvalues of $\boldsymbol{\rho}$ and g_q is defined as in Lemma 4. To prove strict concavity, we have to show that for $0 < q \leq 1$ the following holds

$$f_q(t\boldsymbol{\rho}_A + (1-t)\boldsymbol{\rho}_B) > t f_q(\boldsymbol{\rho}_A) + (1-t) f_q(\boldsymbol{\rho}_B), \tag{2.193}$$

where $0 < t < 1$ and $\boldsymbol{\rho}_A \neq \boldsymbol{\rho}_B$. Suppose that

- $\boldsymbol{\rho}_t = t\boldsymbol{\rho}_A + (1-t)\boldsymbol{\rho}_B$,
- $\boldsymbol{\rho}_t$ has the following eigenvalue decomposition $\boldsymbol{\rho}_t = \mathbf{P}\mathbf{D}\mathbf{P}^\intercal$ with $\mathbf{P}$ an orthogonal matrix and $\mathbf{D}$ a diagonal matrix,
- $\mathbf{S_A} = \mathbf{P}^\intercal \boldsymbol{\rho_A} \mathbf{P}$,
- $\mathbf{S_B} = \mathbf{P}^\intercal \boldsymbol{\rho_B} \mathbf{P}$,
- $\mathbf{S_A^D}$, $\mathbf{S_B^D}$ have the same diagonal elements as $\mathbf{S_A}$ and $\mathbf{S_B}$, respectively, but they have off-diagonal elements equal to zero,

then

$$\begin{aligned} f_q(\boldsymbol{\rho}_t) &= f_q(\mathbf{D}) && (2.194)\\ &= g_q(\operatorname{diag}(\mathbf{D})) && (2.195)\\ &\geq t g_q(\operatorname{diag}(\mathbf{S_A^D})) + (1-t) g_q(\operatorname{diag}(\mathbf{S_B^D})) && (2.196)\\ &= t f_q(\mathbf{S_A^D}) + (1-t) f_q(\mathbf{S_B^D}) && (2.197)\\ &\geq t f_q(\mathbf{S_A}) + (1-t) f_q(\mathbf{S_B}) && (2.198)\\ &= t f_q(\boldsymbol{\rho_A}) + (1-t) f_q(\boldsymbol{\rho_B}), && (2.199) \end{aligned}$$

where $\operatorname{diag}(\mathbf{A})$ is the vector containing the diagonal elements of the square matrix $\mathbf{A}$. Equality (2.194) holds because $\mathbf{D}$ has the same eigenvalues as $\boldsymbol{\rho}_t$.

Equality (2.195) and (2.197) hold because the function f_q of a diagonal matrix equals the function g_q of its diagonal elements. Inequality (2.196) holds because the diagonal matrices $\mathbf{D}$, $\mathbf{S}_\mathbf{A}^\mathbf{D}$ and $\mathbf{S}_\mathbf{B}^\mathbf{D}$ satisfy $\mathbf{D} = t\mathbf{S}_\mathbf{A}^\mathbf{D} + (1-t)\mathbf{S}_\mathbf{B}^\mathbf{D}$, and because g_q is strictly concave for $0 \leq q \leq 1$ (see Lemma 4). Inequality (2.198) holds because of Lemma 2. Equality (2.199) holds because $\boldsymbol{\rho_A}$ and $\boldsymbol{\rho_B}$ have the same eigenvalues as $\mathbf{S_A}$ and $\mathbf{S_B}$, respectively.

Inequality (2.196) becomes an equality if and only if $\mathbf{S}_\mathbf{A}^\mathbf{D} = \mathbf{S}_\mathbf{B}^\mathbf{D}$. Inequality (2.198) becomes and equality if and only if $\mathbf{S_A}$ and $\mathbf{S_B}$ have off-diagonal elements equal to zero (see Lemma 3). Both inequalities thus becomes an equality if and only if $\boldsymbol{\rho_A} = \boldsymbol{\rho_B}$. Therefore, if $\boldsymbol{\rho}_A \neq \boldsymbol{\rho}_B$, then

$$f_q(\boldsymbol{\rho}_t) > t f_q(\boldsymbol{\rho_A}) + (1-t) f_q(\boldsymbol{\rho_B}).$$ ■

Theorem 10. *The diversity* ${}^q\mathrm{D}_{OLC}(\boldsymbol{\rho}(\mathbf{p}))$ *is concave in* $\mathbf{p}$ *for* $q = 0$*, strictly concave for* $0 < q \leq 1$*, strictly quasiconcave for* $1 < q < \infty$ *and quasiconcave for* $q = \infty$*.*

Proof. Lemma 5 states that in the domain of symmetric positive semidefinite matrices, the function

$$f_q(\boldsymbol{\rho}) = \begin{cases} \sum_{i=1}^{n_t} -\operatorname{tr}(\boldsymbol{\rho}\ln(\boldsymbol{\rho})) & q = 1, \\ \sum_{i=1}^{n_t} \operatorname{tr}(\boldsymbol{\rho}^q) & q \in [0,1[\ \cup\ ]1,\infty[\end{cases} \tag{2.200}$$

is concave for $q = 0$, strictly concave for $0 < q \leq 1$ and strictly convex for $1 < q < \infty$. The OLC diversity can be expressed as follows with respect to $f_q(\boldsymbol{\rho})$

$${}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}) = \begin{cases} \exp(f_q(\boldsymbol{\rho})) & q = 1, \\ f_q(\boldsymbol{\rho})^{\frac{1}{1-q}} & q \in [0,1[\ \cup\ ]1,\infty[. \end{cases} \tag{2.201}$$

A strictly increasing convex function of a strictly concave function is strictly concave and therefore ${}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho})$ becomes strictly concave in $\boldsymbol{\rho}$ for $0 < q \leq 1$. A strictly decreasing function of a strictly convex function is strictly quasiconcave function and therefore ${}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho})$ becomes strictly quasiconcave in $\boldsymbol{\rho}$ for $1 < q < \infty$. A nondecreasing convex function of a concave function is concave and therefore ${}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho})$ becomes concave in $\boldsymbol{\rho}$ for $q = 0$. Minimizing the largest eigenvalue is a convex optimization problem, therefore ${}^\infty\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho})$, which is the inverse of the largest eigenvalue, is a quasiconcave function. The concavity and quasiconcavity with respect to $\mathbf{p}$ holds because $\boldsymbol{\rho}(\mathbf{p})$ is linear in $\mathbf{p}$. ■

2.J Properties of the richness

When applying an index in the general framework, we first assign vectors of unit norm - which we call the feature vectors - to each type such the similarity

between each pair of types equals the square inner product of these feature vectors. We then apply the index to the feature vectors. When proving the properties of the OLC richness, we will therefore first translate the properties with respect to the types into properties with respect to the feature vectors (see Table 2.5 for a translation table), and we then prove the latter properties. The first eight properties in Table 2.8 get translated as follows

- **Interpretability:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) = n_t$ if each pair of feature vectors in $\mathcal{F}$ is orthogonal.
- **Monotonically increasing:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \cup \{\mathbf{f}'\}) \geq {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})$
- **Strictly monotonically increasing:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \cup \{\mathbf{f}'\}) > {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})$ if $\mathbf{f}'$ is not a linear combination of the feature vectors belonging to the set $\mathcal{F}$.
- **Non-changing:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \cup \{\mathbf{f}'\}) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})$ if $\mathbf{f}'$ already occurs in the set $\mathcal{F}$.
- **Subadditivity:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \mathcal{F}_2) \leq {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2)$
- **Strict subadditivity:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \mathcal{F}_2) < {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2)$ if not every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$.
- **Additivity:** ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \mathcal{F}_2) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2)$ if every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$.

Note that the upperbound property is trivial to proof. The OLC richness equals the OLC diversity where the relative abundances are chosen in a specific way. The upperbound of the OLC diversity holds for any abundance vector and thus also for the specific abundance vector used when calculating the the OLC richness. Before proving the remaining properties, we show that sets are a natural mathematical representation for a groups of feature vectors.

Lemma 6. *If a certain feature vector occurs multiple times, e.g.,* $\mathbf{f_i} = \mathbf{f_j}$*, then removing one of the duplicates, e.g.,* $\mathbf{f_j}$*, does not change the OLC richness.*

Proof. For every $\mathbf{p}$ there exists a $\mathbf{q}$ such that $\boldsymbol{\rho}(\mathbf{q}) = \boldsymbol{\rho}(\mathbf{p})$ and $\mathbf{q}_j = 0$, namely

$$\mathbf{q}_k = \begin{cases} \mathbf{p}_k, & k \in \{1, \ldots, n_t\}, k \neq i, k \neq j, \\ \mathbf{p}_i + \mathbf{p}_j & k = i, \\ \mathbf{q}_j = 0 & k = j. \end{cases} \tag{2.202}$$

For every $\mathbf{p}$ there thus exists a $\mathbf{q}$ such that $\mathbf{q}_j = 0$ and ${}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{q})) = {}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))$. Therefore, removing $\mathbf{f_j}$ from the set does not lower the richness. ■

It is not important how many times a feature vector is contained by the group of features, but only whether it is contained by the feature vector group or not. Therefore we use sets based on the fundamental binary relation 'is an element of' to represent groups of feature vectors. Note that the sequence in which we proof the properties, does not correspond to the sequence in which we list the properties in the paper. This is because in the proof of certain properties we use the fact that other properties hold. Those other properties then need to be proven first of course.

2.J.1 Monotonically increasing

Theorem 11. *The OLC richness satisfies*

$$ {}^{q}\mathrm{R}_{OLC}(\mathcal{F} \cup \{\mathbf{f}'\}) \geq {}^{q}\mathrm{R}_{OLC}(\mathcal{F}). \tag{2.203} $$

Proof. Suppose that the abundance vector $\hat{\mathbf{p}}$ maximizes the diversity of the feature set $\mathcal{F}$, then

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) = {}^{q}\mathrm{D}_{\mathrm{OLC}}\left(\boldsymbol{\rho}(\hat{\mathbf{p}})\right) \tag{2.204} $$

$$ = {}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}'([\hat{\mathbf{p}}; 0])) \tag{2.205} $$

$$ \leq {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \{\mathbf{f}'\}), \tag{2.206} $$

where $\boldsymbol{\rho}$ is the second moment matrix corresponding to $\mathcal{F}$, and $\boldsymbol{\rho}'$ is the second moment matrix corresponding to $\mathcal{F}' = \mathcal{F}_1 \cup \{\mathbf{f}'\}$. Equation (2.205) holds because adding a type with abundances equal to zero doesn't change the diversity. Equation (2.206) holds because the diversity of a model is never greater than its richness. ■

2.J.2 Non-changing

Theorem 12. *The OLC richness satisfies*

$$ {}^{q}\mathrm{R}_{OLC}(\mathcal{F} \cup \{\mathbf{f}'\}) = {}^{q}\mathrm{R}_{OLC}(\mathcal{F}) \tag{2.207} $$

if $\mathbf{f}'$ *already occurs in the set* $\mathcal{F}$*.*

Proof. The proof is trivial because $\mathcal{F} = \mathcal{F} \cup \{\mathbf{f}'\}$. ■

2.J.3 Strictly monotonically increasing

Order q is greater than one

Theorem 13. *For $q > 1$, the OLC richness does not always satisfy*

$$ {}^{q}\mathrm{R}_{OLC}(\mathcal{F} \cup \{\mathbf{f}'\}) > {}^{q}\mathrm{R}_{OLC}(\mathcal{F}) \tag{2.208} $$

when $\mathbf{f}'$ is not a linear combination of the feature vectors belonging to the set $\mathcal{F}$.

Proof. Suppose that $\mathcal{F} = \{[0.8, 0.6, 0]^\intercal, [0.8, -0.6, 0]^\intercal\}$, $\mathbf{f}' = [\sqrt{d}, 0, \sqrt{1-d}]^\intercal$ and $\mathcal{F}' = \mathcal{F} \cup \{\mathbf{f}'\}$. We will prove that if d satisfies $0.36\left(\frac{0.36}{0.64}\right)^{q-1} + 0.64 \leq d < 1$, then ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \cup \{\mathbf{f}'\}) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})$. Note that for these values d, $\mathbf{f}'$ is not a linear combination of the feature vectors belonging to the set $\mathcal{F}$. For $q > 1$, maximizing ${}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))$ is equivalent to minimizing

$$ \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i(\boldsymbol{\rho}(\mathbf{p}))^q, \tag{2.209} $$

where $\boldsymbol{\lambda}_1(\boldsymbol{\rho}(\mathbf{p})), \ldots, \boldsymbol{\lambda}_{n_t}(\boldsymbol{\rho}(\mathbf{p}))$ and $\mathbf{u}_1(\boldsymbol{\rho}(\mathbf{p})), \ldots, \mathbf{u}_{n_t}(\boldsymbol{\rho}(\mathbf{p}))$ are the eigenvalues and the corresponding eigenvectors of $\boldsymbol{\rho}(\mathbf{p})$. For $\mathbf{p}_1 = \mathbf{p}_2 = 0.5$ and $\mathbf{p}_3 = 0$, the second moment matrix corresponding to $\mathcal{F}'$ becomes

$$ \boldsymbol{\rho}'(\mathbf{p}) = \begin{bmatrix} 0.64 & 0 & 0 \\ 0 & 0.36 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.210} $$

Now we will show that for certain values of d, the OLC diversity is maximized in the relative abundances when $\hat{\mathbf{p}} = [0.5, 0.5, 0]^\intercal$. The eigenvalues and corresponding eigenvectors of the OLC diversity in $\hat{\mathbf{p}} = [0.5, 0.5, 0]^\intercal$ are

$$ \boldsymbol{\lambda}_1(\boldsymbol{\rho}'(\hat{\mathbf{p}})) = 0.64, \qquad \mathbf{u}_1(\boldsymbol{\rho}'(\hat{\mathbf{p}})) = [1, 0, 0]^\intercal, \tag{2.211} $$

$$ \boldsymbol{\lambda}_2(\boldsymbol{\rho}'(\hat{\mathbf{p}})) = 0.36, \qquad \mathbf{u}_2(\boldsymbol{\rho}'(\hat{\mathbf{p}})) = [0, 1, 0]^\intercal, \tag{2.212} $$

$$ \boldsymbol{\lambda}_3(\boldsymbol{\rho}'(\hat{\mathbf{p}})) = 0, \qquad \mathbf{u}_3(\boldsymbol{\rho}'(\hat{\mathbf{p}})) = [0, 0, 1]^\intercal. \tag{2.213} $$

In our scenario $\boldsymbol{\rho}'([0.5, 0.5, 0]^\intercal)$ has a simple spectrum (which means that all eigenvectors have multiplicity one), and we can thus apply the Hadamard first variation formula [53]

$$ \frac{\partial \boldsymbol{\lambda}_i(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_j} = q\boldsymbol{\lambda}_i(\boldsymbol{\rho}'(\mathbf{p}))^{q-1}(\mathbf{f_j}^\intercal \mathbf{u}_i(\boldsymbol{\rho}'(\mathbf{p})))^2. \tag{2.214} $$

This means that in $\hat{\mathbf{p}} = [0.5, 0.5, 0]^\intercal$ the derivative of each eigenvalue with respect to each element of $\mathbf{p}$ becomes

$$ \left.\frac{\partial \boldsymbol{\lambda}_1(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_1}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.64^q, \qquad \left.\frac{\partial \boldsymbol{\lambda}_2(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_1}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.36^q, \quad \left.\frac{\partial \boldsymbol{\lambda}_3(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_1}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = 0, $$

$$\left.\frac{\partial \boldsymbol{\lambda}_1(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_2}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.64^q, \quad \left.\frac{\partial \boldsymbol{\lambda}_2(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_2}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.36^q, \quad \left.\frac{\partial \boldsymbol{\lambda}_3(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_2}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = 0,$$
$$\left.\frac{\partial \boldsymbol{\lambda}_1(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_3}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.64^{q-1}d, \left.\frac{\partial \boldsymbol{\lambda}_2(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_3}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = 0, \quad \left.\frac{\partial \boldsymbol{\lambda}_3(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_3}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = 0. \tag{2.215}$$

The partial derivatives of Objective Function (2.209) with respect to each element of $\mathbf{p}$ equal

$$\left.\frac{\partial \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i((\boldsymbol{\rho}'(\mathbf{p})))^q}{\partial \mathbf{p}_1}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.64^q + q0.36^q, \tag{2.216}$$

$$\left.\frac{\partial \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i((\boldsymbol{\rho}'(\mathbf{p})))^q}{\partial \mathbf{p}_2}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.64^q + q0.36^q, \tag{2.217}$$

$$\left.\frac{\partial \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i((\boldsymbol{\rho}'(\mathbf{p})))^q}{\partial \mathbf{p}_3}\right|_{\mathbf{p}=\hat{\mathbf{p}}} = q0.64^{q-1}d. \tag{2.218}$$

Due to symmetry and convexity, we know that $\mathbf{p}_1 = \mathbf{p}_2$ in the optimum. We have an optimum in $\mathbf{p}_1 = \mathbf{p}_2 = 0.5$ and $\mathbf{p}_3 = 0$ if increasing $\mathbf{p}_3$ and decreasing $\mathbf{p}_1$ or $\mathbf{p}_2$ with an equal amount, doesn't decrease Objective Function (2.209)

$$\frac{\partial \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_3} \geq \frac{\partial \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_1} = \frac{\partial \sum_{i=1}^{n_t} \boldsymbol{\lambda}_i(\boldsymbol{\rho}'(\mathbf{p}))^q}{\partial \mathbf{p}_2} \tag{2.219}$$

$$\Leftrightarrow q0.64^{q-1}d \geq q0.64^q + q0.36^q \tag{2.220}$$

$$\Leftrightarrow d \geq 0.36\left(\frac{0.36}{0.64}\right)^{q-1} + 0.64. \tag{2.221}$$

If $d \geq 0.36\left(\frac{0.36}{0.64}\right)^{q-1}+0.64$, then $\mathbf{p}_3 = 0$ in the optimum and therefore ${}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \cup \{\mathbf{f}'\}) = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})$. If $d < 1$, then $\mathbf{f}'$ is not a linear combination of the feature vectors belonging to $\mathcal{F}$. We can conclude that for $1 > d \geq 0.36\left(\frac{0.36}{0.64}\right)^{q-1} + 0.64$, the property does not hold. ■

Order q is smaller than or equal to one

Lemma 7. *If we have two symmetric matrices $\boldsymbol{\rho} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{\rho}' \in \mathbb{R}^{m\times m}$ where*

$$\boldsymbol{\rho}' = \mathbf{V}^\intercal \boldsymbol{\rho} \mathbf{V}, \tag{2.222}$$

$\mathbf{V} \in \mathbb{R}^{n\times m}$ is a matrix with orthonormal columns, and the columns space of $\boldsymbol{\rho}$ is a subspace of the column space of $\mathbf{V}$, then $\boldsymbol{\rho}'$ has the same non-zero eigenvalues as $\boldsymbol{\rho}$.

Proof. First, we take the truncated Singular Value Decomposition (SVD) of $\boldsymbol{\rho}$

$$\boldsymbol{\rho} = \mathbf{U_t}\boldsymbol{\Lambda_t}\mathbf{U_t}^\intercal, \tag{2.223}$$

where $\mathbf{U_t} \in \mathbb{R}^{n\times k}$ has orthonormal columns, $\boldsymbol{\Lambda_t} \in \mathbb{R}^{k\times k}$ is a diagonal matrix with non-zero diagonal elements, and k is the number of non-zero eigenvalues that $\boldsymbol{\rho}$ has. Note that $\mathbf{U_t}$ and $\boldsymbol{\rho}$ have the same column space. We can now expresses $\boldsymbol{\rho'}$ as follows

$$\boldsymbol{\rho'} = \mathbf{U'_t}\boldsymbol{\Lambda_t}\mathbf{U'_t}^\intercal, \tag{2.224}$$

where $\mathbf{U'_t} = \mathbf{V}^\intercal\mathbf{U_t}$. Because the column space of $\mathbf{U_t}$ is a subspace of the column space of $\mathbf{V}$, projecting the columns of $\mathbf{U_t}$ on the space spanned by the columns of $\mathbf{V}$ doesn't change the columns of $\mathbf{U_t}$, and therefore

$$\mathbf{U_t} = \mathbf{V}\mathbf{V}^\intercal\mathbf{U_t}. \tag{2.225}$$

We can now derive that

$$\mathbf{U'_t}^\intercal\mathbf{U'_t} = \mathbf{U'_t}\mathbf{V}^\intercal\mathbf{V}\mathbf{U'_t} = \mathbf{U_t}^\intercal\mathbf{V}\mathbf{V}^\intercal\mathbf{V}\mathbf{V}^\intercal\mathbf{U_t} = \mathbf{U_t}^\intercal\mathbf{U_t} = I_{k\times k}, \tag{2.226}$$

where $I_{k\times k} \in \mathbb{R}^{k\times k}$ is the identity matrix. The matrix $\mathbf{U'_t}$ thus also has orthonormal columns, and Equation (2.224) is therefore the truncated SVD of $\boldsymbol{\rho'}$. Because the diagonal matrices of the truncated SVD of both $\boldsymbol{\rho'}$ and $\boldsymbol{\rho}$ are equal, $\boldsymbol{\rho'}$ and $\boldsymbol{\rho}$ have the same non-zero eigenvalues. ■

Lemma 8. *If we have a group of entities with abundance vector $\bar{\mathbf{p}}$ where type j has both an abundance equal to zero ($\bar{\mathbf{p}}_j = 0$) and a feature vector that is linearly independent of the feature vectors corresponding to non-zero abundances ($\mathbf{f_j} \not\subset span(F')$ where $\mathcal{F}'$ contains all feature vectors $\mathbf{f_i}$ for which $\bar{\mathbf{p}}_i \neq 0$), then for $0 < q \leq 1$*

$$\lim_{x\to 0_+} \left.\frac{\partial^q \mathrm{D}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_j}\right|_{\mathbf{p}=\bar{\mathbf{p}}'(x)} = \infty, \tag{2.227}$$

where $\bar{\mathbf{p}}'(x)$ has the same abundances as $\bar{\mathbf{p}}$ except for $\bar{\mathbf{p}}'_j(x)$ which equals x. Note that the sum of the elements of $\bar{\mathbf{p}}'(x)$ are not 1 anymore when $x \neq 0$.

Proof. We will do the proof for $q = 1$. The proof for $0 < q < 1$ is similar. If the eigenvalues of $\boldsymbol{\rho}(\mathbf{p})$, denoted by $\boldsymbol{\lambda}_1(\mathbf{p}) \geq \cdots \geq \boldsymbol{\lambda}_{n_y}(\mathbf{p})$, are strictly positive and have multiplicity one, then the partial derivative of the diversity with respect to $\mathbf{p}_j$ becomes

$$\frac{\partial^1 \mathrm{D}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_j} = {}^1\mathrm{D}(\boldsymbol{\rho}(\mathbf{p})) \sum_{i=1}^{n_t} \left(-1 - \ln(\boldsymbol{\lambda}_i(\mathbf{p}))\right) \frac{\partial \boldsymbol{\lambda}_i(\mathbf{p})}{\partial \mathbf{p}_j}. \tag{2.228}$$

The first problem we encounter is that not all eigenvalues of $\boldsymbol{\rho}(\bar{\mathbf{p}})$ are strictly positive (some equal zero) and therefore $\ln(\boldsymbol{\lambda}_i(\mathbf{p}))$ becomes undefined when $\boldsymbol{\lambda}_i(\mathbf{p}) = 0$. If we change the basis of the feature vectors to an orthonormal basis $\mathbf{V} = [\mathbf{v_1}, \ldots, \mathbf{v_{m+1}}]$ where $\mathbf{V}' = [\mathbf{v_1}, \ldots, \mathbf{v_m}]$ spans the same space as $\mathcal{F}'$, and $\mathbf{v_{m+1}}$ is the component of $\mathbf{f_j}$ orthogonal to $\mathcal{F}'$, then the new second moment matrix

$$\boldsymbol{\rho}'(\bar{\mathbf{p}}'(x)) = \mathbf{V}^\intercal \boldsymbol{\rho}(\bar{\mathbf{p}}'(x))\mathbf{V} \tag{2.229}$$

has the same non-zero eigenvalues as $\boldsymbol{\rho}(\bar{\mathbf{p}}'(x))$ (see Lemma 7). Moreover, if $x > 0$, then all eigenvalues of $\boldsymbol{\rho}'(\bar{\mathbf{p}}'(x))$ are strictly positive because the number of non-zero eigenvalues equals the size of the largest subset of linearly independent feature vectors with non-zero abundances, and thus equals $m + 1$. We choose the basis vectors $\mathbf{v_1}, \ldots, \mathbf{v_m}$ equal to the eigenvectors corresponding to the non-zero eigenvalues of $\boldsymbol{\rho}(\bar{\mathbf{p}}'(0))$, and $\boldsymbol{\rho}'(\bar{\mathbf{p}}'(0))$ thus becomes diagonal. However, $\boldsymbol{\rho}'(\bar{\mathbf{p}}'(0))$ has one eigenvalue equal to zero because $\bar{\mathbf{p}}'_j(0)$ is zero. To be able to apply Equation (2.228), we have to take the limit for x going to zero

$$\lim_{x \to 0_+} \left. \frac{\partial^1 \mathrm{D}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_j} \right|_{\mathbf{p}=\bar{\mathbf{p}}'(x)} \tag{2.230}$$

$$= {}^1\mathrm{D}(\boldsymbol{\rho}(\bar{\mathbf{p}})) \lim_{x \to 0_+} \sum_{i=1}^{m+1} (-1 - \ln(\boldsymbol{\lambda}'_i(\bar{\mathbf{p}}'(x)))) \frac{\partial \boldsymbol{\lambda}'_i(\bar{\mathbf{p}}'(x))}{\partial \bar{\mathbf{p}}'_j(x)}, \tag{2.231}$$

where $\boldsymbol{\lambda}'_1(\bar{\mathbf{p}}'(x)) > \cdots > \boldsymbol{\lambda}'_{m+1}(\bar{\mathbf{p}}'(x))$ are the eigenvalues of $\boldsymbol{\rho}'(\bar{\mathbf{p}}'(x))$. The eigenvalue $\boldsymbol{\lambda}_{m+1}(0)$, which equals zero, has multiplicity 1, and therefore we can apply the Hadamard first variation formula [53]

$$\left. \frac{\partial \boldsymbol{\lambda}'_{m+1}(\mathbf{p})}{\partial \mathbf{p}_j} \right|_{\mathbf{p}=\bar{\mathbf{p}}} = (\mathbf{v_{m+1}}^\intercal \mathbf{f_j})^2. \tag{2.232}$$

The space spanned by $\mathbf{v_1}, \ldots, \mathbf{v_m}$ equals the space spanned by $\mathcal{F}'$ and therefore $\mathbf{f_j}$ cannot be written as a linear combination of $\mathbf{v_1}, \ldots, \mathbf{v_m}$. This means that $(\mathbf{v_{m+1}}^\intercal \mathbf{f_j})^2$, and thus Equation (2.232), is strictly positive.

The second problem we encounter is that the derivative of an eigenvalue of the second moment matrix with respect to the abundances, does not exists when the eigenvalue has a multiplicity greater than one. However, the map $\mathbf{A} \to \boldsymbol{\lambda}_i(\mathbf{A})$ is Lipschitz continuous on the space of Hermitian matrices for $1 \leq i \leq m$ and therefore there exists a finite $K \in \mathbb{R}_{>0}$ such that for any eigenvalue, the following holds

$$K > \lim_{\Delta \mathbf{p}_j \to 0_+} \frac{\boldsymbol{\lambda}_i(\bar{p} + \Delta \mathbf{p}_j \mathbf{e^{(j)}}) - \boldsymbol{\lambda}_i(\boldsymbol{\rho}(\bar{\mathbf{p}}))}{\Delta \mathbf{p}_j} > -K. \tag{2.233}$$

We can now conclude that the following statement holds, even if $\boldsymbol{\rho}(\bar{\mathbf{p}})$) has eigenvalues equal to zero and eigenvalues with multiplicity larger than 1,

$$\lim_{x\to 0_+} \frac{\partial^1 \mathrm{D}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_j}\bigg|_{\mathbf{p}=\bar{\mathbf{p}}'(x)} > {}^1\mathrm{D}(\boldsymbol{\rho}(\bar{\mathbf{p}}))\Bigg(\Bigg(\sum_{i=1}^{m} -|-1-\ln(\boldsymbol{\lambda}_i)|\,K\Bigg) \tag{2.234}$$

$$+ \lim_{x\to 0_+} \left(-1-\ln(\boldsymbol{\lambda}_{m+1}(x))\right)\left(\mathbf{v_{m+1}}^{\intercal}\mathbf{f_j}\right)^2\Bigg)$$

$$= \infty, \tag{2.235}$$

where $\boldsymbol{\lambda}_i$ is the i-th greatest eigenvalue of $\boldsymbol{\rho}'(\bar{\mathbf{p}}'(0))$, and $\boldsymbol{\lambda}_{m+1}(x)$ is the smallest eigenvalue of $\boldsymbol{\rho}'(\bar{\mathbf{p}}'(x))$). Equation (2.235) holds because $\lim_{x\to 0_+}\left(-1-\ln(\boldsymbol{\lambda}_{m+1}(x))\right)=\infty$, $\left(\mathbf{v_{m+1}}^{\intercal}\mathbf{f_j}\right)^2>0$ and $-1-\ln(\boldsymbol{\lambda}_i)$ is finite for $1\le i\le m$. ■

Lemma 9. *If we have a group of entities with the abundance vector* $\bar{\mathbf{p}}$ *where type* k *has a non-zero abundance (*$\bar{\mathbf{p}}_k>0$*), then for* $0<q\le 1$ *holds that* $\frac{\partial^q \mathrm{D}_{OLC}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_k}$ *is finite in* $\hat{\mathbf{p}}=\bar{\mathbf{p}}$.

Proof. We will do the proof for $q=1$. The proof for $0<q<1$ is similar. If we change the basis of the feature vectors to an orthonormal basis $\mathbf{V}=[\mathbf{v_1},\ldots,\mathbf{v_m}]$ that spans the same space as the feature vectors corresponding to types with non-zero abundances, then the new second moment matrix

$$\boldsymbol{\rho}'(\bar{\mathbf{p}}) = \mathbf{V}^{\intercal}\boldsymbol{\rho}(\bar{\mathbf{p}})\mathbf{V} \tag{2.236}$$

has the same non-zero eigenvalues as $\boldsymbol{\rho}(\bar{\mathbf{p}})$. Moreover, $\boldsymbol{\rho}'(\bar{\mathbf{p}})$ has only non-zero eigenvalues because its dimensionality equals the size of the largest subset of linearly independent feature vectors with non-zero abundances which in its turn equals the number of non-zero eigenvalues. The derivative of the diversity with respect to $\mathbf{p}_k$ in $\bar{\mathbf{p}}$ is finite and equal to

$$\frac{\partial^1 \mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_k}\bigg|_{\mathbf{p}=\bar{\mathbf{p}}} = -{}^1\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}}))\left(\mathbf{f}_k^{\intercal}\mathbf{V}\Big(\boldsymbol{\rho}'(\bar{\mathbf{p}})\ln(\boldsymbol{\rho}'(\bar{\mathbf{p}}))\Big)\mathbf{V}^{\intercal}\mathbf{f}_k\right). \quad ■$$

Lemma 10. *If we have a group of entities with abundance vector* $\bar{\mathbf{p}}$ *where type* k *corresponds to a non-zero abundance, and type* j *has both an abundance equal to zero (*$\bar{\mathbf{p}}_j=0$*) and a feature vector that is linearly independent of the feature vectors corresponding to non-zero abundances (*$\mathbf{f_j}\not\subset span(F')$ *where* $\mathcal{F}'$ *contains all feature vectors* $\mathbf{f_i}$ *for which* $\bar{\mathbf{p}}_i\neq 0$*), then for* $0<q\le 1$ *holds*

$$\lim_{d\to 0_+}\left(\frac{{}^q\mathrm{D}_{OLC}(\boldsymbol{\rho}(\bar{\mathbf{p}}+d(\mathbf{e^{(j)}}-\mathbf{e^{(k)}})))-\mathrm{D}(\boldsymbol{\rho}(\bar{\mathbf{p}}))}{d}\right)=\infty, \tag{2.237}$$

where $\mathbf{e^{(i)}}\in\mathbb{R}^{n_t}$ *is a vector that only contains zeros, except for the i-th element, which equals one.*

Proof. Lemma 8 applies as long as $\mathbf{f_j} \not\subset \text{span}(F')$, therefore

$$\lim_{d_j \to 0+} \frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} + d_j\mathbf{e^{(j)}} - d_k\mathbf{e^{(k)}})) - {}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} - d_k\mathbf{e^{(k)}}))}{d_j} = \infty \quad (2.238)$$

for any $d_k \leq \bar{p}_k$. We can now conclude

$$\lim_{d \to 0_+} \frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} + d(\mathbf{e^{(j)}} - \mathbf{e^{(k)}}))) - {}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))}{d}$$

$$= \lim_{d_k \to 0+} \lim_{d_j \to d_k} \left(\frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} + d_j\mathbf{e^{(j)}} - d_k\mathbf{e^{(k)}})) - {}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} - d_k\mathbf{e^{(k)}}))}{d_k} \right. \quad (2.239)$$

$$\left. + \frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} - d_k\mathbf{e^{(k)}})) - {}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))}{d_k} \right)$$

$$= \left(\lim_{d_j \to 0+} \lim_{d_k \to d_j} \frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} + d_j\mathbf{e^{(j)}} - d_k\mathbf{e^{(k)}})) - {}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\bar{\mathbf{p}} - d_k\mathbf{e^{(k)}}))}{d_j} \right) \quad (2.240)$$

$$- \left. \frac{\partial\, \mathrm{D}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_k} \right|_{\mathbf{p}=\bar{\mathbf{p}}} \quad (2.241)$$

$$= \infty, \quad (2.242)$$

where Lemma 9 states that $\left. \frac{\partial {}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho}(\mathbf{p}))}{\partial \mathbf{p}_k} \right|_{\mathbf{p}=\bar{\mathbf{p}}}$ exists and is finite. ■

Theorem 14. *For $0 < q \leq 1$, the OLC richness satisfies*

$${}^q\mathrm{R}_{OLC}(\mathcal{F}) > {}^q\mathrm{R}_{OLC}(\mathcal{F} \setminus \mathbf{f_j}) \quad (2.243)$$

if $\mathbf{f_j}$ is not a linear combination of the other feature vectors belonging to the set $\mathcal{F}$.

Proof. Suppose that we have a group of entities with abundance vector $\bar{\mathbf{p}}$ where type k corresponds to a non-zero abundance, and type j has both an abundance equal to zero ($\bar{\mathbf{p}}_j = 0$) and a feature vector that is linearly independent of the feature vectors corresponding to non-zero abundances ($\mathbf{f_j} \not\subset \text{span}(F')$ where $\mathcal{F}'$ contains all feature vectors $\mathbf{f_i}$ for which $\bar{\mathbf{p}}_i \neq 0$). We can conclude from Lemma 10 and the epsilon-delta definition of the limit that for all $\epsilon > 0$, there exists a δ such that for all $0 < \bar{d} < \delta$

$$\left. \frac{\partial^1 \mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\Omega}(\bar{\mathbf{p}}'(d)))}{\partial d} \right|_{d=\bar{d}} > \epsilon, \quad (2.244)$$

where $\bar{\mathbf{p}}'(d) = \bar{\mathbf{p}} + d(\mathbf{e^{(j)}} - \mathbf{e^{(k)}})$. Concavity of ${}^1\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega}(\bar{\mathbf{p}}'(d)))$ in d and the fact there exists a $\bar{d} > 0$ for which the derivative is strictly positive imply ${}^1\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega}(\bar{\mathbf{p}}'(0))) < {}^1\mathrm{D}_{\mathrm{OLC}}(\mathbf{\Omega}(\bar{\mathbf{p}}'(\bar{d})))$. This means there does not exist an optimum for which $\mathbf{p}_j = 0$ and the richness thus strictly decreases when removing $\mathbf{f_j}$ or, equivalently, strictly increases when adding $\mathbf{f_j}$ to the set $\mathcal{F} \setminus \mathbf{f_j}$. ■

2.J.4 Additivity

Theorem 15. *The OLC richness satisfies*

$$ {}^q\mathrm{R}_{OLC}(\mathcal{F}_1 \cup \mathcal{F}_2) = {}^q\mathrm{R}_{OLC}(\mathcal{F}_1) + {}^q\mathrm{R}_{OLC}(\mathcal{F}_2) \tag{2.245}$$

if every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$

Proof. Suppose that $\mathbf{\Omega}_1(\mathbf{p}_1)$, $\mathbf{\Omega}_2(\mathbf{p}_2)$ and $\mathbf{\Omega}(\mathbf{p})$ are the weighted kernel matrices of $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$, respectively. We can formulate $\mathbf{p}$ as a weighted concatenation of $\mathbf{p^{(1)}}$ corresponding to $\mathbf{\Omega}_1$ and $\mathbf{p^{(2)}}$ corresponding to $\mathbf{\Omega}_2$ as follows

$$ \mathbf{p} = [w\mathbf{p^{(1)}}, (1-w)\mathbf{p^{(2)}}], \tag{2.246}$$

where

$$ 0 \leq w \leq 1, \;\; \mathbf{p^{(1)}} > 0, \;\; \mathbf{p^{(2)}} > 0, \;\; \sum_{i=1}^{n_t^{(1)}} \mathbf{p^{(1)}}_i = \sum_{i=1}^{n_t^{(2)}} \mathbf{p^{(2)}}_i = 1. \tag{2.247}$$

By applying the modularity property, we can write the objective function in optimization problem (2.43) as

$$ {}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p}) \tag{2.248}$$

$$ = \begin{cases} {}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(1)}})w^0 + {}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(2)}})(1-w)^0 & q = 0 \\ \left(\frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(1)}})}{w}\right)^w \left(\frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(2)}})}{1-w}\right)^{1-w} & q = 1 \\ = \min\left\{\frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(1)}})}{w}, \frac{{}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(2)}})}{1-w}\right\} & q = \infty \\ = ({}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(1)}})^{1-q}w^q + {}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(2)}})^{1-q}(1-w)^q)^{\frac{1}{1-q}} & \text{otherwise.} \end{cases} \tag{2.249}$$

This diversity is nondecreasing in ${}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(1)}})$ and ${}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(2)}})$. To maximize the diversity we have to choose $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$ such that ${}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(1)}})$ and ${}^q\mathrm{D}_{\mathrm{OLC}}(\mathbf{p^{(2)}})$ are maximized and thus equal to ${}^q\mathrm{R}_{\mathrm{OLC}}{}^{(1)}$ and ${}^q\mathrm{R}_{\mathrm{OLC}}{}^{(2)}$,

respectively. The derivative with respect to w for $0 < q < \infty$, after maximizing the diversity, becomes

$$\frac{\partial {}^{q}\mathrm{D}_{\mathrm{OLC}}([w\hat{\mathbf{p}}^{(1)}, (1-w)\hat{\mathbf{p}}^{(2)}])}{\partial w}$$

$$= \begin{cases} \left(\frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)}}{w}\right)^{w} \left(\frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}}{1-w}\right)^{1-w} \log\left(\frac{(1-w)^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)}}{w^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}}\right) & q = 1 \\ \frac{q}{1-q}\left(\left(\frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)}}{w}\right)^{1-q} - \left(\frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}}{1-w}\right)^{1-q}\right)^{\frac{q}{1-q}} & \text{otherwise,} \end{cases} \tag{2.250}$$

where $\hat{\mathbf{p}}^{(1)}$ and $\hat{\mathbf{p}}^{(2)}$ are the optimal values for $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$, respectively. For $0 < w < 1$, this derivative equals zero if and only if

$$w = \frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)}}{{}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)} + {}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}}. \tag{2.251}$$

The remaining critical points $w = 0$ and $w = 1$ aren't a maximum as we will show at the end of this proof. Therefore Equation (2.251) expresses the $\hat{w}$ that maximizes the diversity. One can verify that for $q = 0$ and $q = \infty$, the diversity is also maximized when Equation (2.251) holds. If we substitute Equation (2.251) in Equation (2.249 and choose $\mathbf{p^{(1)}}$ and $\mathbf{p^{(2)}}$ such that the corresponding diversities are maximized, then we get the optimal richness which equals

$${}^{q}\mathrm{R}_{\mathrm{OLC}} = {}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)} + {}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}. \tag{2.252}$$

For $w = 0$ and $w = 1$ the richness becomes ${}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}$ and ${}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)}$, respectively. These value of w do not maximize the diversity because

$${}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}, {}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)} < {}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(1)} + {}^{q}\mathrm{R}_{\mathrm{OLC}}{}^{(2)}. \qquad \blacksquare$$

2.J.5 Subadditivity

Lemma 11. *Of all positive semidefinite symmetric matrices that have main diagonal blocks equal to* $\{\mathbf{A}_{1,1}, \ldots, \mathbf{A}_{m,m}\}$*, the block diagonal matrix maximizes* $f(\boldsymbol{\lambda}(\mathbf{A}))$ *in* $\mathbf{A}$ *where* $\boldsymbol{\lambda}(\mathbf{A})$ *is the vector containing the eigenvalues of* $\mathbf{A}$*, and* f *is a quasiconcave function that is symmetric in these eigenvalues.*

Proof. We can diagonalize the diagonal blocks

$$\mathbf{A_{ii}} = \mathbf{P_i D_i P_i}^{\mathsf{T}} \text{ for } i = 1, \ldots, m \tag{2.253}$$

where $\mathbf{P_i}$ is an orthogonal matrix, and $\mathbf{D_i}$ is a diagonal matrix. We can now decompose $\mathbf{A}$ as follows $\mathbf{A} = \mathbf{PDP}^\intercal$ where

$$\mathbf{P} = \begin{bmatrix} \mathbf{P_1} & 0 & \cdots & 0 \\ 0 & \mathbf{P_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{P_m} \end{bmatrix}, \tag{2.254}$$

$$\mathbf{D} = \begin{bmatrix} \mathbf{D_1} & \mathbf{P_1}^\intercal \mathbf{A_{1,2}} \mathbf{P_2} & \cdots & \mathbf{P_1}^\intercal \mathbf{A_{1,m}} \mathbf{P_m} \\ \mathbf{P_2}^\intercal \mathbf{A_{2,1}} \mathbf{P_1} & \mathbf{D_2} & \cdots & \mathbf{P_2}^\intercal \mathbf{A_{2,m}} \mathbf{P_m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_m}^\intercal \mathbf{A_{m,1}} \mathbf{P_1} & \mathbf{P_m}^\intercal \mathbf{A_{m,1}} \mathbf{P_2} & \cdots & \mathbf{D_m} \end{bmatrix}. \tag{2.255}$$

Because $\mathbf{P}$ is orthogonal, the matrices $\mathbf{D}$ and $\mathbf{A}$ have the same eigenvalues. From Lemma 2 we can conclude that $f(\mathbf{D})$ is maximized for $\mathbf{D}$ a diagonal matrix. This occurs when $\mathbf{A}_{ij} = 0$ for $i, j \in \{1, \ldots, m\}$ and $i \neq j$, and thus occurs when $\mathbf{A}$ is a block diagonal matrix. ■

Lemma 12. *If $\boldsymbol{\lambda}(\mathbf{A})$ is the vector containing the eigenvalues of $\mathbf{A}$ and $f(\boldsymbol{\lambda}(\mathbf{A}))$ is a strictly quasiconcave function that is symmetric in these eigenvalues, then the positive semidefinite symmetric partitioned matrix with main diagonal blocks $\{\mathbf{A}_{1,1}, \ldots, \mathbf{A}_{m,m}\}$ that maximizes $f(\boldsymbol{\lambda}(\mathbf{A}))$ in $\mathbf{A}$, does not contain non-zero off-diagonal blocks.*

Proof. Setting at least one off-diagonal block to non-zero, causes $\mathbf{D}$ in the proof of Lemma 11 to have at least one off-diagonal block different from zero. Applying Lemma 3 on $\mathbf{D}$ now proofs the lemma. ■

Theorem 16. *The OLC richness satisfies*

$${}^q\mathrm{R}_{OLC}(\mathcal{F}_1 \cup \mathcal{F}_2) \leq {}^q\mathrm{R}_{OLC}(\mathcal{F}_1) + {}^q\mathrm{R}_{OLC}(\mathcal{F}_2) \tag{2.256}$$

Proof. Theorem 15 states that if the off-diagonal blocks of $\boldsymbol{\Omega}$ are 0

$$\boldsymbol{\Omega} = \begin{bmatrix} \boldsymbol{\Omega}_1 & 0 \\ 0 & \boldsymbol{\Omega}_2 \end{bmatrix}, \tag{2.257}$$

then the richness becomes

$${}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \mathcal{F}_2) = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2). \tag{2.258}$$

According to Lemma 11, for any value of $\mathbf{p}$, the objective function ${}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\lambda}(\boldsymbol{\Omega}(\mathbf{p})))$ is maximized when the off-diagonal blocks of $\boldsymbol{\Omega}(\mathbf{p})$ are zero. In all other cases the objective function ${}^q\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\lambda}(\boldsymbol{\Omega}(\mathbf{p})))$ isn't higher which proves the theorem. ■

2.J.6 Strict subadditivity

Order q is greater than one

Theorem 17. *For $q = 0$ and $1 < q \leq \infty$, ${}^{q}\mathrm{R}_{OLC}$ does not satisfy*

$$ {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_1 \cup \mathcal{F}_2) < {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_2) \tag{2.259} $$

if not every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$.

Proof. The proof for $q = 0$ is trivial. For $1 < q \leq \infty$, we choose

$$ \mathcal{F}_1' = \{[0.8, 0.6, 0]^\intercal, [0.8, -0.6, 0]^\intercal\}, \tag{2.260} $$

$$ \mathcal{F}_1 = \mathcal{F}_1' \cup \{[\sqrt{d}, 0, \sqrt{1-d}]^\intercal\}, \tag{2.261} $$

$$ \mathcal{F}_2 = \{[0, 0, 1]^\intercal\}, \tag{2.262} $$

$$ \mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2, \tag{2.263} $$

$$ \mathcal{F}' = \mathcal{F}_1' \cup \mathcal{F}_2, \tag{2.264} $$

where $d = 0.36\left(\frac{0.36}{0.64}\right)^{q-1} + 0.64$. As shown in the proof of Theorem 13

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) = {}^{q}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho_1}([0.5, 0.5, 0]^\intercal)) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1'). \tag{2.265} $$

Theorem 15 states that

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}') = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1') + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2). \tag{2.266} $$

Substituting Equation (2.265) in Equation (2.266) results in

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}') = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2). \tag{2.267} $$

According to Theorem 11 the following holds

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) \geq {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}'). \tag{2.268} $$

If we now substitute Equation (2.267) in Inequality (2.268), then we get

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) \geq {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2). \tag{2.269} $$

Theorem 16 states that

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) \leq {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2). \tag{2.270} $$

From Equation (2.269) and Equation (2.270), we can conclude that ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2)$ despite the fact that not every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$. ■

Order q is smaller than or equal to one

Theorem 18. *For $0 < q \leq 1$, the OLC richness satisfies*

$$ {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_1 \cup \mathcal{F}_2) < {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_2) \tag{2.271} $$

if not every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$.

Proof. If not every feature vector belonging to $\mathcal{F}_1$ is orthogonal to every feature vector belonging to $\mathcal{F}_2$, then

$$ \mathbf{\Omega} = \begin{bmatrix} \mathbf{\Omega}_1 & \mathbf{A} \\ \mathbf{A}^\intercal & \mathbf{\Omega}_2 \end{bmatrix}, \tag{2.272} $$

where $\mathbf{A}$ is a non-zero matrix and $\mathbf{\Omega}$, $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ are the kernel matrices corresponding to $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$, $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. The weighted kernel matrix becomes

$$ \mathbf{\Omega}(\mathbf{p}) = \mathbf{\Omega}([w\mathbf{p}^{(\mathbf{1})}, (1-w)\mathbf{p}^{(\mathbf{2})}]^\intercal) \tag{2.273} $$

$$ = \begin{bmatrix} \mathbf{\Omega}_1(w\mathbf{p}^{(\mathbf{1})}) & \mathbf{A}(w\mathbf{p}^{(\mathbf{1})}, (1-w)\mathbf{p}^{(\mathbf{2})}) \\ \mathbf{A}^\intercal(w\mathbf{p}^{(\mathbf{1})}, (1-w)\mathbf{p}^{(\mathbf{2})}) & \mathbf{\Omega}_2((1-w)\mathbf{p}^{(\mathbf{2})}) \end{bmatrix}, \tag{2.274} $$

where

$$ 0 \leq w \leq 1, \ \sum_{i=1}^{n_t^{(1)}} {\mathbf{p}^{(\mathbf{1})}}_i = \sum_{i=1}^{n_t^{(2)}} {\mathbf{p}^{(\mathbf{2})}}_i = 1. \tag{2.275} $$

We distinguish two cases. One in which $\mathbf{A}(w\mathbf{p}^{(\mathbf{1})}, (1-w)\mathbf{p}^{(\mathbf{2})})$ is the zero matrix and one in which it isn't. First, we will prove that the theorem holds for the first case. If $\mathcal{F}_1' \subset \mathcal{F}_1$ and $\mathcal{F}_2' \subset \mathcal{F}_2$ are the subsets corresponding to the non-zero elements of $\mathbf{p}$, then because $\mathbf{A}(w\mathbf{p}^{(\mathbf{1})}, (1-w)\mathbf{p}^{(\mathbf{2})})$ is the zero matrix, this implies that $\mathrm{span}(\mathcal{F}_1') \perp \mathrm{span}(\mathcal{F}_2')$ and ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1' \cup \mathcal{F}_2') = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1') + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2')$. Because $\mathrm{span}(\mathcal{F}_1) \not\perp \mathrm{span}(\mathcal{F}_2)$, there has to be at least one feature vector belonging to $\mathcal{F}_1 \setminus \mathcal{F}_1'$ or $\mathcal{F}_2 \setminus \mathcal{F}_2'$ that does not belong to the span of $\mathcal{F}_1'$ or $\mathcal{F}_2'$, respectively. According to Theorem 14, this means that ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) > {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1')$ or ${}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2) > {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2')$, which implies

$$ {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \mathcal{F}_2) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1' \cup \mathcal{F}_2') \tag{2.276} $$

$$ = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1') + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2') \tag{2.277} $$

$$ < {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2). \tag{2.278} $$

Now we still have to prove that the statement holds when $\mathbf{A}(w\mathbf{p}^{(\mathbf{1})}, (1-w)\mathbf{p}^{(\mathbf{2})})$ is not equal to the zero matrix. According to Lemma 12, the OLC diversity of a

weighted kernel matrix with diagonal blocks $\mathbf{\Omega}_1(w\mathbf{p^{(1)}})$ and $\mathbf{\Omega}_2((1-w)\mathbf{p^{(2)}})$ is maximized if and only if $\mathbf{A}(w\mathbf{p^{(1)}}, (1-w)\mathbf{p^{(2)}})$ is the zero matrix. In this case ${}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1 \cup \mathcal{F}_2) = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2)$. In case $\mathbf{A}(w\mathbf{p^{(1)}}, (1-w)\mathbf{p^{(2)}})$ is not the zero matrix, the following thus holds $\mathrm{R}(\mathcal{F}_1 \cup \mathcal{F}_2) < \mathrm{R}(\mathcal{F}_1) + \mathrm{R}(\mathcal{F}_2)$. ■

2.J.7 Interpretability

Theorem 19. *The OLC richness satisfies*

$$ {}^q\mathrm{R}_{OLC}(\mathcal{F}) = n_t \tag{2.279}$$

if each pair of feature vectors in $\mathcal{F}$ is orthogonal.

Proof. We will prove the Theorem by induction. If $n_t = 1$, then the weighted kernel matrix becomes $\mathbf{\Omega}(\hat{\mathbf{p}}) = \mathbf{\Omega}([1]) = [1]$, and therefore ${}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})$ equals 1. This is the base case. If we add a feature vector orthogonal to the current set, denoted by $\mathbf{f}_\perp$, then according to Theorem 15: ${}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \cup \{\mathbf{f}_\perp\}) = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) + 1$. This is the inductive step that concludes the proof. ■

2.K Properties of the contribution vector

2.K.1 Partitioning

Theorem 20. *If ${}^{\mathbf{q}}\mathbf{c}_{OLC}$ is the OLC contribution vector corresponding to $\mathcal{F}$, then the sum of the contributions of all types equals the richness of $\mathcal{F}$*

$$ \sum_{i=1}^{n_t} {}^{\mathbf{q}}\mathbf{c}_{OLC,i} = {}^q\mathrm{R}_{OLC}(\mathcal{F}). \tag{2.280}$$

Proof. The contribution vector is defined as ${}^{\mathbf{q}}\mathbf{c}_{\mathrm{OLC}} = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})\hat{\mathbf{p}}$. We can now conclude that

$$ \sum_{i=1}^{n_t} {}^{\mathbf{q}}\mathbf{c}_{\mathrm{OLC},i} = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) \sum_{i=1}^{n_t} \hat{\mathbf{p}}_i = {}^q\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}). \tag{2.281}$$

because $\sum_{i=1}^{n_t} \hat{\mathbf{p}}_i = 1$. ■

2.K.2 Consistent with richness of subsets

Lemma 13. *If $\boldsymbol{\rho_1} \in \mathbb{R}^{n_y \times n_y}$ and $\boldsymbol{\rho_2} \in \mathbb{R}^{n_y \times n_y}$ are two second moment matrices that have the same dimensions, then*

$$ {}^2\mathrm{D}_{OLC}(\boldsymbol{\rho_1}) \geq {}^2\mathrm{D}_{OLC}(\hat{w}\boldsymbol{\rho_1} + (1-\hat{w})\boldsymbol{\rho_2})\hat{w}, \tag{2.282}$$

where

$$\hat{w} = arg\max_{w} {}^{2}\mathrm{D}_{OLC}\left(w\boldsymbol{\rho_1} + (1-w)\boldsymbol{\rho_2}\right). \tag{2.283}$$

Proof. We can write the quadratic OLC diversity of $w\boldsymbol{\rho_1} + (1-w)\boldsymbol{\rho_2}$ as

$$\tag{2.284}$$

$${}^{2}\mathrm{D}_{\mathrm{OLC}}(w\boldsymbol{\rho_1} + (1-w)\boldsymbol{\rho_2}) = \big(w^2 \operatorname{tr}(\boldsymbol{\rho_1}^2) + (1-w)^2 \operatorname{tr}(\boldsymbol{\rho_2}^2) \tag{2.285}$$

$$+ 2w(1-w)\operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})\big)^{-1}. \tag{2.286}$$

The derivative with respect to w equals

$$\frac{d^2\mathrm{D}_{\mathrm{OLC}}(w\boldsymbol{\rho_1} + (1-w)\boldsymbol{\rho_2})}{dw} = -{}^{2}\mathrm{D}_{\mathrm{OLC}}(w\boldsymbol{\rho_1} + (1-w)\boldsymbol{\rho_2})^2\big(2w\operatorname{tr}(\boldsymbol{\rho_1}^2)$$

$$+ 2(w-1)\operatorname{tr}(\boldsymbol{\rho_2}^2) + (2-4w)\operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})\big). \tag{2.287}$$

This derivative equals zero when

$$w = \frac{\operatorname{tr}(\boldsymbol{\rho_2}^2) - \operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})}{\operatorname{tr}(\boldsymbol{\rho_1}^2) + \operatorname{tr}(\boldsymbol{\rho_2}^2) - 2\operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})}. \tag{2.288}$$

We only consider the case where $\hat{w} = \arg\max_w {}^{2}\mathrm{D}_{\mathrm{OLC}}\left(w\boldsymbol{\rho_1} + (1-w)\boldsymbol{\rho_2}\right)$ satisfies $0 < \hat{w} < 1$. For $\hat{w}$ equal to 0 or 1, the proof is trivial. If the optimal weight vector $\hat{w}$ is different from 0 and 1, then the derivative of the diversity with respect to w equals 0 in $w = \hat{w}$ where $0 < \hat{w} < 1$. This means that

$$0 < \frac{\operatorname{tr}(\boldsymbol{\rho_2}^2) - \operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})}{\operatorname{tr}(\boldsymbol{\rho_1}^2) + \operatorname{tr}(\boldsymbol{\rho_2}^2) - 2\operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})} < 1 \Rightarrow \operatorname{tr}(\boldsymbol{\rho_1}^2), \operatorname{tr}(\boldsymbol{\rho_2}^2) \geq \operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2}). \tag{2.289}$$

We will now evaluate the quadratic diversity for this optimal $\hat{w}$ by substituting Equation (2.288) in Equation (2.286)

$${}^{2}\mathrm{D}_{\mathrm{OLC}}(\hat{w}\boldsymbol{\rho_1} + (1-\hat{w})\boldsymbol{\rho_2}) = \frac{\operatorname{tr}(\boldsymbol{\rho_1}^2) + \operatorname{tr}(\boldsymbol{\rho_2}^2) - 2\operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})}{\operatorname{tr}(\boldsymbol{\rho_1}^2)\operatorname{tr}(\boldsymbol{\rho_2}^2) - \operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})^2}. \tag{2.290}$$

Note that due to the Cauchy-Schwarz inequality $\operatorname{tr}(\boldsymbol{\rho_1}^2)\operatorname{tr}(\boldsymbol{\rho_2}^2) - \operatorname{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})^2 \geq 0$. We can now conclude that

$${}^{2}\mathrm{D}_{\mathrm{OLC}}(\boldsymbol{\rho_1}) \geq {}^{2}\mathrm{D}_{\mathrm{OLC}}(\hat{w}\boldsymbol{\rho_1} + (1-\hat{w})\boldsymbol{\rho_2})\hat{w} \tag{2.291}$$

$$\Leftrightarrow \frac{1}{\mathrm{tr}(\boldsymbol{\rho_1}^2)} \geq \frac{\mathrm{tr}(\boldsymbol{\rho_2}^2) - \mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})}{\mathrm{tr}(\boldsymbol{\rho_1}^2)\,\mathrm{tr}(\boldsymbol{\rho_2}^2) - \mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})^2} \tag{2.292}$$

$$\Leftrightarrow \mathrm{tr}(\boldsymbol{\rho_1}^2)\,\mathrm{tr}(\boldsymbol{\rho_2}^2) - \mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})^2 \geq \mathrm{tr}(\boldsymbol{\rho_1}^2)\,\mathrm{tr}(\boldsymbol{\rho_2}^2) - \mathrm{tr}(\boldsymbol{\rho_1}^2)\,\mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2}) \tag{2.293}$$

$$\Leftrightarrow \mathrm{tr}(\boldsymbol{\rho_1}^2) \geq \mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2}) \tag{2.294}$$

holds because Equation (2.294) is implied by equation (2.289). The direction of the inequality sign in Inequality 2.294 didn't change after we divided by $\mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})$ because $\mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2})$ is positive as we show next

$$\mathrm{tr}(\boldsymbol{\rho_1}\boldsymbol{\rho_2}) = \mathrm{tr}(\mathbf{F_1}\mathbf{P_1}\mathbf{F_1}^\intercal\mathbf{F_2}\mathbf{P_2}\mathbf{F_2}^\intercal) \tag{2.295}$$

$$= \mathrm{tr}(\mathbf{P_1}^{\frac{1}{2}}\mathbf{F_1}^\intercal\mathbf{F_2}\mathbf{P_2}^{\frac{1}{2}}\mathbf{P_2}^{\frac{1}{2}}\mathbf{F_2}^\intercal\mathbf{F_1}\mathbf{P_1}^{\frac{1}{2}}) \tag{2.296}$$

$$= \|\mathbf{P_2}^{\frac{1}{2}}\mathbf{F_2}^\intercal\mathbf{F_1}\mathbf{P_1}^{\frac{1}{2}}\|_F^2 \geq 0, \tag{2.297}$$

where $\mathbf{F_1}$ is the matrix with columns equal to the feature vectors in $\mathcal{F}_s$, and $\mathbf{F_2}$ is the matrix with columns equal to the feature vectors in $\mathcal{F} \setminus \mathcal{F}_s$. Equation (2.296) holds because $\mathrm{tr}(\mathbf{AB}) = \mathrm{tr}(\mathbf{BA})$ where $\mathbf{A} = \mathbf{F_1}\mathbf{P_1}^{\frac{1}{2}}$ and $\mathbf{B} = \mathbf{P_1}^{\frac{1}{2}}\mathbf{F_1}^\intercal\mathbf{F_2}\mathbf{P_2}\mathbf{F_2}^\intercal$. ■

Theorem 21. *The quadratic OLC contribution of a subset does not exceed the richness of that subset*

$$\sum_{[i\in\{1,\ldots,n_t\}]\mathbf{f_i}\in\mathcal{F}_s} {}^{\mathbf{2}}\mathbf{c}_{OLC,i} \leq {}^2\mathrm{R}_{OLC}(\mathcal{F}_s), \tag{2.298}$$

where ${}^{\mathbf{2}}\mathbf{c}_{OLC}$ *is the quadratic OLC contribution vector corresponding to* $\mathcal{F}$*. Also, the contribution of a subset is larger than or equal to the total richness minus the richness of the complementary set*

$$\sum_{[i\in\{1,\ldots,n_t\}]\mathbf{f_i}\in\mathcal{F}_s} {}^{\mathbf{2}}\mathbf{c}_{OLC,i} \geq {}^2\mathrm{R}_{OLC}(\mathcal{F}) - {}^2\mathrm{R}_{OLC}(\mathcal{F} \setminus \mathcal{F}_s). \tag{2.299}$$

Proof. We can formulate $\mathbf{p}$, the abundance vector corresponding to $\mathcal{F}$, as a weighted concatenation of $\mathbf{p^{(1)}}$, the abundance vector corresponding to $\mathcal{F}_s$, and $\mathbf{p^{(2)}}$, the abundance vector corresponding to $\mathcal{F} \setminus \mathcal{F}_s$, as follows

$$\mathbf{p} = [w\mathbf{p^{(1)}}; (1-w)\mathbf{p^{(2)}}], \tag{2.300}$$

where

$$0 \leq w \leq 1,\ \ \mathbf{p^{(1)}} > 0,\ \ \mathbf{p^{(2)}} > 0,\ \ \sum_{i=1}^{n_t(1)} {\mathbf{p^{(1)}}}_i = \sum_{i=1}^{n_t^{(2)}} {\mathbf{p^{(2)}}}_i = 1. \tag{2.301}$$

We denote the abundance vector that optimizes the richness objective function as $\hat{\mathbf{p}} = [\hat{w}\bar{\mathbf{p}}^{(1)}, (1-\hat{w})\bar{\mathbf{p}}^{(2)}]$. Note that

$$^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_s) \geq {}^{2}\mathrm{D}_{\mathrm{OLC}}(\bar{\boldsymbol{\rho}}_1), \tag{2.302}$$

where $\bar{\boldsymbol{\rho}}_1 = \boldsymbol{\rho}^{(1)}(\bar{\mathbf{p}}^{(1)})$ is the second moment matrix corresponding to feature set $\mathcal{F}_s$ and abundance vector $\bar{\mathbf{p}}^{(1)}$, and

$$^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) = {}^{2}\mathrm{D}_{\mathrm{OLC}}(\hat{w}\bar{\boldsymbol{\rho}}_1 + (1-\hat{w})\bar{\boldsymbol{\rho}}_2)\ , \tag{2.303}$$

where $\bar{\boldsymbol{\rho}}_2 = \boldsymbol{\rho}^{(2)}(\bar{\mathbf{p}}^{(2)})$ is the second moment matrix corresponding to the feature set $\mathcal{F} \setminus \mathcal{F}_s$ and abundance vector $\bar{\mathbf{p}}^{(2)}$. Lemmma 13 states that

$$^{2}\mathrm{D}_{\mathrm{OLC}}(\bar{\boldsymbol{\rho}}_1) \geq {}^{2}\mathrm{D}_{\mathrm{OLC}}(\hat{w}\bar{\boldsymbol{\rho}}_1 + (1-\hat{w})\bar{\boldsymbol{\rho}}_2)\hat{w}\ . \tag{2.304}$$

Substituting Inequality (2.302) and Equation (2.303) in Inequality (2.304) results in

$$^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_s) \geq {}^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})\hat{w} = \sum_{i=1}^{n'_t} {}^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})\hat{w}\bar{\mathbf{p}}_i^{(1)} = \sum_{[i\in\{1,\ldots,n_t\}]\mathbf{f_i}\in\mathcal{F}_s} {}^{\mathbf{2}}\mathbf{c}_{\mathrm{OLC},i}, \tag{2.305}$$

which proves the first part of the theorem. The upperbound

$$^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \setminus \mathcal{F}_s) \geq {}^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})(1-\hat{w}) \tag{2.306}$$

implies

$$^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})\hat{w} \geq {}^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}) - {}^{2}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F} \setminus \mathcal{F}_s). \tag{2.307}$$

and the lowerbound thus holds as well. ■

2.K.3 Consistent with richness of orthogonal subsets

Theorem 22. *The vector* $^{\mathbf{q}}\mathbf{c}_{OLC} = {}^{q}\mathrm{R}_{OLC}(\mathcal{F})\hat{\mathbf{p}}$ *with* $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ *satisfies*

$$\sum_{[i\in\{1,\ldots,n_t\}]\mathbf{f_i}\in\mathcal{F}_1} {}^{\mathbf{q}}\mathbf{c}_{OLCi} = {}^{q}\mathrm{R}_{OLC}(\mathcal{F}_1) \tag{2.308}$$

if span$(\mathcal{F}_1) \perp$ *span*$(\mathcal{F}_2)$.

Proof. Similar to the proof of Theorem 21, we express the optimal abundance vector as a weighted concatenation $\hat{\mathbf{p}} = [\hat{w}\bar{\mathbf{p}}^{(1)}, (1-\hat{w})\bar{\mathbf{p}}^{(2)}]$. We now have to show that

$$^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) = {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F})\hat{w} = ({}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2))\hat{w}. \tag{2.309}$$

In the proof of Theorem 15 it is already shown that

$$\hat{w} = \frac{{}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1)}{{}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_1) + {}^{q}\mathrm{R}_{\mathrm{OLC}}(\mathcal{F}_2)}.$$

■

2.K.4 Twin property

Theorem 23. *If type i is equal to type j, then removing type j doesn't change the OLC contribution of the types different from type j.*

Proof. Duplicating a type doesn't change the OLC richness (see the non-changing property of the OLC richness). If $\hat{\mathbf{p}}$ is the abundance vector that optimizes the OLC diversity before removing the duplicate, then $\hat{\mathbf{q}}$ is the optimal abundance vector after removing the duplicate, where

$$\hat{\mathbf{q}}_k = \begin{cases} \hat{\mathbf{p}}_k, & k \in \{1, \dots, n_t\}, k \neq i, k \neq j, \\ \hat{\mathbf{p}}_i + \hat{\mathbf{p}}_j & k = i, \\ 0 & k = j. \end{cases} \tag{2.310}$$

Note the relative abundance of the duplicate equals 0 in $\hat{\mathbf{q}}$, which is equivalent to removing the duplicate. Both the OLC richness and the relative abundances of types different from type j remain unchanged after removing the duplicate, and therefore the OLC contributions of the types different from type j do not change either. ■

2.L Alpha, beta and gamma

Theorem 24. *If we have pooled group consisting of n_w subgroups with entity models $\mathcal{E} = \{E_i\}_{i=1}^{n_w}$ and relative sizes $\hat{w} = [\mathbf{w}_1, \dots, \mathbf{w}_{n_w}]^\intercal$, then the decomposition in alpha and beta that satisfies the following axioms*

1. *The alpha diversity is expressed in the effective number of types per subgroup, the beta diversity in the effective number of subgroups, and the gamma diversity in the effective number of types. Because the effective number of types equals the effective number of types per subgroup multiplied by he effective number of subgroups, this implies that gamma is alpha multiplied by beta;*
2. *Alpha only depends on the relative sizes and the diversities of the subgroups. It does not depend on differences between subgroups (i.e., on how dissimilar types belonging to different subgroups are). Moreover, alpha is some type of average of the diversities of the subgroups;*
3. *The range to which beta belongs, is determined by the relative sizes of the subgroups. It does not depend on the subgroup diversities. Moreover, the beta component is maximized when the subgroups are completely distinct, in which case it becomes the type-neutral diversity applied to the relative sizes of the subgroups, and is minimized when the subgroups are identical, in which case it becomes one;*

is the following decomposition

$$ {}^{q}\mathrm{D}^{\gamma}_{OLC}(\mathcal{E},\mathbf{w},s) = {}^{q}\mathrm{D}_{OLC}\left(\sum_{i=1}^{n_w}\mathbf{w}_i E_i, s\right), \tag{2.311} $$

$$ {}^{q}\mathrm{D}^{\alpha}_{OLC}(\mathcal{E},\mathbf{w},s) = \left(\frac{\sum_{i=1}^{n_w}\mathbf{w}_i^q\left({}^{q}\mathrm{D}_{OLC}(E_i,s)\right)^{1-q}}{\sum_{i=1}^{n_w}\mathbf{w}_i^q}\right)^{\frac{1}{1-q}}, \tag{2.312} $$

$$ {}^{q}\mathrm{D}^{\beta}_{OLC}(\mathcal{E},\mathbf{w},s) = \frac{{}^{q}\mathrm{D}^{\gamma}_{OLC}(\mathcal{E},\mathbf{w},s)}{{}^{q}\mathrm{D}^{\alpha}_{OLC}(\mathcal{E},\mathbf{w},s)}. \tag{2.313} $$

Because ${}^{1}\mathrm{D}^{\alpha}_{OLC}(\mathcal{E},\mathbf{w},s)$*, as expressed above, is undefined, we instead take the limit for* q *going to* 1

$$ {}^{1}\mathrm{D}^{\alpha}_{OLC}(\mathcal{E},\mathbf{w},s) = \lim_{q\to 1}{}^{q}\mathrm{D}^{\alpha}_{OLC}(\mathcal{E},\mathbf{w},s) = \exp\left(\sum_{i=1}^{n_w}\mathbf{w}_i \ln\left({}^{1}\mathrm{D}_{OLC}\left(E_i,s\right)\right)\right). \tag{2.314} $$

Proof. The proof listed here is similar to the proof for the type-neutral case which can be found in the digital appendix of Jost [26]. Gamma equals the diversity applied to the pooled group

$$ {}^{q}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E},\mathbf{w},s) = {}^{q}\mathrm{D}_{\mathrm{OLC}}\left(\sum_{i=1}^{n_w}\mathbf{w}_i E_i, s\right). \tag{2.315} $$

If the subgroups $\mathcal{E}_{\mathrm{dist}} = \{E_i\}_{i=1}^{n_w}$ are completely distinct, then according to the modularity property

$$ \left({}^{q}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E}_{\mathrm{dist}},\mathbf{w}),s\right)^{1-q} = \sum_{i=1}^{n_w}\mathbf{w}_i^q\left({}^{q}\mathrm{D}_{\mathrm{OLC}}(E_i,s)\right)^{1-q}. \tag{2.316} $$

Because gamma is the product of alpha and beta, we get

$$ \left({}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}_{\mathrm{dist}},\mathbf{w},s)\right)^{1-q} = \frac{\sum_{i=1}^{n_w}\mathbf{w}_i^q\left({}^{q}\mathrm{D}_{\mathrm{OLC}}(E_i,s)\right)^{1-q}}{\left({}^{q}\mathrm{D}^{\beta}_{\mathrm{OLC}}(\mathcal{E}_{\mathrm{dist}},\mathbf{w},s)\right)^{1-q}}. \tag{2.317} $$

Beta equals the type-neutral diversity applied to the relative sizes of the subgroups when the subgroups are completely distinct

$$ \left({}^{q}\mathrm{D}^{\beta}_{\mathrm{OLC}}(\mathcal{E}_{\mathrm{dist}},\mathbf{w},s)\right)^{1-q} = \sum_{i=1}^{n_w}\mathbf{w}_i^q. \tag{2.318} $$

Equation (2.318) can now be substituted in Equation (2.317)

$$\left({}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}_{\mathrm{dist}}, \mathbf{w}, s)\right)^{1-q} = \frac{\sum_{i=1}^{n_w} \mathbf{w}_i^q \left({}^{q}\mathrm{D}_{\mathrm{OLC}}(E_i, s)\right)^{1-q}}{\sum_{i=1}^{n_w} \mathbf{w}_i^q}. \tag{2.319}$$

Because alpha only depends on the relative size and the diversity of each subgroup, this equation also holds when the subgroups are not completely distinct, and therefore

$${}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) = \left(\frac{\sum_{i=1}^{n_w} \mathbf{w}_i^q \left({}^{q}\mathrm{D}_{\mathrm{OLC}}(E_i, s)\right)^{1-q}}{\sum_{i=1}^{n_w} \mathbf{w}_i^q}\right)^{\frac{1}{1-q}}. \qquad \blacksquare$$

Theorem 25. *The decomposition proposed in Theorem 24 additionally satisfies the fifth axiom listed in Jost [26], which states that alpha can never be greater than gamma, if and only if (i) the subgroups are equally weighted, or (ii) q equals* 0 *or* 1.

Proof. The OLC diversity is more general than the type-neutral diversity and therefore, for unequally weighted subgroups, the axiom also doesn't hold either when q is different from 0 and 1. Because f_q, as defined in Lemma 5, is concave for $0 \leq q < 1$, the following holds

$$f_q\left(\sum_{i=1}^{n_w} \frac{1}{n_w} \boldsymbol{\rho}^{(\mathbf{i})}\right) \geq \sum_{i=1}^{n_w} \frac{1}{n_w} f_q\left(\boldsymbol{\rho}^{(\mathbf{i})}\right). \tag{2.320}$$

For equally weighted subgroups this implies

$${}^{q}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) = \left(f_q\left(\sum_{i=1}^{n_w} \frac{1}{n_w} \boldsymbol{\rho}^{(\mathbf{i})}\right)\right)^{\frac{1}{1-q}} \tag{2.321}$$

$$\geq \left(\sum_{i=1}^{n_w} \frac{1}{n_w} f_q\left(\boldsymbol{\rho}^{(\mathbf{i})}\right)\right)^{\frac{1}{1-q}} = {}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s). \tag{2.322}$$

Because f_q is convex for $1 \leq q < \infty$, it satisfies

$$f_q\left(\sum_{i=1}^{n_w} \frac{1}{n_w} \boldsymbol{\rho}^{(\mathbf{i})}\right) \leq \sum_{i=1}^{n_w} \frac{1}{n_w} f_q\left(\boldsymbol{\rho}^{(\mathbf{i})}\right). \tag{2.323}$$

For equally weighted subgroups this also means that

$${}^{q}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) = \left(f_q\left(\sum_{i=1}^{n_w} \frac{1}{n_w} \boldsymbol{\rho}^{(\mathbf{i})}\right)\right)^{\frac{1}{1-q}} \tag{2.324}$$

$$\geq \left(\sum_{i=1}^{n_w} \frac{1}{n_w} f_q \left(\boldsymbol{\rho}^{(\mathbf{i})} \right) \right)^{\frac{1}{1-q}} = {}^{q}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s). \tag{2.325}$$

Because $\mathbf{f_1}$, as defined in Lemma 5, is concave, it meets the following equation:

$$f_1 \left(\sum_{i=1}^{n_w} \mathbf{w}_i \boldsymbol{\rho}^{(\mathbf{i})} \right) \geq \sum_{i=1}^{n_w} \mathbf{w}_i f_1 \left(\boldsymbol{\rho}^{(\mathbf{i})} \right). \tag{2.326}$$

This implies

$${}^{1}\mathrm{D}^{\gamma}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s) = \exp \left(f_1 \left(\sum_{i=1}^{n_w} \mathbf{w}_i \boldsymbol{\rho}^{(\mathbf{i})} \right) \right) \tag{2.327}$$

$$\geq \exp \left(\sum_{i=1}^{n_w} \mathbf{w}_i f_q \left(\boldsymbol{\rho}^{(\mathbf{i})} \right) \right) = {}^{1}\mathrm{D}^{\alpha}_{\mathrm{OLC}}(\mathcal{E}, \mathbf{w}, s). \tag{2.328}$$

For $q = 0$ the weights are irrelevant and the property thus also holds for unequally weighted subgroups. ■

2.M Phylogenetic positive semidefinite kernel

Theorem 26. *The phylogenetic kernel*

$$k(d_i, d_j) = \sqrt{\sum_{k \in (B_i \cap B_j)} \mathbf{x}_k} \tag{2.329}$$

is positive semidefinite, where $\mathbf{x}_k$ is the length of branch k, and B_i and B_j are sets containing the indices of the branches from which species i and species j are descendants, respectively.

Proof. Suppose that s_k and e_k are the distances from the root of the tree to the start point and the end point of branch k, respectively. The length of branch k thus equals

$$\mathbf{x}_k = e_k - s_k. \tag{2.330}$$

If we choose the feature mapping $\varphi(d_i) \in \mathbb{R}^n$ as follows

$$\varphi_k(d_i) = \begin{cases} \sqrt{\sqrt{e_k} - \sqrt{s_k}} & k \in B_i, \\ 0 & k \notin B_i, \end{cases} \tag{2.331}$$

for $k = [1, \ldots, n]$ where n is the number of branches, then

$$k(d_i, d_j) = \varphi_i(d_i)^\intercal \varphi_i(d_j) = \sum_{k \in (B_i \cap B_j)} \sqrt{e_k} - \sqrt{s_k} \tag{2.332}$$

$$= \max_{k \in (B_i \cap B_j)} \sqrt{e_k} \tag{2.333}$$

$$= \sqrt{\sum_{k \in (B_i \cap B_j)} \mathbf{x}_k}. \tag{2.334}$$

Equation (2.333) holds because the branches shared by two species are linked branches in the tree where, except for the shared branch that starts in the root, each start point of a shared branch is the end point of another shared branch. This means that, except for the shared branch that is the furthest from the root, the term $\sqrt{e_k}$ is eliminated by the term $-\sqrt{s_k}$ of the shared branch one step further from the root. Equation (2.334) holds because the endpoint of the shared branch that is the furthest from the root, equals the sum of the lengths of the shared branches that connect this endpoint to the root. ■

Theorem 27. *The phylogenetic kernel*

$$k'(d_i, d_j) = \sqrt{\frac{\sqrt{\sum_{k \in (B_i \cap B_j)} \mathbf{x}_k}}{\sqrt{\sum_{l \in B_i} \mathbf{x}_l \sum_{m \in B_j} \mathbf{x}_m}}}, \tag{2.335}$$

is positive semidefinite.

Proof. If we set the feature mapping equal to

$$\varphi'(d_i) = \frac{\varphi(d_i)}{\sqrt[4]{\sum_{k \in B_i} \mathbf{x}_k}}, \tag{2.336}$$

where $\varphi(d_i)$ is the feature mapping corresponding to the kernel defined in Equation (2.329), then

$$k'(d_i, d_j) = \varphi'(d_i)^\intercal \varphi'(d_j). \qquad ■$$

2.N Nucleotide probabilities in phylogenetic trees

2.N.1 Three equally distant species

If we have a phylogenetic tree that assumes a nucleotide substitution model which is a Markov chain (e.g., the JC69 model [23]), then we will explain here a

method that can be used to calculate the probability that three equally distant species in this phylogenetic tree have a certain nucleotide at a certain site. The method consists of two steps. First we locate the nodes that separate the species in the sense that the node lies on each path going from one of the three species to another of the three species. In Figure 2.N.1 node D separates species A, B and C. Second we calculate the probability that species A has nucleotide c_i, species B has nucleotide c_j and species C has nucleotide c_k at a certain site as follows

$$P(Y_A = c_i, Y_B = c_j, Y_C = c_k)$$

$$= \sum_{l=1}^{4} P(Y_A = c_i, Y_B = c_j, Y_C = c_k, Y_D = c_l) \tag{2.337}$$

$$= \sum_{l=1}^{4} P(Y_A = c_i; Y_B = c_j, Y_C = c_k, Y_D = c_l) P(Y_B = c_j; Y_C = c_k, Y_D = c_l)$$

$$P(Y_C = c_k; Y_D = c_l) P(Y_D = c_l) \tag{2.338}$$

$$= \sum_{l=1}^{4} P(Y_A = c_i; Y_D = c_l) P(Y_B = c_j; Y_D = c_l) P(Y_C = c_k; Y_D = c_l) P(Y_D = c_l), \tag{2.339}$$

where $P(X = x; Y = y)$ is a conditional probability (i.e., the probability that $X = x$ given $Y = y$), Y_X is the random variable that represents the nucleotide of species X, and c_1, c_2, c_3 and c_4 represent guanine, adenine, thymine and cytosine, respectively. Equation (2.337) holds because of the law of total probability. A Markov chain has the property that the probability on the value of the current state is entirely determined by the value of the last known state. Because the Markov chains that we use are time reversible, this means that if state i is at the end of a path, then of all known states on this path, only the state closest the state i determines the probability on the value of state i. Equation (2.339) thus holds because species D lies on the path that connects each pair of species. Note that $P(Y_V = c_i; Y_W = c_j)$ equals

$$\mathrm{P}_{ij}(d) = \begin{cases} \frac{1}{4} + \frac{3}{4} \exp\left(-\frac{4}{3}d\right) & \text{if } i = j, \\ \frac{1}{4} - \frac{1}{4} \exp\left(-\frac{4}{3}d\right) & \text{if } i = j \end{cases} \tag{2.340}$$

in the JC69 model, where $\mathrm{P}(d)$ is the transition matrix, and d is the phylogenetic distance between species V and species W. We can now apply this method on Figure 2.N.1 to calculate the proportion of sites that are, on the one hand, different for species A and species C and, on the other hand, equal for species B and species C. The probabilities on the relevant states become

$$P(Y_A = c_i, Y_B = c_j, Y_C = c_k, Y_D = c_l)$$

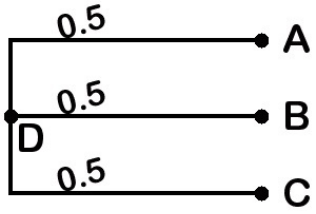


Figure 2.N.1: Phylogenetic tree of three species (species A, species B and Species C) where the phylogenetic distance between each pair of species equals 1. Node D separates these species.

$$= P(Y_A = c_i; Y_D = c_l)P(Y_B = c_j; Y_D = c_l)P(Y_C = c_k; Y_D = c_l)P(Y_D = c_l) \tag{2.341}$$

$$= \begin{cases} 0.635 \cdot 0.122 \cdot 0.122 \cdot 0.25 = 0.0023 & \text{if } c_i \neq c_k = c_j, c_i = c_l, c_j \neq c_l, c_k \neq c_l, \\ 0.122 \cdot 0.635 \cdot 0.635 \cdot 0.25 = 0.0123 & \text{if } c_i \neq c_k = c_j, c_i \neq c_l, c_j = c_l, c_k = c_l, \\ 0.122 \cdot 0.122 \cdot 0.122 \cdot 0.25 = 0.0045 & \text{if } c_i \neq c_k = c_j, c_i \neq c_l, c_j \neq c_l, c_k \neq c_l. \end{cases} \tag{2.342}$$

When applying the law of total probability with respect to all four random variables, the number of terms corresponding to states that meet the first, second and third condition expressed in Equation (2.342) equals 12, 12 and 24, respectively. This means that

$$P(Y_A \neq Y_C = Y_B) = 12 \cdot 0.0023 + 12 \cdot 0.0123 + 24 \cdot 0.0045 = 0.186. \tag{2.343}$$

2.N.2 Skewed and unskewed tree

First we will calculate the diversity of the skewed tree (see Figure 2.2b). In Figure 2.N.2 node G isolates species A, B, C and D and node H isolates E and F. Species A, B, C, D, E and F are called real species because they correspond to actual species, and species G and H are called virtual species. We can now write the probabilities on the nucleotides as follows

$$P(Y_A = c_i, Y_B = c_j, Y_C = c_k, Y_D = c_l, Y_E = c_m, Y_F = c_n, Y_G = c_p, Y_H = c_q)$$

$$= P(Y_A = c_i; Y_G = c_p)P(Y_B = c_j; Y_G = c_p)P(Y_C = c_k; Y_G = c_p)$$

$$P(Y_D = c_l; Y_G = c_p)P(Y_E = c_m; Y_H = c_q)P(Y_F = c_n; Y_H = c_q)$$

$$P(Y_G = c_p; Y_H = c_q)P(Y_H = c_q) \tag{2.344}$$

$$= t_1^{u_1} f_1^{6-u_1} t_2^{u_2} f_2^{1-u_2} 0.25, \tag{2.345}$$

where u_1 is the number of real species that have a nucleotide equal to the nucleotide of the virtual species that isolates it, u_2 equals 1 when both virtual

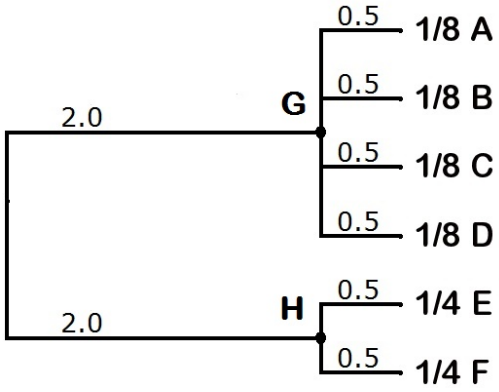


Figure 2.N.2: Same tree as in Figure 2.2b, except that we added node G and node H.

species have the same nucleotide and equals 0 otherwise, and

$$t_1 = \frac{1}{4} + \frac{3}{4}\exp\left(-\frac{4}{3}\cdot 0.5\right), \tag{2.346}$$

$$f_1 = \frac{1}{4} - \frac{1}{4}\exp\left(-\frac{4}{3}\cdot 0.5\right), \tag{2.347}$$

$$t_2 = \frac{1}{4} + \frac{3}{4}\exp\left(-\frac{4}{3}\cdot 4\right), \tag{2.348}$$

$$f_2 = \frac{1}{4} - \frac{1}{4}\exp\left(-\frac{4}{3}\cdot 4\right). \tag{2.349}$$

For all 4^8 possible values of the nucleotides, we can calculate the diversity of the nucleotides

$${}^{q}\mathrm{D}(\mathcal{C}) \tag{2.350}$$

$$= \left(\left(\sum_{\forall i, c_i = c_1} \mathbf{p}_i\right)^q + \left(\sum_{\forall j, c_j = c_2} \mathbf{p}_j\right)^q + \left(\sum_{\forall k, c_k = c_3} \mathbf{p}_k\right)^q + \left(\sum_{\forall l, c_l = c_4} \mathbf{p}_l\right)^q\right)^{\frac{1}{1-q}}, \tag{2.351}$$

where $\mathcal{C} = \{c_i, c_j, c_k, c_l, c_m, c_n, c_p, c_q\}$ contains the nucleotides of species A, B, C, D, E ,F ,G and H, respectively, and $\mathbf{p} = [1/8, 1/8, 1/8, 1/8, 1/4, 1/4, 0, 0]^\intercal$ is the corresponding abundance vector. Virtual species have a relative abundance equal to zero because they don't actually occur. The nucleotide diversity now becomes the following weighted average over these diversities

$${}^{q}\mathrm{D}_{\mathrm{DNA}} = \left(\sum_{\mathcal{C}} P(\mathcal{Y} = \mathcal{C})\ {}^{q}\mathrm{D}(\mathcal{C})^{1-q}\right)^{\frac{1}{1-q}}, \tag{2.352}$$

where $P(\mathcal{Y} = \mathcal{C}) = P(Y_A = c_i, Y_B = c_j, Y_C = c_k, Y_D = c_l, Y_E = c_m, Y_F = c_n, Y_G = c_p, Y_H = c_q)$. In the case of the unskewed tree (see Figure 2.2a), the nucleotide diversity can be calculated similarly.

Chapter 3

Quality of Service Prediction of Workflow Compositions

QoS prediction for web service compositions using kernel-based quantile estimation with online adaptation of the constant offset

Dries Geebelen, Kristof Geebelen, Eddy Truyen,
Sam Michiels, Johan A.K. Suykens, Joos Vandewalle,
Wouter Joosen

Published in *Information Sciences*, vol. 268, pp 397-424, 2014.

Author contributions: D.G. and K.G. contributed equally to this work. D.G. proposed the performance indices, designed and implemented the QoS prediction algorithm, performed most experiments concerning the QoS prediction, and introduced the QoS aggregation method. K.G. formalized the QoS aggregation method by using petri nets, ran most experiments concerning the QoS aggregation, collected the real-world data and did the literature study with respect to related work. D.G. and K.G. wrote approximately an equal part of the paper. The remaining authors provided feedback.

Abstract

Services offered in a commercial context are expected to deliver certain levels of quality, typically contracted in a service level agreement (SLA) between the service provider and consumer. To prevent monetary penalties and loss of reputation by violating SLAs, it is important that the service provider can accurately estimate the Quality of Service (QoS) of all its provided (composite) services. This paper proposes a technique for predicting whether the execution of a service composition will be compliant with service level objectives (SLOs). We make three main contributions. First, we propose a simulation technique based on Petri nets to generate composite time series using monitored QoS data of its elementary services. This techniques preserves time related information and takes mutual dependencies between participating services into account. Second, we propose a kernel-based quantile estimator with online adaptation of the constant offset to predict future QoS values. The kernel-based quantile estimator is a powerful non-linear black-box regressor that (i) solves a convex optimization problem, (ii) is robust, and (iii) is consistent to the Bayes risk under rather weak assumptions. The online adaption guarantees that under certain assumptions the number of times the predicted value is worse than the actual value converges to the quantile value specified in the SLO. Third, we introduce two performance indicators for comparing different QoS prediction algorithms. Our validation in the context of two case studies shows that the proposed algorithms outperform existing approaches by drastically reducing the violation frequency of the SLA while maximizing the usage of the candidate services.

3.1 Introduction

Context. Workflow languages, such as WS-BPEL[1], focus on combining web services into aggregate services that satisfy the needs of clients. A service composition consists of a collection of related, structured activities or tasks that produce a specific service by combining services provided by multiple business partners. Tasks can be delegated to globally available software services and may require human interaction. For example, an integrated travel planning web service can be created by composing services for hotel booking, airline booking, payment, etc.

In a global service market, many available web services can provide similar functionality, but have different Quality of Service (QoS). QoS can be defined in terms of attributes such as price, response time, availability and reputation [1]. QoS-aware service composition refers to the process of composing services in function of their QoS properties such that the overall composition meets the QoS constraints, specified in the service level agreement (SLA) [2]. Violating SLAs

is often associated with a decrease in reputation or monetary penalties for the service provider [3]. It is therefore in the providers' best interest to make for each potential composition an accurate assessment of the QoS that will be provided during its execution by the client. In this context, two important challenges arise: *QoS aggregation* [4, 5, 6] and *QoS prediction* [7, 8, 9]. Composite services typically consist of different activities such as sequences, conditions, loops and parallel invocations. To calculate the aggregated QoS of the composition, these different composition patterns have to be taken into account. Furthermore, web services typically operate autonomously within a rapidly changing environment. As a result, their QoS may change frequently, either because of internal changes or because of changes in their environment [10]. To know whether a composition will respect the SLA during its execution, the provider must thus predict what the QoS of the composition will be during its actual execution by the client.

Problem Statement. The problem tackled in this paper is to produce future QoS values of the composition as accurate as possible given historical QoS values of the individual services. A selected composition can then be rejected or accepted to address the customers' requests depending on whether the prediction indicates if the composition will be executed according to the service level objectives (SLOs) of the SLA. In the context of this paper, an SLO states that an accepted service should satisfy a QoS constraint with a probability greater than τ (e.g. the service must respond in less than 1 second in 99,9% of the cases). Two types of errors can occur in this scenario: a type I error occurs when the service is rejected and would have satisfied the SLO, a type II error occurs when the service is accepted and will violate the objective. Suppose f_τ is a function that predicts a future QoS attribute given a certain input (e.g. past values of that QoS attribute) and f_τ belongs to a certain hypothesis space $\mathcal{H}$. If based on this prediction we decide whether or not to accept the request (e.g. we accept if the predicted value is below or above a certain threshold value), then the problem we try to optimize is

$$\min_{f_\tau \in \mathcal{H}} P(\text{Type I error}|f_\tau)(1-\tau) + P(\text{Type II error}|f_\tau)\tau \tag{3.1a}$$

$$\text{such that } P(\text{Type II error}|\text{accepted}, f_\tau) \leq 1-\tau \tag{3.1b}$$

where P expresses a probability and 'accepted' means the service was accepted to execute the task. Constraint (3.1b) states that the frequency of accepted services that violate the SLO should not exceed $1-\tau$. As long as constraint (3.1b) is satisfied, we have chosen the cost we want to minimize in equation (3.1a) to be linear in the number of type I and type II errors: an error of type I has cost $1-\tau$ and an error of type II has cost τ. We will show in this paper

[1]Web Services Business Process Execution Language Version 2.0, April 2007, OASIS Technical Committee,
http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html

that in order to minimize optimization problem (3.1), we need to know the conditional τ-quantile value. From an economical point of view it is important to accept as many requests as possible, while the proportion of violations of the accepted request should be as small as possible. Therefore it is interesting to reformulate problem (3.1) as an optimization problem in the rejection rate (P(rejected)) and the violation rate (P(Type II error|accepted))

$$\min_{f_\tau \in \mathcal{H}} P(\text{Type II error}|\text{accepted}, f_\tau)$$

$$+ P(\text{rejected}|f_\tau)((1-\tau) - P(\text{Type II error}|\text{accepted}, f_\tau)) \quad (3.2a)$$

$$\text{such that } (1-\tau) - P(\text{Type II error}|\text{accepted}, f_\tau) \geq 0 \quad (3.2b)$$

where *rejected* means the service was rejected to execute the task. A proof for this reformulation can be found in Appendix 3.A. Given Constraint (3.2b) holds, the cost we want to minimize decreases for a decreasing rejection rate and a decreasing violation rate as desired.

Related Work. With respect to *QoS aggregation*, the current state-of-the-art techniques have some important drawbacks. As explained in [8], some existing approaches for aggregation use *hard* composition rules such as addition, maximum or conjunction [1, 7, 10, 11, 2, 12, 13]. In case one is interested in quantile values, hard contract rules can be overly pessimistic because they do not take the probability distribution of the individual services into account. This problem can be solved by estimating probability distributions for the QoS values of the elementary services which are combined according to *soft* composition rules [5, 8]. Soft contract rules, on the other hand, assume that QoS values of the different elementary services are independently distributed. In practice, however, quality attributes such as response time of different automated services are often dependent due to several reasons: both services run on the same server; during the day certain services are executed more often than during the night; users choose the fastest service out of a group of services which causes the services to become equally fast; etc. These dependencies are even more distinct with non-automatic services that require human interaction to execute a task. They are often induced due to the fact that outside working hours less people are available to execute a task.

Concerning *QoS prediction* it is important that the predicted quantile value approximates the true conditional quantile function as good as possible. We didn't find any seminal work that uses state-of-the-art prediction methods to predict quantile values in this context.

Contributions. This paper has the following contributions. First, we present a *QoS aggregation* technique that generates composite time series using monitored QoS data of its elementary services. Our proposed simulation technique based

on Petri nets preserves time related information and takes mutual dependencies between participating services into account. Second, we address *QoS prediction* by solving optimization problem (3.1) using a kernel-based quantile regressor with online adaptation of the constant offset. This estimator has the following properties:

- It can guarantee that, under certain assumptions, the number of times the predicted value is worse than the actual value converges to the agreed quantile value for the number of data points going to infinity.
- Kernel-based methods like SVM [14] and LS-SVM [15] have shown to be powerful non-linear black-box methods successful for various applications such as optical character recognition [16] and electricity load prediction[17].
- The solved optimization problem is convex as opposed to other non-linear black-box methods such as neural networks. This guarantees the global optimum can be found efficiently.
- It is robust and thus resistant to outliers. Its breakdown point equals $1 - \tau$ if $\tau \geq 0.5$.
- It is risk consistent to the Bayes risk under rather weak assumptions and it approximates the conditional quantile function [18].

Third, we introduce two performance indicators to quantify and compare our results: the first expresses, given certain assumptions, the cost we want to minimize in (3.1a) and the second expresses, given certain assumptions, the likelihood (3.1b) holds.

Structure of the Paper. This paper is organized as follows: Section 2 clarifies the problem statement and motivates our approach. Section 3 elaborates on related work. Section 4 provides an overview of the QoS model we use for this work. We propose the simulation technique based on Petri nets for calculating the QoS attributes of composite services, given QoS attributes of its elementary services in Section 5. Section 6 explains the performance indicators and describes the underlying prediction mechanism that is used to predict violations of the SLA between customer and service provider. An experimental evaluation of our approach and comparison with existing work is documented in Section 7. Section 8 provides a conclusion. Finally, Section 9 elaborates on future work.

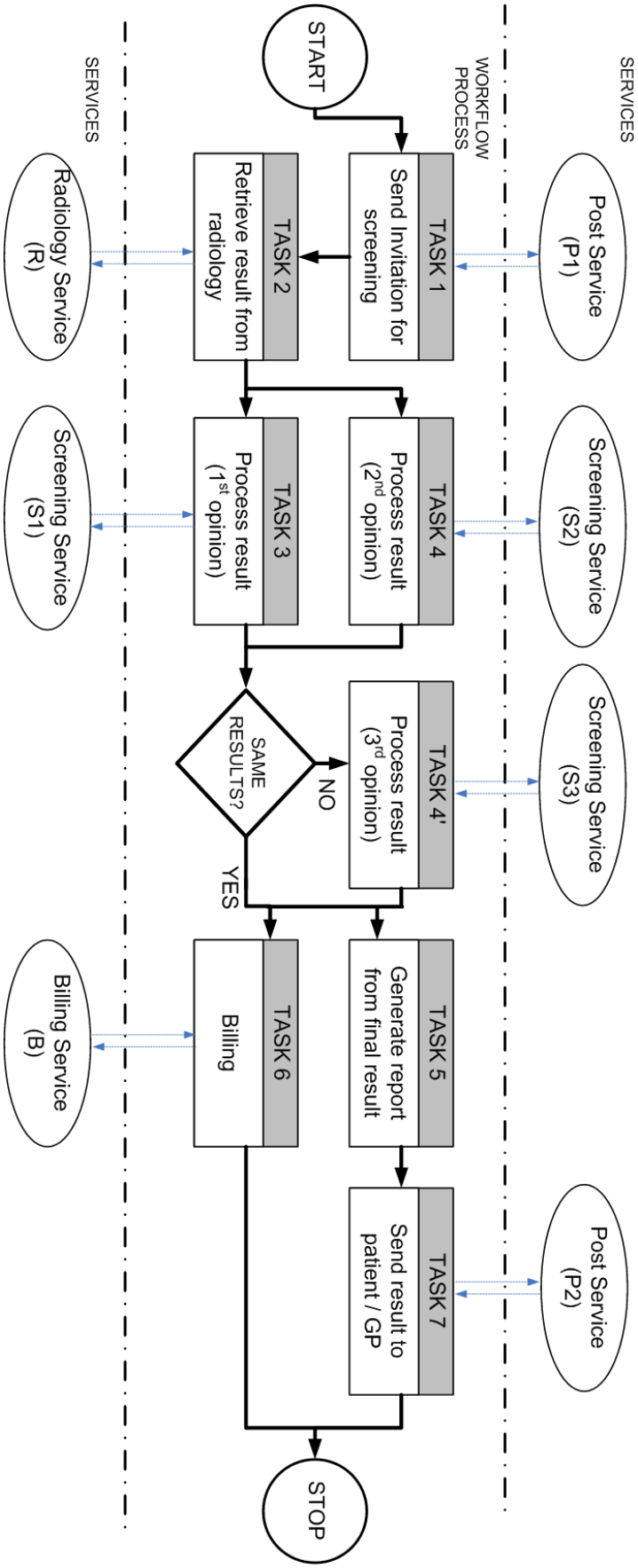


Figure 3.1: Example e-health workflow process

Table 3.1: Probability density, average and 99%-quantile values of RTs for 2 composite services CS1 & CS2.

Services	Monitored RT (Prob. Density)					99% Q-Value	Average Value
	0-5	5-10	10-15	15-20	20-25		
CS 1	0%	1%	98%	1%	0%	15	12.5
CS 2	15%	75%	6%	3%	1%	20	7.5

3.2 Motivation

3.2.1 Running Example

This section presents a case study situated in the health care environment. The case study consists of a composite service (workflow), initiated by the government, that realizes a mammography screening program in order to reduce breast cancer mortality. The workflow is illustrated in Figure 3.1.

The first task of the workflow consists of sending out invitations to all women that qualify for the program. A radiologist will take images needed for screening and uploads them to the system (task 2). Next, the images need to be analysed by specialized screening centers. There are always two independent readings, represented by tasks 3 and 4. These readings can be performed in parallel. In a next step, the two results of the readings are compared. When the results are identical, it is unlikely that the two physicians made the same mistake. Therefore it can be safely assumed that results are correct and the workflow can proceed with task 5. However, when the results are different, a concluding reading is performed (task 4'). Once the results of the screening of a particular screening subject are formulated, a report is generated (task 5) and a report is sent to the screening subject and her general practitioner in task 7. In parallel, different parties are billed (task 6).

Suppose that the government, who finances this initiative, wants some quality guarantees and specifies a service level agreement with the company (service provider) responsible for executing the workflow. In this agreement, the company specifies, for example, that in 99% of the cases the duration of the workflow composition will take no longer than 15 working days. We will now motivate why quantile estimation, time series prediction and taking into account mutual QoS dependencies are important for accurate SLO compliance estimation.

Table 3.2: Past (monitored) and future (to be predicted) RTs for two composite services.

Services	Monitored RT (days)				Real RT (days)	
	t1	t2	t3	t4	t5	t6
CS A	10	25	10	25	**10**	**25**
CS B	20	10	20	10	**20**	**10**

3.2.2 Quantile vs. Average SLO Compliance Estimation

The algorithms that we propose are based on quantile estimation. We explain its benefit by means of a simple example. Suppose that two different service selections on the e-health workflow leads to two composite services CS1 & CS2. The response time of these services have, at a certain time, probability densities for RTs of 0 to 25 days as presented in Table 3.1. We can use these values to make a quality estimate for an SLO. For example, which service would be the best candidate to have 99% certainty that its response time will not exceed 15 days? Using the average RTs of 12.5 days for CS1 and 7.5 days for CS2, it seems that CS2 is more reliable than CS1 and thus the best choice. However, using 99% quantile values of 15 days for CS1 and 20 days for CS2, we rightly conclude the opposite: CS1 has a higher probability that it will not exceed the threshold of 15 days because it is less volatile.

3.2.3 Time Series Prediction

Again, consider a scenario where one has to estimate which of two composite services, as shown in Table 3.2, is the best candidate to comply with an SLO stating that the total duration of the composite service will take no longer than 15 working days. If we expand the pattern present in $t1$, $t2$, $t3$ and $t4$ to $t5$ and $t6$ as shown in Table 3.2, then at $t5$ 'CS A' complies with the SLO while 'CS B' violates the SLO and at $t6$ the opposite happens. The estimation for this scenario, as for many real-life scenario's, is thus time dependent. Possible causes for varying quality of service attributes are temporary over -or underload, infrastructure failures, seasonality due to fixed working hours of non-automatic services, etc. The algorithms we propose to predict time varying QoS attributes are discussed in Section 3.6.

Table 3.3: Monitored response times for the screening service (SS), the post service (PS) and the corresponding response time for the composite service. The response time for SS equals one working day and the response time for PS equals 2 working days. During the weekend (when no personnel is available to execute the tasks) the progress freezes and the response time will be extended by up to 2 days.

	Mo	Tu	We	Th	Fr	Sa	Su	**(Max)**
SS	1	1	1	1	3	2	1	**3**
PS	2	2	2	4	4	3	2	**4**
CS	3	3	5	5	5	4	3	5

3.2.4 Mutual Dependencies

In this example we will show why taking into account mutual dependencies (e.g. as a consequence of time dependencies) is important. Suppose the composite service (CS) consists of a screening services and a post service (SS and PS) executed in sequence as shown in Table 3.3. The response times are dependent in the sense that if the execution of SS contains a weekend, then the execution of PS does not contain a weekend. The true worst-case response time is 5 days (3 working days and an extension of 2 days due to the weekend). Algorithms assuming independence of response times will estimate the worst-case response time as 7 days (3 days for SS plus 4 days for PS) because they add the penalty due to weekends twice. Using these values, for example to calculate the 99% QoS quantile, will result in a too pessimistic value (7 days instead of 5 days), causing a competitive disadvantage to the service provider. This example shows why in real scenario's assuming independence can be dangerous. Mutual dependencies as a consequence of sharing resources or other dependencies give similar results.

3.3 Related Work

In literature, several works can be found that address QoS-aware service composition. Surveys are reported in [19, 20, 21]. Important research challenges in this domain are **QoS aggregation, QoS-based service selection** and **QoS prediction**. Figure 3.1 and Table 3.1 give an overview of related work for each challenge. We discuss them briefly.

QoS aggregation, which is typically part of service selection and prediction, involves calculating a global QoS value based on the QoS of the participating services. Jaeger et al. [4] introduce an aggregation method to calculate the QoS of the composition, given the QoS values of the individual services. The model identifies several structural elements called composition patterns. These

Table 3.1: Related work - overview

Related work	Challenge	Approach	Implementation choices
Cardoso(2002, 2004)[1, 7]	prediction	stochastic workflow reduction	hqv (avg, min, max), mutually independent
Jaeger(2004)[4]	aggregation	composition patterns	deterministic
Zeng(2004)[10]	selection	linear integer programming	deterministic
Aggarwal(2004)[11]	selection	linear integer programming	deterministic
Canfora(2005)[2]	selection	genetic algorithms	deterministic
Hwang(2007)[5]	aggregation	probabilistic composition patterns	stochastic, mutually independent
Harney(2007)[12]	selection	value of changed information	deterministic
Yu (2007)[13]	selection	efficient heuristic algorithms	deterministic
Wiesemann(2008)[23]	selection	stochastic programming	stochastic, mutually independent
Rosario(2008)[8]	prediction	monte carlo sampling	hqv (distribution), mutually independent
Mukherjee(2008)[6]	aggregation	composition patterns for WS-BPEL	deterministic, mutually independent
Leitner(2010)[9]	prediction	regression	other (runtime data of composition)
This paper	prediction	petri nets, support vector machines	hqv (time series), mutually dependent

hqv:	historical values of qos attributes
deterministic:	qos values of participating services are fixed point estimates
stochastic:	qos values of participating services are stochastic variables that change in time
mutually independent:	qos values among participating services do not depend on each other

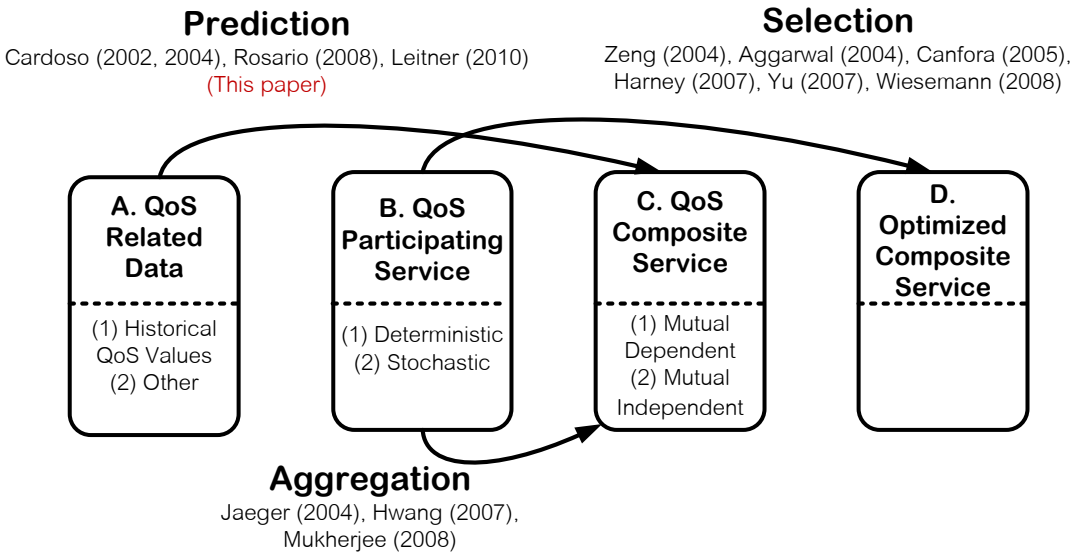


Figure 3.1: Related work - positioning

structural elements were derived from a set of workflow patterns defined in Van der Aalst [22]. A practical approach is taken by Mukherjee [6]. They propose a model for estimating three key QoS parameters (Response Time, Cost and Reliability) of an executable BPEL process from the QoS information of its partner services and certain control flow parameters. Hwang et. al. [5] approaches the aggregation challenge as a stochastic problem. Each QoS parameter for a web service is represented as a discrete random variable with a probability mass function. They propose a probabilistic framework to derive a QoS measure of a composite service from those of its constituent services and explore algorithms for computing the probability distribution functions of the QoS of the service composition. Their theoretical rules for composing the probability mass function are based on the assumption that QoS values of each constituent web service of a composition construct are mutually independent.

QoS-based service selection deals with finding an assignment of services to workflow tasks which maximizes a customer related utility function. Typically this comes down to the following optimization problem: given an abstract composite service and a set of candidate services with different QoS values for each task, find a service for each task such that the utility is maximized and the global composite QoS values satisfy certain Service Level Objectives (SLOs). Popular techniques in literature to solve this challenge efficiently are integer programming [10, 11], efficient heuristic algorithms [13] and genetic algorithms [2]. These works tackle the composition problem assuming deterministic QoS attributes for the elementary services. Harney et al. [12] present a composition solution that intelligently adapts workflow processes to changes in quality parameters of participating services providers. Their approach

assumes that QoS values remain fixed for a certain amount of time. Changes are introduced by means of expiration times, i.e. service providers provide their current reliability rates and duration of time for which the current reliability rates are guaranteed to remain unchanged. Wiesemann et al. [23] focus on the stochastic service composition problem. They formulate the service composition problem as a multi-objective stochastic program which simultaneously optimizes QoS parameters which are modelled as decision-dependent random variables. Their model minimizes the average value-at-risk (AVaR) of the workflow duration and costs while imposing constraints on the workflow availability and reliability.

Driven by the fact that QoS attributes, such as response time, can be very volatile with respect to time, the challenge of **QoS prediction** has recently gained popularity. Its main goal is to bridge the time gap between the service selection process and the actual execution of the composite service. By predicting accurate expected values for quality measures in the near feature, this technique is able to improve the probability that the selected composition of services will still respect the SLO during the execution of the workflow. This challenge typically starts from QoS related data and assumes that a selection of services for the composition has taken place. Cardoso's PhD. thesis [1] is a seminal work that proposes a framework that uses Stochastic Workflow Reduction to arrive at QoS estimates for the overall workflow. Using different workflow patterns, they defined QoS aggregation with four attributes: response time, cost, reliability, and fidelity. These aggregation patterns make it possible to predict the QoS performance of service processes by performing the substitution repeatedly until the whole process is transformed into a composite service node. Although Cardoso mentioned the possibility of deriving distribution functions for QoS of workflow tasks, the proposed reduction rules were only implemented for point estimates, such as as minimum, average, and maximum, of historical QoS values. Leitner et al. [9] present a framework called PREvent for event-based monitoring and prediction of SLA violations and automated runtime prevention by triggering adaptation actions in service compositions. Monitored runtime data of the composition is used by a predictor to identify problematic instances at defined checkpoints in the composition execution via regression. Because their approach does not use historical QoS values of the participating services, there is no need for aggregation or for taking service dependencies into account. Rosario et al. [8] propose QoS estimation based on soft contracts. Soft contracts are characterized through probability distributions for QoS parameters. To yield a global contract for the composition, they use a tool called TOrQuE to unfold a composition and estimate its response time using Monte Carlo simulation. Their simulation technique assumes that the probability distribution of QoS values of the elementary services are independently distributed. As discussed in this work, this assumption is often violated in practice with as consequence that their approach leads to overoptimistic or overpessimistic results.

In this paper, we present a two-step approach to predict QoS values of

composite services. The first step, which is based on Petri nets, deals with **QoS aggregation** by deriving QoS values of a workflow composition from those of its participating elementary services. In contrast to related work, this technique preserves both mutual and time dependencies to allows us, in a second step, to effectively deal with the **QoS prediction** challenge. We apply time series prediction on the aggregated historical QoS to accurately predict if a composition will comply with an SLA. The accuracy of our approach is compared with the approach of Rosario [8] in Section 3.7.2.

3.4 QoS Considerations

3.4.1 QoS for Elementary Service

In the domain of web services, QoS parameters can be used to determine non-functional properties of the service. QoS attributes can be divided into quantitative and qualitative attributes. Examples of the latter are security and privacy. Popular quantitative attributes are response time, throughput, reputation, reliability, availability, and cost:

- Response Time (RT): the time taken to send a request and receive a response (expressed in milliseconds). The response time is the sum of the processing time and the transmission time. For short running processes they are usually of the same order. For long running processes that can take hours, days or even weeks to complete, the transmission time is usually negligible.
- Throughput (TP): the maximum requests that can be handled at a given unit in time (expressed in requests/minute).
- Reputation (RP): the reputation of a service is a measure of its trustworthiness (expressed as scalar with higher value being better). The value is defined as the average ranking given to the service by end users.
- Reliability (RL): the probability that a task is satisfactorily fulfilled (expressed as a percentage). The reliability can be calculated from past data by dividing the number of successful executions by the total number of executions.
- Availability (A): the probability that a web service is available (expressed in available time/total time). It is computed by dividing the total amount of time in which a service is available through the total monitoring time. In the scope of this work, we define an available service as a service that is able to respond within a predefined time interval.

Table 3.1: QoS computations for composite services. RT_i, TP_i, RP_i, RL_i, A_i and C_i are respectively the response time, throughput, reliability, availability and cost of the i'th service. There are a total of m services which are part of a sequence, parallel execution or switch. In case of a switch the expected value is calculated where p_i is the probability that service i is executed. In case of a loop one service is executed k times.

QoS Attribute	Composition Patterns			
	Sequence	**Parallel**	**Switch**	**Loop**
Response Time	$\sum_{i=1}^{m} \mathrm{RT}_i$	$\max_{i=1}^{m}(\mathrm{RT}_i)$	$\sum_{i=1}^{m} p_i \cdot \mathrm{RT}_i$	$\mathrm{RT} \cdot k$
Throughput	$\min_{i=1}^{m}(\mathrm{TP}_i)$	$\min_{i=1}^{m}(\mathrm{TP}_i)$	$\sum_{i=1}^{m} p_i \cdot \mathrm{TP}_i$	TP
Reputation	$\sum_{i=1}^{m} \frac{\mathrm{RP}_i}{m}$	$\sum_{i=1}^{m} \frac{\mathrm{RP}_i}{m}$	$\sum_{i=1}^{m} p_i \cdot \mathrm{RP}_i$	RP
Reliability	$\prod_{i=1}^{m} \mathrm{RL}_i$	$\prod_{i=1}^{m} \mathrm{RL}_i$	$\sum_{i=1}^{m} p_i \cdot \mathrm{RL}_i$	RL^k
Availability	$\prod_{i=1}^{m} \mathrm{A}_i$	$\prod_{i=1}^{m} \mathrm{A}_i$	$\sum_{i=1}^{m} p_i \cdot \mathrm{A}_i$	A^k
Cost	$\sum_{i=1}^{m} \mathrm{C}_i$	$\sum_{i=1}^{m} \mathrm{C}_i$	$\sum_{i=1}^{m} p_i \cdot \mathrm{C}_i$	$\mathrm{C} \cdot k$

- Cost (C): the cost that a service requester has to pay for invoking a specific operation of a service (expressed in cents/request). Other pricing schemes are sometimes used such as membership fee or monthly fee.

Because the focus of this paper is on the prediction of quality of service attributes with a volatile nature, we do not consider static attributes like fixed costs such as membership fees. Also reputation is an attribute that gives a general impression about users opinions of a service, and is not meant to frequently change in time. System-level QoS attributes, such as throughput, often largely depend on hardware and computing power of the underlying infrastructure of the composite service and need to be evaluated over multiple instances. In this work, we concentrate on instance-level QoS attributes of composite services that directly relate to the QoS values of its constituent web services and moreover are non-stationary. Interesting attributes for our approach are response time, reliability, availability and cost as pay-per-service. For the sake of simplicity we limit the QoS prediction algorithms in Section 3.6 and their evaluation in Section 3.7 to the response time attribute.

3.4.2 QoS for Composite Services

The QoS of a service composition is calculated based on the QoS values of its constituents. In contrast to the measurement of QoS for elementary services, composite services consist of different activities such as sequences, if-conditions, loops and parallel invocations. We need to take into account these different composition patterns to calculate the QoS of a composite service. Example QoS computations are summarized in Table 3.1. WS-BPEL elements relevant to QoS computation are simple elements as receive, reply, invoke, assign, throw, wait and complex elements like sequence, flow, if, while and foreach. Similar to Kiepuszewski et al. [24], we define a structured model that consists of four constructs that allow for recursive construction of larger workflows:

- Sequence: multiple tasks that are sequentially executed.
- Parallel execution (and-split/and-join): multiple paths that are executed concurrently en merged synchronously.
- Exclusive choice (or-split/or-join): multiple possible paths, among which only one can be executed.
- Loop: a path that is repeatedly executed a fixed number of times c.

Various standards for service composition include more constructs in addition to the four basic constructs described above. Jaeger et al. [4] summarizes the workflow patterns that cover most control constructs proposed in existing standards and products. In this paper we limit ourselves to the four basic constructs that are able to cover the most important activities offered in WS-BPEL, a popular workflow language for orchestrating web services. How the composite activities of WS-BPEL are mapped to the basic constructs will be explained in the next section. A Petri net graph and a Petri net execution time system (PNET-system) is used to reason over the workflow and to estimate its total response time.

Besides the overhead generated by the service calls, the QoS of a composite service is influenced by events internal to the orchestration. Usually, the delay caused by internal events is negligible compared to that of the service calls. This is certainly the case for medium and long running processes, which are the main targets for our approach. For further analysis, we assume that the overall delay of the orchestration depends solely on the response times of the services it calls during execution. The inclusion of internal delays is a trivial extension.

3.5 QoS Aggregation

In the motivation, we emphasized that time related information and mutual dependencies are important to make accurate composite QoS estimates. In this section we present a QoS aggregation technique that takes both into account by generating a simulated time series of the QoS attributes of the workflow as if the composition was executed several times in the past. The quantile values will be predicted using this simulated dataset as will be explained in Section 3.6.

3.5.1 Petri Net Graph

Various activity-based process models have been suggested for workflow systems. We represent the workflow as a non-deterministic Petri net graph (PNG), which is a commonly used representation for workflow systems [25, 26]. Compared to a structured workflow model, expressing our QoS model as a formal model offers more expressive power and is equipped with strong analysis capabilities. Furthermore, the Petri net functions as an intermediate representation that allows generalization of our approach to different workflow languages. We apply our technique in the context of WS-BPEL, however, it can be applied on any workflow language that can be represented by our Petri net representation.

Our Petri net consists of places and transitions. Places correspond to workflow states and allow the representation of conditional execution. There are two kinds of transitions in our Petri net: timed and immediate transitions. Timed transitions represent services and the firing delay of the transition corresponds to the response time of these services. Immediate transitions are needed to enable the representation of internal events of the composite service.

A generalization of the Petri net graph we use for our analysis of WS-BPEL, is a 8-tuple ($\mathcal{P}$, $\mathcal{T}^1$, $\mathcal{T}^2$, $\mathcal{T}$, $\mathcal{D}$, W^-, W^+, s_0), where

- $\mathcal{P} = \{p_1, p_2, ..., p_{n_1}\}$ is a finite set of places. The set contains exactly one starting place (p_1) and exactly one ending place (p_{n_1}).
- $\mathcal{T}^1 = \{t_1^1, t_2^1, ..., t_{n_2}^1\}$ is a finite set of immediate transitions. Immediate transitions have no firing delay.
- $\mathcal{T}^2 = \{t_1^2, t_2^2, ..., t_{n_3}^2\}$ is a finite set of timed transitions. Timed transitions have a firing delay.
- $\mathcal{T} = \{t_1, t_2, ..., t_{n_4}\}$ is a finite set of transitions. All transitions are immediate or timed transitions ($\mathcal{T} = \mathcal{T}^1 \cup \mathcal{T}^2$) but cannot be both ($\mathcal{T}^1 \cap \mathcal{T}^2 = \emptyset$).

- $\mathcal{D} = \{d_1, d_2, ..., d_{n_3}\}$ is finite set of positive real functions representing the firing delay of the corresponding timed transitions with respect to the time. $d_i(s)$ is the firing delay of t_i^2 at time s.

- W^-, W^+ are the backward and forward incidence matrices, respectively. They contain boolean values. If $W^-(i,j) = 1$, then there is an arc going from p_i to t_j. If $W^+(i,j) = 1$, then there is an arc going from t_i to p_j. If $W^-(i,j)$ or $W^+(i,j)$ equals 0, then there is no corresponding arc. Places, except for the starting and ending place, can have multiple incoming arcs ('OR-join') or multiple outgoing arcs ('OR-split'). The starting place differs in the sense it has no incoming arcs and the ending place differs in the sense it has no outgoing arcs. Timed transitions have one incoming arc and one outgoing arc. Immediate transitions can have multiple incoming arcs ('AND-join') or multiple outgoing arcs ('AND-split').

- The initial marking of the Petri net is always one token present at the starting place. The initial time of the Petri net equals s_0. Time is continuous and can take any real value.

- An additional constraint on our Petri net is that each 'AND-split'-transition, 'OR-split'-place has to have exactly one corresponding 'AND-join'-transition, 'OR-join'-place respectively and vice versa. With corresponding we mean that all outgoing paths of the 'split'-node are disconnected until they reach the corresponding 'join'-node in which they *all* come together. The paths connecting the 'split'-node and corresponding 'join'-node are called the connecting paths.

The mapping of WS-BPEL activities, represented by their Business Process Modeling Notation (BPMN)[2], to their corresponding Petri net representation is shown in Figure 3.1. If we apply this mapping on the case study introduced in Section 3.2.1, we get the Petri net representation illustrated in Figure 3.2. The firing delays introduced by the timed transitions t_1^2 to t_7^2 correspond to the response times of the post 1, radiology, screening 1 to 3, report, billing and post 2 service respectively. Remark that we have truncated the Petri net graph by removing 'a place followed by an immediate transition with one incoming and one outgoing arc'. These constructs have no influence on further analysis and calculations.

Similar to Colored Petri nets, we add an extension to the elements of the net to deal with the other QoS attributes besides response time: a Petri net token is associated with 3 data values C, A and RL of type integer to hold the current aggregated cost, availability and reliability of a composition. The extension is a 4-tuple ($\mathcal{Q}$, $\mathcal{C}$, $\mathcal{A}$, $\mathcal{RL}$), where

[2]Business Process Modeling Notation (BPMN) Version 2.0, January 2011, OMG Specification, `http://bpmn.org/`

- $\mathcal{Q} = \{q_1, q_2, ..., q_{n_3}\}$ is a finite set of positive real numbers representing the maximal allowed delay for each timed transition before it is considered as unavailable.
- $\mathcal{C} = \{c_1, c_2, ..., c_{n_3}\}$ is a finite set of positive real functions representing the cost of the corresponding timed transitions with respect to the time. $c_i(s)$ is the cost of t_i^2 at time s.
- $\mathcal{A} = \{a_1, a_2, ..., a_{n_3}\}$ is a finite set of functions where $a_i(s) \in \{0, 1\}$ represents the availability of the corresponding timed transitions with respect to the time. $a_i(s)$ is the availability of t_i^2 at time s. There is a direct relation between a_i and d_i: if $(d_i <= q_i)\{a_i = 1\}$ else $\{a_i = 0\}$
- $\mathcal{RL} = \{rl_1, rl_2, ..., rl_{n_3}\}$ is a finite set of functions where $rl_i \in \{0, 1\}$ represents the reliability of the corresponding timed transitions with respect to the time. $rl_i(s)$ is the reliability of t_i^2 at time s.

3.5.2 Petri Net Execution Semantics

The execution of the Petri net graph is done by passing tokens from the initial marking to the end marking. These markings correspond to a token present at start place and end place respectively. An execution of the Petri net graph simulates the execution of the workflow. The execution (response) time of the workflow at time s is the time elapsed between a token present at the starting place at time s and a token reaching the ending place in the same execution cycle. During execution of the Petri net, time increases in a continuous way, starting from the initial value s_0. For each time, the following rules are executed in a non-deterministic order until no more rules can be executed:

- Rule 1: if a token is present at the incoming place of a timed transition t_i^2, then that token is consumed and the fire delay counter of that transition is activated. The counter starts counting down from $d_i(s)$ where s is the current time. As time increases, the counter value decreases with an equal amount.
- Rule 2: if an active fire delay counter of a timed transition reaches zero, then a token is generated at the outgoing place and the fire delay counter is deactivated.
- Rule 3: if tokens are present at *all* incoming places of an immediate transition, then these tokens are consumed and new tokens are generated at all outgoing places.

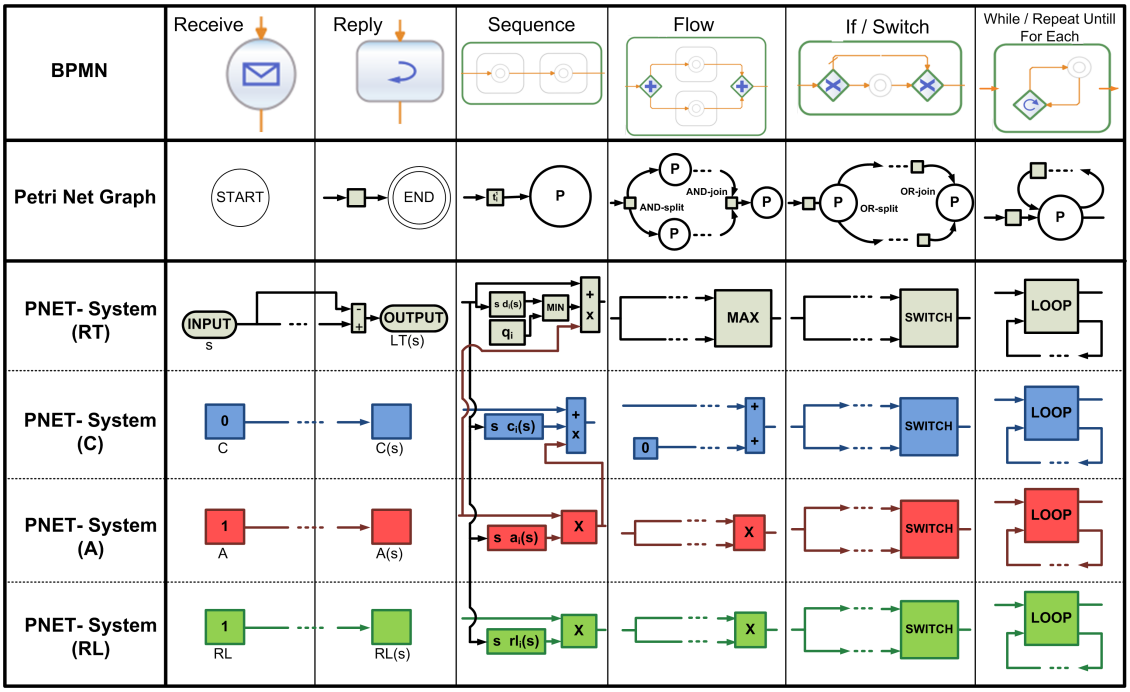


Figure 3.1: Mapping: Business Process Modeling Language (BPMN) - Petri net graph - PNET-system. The gray, blue, red and green building blocks of the PNET-system correspond to respectively the response time, cost, availability and reliability.

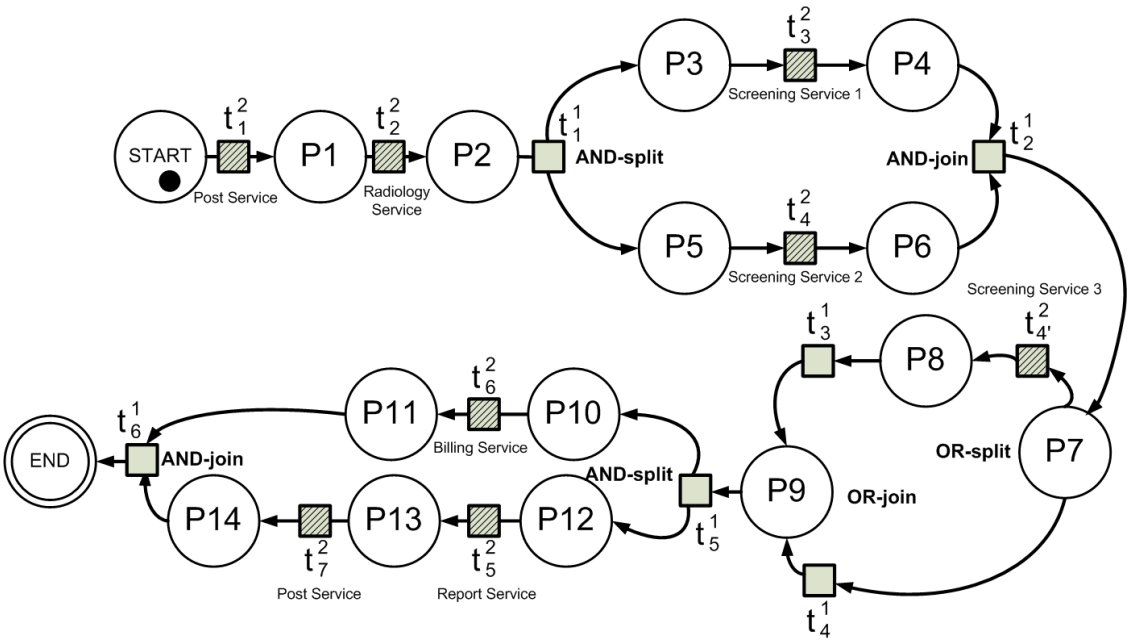


Figure 3.2: Petri net graph for e-health workflow

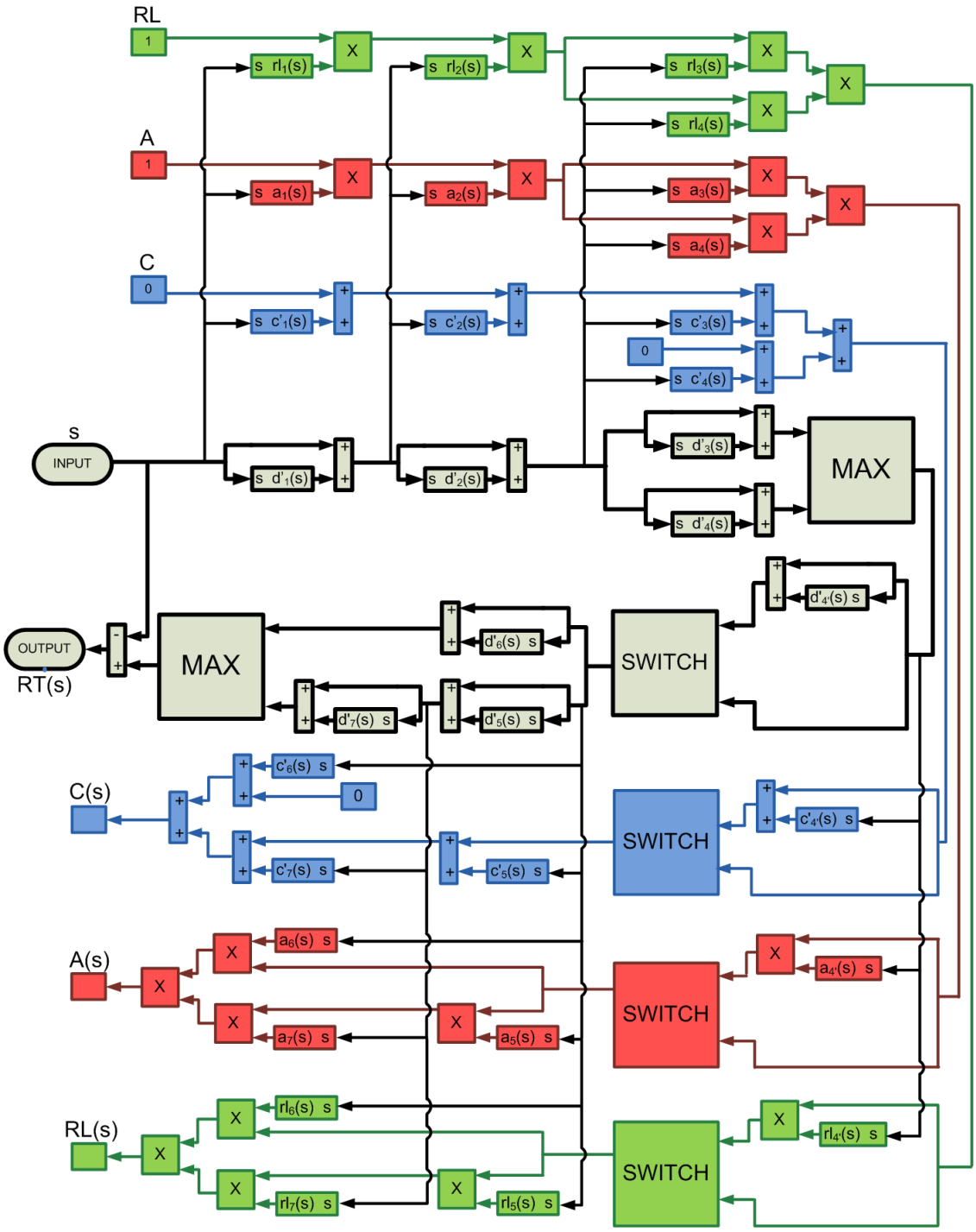


Figure 3.3: Petri net execution time system for e-health workflow.

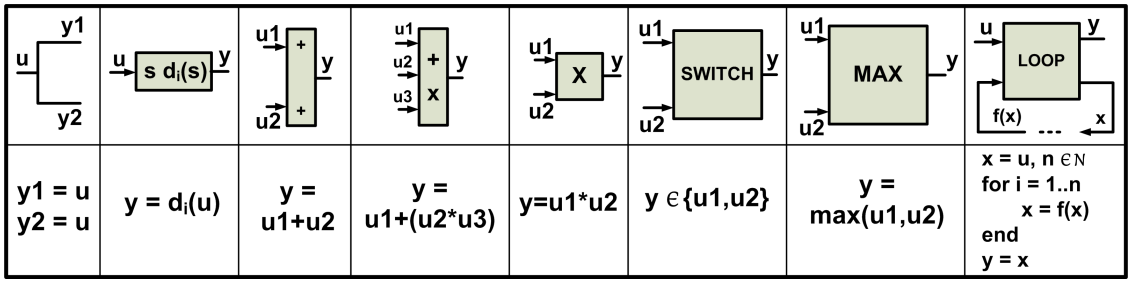


Figure 3.4: Overview of the PNET-system building blocks. The top row visualises the building blocks and the bottom row describes the input-output behavior.

- Rule 4: if a token reaches the ending place, then the execution stops. Time freezes and no more rules are executed. The execution time of a Petri net graph equals the time at which the execution stops minus the initial time.

Applied on our case study, the token starts at the start place at s_0. The only applicable rule is rule 1 where the fire delay counter of the timed transition t_1^2 is activated. When the counter reaches zero at time $s_0 + d_1$, a new token is generated according to rule 2 and arrives at P1. Analogue, the token reaches P2. At P2 rule 3 is executed and new tokens are generated at both outgoing places of t_1^1. The parallel execution is done according to rule 1 followed by rule 2 for each branch in a non-deterministic order. The parallel execution ends when both tokens are consumed and a new token is generated at the immediate transition t_2^1 according to rule 3. When this token arrives at P7, both rule 1 and rule 3 can be executed corresponding to the upper and lower path respectively. Again, a rule is chosen non-deterministically. If rule 3 fires, a token is generated at the lower path and sent to the immediate transition. If rule 1 fires, a token is generated at the upper path, going to the timed transition $t_{4'}^2$. The next steps are similar to what we explained already. The execution ends after the 'AND-join', where rule 4 is applied and the token reaches its ending state. The execution time of the Petri net graph, where tokens are passed from start to end, corresponds to the simulated response time of the composite service.

To keep track of the cost, availability and reliability during the execution of the Petri net, we add the following rules as an extension to the existing ones:

- Rule 0: if a token is present at the start place, all associated data values are initialized as follows: $C \leftarrow 0, A \leftarrow 1, RL \leftarrow 1$.
- Rule 1': if the firing delay $d_i(s)$ of a timed transition t_i^2 is above a predefined value q_i, we consider the timed transition as unavailable. The data values associated with the token are updated as follows: $A \leftarrow 0, C \leftarrow$

$C, RL \leftarrow 0$ and the Petri net execution is halted. The Petri net execution time for the halted service then equals q_i. The response time now equals the time elapsed between a token present at the starting place and the time at which the execution is halted.

- Rule 2': if an active fire delay counter of a timed transition t_i^2 reaches zero, then a token is generated at the outgoing place and the fire delay counter is deactivated. The data values associated with the token are updated as follows: $A \leftarrow 1, C \leftarrow C + c_i(s), RL \leftarrow RL \times rl_i(s)$.

Taking into account these extension rules for the case study, the data values C, A and RL contain the current aggregated QoS values for the cost, availability and reliability respectively.

3.5.3 Petri Net Execution Time System

Instead of simulating the above Petri net graph, we derive in this subsection a transformation of a Petri net graph into a Petri net execution time system (PNET-system). A PNET-system immediately outputs the time at which the execution of the corresponding Petri net graph stops, given the time at which the execution starts. The extended PNET-system also outputs the corresponding cost, availability and reliability of the Petri net execution. Because the Petri net is non-deterministic, the PNET-system is non-deterministic as well. An overview of the building blocks is illustrated in Figure 3.4. We define $S_i^-, C_i^-, A_i^-, RL_i^-$ and $S_i^+, C_i^+, A_i^+, RL_i^+$ as the values of the simulation time, cost, availability and reliability right before, respectively right after the execution of the timed transition t_i^2. The simulation time s represents the virtual time during which the Petri net is executed. The PNET-system is generated as follows out of a Petri net graph for the different QoS attributes (see also figure 3.1 for the relationship between their constructs):

Response Time:

- Instead of calculating the execution (response) time, we are, until now, calculating the time at which the execution stops. For that reason we subtract the initial time (s_{start}) from the time at which the execution stops: $RT(s) = s - s_{start}$.
- The time at which a token is generated at the outgoing place of a timed transition t_i^2 equals the time at which a token is consumed at the incoming place added with the firing delay $d_i(s)$. A timed transition in a PNG corresponds to an addition with the delay at that time in a PNET-system. In the extended Petri net execution, any delay above the predefined value q_i is not taken into account because the workflow is then considered

unavailable. Delays on an unavailable path are also not taken into account to calculate the simulated response time. The simulation time after the timed transition t_i^2 is fired at time s is thus calculated as $S_i^+ = S_i^- + \min(d_i(s), q_i) * A_i^-$.

- An 'AND-split' means all paths are executed simultaneously. The time at which the 'AND-split' is fired is copied to all outgoing places. An 'AND-split' in a PNG corresponds to a 'fork' in a PNET-system. The time at which a token is fired at the 'AND-join'-transition equals the maximum of the times on which a token is generated at the incoming places. An 'AND-join'-transition in a PNG corresponds to a 'MAX'-building block in a PNET-system.
- An 'OR-split' with corresponding 'OR-join' means a token is send into one of the connecting paths. This is equivalent to sending tokens into all connecting paths, but all but one of the former connecting paths are disconnected from the former 'OR-join'. An 'OR-split' in a PNG corresponds to a 'fork' in a PNET-system and an 'OR-join' in a PNG corresponds to a 'SWITCH'-building block in a PNET-system.
- A loop in a PNG corresponds to a 'LOOP'-building block in a PNET-system.

Availability & Reliability:

- The initial availability (reliability) is initialized to 1 (100%).
- An 'AND-split' in a PNG corresponds to a 'fork' in a PNET-system. An 'AND-join'-transition in a PNG is where two parallel paths meet. The resulting availability (reliability) is the product of availability (reliability) of both paths. An 'AND-join'-transition in a PNG corresponds to a multiplication-building block in a PNET-system.
- An 'OR-split' in a PNG corresponds to a 'fork' in a PNET-system and an 'OR-join' in a PNG corresponds to a 'SWITCH'-building block in a PNET-system.
- The availability (reliability) at which a token is generated at the outgoing place of a timed transition equals the availability (reliability) at which a token is consumed at the incoming place multiplied with the availability (reliability) at the simulated execution time (starting time plus the time induced by the firing delays of the previous timed transition). Current simulated execution time is thus used as an input to retrieve the availability (reliability) at a specific simulated execution point.
- A loop in a PNG corresponds to a 'LOOP'-building block in a PNET-system.

Cost:

- The initial cost is initialized to 0.
- The cost of a parallel execution equals the summation of the costs generated on all paths. To take into account the cost generated before the parallel execution, one path is connected with the previous execution path by a straight line. All other paths start counting form zero. An 'AND-join'-transition in a PNG corresponds to a summation-building block in a PNET-system. The result is a summation of all prior costs.
- An 'OR-split' in a PNG corresponds to a 'fork' in a PNET-system and an 'OR-join' in a PNG corresponds to a 'SWITCH'-building block in a PNET-system.
- The cost at which a token is generated at the outgoing place of a timed transition equals the cost at which a token is consumed at the incoming place plus the cost at the current simulated execution time. Also here the current simulated execution time is a necessary input to retrieve the cost. When a service is not available, the execution of the PNG stops and no further costs are made. In the PNET-system this is achieved by multiplying the cost generated by the timed transition with the aggregated availability of the system. The resulting cost after the timed transition t_i^2 that is fired at time s is thus calculated as $C_i^+ = C_i^- + c_i(s) * A_i^+$.
- A loop in a PNG corresponds to a 'LOOP'-building block in a PNET-system.

The extended PNET-system for the e-health workflow is illustrated in Figure 3.3. To keep the overview, we did not include all the building blocks as defined in Figure 3.4 but used the modified functions $d_i'(s) = \min(d_i(s), q_i) * A_i^-$ and $c_i'(s) = c_i(s) * A_i^+$ instead.

3.5.4 From Non-Deterministic to Deterministic

To make our approach practically usable, we need to find a way to simulate a non-deterministic system using a deterministic algorithm. To allow ex-ante estimation of QoS values, we need to make assumptions on how many times a loop will be executed and which path will be followed after a conditional execution. This is not always trivial because the number of loop executions or the value of a condition is often only known at run-time. Possible strategies for resolving the non-deterministic constraints by making deterministic assumptions are:

- Assume the worst-case scenario. In case of a loop, this means to use the maximum number of times it can be executed in practice. For a conditional execution, the worst-case path depends on the QoS attribute. The worst response time is on the path that takes the longest time to execute. This can be modelled by replacing the 'OR-split' and 'OR-join' by an 'AND-split' and 'AND-join' (parallel execution with synchronization) in the Petri net graph and replacing the 'SWITCH' by a 'MAX' building block in the PNET-system. The worst-case cost is on the most expensive path and for availability (reliability), it is the least available (reliable) path. In practice, one can simulate the QoS values for all paths and take the worst values for each attribute. For a more general approach to worst case execution analysis, we refer to specialized literature in this domain [27].
- Use a probabilistic model by assigning probabilities to the number of times a loop is executed or to each path that is implied by a condition. In practice, this strategy can be realized by doing the simulation for all paths considered and assign probabilities to the resulting QoS values. An important remark here is that the resulting values cannot simply be added after multiplication with their corresponding probabilities if one wants to estimate quantile values.

3.5.5 Example Simulated QoS Calculation

In this section, we explain more in detail how we generate a composite time series from individual time series of the constituting services using the PNET-system. Suppose we have monitored the response times for several consequent time periods of the 8 services used in the mammography workflow as shown in Figure 3.5. We generate a virtual time series by simulating the execution of the composite service according to the PNET-system discussed in the previous subsection. The inputs are fixed time steps in the past. For example, if the workflow would be executed at time 0.8, we can see that the execution of the first service (post service) takes approximately 1.31 time units. This implies that the second service (radiology service) will be executed at time 2.11 (sum of 0.8 and 1.31). The closest monitored result is at t=2 with a response time of 1.27 units. The resulting execution time is 3.38 (sum of 2.11 and 1.27). At this point, the two screening services S1 and S2 will be executed in parallel with corresponding RTs of 3.22 and 1.78. Because this is a parallel execution where the workflow has to wait for the slowest service to finish, we will arrive at the next service at time 6,61 (maximum ending time of SS1 and SS2). The other calculations are similar. For the *if condition*, we have to make a deterministic assumption concerning the path that will be chosen. In practice, in most of the cases of a mammography screening the results of SS1 and SS2 will probably

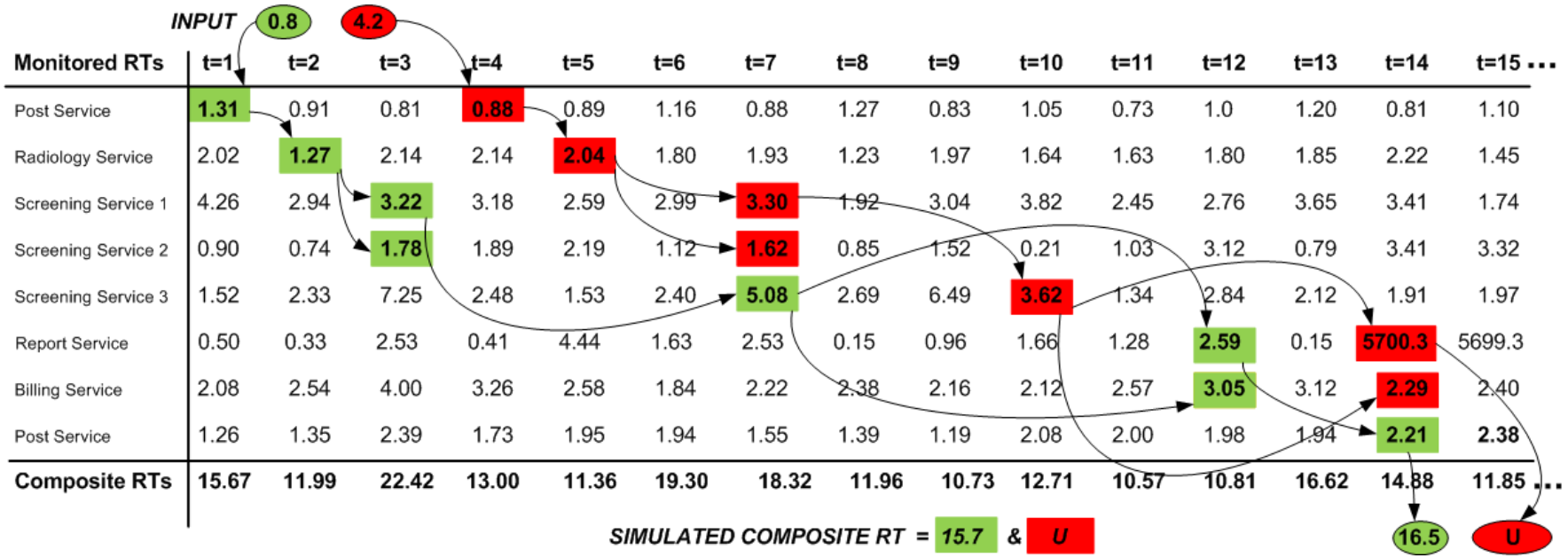


Figure 3.5: Example simulated RT calculation for the e-health workflow. The response times that are highlighted in green are the response times corresponding to a composition that would have been executed successfully. The response times that are highlighted in red are the response times corresponding to a composition that would not have been executed successfully.

Table 3.1: Monitored QoS values and worst case composite QoS calculations for the 8 services used in the mammography workflow. For each services there are two rows: the first row corresponds to the composition that would have been executed at time 0.8 and the second row corresponds to the composition that would not have been executed at time 4.2. The maximal allowed delay before the service is considered unavailable is denoted by q. The time period at which the service and the workflow is executed is denoted by t and s respectively.

Monitored QoS	***q***	**t**	**RT**	**C**	**A**	**RL**
Post Service	2.10	1	1.31	6.72	1	1
		4	0.88	5.77	1	1
Radiology Service	6.14	2	1.27	203.30	1	1
		5	2.04	235.33	1	1
Screening Service 1	6.32	3	3.22	110.37	1	1
		7	3.30	110.89	1	1
Screening Service 2	20.15	3	1.78	103.17	1	1
		7	1.62	102.62	1	1
Screening Service 3	15.95	7	5.08	125.81	1	1
		10	3.62	113.10	1	1
Report Service	49.35	12	2.59	6.71	1	1
		14	5700.3	8.57	0	0
Billing Service	4.82	12	3.05	9.30	1	1
		14	2.29	5.24	1	0
Post Service	5.15	14	2.21	9.88	1	1
		-	-	-	-	-
Composite QoS	$s = 0.8$		15.67	575.26	1	1
	$s = 4.2$		63.6	572.95	0	0

match, meaning that a third opinion is not necessary. This implies that the upper path of the conditional execution in Figure 3.3 rarely will be chosen. Nevertheless, the safest strategy is to calculate the worst case scenario. This means we take into account the path that generates the maximum response time of all the paths that could be implied by the *if condition*. Worst case calculations of the response time of the composite service using input times (s_{start}) of 0.8 and 4.2 are

$$\begin{aligned} RT_{0.8} = \; & 0.8 + 1.31 + 1.27 + \max(3.22; 1.78) + \max(5.08; 0) \\ & + \max(2.59 + 2.21; 3.05) - 0.8 = 15.7 \end{aligned}$$

$$\begin{aligned} RT_{4.2} = \; & 4.2 + 0.88 + 2.04 + \max(3.30; 1.62) + \max(3.62; 0) \\ & + \max(5700.3 + U; 2.29) - 4.2 = U. \end{aligned}$$

Remark that for the second calculation, the report service is unavailable and takes 5700.3 s to recover from failure. For services further in the execution chain, QoS values cannot be retrieved because no data is available that far in the future. An unavailable value is marked as 'Unknown (U)' and has the following properties: $0 \times U = 0; 1 \times U = U; \max(U, x) = U; \min(U, x) = U; U + x = U$. In our extended calculation below, we tackle this problem by taking into account the other QoS attributes and their interrelations.

To simulate the composite QoS values for cost, availability and reliability, we need their monitored values for all participating services at the times they are actually executed. Table 3.1 shows some fictive QoS values we use to illustrate our calculations. When a service on the path of execution is not available, the aggregated availability and reliability will become 0 and the costs and response times of the remaining services will not be taken into account due to its multiplication with the aggregated availability in the PNET-system. Worst case calculation for input times of 0.8 and 4.2 are

$$\begin{aligned}
RT_{0.8} &= 0.8 + \min(1.31, 2.10) \times 1 + \min(1.27, 6.14) \times 1 \\
&\quad + \max(\min(3.22, 6.32) \times 1; \min(1.78, 20.15) \times 1) \\
&\quad + \max(\min(5.08, 15.95) \times 1; 0) \\
&\quad + \max(\min(2.59, 49.35) \times 1 + \min(2.21, 5.18) \times 1; \min(3.05, 4.82) \times 1) \\
&\quad - 0.8 = 15.7, \\
A_{0.8} &= 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 = 1, \\
C_{0.8} &= 6.72 \times 1 + 203.30 \times 1 + 110.37 \times 1 + 103.17 \times 1 + 125.81 \times 1 \\
&\quad + 6.71 \times 1 + 9.30 \times 1 + 9.88 \times 1 = 575.26, \\
RL_{0.8} &= 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 1 = 1,
\end{aligned}$$

$$\begin{aligned}
RT_{4.2} &= 4.2 + \min(0.88, 2.10) \times 1 + \min(2.04, 6.14) \times 1 \\
&\quad + \max(\min(3.30, 6.32) \times 1; \min(1.62, 20.15) \times 1) \\
&\quad + \max(\min(3.62, 15.95) \times 1; 0) \\
&\quad + \max(\min(5700.3, 49.35) \times 1 + \min(U, 5.18) \times 0; \min(2.29, 4.82) \times 1) \\
&\quad - 4.2 = 63.6,
\end{aligned}$$

$$A_{4.2} = 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 0 \times 1 \times 1 = 0,$$

$$C_{4.2} = 5.77 \times 1 + 235.33 \times 1 + 110.89 \times 1 + 102.62 \times 1 + 113.10 \times 1 + 8.57 \times 0 + 5.24 \times 1 + U \times 0 = 572.95,$$

$$RL_{4.2} = 1 \times 1 \times 1 \times 1 \times 1 \times 1 \times 0 \times 0 \times U = 0.$$

Using this technique to calculate a time series for the composite service, we can apply the same prediction algorithm to estimate the QoS for elementary and composite services.

3.5.6 Discussion

Related works on QoS prediction typically make predictions for the individual services [1, 7, 8] and combine these predictions using QoS aggregation strategies such as Monte Carlo simulation [8] or probabilistic composition rules [5]. In our approach, we simulate a complete time series for the composite service, as it was executed several times in the past, and apply prediction on the resulting composite time series. Existing aggregation techniques have the following disadvantages compared to our simulation approach:

- Quantile values of QoS attributes of the individual service are not sufficient to calculate a realistic global quantile value for the composite service. One needs to predict the probability density for the QoS attributes of each individual service.
- The quantile values depend not only on the probability density of the QoS values of the individual services but on the dependencies between these QoS values as well. These mutual dependencies need to be modelled explicitly. Our strategy implicitly takes them into account as discussed below.
- QoS of individual services are often modelled as distributions that do not preserve information on how the QoS evolves in time. We believe that these time dependencies are valuable to make accurate estimations of future QoS values by means of time series prediction.

We elaborate on how mutual and time dependencies are implicitly taken into account by our aggregation approach. The example in Section 3.2.4 illustrates that aggregation causes distorted results when mutual dependencies (due to time dependencies) aren't taken into account. Our approach, however, gives the correct response times for the composite service (CS) equivalent to those shown in Table 3.3. The starting time of the post service (PS) is aggregated by the

PNET-system after the execution of the screening service (SS) to generate the composite time series and thus the weekend will never be counted twice. In case of mutual dependencies due to resource dependencies we get similar results. The result of our simulation approach is a time series for the composite QoS. The time information of this series can be exploited, as illustrated in Section 3.2.3, to make more accurate predictions for the future composite QoS. How this time series prediction can be done will be discussed in the next section.

3.6 QoS Prediction

The goal we want to achieve with our estimation algorithm is to predict whether the chance that a certain SLO will be violated is larger than a predefined value. First we define in Section 3.6.1 the problem we try to solve by discussing two performance indicators we want to minimize out-of-sample. Then we propose a kernel-based batch learning algorithm designed to minimize the first performance indicator and an online algorithm designed to minimize the second performance indicator in Section 3.6.2 and Section 3.6.3, respectively. Finally Section 3.6.4 combines both of these algorithms such that a good performance is achieved on both indicators.

3.6.1 Performance Indicators

At time i, we want to check whether a service has a chance of at least τ to have a response time smaller than the maximal response time $f_{i,max}$. This can be achieved by first obtaining the confidence interval $[0, f_\tau(x_i)]$ such that the chance the response time belongs to this interval equals τ. The service is then selected if $f_\tau(x_i) \leq f_{i,max}$. The true conditional quantile value is denoted by $\mu_\tau(x_i)$ and satisfies

$$P(y_i < \mu_\tau(x_i)) \leq \tau$$

$$P(y_i \leq \mu_\tau(x_i)) \geq \tau. \tag{3.3}$$

For simplicity reasons we assume that the quantile value is unique (which is in theory not necessary always true).

Performance Indicator I

To be able compare different quantile estimators we have to quantify how good the estimated quantile value approximates the true quantile value. For a location estimator the mean value minimizes the square error and the median value

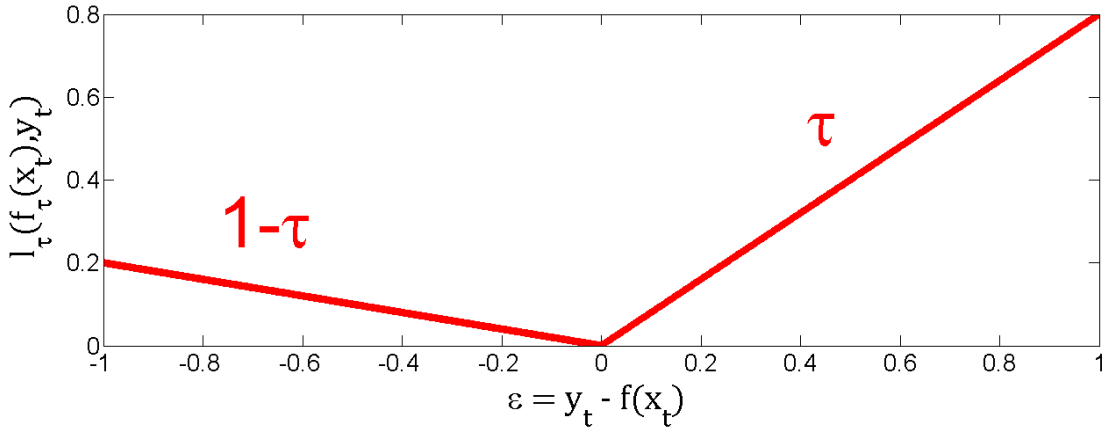


Figure 3.1: Pinball loss function used for quantile estimation. On the figure τ equals 0.8.

minimizes the absolute value of the error. Similarly it can be shown the quantile value minimizes the following pinball loss function [28] (Figure 3.1):

$$l_\tau(y_i - f_\tau(x_i)) = \begin{cases} \tau(y_i - f_\tau(x_i)), & y_i - f_\tau(x_i) \geq 0 \\ (\tau - 1)(y_i - f_\tau(x_i)), & y_i - f_\tau(x_i) < 0. \end{cases} \tag{3.4}$$

This loss is however not unique in the sense that there exists other loss-functions that have an optimum in the quantile value as well, such as

$$l_{\tau,\log} = l_\tau(\log(y_i) - \log(f_\tau(x_i))). \tag{3.5}$$

It is moreover possible that one estimator performs better according to one of these loss functions (e.g. l_τ) and another estimator performs better according to another one (e.g. $l_{\tau,\log}$). That's why, in this section, we further specify the problem of estimating the quantile value to a problem of minimizing a well chosen cost. Two types of error can occur in our applications:

- Type I error: a (composite) service is rejected to execute a customer's request in which the actual response time is smaller than the maximal response time ($y_i < f_{i,max}$).
- Type II error: a (composite) service is accepted to execute a customer's request in which the actual response time is larger than the maximal response time ($y_i > f_{i,max}$).

For the first performance indicator we assign costs to both errors: the cost of a type I error equals $1 - \tau$ and the cost of a type II error equals τ. In Theorem 28 and Corollary 1 we will show that the expected cost can be minimized if one knows the true conditional quantile value.

Theorem 28. *If the cost of a type I error equals $1-\tau$ and the cost of a type II error equals τ, then the expected conditional cost for accepting is lower than the expected conditional cost for rejecting if and only if $f_{i,max} > \mu_\tau(x_i)$.*

Proof. The expected conditional cost for accepting a service equals

$$E(\text{cost}|\text{accept}, x_i) = P(y_i > f_{i,max}|x_i)\tau, \tag{3.6}$$

where $P(y_i > f_{i,max}|x_i)$ equals the probability that $y_i > f_{i,max}$, and the expected conditional cost for rejecting a service equals

$$E(\text{cost}|\text{reject}, x_i) = P(y_i < f_{i,max}|x_i)(1-\tau). \tag{3.7}$$

Accepting is better than rejecting if and only if

$$\begin{aligned} & E(\text{cost}|\text{accept}, x_i) - E(\text{cost}|\text{reject}, x_i) < 0 \\ \Leftrightarrow\ & P(y_i > f_{i,max}|x_i)\tau - P(y_i < f_{i,max}|x_i)(1-\tau) < 0 \\ \Leftrightarrow\ & P(y_i > f_{i,max}|x_i) < 1-\tau \\ \Leftrightarrow\ & f_{i,max} > \mu_\tau(x_i). \end{aligned} \tag{3.8}$$

■

Corollary 1. *If $f_\tau(x_i)$ equals the true conditional quantile value $\mu_\tau(x_i)$ and a service is accepted if and only if $f_\tau(x_i) < f_{i,max}$, then the expected conditional cost is minimized for all possible values of $f_{i,max}$.*

A type I error occurs when $y_i < f_{i,max} < f_\tau(x_i)$ and a type II error occurs when $f_\tau(x_i) < f_{i,max} < y_i$. Given a test set only containing the true response times y_t, we cannot calculate the cost because we have no values for $f_{i,max}$. We need extra assumptions to handle the uncertainty over $f_{i,max}$. The first performance indicator (PI_1) will be defined as the cumulative expected cost, given these assumptions.

Theorem 29. *If we receive one service request per time step, if the probability on a certain f_{max} is time independent and uniformly distributed over the interval $[f^-, f^+]$ with f^-, f^+ finite and if all y_t, $f_\tau(x_t)$ belong to this interval, then the expected cost, given y_t and $f_\tau(x_t)$, becomes proportional to the pinball loss (l_τ)*

$$E(cost|y_t, f_\tau(x_t)) \sim l_\tau(y_t - f_\tau(x_t)). \tag{3.9}$$

Proof. When $f_\tau(x_t) > y_t$ a type I error (with cost $1-\tau$) can occur and when $y_t < f_\tau(x_t)$ a type II error (with cost τ) can occur. Because f_{max} is uniformly distributed, the probability on these errors becomes proportional to the distance between $f_\tau(x_t)$ and y_t and the expected cost becomes proportional to the pinball loss. ■

Corollary 2. *Given the same assumptions as in Theorem 29, the first performance indicator becomes*

$$PI_1(\mathcal{T}, f_\tau) = \sum_{t=1}^{n_{test}} l_\tau(y_{t,test} - f_\tau(x_{t,test})). \tag{3.10}$$

A disadvantage of using the pinball loss as performance measure is that the loss of one datapoint is not bounded (it can become infinite), despite the fact that a datapoint can only cause one error of type I or II. This is caused by the assumption that all y_t and $f_\tau(x_t)$ belong to the interval $[f^-, f^+]$ and the interval $[f^-, f^+]$ is thus not a priori determined and can become arbitrary large.

Theorem 30. *If we receive one service request per time step and if the probability on a certain f_{max}, denoted by $P(f_{max})$, is time independent and a priori determined, then the first performance indicator (PI_1) equals the following cumulative expected cost*

$$PI_1(\mathcal{T}, f_\tau, F) = \sum_{t=1}^{n_{test}} l_\tau \left(F(y_{t,test}) - F(f_\tau(x_{t,test}))\right) \tag{3.11}$$

where F is the cumulative distribution function of P

$$F(y) = \int_0^y P(f_{t,max}) df_{t,max}. \tag{3.12}$$

Proof. If $y_{t,test} \geq f_\tau(x_{t,test})$, then an error of type II can occur with probability $F(y_{t,test}) - F(f_\tau(x_{t,test}))$. The expected cost becomes $\tau(F(y_{t,test}) - F(f_\tau(x_{t,test})))$. If $y_{t,test} < f_\tau(x_{t,test})$, then an error of type I can occur with probability $F(f_\tau(x_{t,test})) - F(y_{t,test})$. The expected cost becomes $(1-\tau)(F(f_\tau(x_{t,test})) - F(y_{t,test}))$. ■

The influence of one data point on PI_1 is bounded because

$$l_\tau \left(F(y_{t,test}) - F(f_\tau(x_{t,test}))\right) \leq \max(\tau, 1-\tau). \tag{3.13}$$

Different probability distributions for f_{max} cause different costs and in practice the performance indicator should equal the cumulative expected cost in which the chosen probability distribution of f_{max} matches the true probability distribution as good as possible. In our experiments we will use the performance indicator as defined in corollary 2 in which the influence of one datapoint is unbounded because we have no data that allows us to make a reasonable estimate of $P(f_{max})$.

Performance Indicator II

In practice costs are not always linear in the number of errors. It is possible that causing more violations than agreed invokes huge fines while causing fewer violations than agreed does not cause equally large profits. For that reason it is important that the number of violations does not exceed the agreed number of violations too much. Constraining the frequency of violations not to exceed $1-\tau$ is in our opinion too severe because the test set contains only a sample of the total population and it is possible the frequency of violations of this sample exceeds $1-\tau$ despite the frequency of violations of the entire population to be smaller or equal to $1-\tau$. That's why we choose performance indicator II to express the probability that the frequency of violations of the test set is larger than or equal to the actual number of violations in the test set, given the frequency of violations of the entire population equals $1-\tau$.

Theorem 31. *Assuming the expected number of violations equals* $1-\tau$ *and assuming the data is independent and identically distributed (i.i.d.), the probability of observing at least* $n_{v,test}$ *violations on a test set where* $n_{req,test}$ *service requests are accepted becomes*

$$P(n_v \geq n_{v,test}|\tau, n_{req,test}) = \sum_{i=n_{v,test}}^{n_{req,test}} (1-\tau)^i \tau^{(n_{req,test}-i)} \binom{n_{req,test}}{i} \quad (3.14)$$

where $\binom{n}{i}$ *is a binomial coefficient.*

Proof. The discrete probability distribution of the number of violations n_v in a sequence of $n_{req,test}$ independent experiments, each of which yields a violation with probability $1-\tau$, equals a binomial distribution. The probability on $n_{v,test}$ or more violations given a binomial distribution is expressed in equation (3.14). ■

To be able to measure the number of violations, we need test data containing times on which service requests are send together with the corresponding maximal response times. In case no such data is available, we have to make extra assumptions.

Corollary 3. *Under the extra assumptions of one service request per time step and* 100% *acceptance with* $f_{i,max}$ *equal to* $f_\tau(x_i)$ *(the latter assumption implies the worst case value of* $f_{i,max}$ *such that the service is accepted), the probability of observing at least* $n_{v,test}$ *violations is denoted by* PI_2 *and equals*

$$PI_2(\mathcal{T}, f_\tau) = \sum_{i=n_{v,test}}^{n_{test}} (1-\tau)^i \tau^{(n_{test}-i)} \binom{n_{test}}{i} \quad (3.15)$$

where $\binom{n}{i}$ *is a binomial coefficient.*

The true quantile value of an estimator is defined as

$$\tau_f(f_\tau) = P(y_{t,test} < f_\tau(x_{t,test})). \tag{3.16}$$

The expected number of observed violations equals $1 - \tau_f(f_\tau)$. The smaller PI_2, the more unlikely the observed number of violations are caused by an estimator with true quantile value larger than or equal to τ. A high PI_2, however, does not imply having a conditional quantile estimator. An unconditional, constant function can perform very well on this performance indicator, but will, luckily, perform poorly on PI_1.

3.6.2 Kernel-Based Quantile Estimation

In this Section we will explain why the kernel-based quantile estimator discussed in [29] suits our problem. The first algorithm we propose is designed to minimize PI_1 and thus the following *expected risk*

$$R[f_\tau] = \int l_\tau \left(y - f_\tau(x)\right) dP(x, y). \tag{3.17}$$

The probability density function $P(x, y)$ is unknown. We have only access to training data. A naive way of minimizing the *expected risk* is minimizing the *empirical risk*

$$R_{emp}[f_\tau, S] = \frac{1}{n} \sum_{t=1}^{n_{tr}} l_\tau(y_{t,tr} - f_\tau(x_{t,tr})). \tag{3.18}$$

When the hypothesis space $\mathcal{H}$ to which f_τ belongs is very rich, this can lead to overfitting. For that reason we will minimize the *regularized risk*

$$R_{reg}[f_\tau, S] = R_{emp}[f_\tau, S] + \frac{\lambda}{2}\|w\|_2^2 = \frac{1}{n} \sum_{t=1}^{n_{tr}} l_\tau(y_{t,tr} - f_\tau(x_{t,tr})) + \frac{\lambda}{2}\|w\|_2^2 \tag{3.19}$$

where λ is a regularization parameter that must be positive. We assume f_τ can be written as

$$f_\tau(x) = w^T \varphi(x) + b \tag{3.20}$$

where w is the weight vector, φ maps the input space into a feature space (which can be infinite dimensional) and b is the constant offset. Minimizing the *regularized risk* can be written as the following optimization problem

$$\min_{w,b,\xi,\xi^*} \sum_{t=1}^{n_{tr}} \tau\xi_t + (1-\tau)\xi_t^* + \frac{1}{2}\lambda\|w\|_2^2 \tag{3.21}$$

$$\text{subject to } \begin{cases} y_{t,tr} - w^T\varphi(x_{t,tr}) - b \leq \xi_t, & t = 1, ..., n_{tr} \\ w^T\varphi(x_{t,tr}) + b - y_{t,tr} \leq \xi_t^*, & t = 1, ..., n_{tr} \\ \xi_t, \xi_t^* \geq 0, & t = 1, ..., n_{tr}. \end{cases} \tag{3.22}$$

Because $\varphi(x)$ can be huge or even infinite dimensional, directly solving this optimization problem can become very expensive. The dual problem, after applying the kernel trick, becomes

$$\min_{\alpha} \frac{1}{2}\lambda\alpha^T\Omega\alpha - \alpha^T y \text{ subject to } \begin{cases} \frac{\tau}{\lambda} \geq \alpha_t \geq \frac{\tau-1}{\lambda}, & t = 1, ..., n_{tr} \\ \sum_{t=1}^{n} \alpha_t = 0 \end{cases} \tag{3.23}$$

where $k(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$, $\Omega \in \mathbb{R}^{n_{tr} \times n_{tr}}$ is the kernel matrix defined as $\Omega_{i,j} = k(x_{i,tr}, x_{j,tr})$, $\alpha = [\alpha_{1,tr}, ..., \alpha_{n_{tr},tr}]^T$ and $y = [y_{1,tr}, ..., y_{n_{tr},tr}]^T$. This optimization problem is a quadratic programming (QP) problem. The kernel trick ($k(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$) makes explicit calculation of the mapping $\varphi(x)$ into a possible infinite dimensional feature space no longer necessary. The function f_τ can now be expressed as

$$f_\tau(x) = \sum_{t=1}^{n_{tr}} \alpha_t k(x_{t,tr}, x) + b. \tag{3.24}$$

The constant offset b can be found using the fact the following holds $f_\tau(x_{t,tr}) = y_{t,tr}$ for α_t belonging to the open interval $]\frac{\tau-1}{\lambda}, \frac{\tau}{\lambda}[$. The pseudo-code for the algorithm explained in this section is shown in Algorithm 1.

Algorithm 1 Given the training set $\mathcal{S}$, the targeted quantile value τ, a kernel k and the regularization parameter λ, find α and b which are parameters of the predicting function f_τ.

for $t_1 = 1 \rightarrow n_{tr}$ **do**
  for $t_2 = 1 \rightarrow n_{tr}$ **do**
    $\Omega_{t_1,t_2} \leftarrow k(x_{t_1,tr}, x_{t_2,tr})$
  end for
end for
$\hat{\alpha} \leftarrow \arg\min_\alpha \frac{1}{2}\lambda\alpha^T\Omega\alpha - \alpha^T y_{tr}$
  subject to $\begin{cases} \frac{\tau}{\lambda} \geq \alpha_t \geq \frac{\tau-1}{\lambda}, & t = 1, ..., n_{tr} \\ \sum_{t=1}^{n} \alpha_t = 0 \end{cases}$
for $t = 1 \rightarrow n_{tr}$ **do**
  if $\hat{\alpha}_t \in \ ]\frac{\tau-1}{\lambda}, \frac{\tau}{\lambda}[$ **then**
    $b \leftarrow y_{t,tr} - \sum_{t_1=1}^{n_{tr}} \hat{\alpha}_{t_1} k(x_{t_1,tr}, x_{t,test})$
  end if
end for

3.6.3 Online Adaptation of an Unconditional Quantile Estimator

The second algorithm we propose is designed to minimize PI_2. It is a simple online algorithm that is updated as follows

$$f_{\tau,t} = \begin{cases} f_{\tau,start}, & t = 1 \\ f_{\tau,t-1} + \eta\tau, & t > 1, y_{t-1} > f_{\tau,t-1} \\ f_{\tau,t-1} + \eta\left(\tau - 1\right), & t > 1, y_{t-1} < f_{\tau,t-1} \\ f_{\tau,t-1}, & t > 1, y_{t-1} = f_{\tau,t-1}. \end{cases} \tag{3.25}$$

where η is the learning rate which must be strictly positive. The start value $f_{\tau,start}$ can be set equal to 0, equal to the first data point, or equal to the quantile value of a number of datapoints. Datapoints used to determine $f_{\tau,start}$ should not be used for the online updating. The number of times y_{t_1} exceeds f_{τ,t_1} for t_1 going from 1 to t is denoted by v_t. We can now express $f_{\tau,t+1}$ as

$$f_{\tau,t+1} = f_{\tau,1} + \eta\left(\tau v_t + (\tau - 1)(t - v_t)\right) = f_{\tau,1} + \eta t\left(\frac{v_t}{t} - (1 - \tau)\right). \tag{3.26}$$

Conversely, v_t can be expressed in function of $f_{\tau,t+1}$ as follows

$$v_t = \frac{f_{\tau,t+1} - f_{\tau,1}}{\eta} + (1 - \tau)t. \tag{3.27}$$

Theorem 32. *Assuming there exists an interval $[y_{min}, y_{max}]$ with y_{min}, y_{max} finite such that $f_{\tau,start}$ and all y_t belong to this interval, the frequency of violations converges to $1 - \tau$ for t going to ∞*

$$\lim_{t\to\infty} \frac{v_t}{t} = 1 - \tau. \tag{3.28}$$

Proof. Because all y_t belong to the interval $[y_{min}, y_{max}]$, all $f_{\tau,t}$ belong to the interval $[y_{min} + \eta\left(\tau - 1\right), y_{max} + \eta\tau]$ and

$$\begin{aligned} &|f_{\tau,t+1} - f_{\tau,1}| \leq C \\ \Rightarrow &\eta t|\frac{v_t}{t} - (1 - \tau)| \leq C \\ \Rightarrow &|\frac{v_t}{t} - (1 - \tau)| \leq \frac{C}{\eta t} \end{aligned} \tag{3.29}$$

where $C = y_{max} - y_{min} + \eta\max(\tau, 1 - \tau)$. Expression 3.29 shows that the distance between $\frac{v_t}{t}$ and $(1 - \tau)$ converges to zero for t going to ∞. ■

Theorem 33. *Assuming that all y_t belong to the interval $[y_{min}, y_{max}]$ with y_{min}, y_{max} finite, the performance indicator PI_2 of the test set converges to 0.5 for $t \to \infty$.*

Proof. Suppose that the random variable X_t follows a binomial distribution which yields a violation with probability $1 - \tau$ in a sequence containing t datapoints, then PI_2 expresses the probability the number of violations and thus X_t is at least v_t. The mean and the standard deviation of X_t equal

$$\mu_{X,t} = t(1 - \tau) \tag{3.30}$$

$$\sigma_{X,t} = \sqrt{t\tau(1 - \tau)}. \tag{3.31}$$

From equation (3.29) we can now conclude

$$\frac{|v_t - \mu_{X,t}|}{\sigma_{X,t}} \leq \frac{C}{\eta\sqrt{t\tau(1 - \tau)}}, \tag{3.32}$$

which implies

$$\lim_{t \to \infty} \frac{v_t - \mu_{X,t}}{\sigma_{X,t}} = 0. \tag{3.33}$$

According to the central limit theorem, the cumulative distribution function (CDF) of X_t converges pointwise to the CDF of a normal distribution with mean $\mu_{X,t}$ and standard deviation $\sigma_{X,t}$ for n going to ∞. Because, as n approaches ∞, v_t is distanced 0 standard deviations from $\mu_{X,t}$, the CDF of X_t in v_t equals 0.5 and PI_2 becomes 0.5. ■

The pseudo-code for the algorithm explained in this section is shown in Algorithm 2.

Algorithm 2 Given x_{test}, y_{test}, η and $f_{\tau,start}$, calculate $f_{\tau,t}(x_{t,test})$ and update $f_{\tau,t}$ for $t = 1, ..., n_{test}$.

```
f_{τ,1} ← f_{τ,start}
for t = 1 → n_test do
  if y_{t,test} > f_{τ,t} then
    f_{τ,t+1} ← f_{τ,t} + τη
  else if y_{t,test} < f_{τ,t} then
    f_{τ,t+1} ← f_{τ,t} + (τ − 1)η
  else
    f_{τ,t+1} ← f_{τ,t}
  end if
end for
```

3.6.4 Kernel-Based Quantile Estimation with Online Adaptation of the Constant Offset

In this section we will present the final algorithm that combines the algorithms explained in the previous two Sections. Additionally we define the input vectors

and explain how the data is preprocessed. The final algorithm consists of three stages.

- Preprocessing phase: in this phase the inputs and corresponding outputs are generated from the response times RT_t. We will predict the logarithm of response times instead of the response times themselves because this avoids our algorithm to make predictions below zero. The input vectors can be past response times (autoregressive), the quantile values of sets of past response times or the time at which the service starts. The best choice for the input vectors depends on the specific dataset one wants to make predictions on.
- Training phase: secondly we optimize α and b on the training set using Algorithm 1 to predict the logarithm of the response time.
- Online updating phase: finally we try to predict $\log(y_t) - f_\tau(x_t)$ with a function denoted by $d_{\tau,t}$ using Algorithm 2. The optimal $d_{\tau,start}$ can be determined using a validation set or can be set to 0. We add d_t to $f_\tau(x_t)$ to make the final prediction. The latter is equivalent to updating the constant offset b_t such that b_t equals $b + d_t$.

This final algorithm performs well on both PI_1 and PI_2. To further clarify how this algorithm works, we will illustrate it using the toy example shown in Table 3.1. First the estimator f is learned using the training data and d_5 is set to zero. At time 5 the predicted value $g_5(5) = 5.0$ is smaller than the true value $y_t = 5.1$ and therefore the online adaptation parameter is increased as follows $d_6 = d_5 + \tau \cdot \eta = 0 + 0.75 \cdot 0.1$. At time 6 the predicted value equal to 5.975 is still smaller than the true value equal to 6.2 and thus $d_7 = d_6 + \tau \cdot \eta$. At time 7 the predicted value is larger than the true value and $d_8 = d_7 + (\tau - 1) \cdot \eta$.

3.7 Experimental Evaluation

We limit the experimental evaluation to predicting the response time. In a first scenario we predict the response time of an individual service, in a second scenario we predict the response time of an entire workflow.

3.7.1 Prediction For Individual Services

The kernel-based quantile estimator (KQ) explained in Section 3.6.2 and the kernel-based quantile estimator with online adaptation of the constant offset (KQOA) explained in Section 3.6.4 will be evaluated using a real-life data set discussed in Appendix 3.B: the Movie Dataset. We also compare the

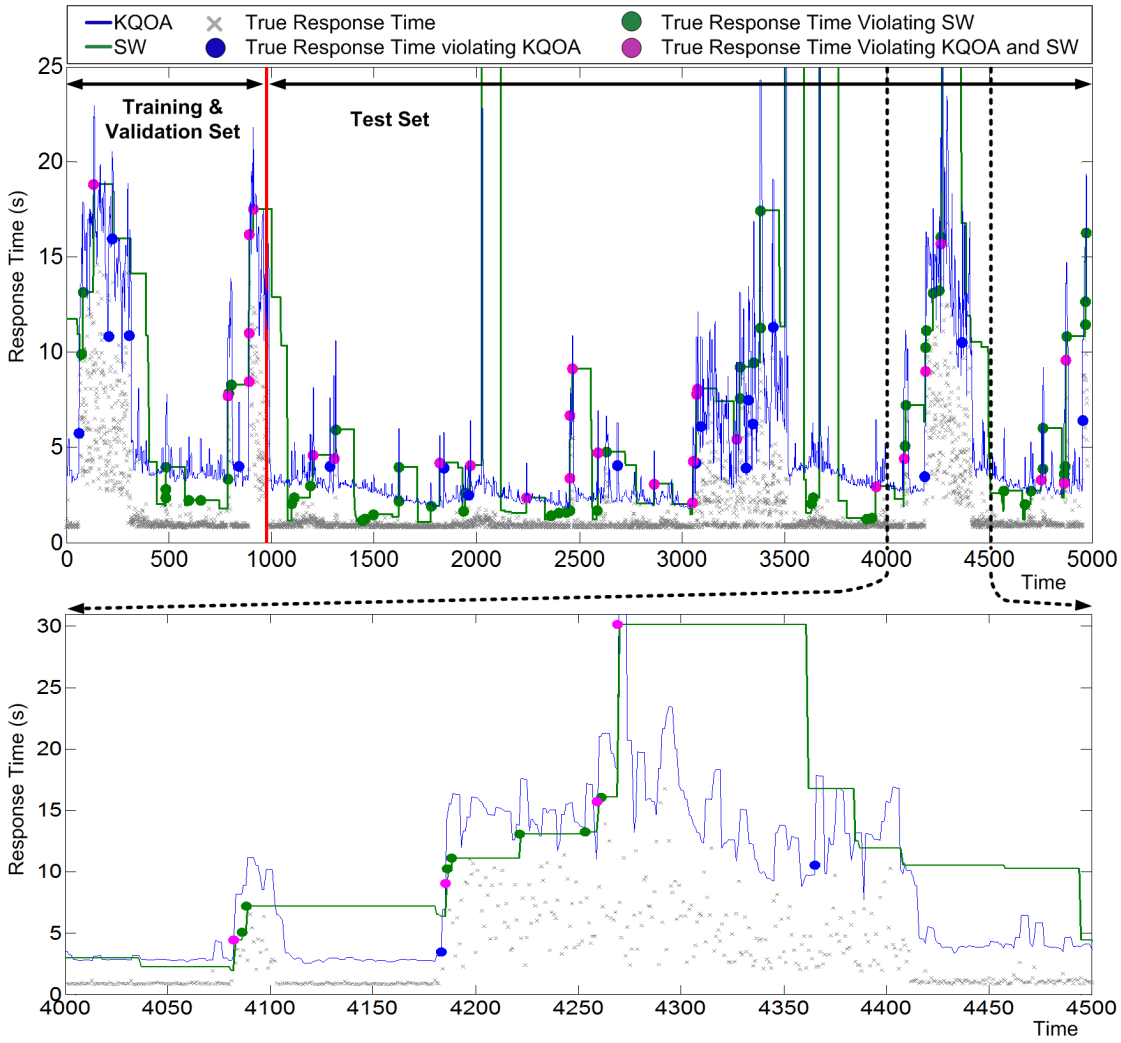


Figure 3.2: Quantile estimation for the Movie Dataset using the kernel-based quantile estimator with online adaptation of the constant offset (KQOA) and the sliding window estimator (SW). The upper figure shows the results for data point 0 to 5000. The lower figure zooms in on data point 4000 to 4500.

Table 3.1: Toy dataset used to illustrate the kernel-based quantile estimator with online adapation of the constant offset. The time, input vector and output vector are denoted respectively as t, x_t and y_t. The kernel-based quantile estimator without and with online adaptation are denoted respectively as f and g_t. The function g_t equals f plus the online adaptation function d_t. The quantile parameter τ equals 0.75. For simplicity we use fixed hyperparameters: the regularization parameter λ equals 0, the kernel is linear and the learning rate η equals 0.1. The first four datapoints are training data and the last four data points are test data. The initial value of the online adaptation parameter (d_5) is set to 0.

	Training Data				**Test Data**			
t	1	2	3	4	5	6	7	8
$x_t(=t)$	1	2	3	4	5	6	7	8
y_t	0.9	2.3	2.9	4.1	5.1	6.2	6.9	7.8
$f(x_t)$	1.4	2.3	3.2	4.1	5.0	5.9	6.8	7.7
d_t	\	\	\	\	0	0.075	0.15	0.125
$g_t(x_t)$	\	\	\	\	5.0	5.975	6.95	7.825

performance of our algorithms to a sliding window approach, which is used in related work [8].

The kernel we use is a RBF-kernel. The input vectors are quantile values of sets of past response time as shown in Algorithm 3. KQ has three hyperparameters: the regularization parameter λ, the kernel parameter σ and the lag L. KQOA has an extra parameter: the online learning rate η. The training set consists of the first 700 datapoints and will be used to find the optimal α and b, given

Algorithm 3 Given the true response times RT_t for $t = 1, ..., n + L$ generate the input vectors x_t and the output values y_t for $t = 1, ..., n$. RT_t is the true response time at time t, x_t is the input vector at time $t + L$ and y_t is the output vector at time $t + L$. The function $quantile\,(\tau, \mathcal{X})$ expresses the τ-quantile value of the set $\mathcal{X}$.

for $t = 1 \rightarrow n$ **do**

$q_{1,t} \leftarrow quantile\left(\tau, \{RT_i\}_{i=t}^{t+\frac{L}{4}-1}\right)$

$q_{2,t} \leftarrow quantile\left(\tau, \{RT_i\}_{i=t+\frac{L}{4}}^{t+\frac{L}{2}-1}\right)$

$q_{3,t} \leftarrow quantile\left(\tau, \{RT_i\}_{i=t+\frac{L}{2}}^{t+\frac{3L}{4}-1}\right)$

$q_{4,t} \leftarrow quantile\left(\tau, \{RT_i\}_{i=t+\frac{3L}{4}}^{t+L-1}\right)$

$x_t \leftarrow [\log(q_{1,t}), \log(q_{2,t}), \log(q_{3,t}), \log(q_{4,t})]^T$

$y_t \leftarrow \log(RT_{t+L})$

end for

the hyperparameters. The validation set consists of the next 300 datapoints and will be used to optimize the hyperparameters λ, σ and L. The learning rate η of the second algorithm allows adaptation to unexpected events and should not be chosen too small. Unexpected behaviour that might happen in the test set might not happen in the validation set. We have chosen η as big as possible such that the performance according to performance indicator I on the validation set does not increase with more than 2%.

When the optimal hyperparameters are found, the algorithm is retrained on both the training and the validation set. The performance will then be evaluated using the test set which contains the remaining 4000 datapoints. We compare our algorithms to a sliding window quantile estimator with windowsize w. The latter estimator uses the τ-quantile value of the lasts w datapoints as τ-quantile estimation for the next datapoint. The parameter w is optimized using the first 1000 datapoints as training set and evaluated using the next 4000 datapoints as test set. The KQOA and SW quantile estimations for the Movie Dataset are shown in Figure 3.2. The lower figure zooms in on the area between data point 4000 and 4500. The performances of the estimators for a quantile value of 99% are summarized in Table 3.1(a) . According to PI_1 the expected cost using the kernel-based algorithms is approximately 30% lower compared to the sliding window algorithm. According to PI_2 the probability an estimator with true quantile value 99% causes equal or more violations than the estimator is quite big for the kernel-based algorithms and almost non-existing for the sliding window estimator.

We can conclude our algorithms performs significantly better than the sliding window estimator on both PI_1 and PI_2. Without online adaptation, however, we cannot guarantee that PI_2 will not converge to zero. With online adaptation, given certain assumptions, we can. To show the effect of the online adaptation, we increased the response time of 50 randomly selected datapoints of the test set by 15 seconds. This causes the KQ algorithm to perform bad because these spikes only appear in the test set and are thereby not learned during the training period. In Table 3.1(b), we can see that for this experiment only KQOA has a value for PI_2 that differs significantly from zero. The optimal hyperparameters for our algorithms are: $L = 24$, $\sigma = 500$, $\lambda = 0.005$ and $\eta = 0.1$. The optimal window size for the sliding window quantile estimator equals 92.

3.7.2 Prediction For Composite Services

For predicting the response time of composite services, we need to calculate the simulated response times according to the technique explained in Section 3.5. We use the e-health case study described in Section 3.2.1, where each elementary service has a response time according to simulated data for a long-running process. Rosario [8] validated the use of certain families of distributions and

Table 3.1: Comparison of the performance of the Kernel-based Quantile Estimator with Online Adaptation of Constant Offset (KQOA) and the Sliding Window Quantile Estimator (SWQ) on the Movie Dataset (a) and the Movie Dataset with an increase of 15s in response time of 50 randomly selected test set datapoints (b). PI_1 equals the cumulative pinball loss, $PI_2(\%)$ is the probability an estimator with true quantile value 99% causes equal or more violations and $f_{v,test}$ is the number of violations in the test set per 100 datapoints.

(a) Movie Dataset

	PI_1	$PI_2(\%)$	$f_{v,test}$
KQOA	0.0635	58.5	0.98
KQ	0.0637	99.6	0.63
SWQ	0.0974	0.0	1.85

(b) Adjusted Movie Dataset

	$PI_2(\%)$	$f_{v,test}$
KQOA	6.9	1.25
KQ	0.0	1.75
SWQ	0.0	2.95

performed their best fit on measured data real-life web services. They observed that the Gamma and the Log-Logistic distributions were a reasonably good fit for the response times.

To simulate data for the elementary services, we will assume the number of working hours needed to complete a task is modelled as log-logistic distributed random variables. The log-logistic distribution is similar in shape to the log-normal distribution but has heavier tails. The probability density function depends on α and β:

$$f(x;\alpha,\beta) = \frac{(\frac{\beta}{\alpha})(\frac{x}{\alpha})^{\beta-1}}{[1+(\frac{x}{\alpha})^{\beta}]^2}. \tag{3.34}$$

Some of the services in our case study require human interaction and thus we assume the completion time of a task is subject to working hours of the service. Depending on the service, a working day starts and ends at a specific hour and are open or closed on Saturdays and Sundays. An overview of our assumptions is shown in Table 3.2. Outside the range of working hours the progress of the task is freezed. For example, if a working day starts at 8h and ends at 16h, then a task taking 9 working hours to complete that starts at 10h on Monday will end at 11h on Tuesday. The response time of 25 hours consist of 6 working hours within the same day, 16 non-working hours and 3 working hours the next day. We assume monitoring is done every hour. The generated response times for all elementary services of the e-health workflow are shown in Figure 3.1. The graph at the bottom represents the resulting composite response time, calculated using the PNET-system described in Section 3.5.

We will compare the Kernel-Based Quantile estimator (KQ) explained in Section 3.6.2 and the Kernel-Based Quantile estimator with Online Adaptation of the constant offset (KQOA) explained in Section 3.6.4 to the probabilistic



(a) Post Service 1

(b) Radiology Service

(c) Screening Service 1

(d) Screening Service 2

(e) Screening Service 3

(f) Report Service

(g) Billing Service

(h) Post Service 2

(i) Composite Application: e-health workflow

Figure 3.1: Overview of response times of elementary service (a-h) and resulting simulated response time of composite service (i) for e-health case-study (horizontal axis: time (in hours) - vertical axis: response time (in hours))

Table 3.2: Parameter assumptions for elementary services of e-health case study

Services	Parameters				
	Start	**End**	**Weekend**	α	β
Post 1	8h	15h	Closed	log(10)	1/8
Radiology	9h	18h	Closed	log(14)	1/10
Screening 1	8h	20h	Saturday	log(12)	1/9
Screening 2	8h	22h	Closed	log(12)	1/7
Screening 3	8h	20h	Saturday	log(15)	1/7
Report	0h	24h	Open	log(2)	1/10
Billing	0h	24h	Open	log(5)	1/4
Post 2	8h	15h	Closed	log(10)	1/8

approach using Boostrap-Based Simulations explained in the paper of Rosario [8]. In this work, they use bootstrap based Monte Carlo (BBMC) simulations to estimate a composite QoS value by repeatedly drawing QoS values from the probability distributions of the constituting elementary services and composing them using soft contract rules. The training set consists of the first 4 weeks, the validation set (used to optimize hyperparameters) of the next 4 weeks and the test set consists of the last 8 weeks. After optimizing the hyperparameters the algorithm is retrained on both the validation and training set. BBMC uses past response time as input which has the problem that recent executions of individual services may not have ended yet. Their response time is thus unknown. We solve this by sliding the window backwards in time until all response times within and before the window are known. As shown in Algorithm 4, we take the time (in hours) within a week as input vector for KQ and KQOA, because we a priori know there is weekly seasonality. The KQOA and BBMC quantile estimations for the e-health application are shown in Figure 3.2. Table 3.3 shows KQ and KQOA perform better than BBMC and shows the number of violations for BBMC is very low.

The estimated quantile value of BBMC overestimates the true quantile value which can be explained as follows: the BBMC does not take into account the mutual dependencies due to time dependencies as we illustrated in Section 3.2.4. For some services the progress freezes during the weekends. The response time of these services on Friday evening will be very high. The Monte Carlo simulation combines random samples of the services response times, without considering if this combination can occur in practice. The worst case scenario using Monte Carlo thus becomes worse than the 'actual' worst case scenario which causes the estimated 99%-quantile value to be too high. The approach used in KQ and KQOA of first composing the response times as explained in Section 3.5 and then estimating the quantile values avoids this problem. Another reason for the

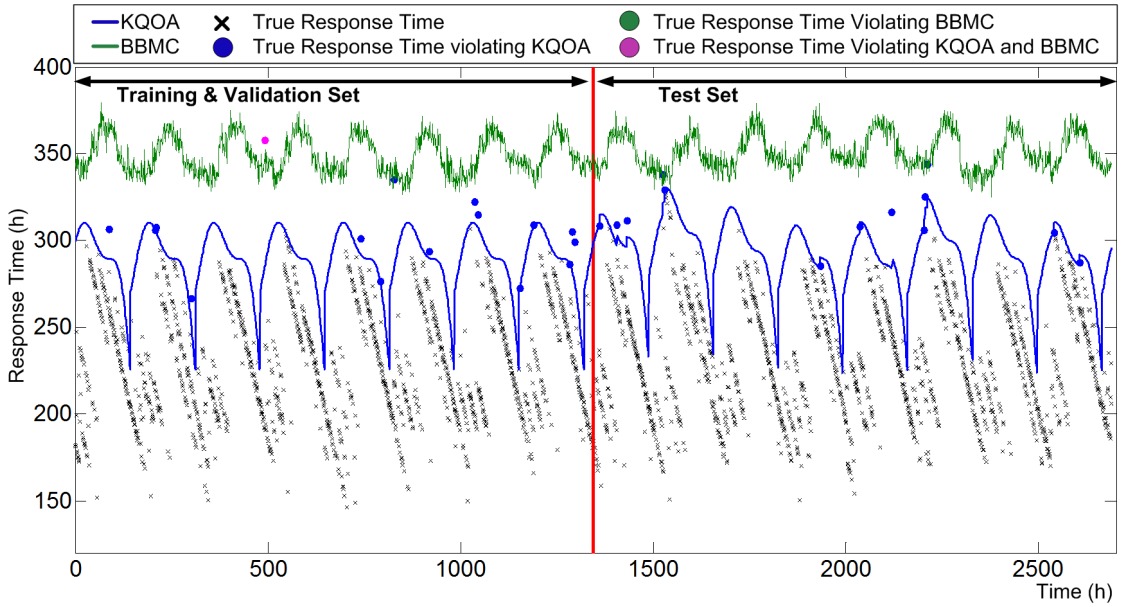


Figure 3.2: Quantile estimation for the e-health application using the kernel-based quantile estimator with online adaptation of the constant offset (KQOA) and the bootstrap based Monte Carlo estimator (BBMC).

Table 3.3: Comparison of the performance of the Kernel-based Quantile Estimator (KQ), Kernel-based Quantile estimator with Online Adaptation of Constant Offset (KQOA) and the probabilistic approach using Boostrap-Based Simulations (BBMC) on the E-Health Dataset. PI_1 equals the cumulative pinball loss, $\mathrm{PI}_2(\%)$ is the probability an estimator with true quantile value 99% causes equal or more violations and $f_{v,test}$ is the number of violations in the test set per 100 datapoints. The chosen hyperparameters for KQOA and/or KQ are: $\sigma = 50$, $\lambda = 0.01$ and $\eta = 0.02$. The chosen window-size for BBMC equals 300.

	PI_1	$\mathrm{PI}_2(\%)$	$f_{v,test}$
KQOA	0.795	69.1	0.89
KQ	0.857	3.3	1.56
BBMC	1.212	100.0	0.0

poor performance of BBMC is that it cannot learn the seasonality patterns.

Algorithm 4 Given the true response times RT_t for $t = 1, ..., n$ generate the input vectors x_t and the output values y_t for $t = 1, ..., n$.

for $t = 1 \rightarrow n$ **do**
  $x_t \leftarrow t \mod 24 \cdot 7$
  $y_t \leftarrow RT_t$
end for

3.8 Conclusion

In this paper we developed a two-step approach to accurately predict QoS quantiles for composite services. The first step, which is based on Petri nets, derives simulated QoS-values of a workflow composition from those of its constituent elementary services. In a second step we focus on the response time and try to predict future response times of composite services based on the simulated response times using a kernel-based quantile estimator with online adaptation of the constant offset. The latter algorithm has a batch and an online learning phase. During the batch learning phase it tries to minimize, in a regularized manner and given certain assumptions, the occurrence of two types of errors: a type I error occurs when a service is rejected in which the real response time is smaller than the agreed response time and a type II error occurs when a service is accepted in which the real response time exceeds the agreed response time. The online learning phase is necessary to assure that the probability an estimator with true quantile value equal to the agreed quantile value causes equal or more violations, converges to 50% for the number of test datapoints going to infinity. The latter implies the number of times the true response time exceeds the estimated response time converges to the agreed quantile value.

We evaluated our prediction algorithm on one elementary service, an automated service using real-life web service data, and on one composite service, a service with human interaction using simulated data. On the first dataset our algorithm caused a loss 30% lower than a sliding window estimator and on the second dataset our algorithm significantly outperformed the probabilistic approach using Boostrap-Based Simulations (BBMC) explained in the paper of Rosario et al. [8].

Despite the promising results our approach has some limitations:

- When making small changes to the composition, the kernel-based quantile estimator needs to be retrained completely. This makes it expensive to use this algorithm as a part of a composition selection procedure. It can however still be used to verify whether a chosen composition satisfies the SLO. Once the algorithm is trained, however, doing prediction on the

chosen composition for future time steps is relatively cheap because the training does not need to be redone.

- The prediction algorithms are most effective for processes with variable QoS that changes according to recognizable patterns. Typically this is the case for long-running processes that require human interaction. When the QoS attributes of candidate services are close to random, the use of prediction algorithms is not effective and will only slow down the composition process. The first step, where composite QoS-values of a workflow composition are derived from those of its elementary services, however, is suitable for all types of processes, and can easily be combined with other estimation techniques.

3.9 Future Work

Our future work includes a practical implementation of QoS-aware service composition using the algorithms presented in this paper. Currently we are evaluating our approach in the context of Cloud Computing. Software as a service (SaaS), sometimes referred to as “Software on Demand", is an emerging mechanism of releasing software applications to customers. From a technical perspective, an import issue in “software on demand" is the customization for the SaaS application to serve multiple tenants, each having their own service level requirements. Our goal is to have a practically efficient solution to address customization of business processes with a focus on run-time assurance of service quality and service level agreements. For the implementation we use the framework discussed in [30]. The framework extends the WS-BPEL language with policy-based reconfiguration capabilities. The framework is based on the Model-View-Controller (MVC) pattern and is implemented in Ruby On Rails (RoR). To achieve accurate estimates of the response times of the individual services and extend them to an estimate for the composite service, we will use the techniques described in this work. The logic to efficiently select appropriate services among available services will be implemented using the reconfiguration support of the framework. A challenge here is to implement an efficient selection algorithm that takes into account the stochastic behaviour of the QoS estimates of candidate services by integrating the QoS estimation techniques presented in this work.

3.10 Acknowledgements

Research supported by: IBBT, Research Council KUL: GOA/10/09 MaNet , PFV/10/002 (OPTEC), several PhD/postdoc & fellow grants; Flem-

ish Government: IOF: IOF/KP/SCORES4CHEM; FWO: PhD/postdoc grants, projects: G.0588.09 (Brain-machine), G.0377.09 (Mechatronics MPC), G.0377.12 (Structured systems); IWT: PhD Grants, projects: SBO LeCoPro, SBO Climaqs, SBO POM, EUROSTARS SMART: iMinds 2013; Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012-2017); EU: FP7-EMBOCON (ICT-248940), FP7-SADCO (MC ITN-264735), ERC ST HIGHWIND (259 166), ERC AdG A-DATADRIVE-B; COST: Action ICO806: IntelliCIS. The scientific responsibility is assumed by its authors.

References

[1] J. Cardoso, Quality of Service and Semantic Composition of Workflows, Ph.D. thesis, University of Georgia, 2002.

[2] G. Canfora, M. Di Penta, R. Esposito, M. L. Villani, An approach for qos-aware service composition based on genetic algorithms, in: 2005 Conference on Genetic and Evolutionary Computation, Proceedings, GECCO'05, ACM, Washington DC, USA, 2005, pp. 1069–1075.

[3] O. F. Rana, M. Warnier, T. B. Quillinan, F. Brazier, D. Cojocarasu, Managing violations in service level agreements, Grid Middleware and Services (2008) 349–358.

[4] M. C. Jaeger, G. Rojec-Goldmann, G. Muhl, Qos aggregation for web service composition using workflow patterns, in: 8th IEEE International Conference on Enterprise Distributed Object Computing, Proceedings, EDOC'04, IEEE, Monterey, California, USA, 2004, pp. 149–159.

[5] S.-Y. Hwang, H. Wang, J. Tang, J. Srivastava, A probabilistic approach to modeling and estimating the qos of web-services-based workflows, Information Sciences 177 (2007) 5484–5503.

[6] D. Mukherjee, P. Jalote, M. Gowri Nanda, Determining qos of ws-bpel compositions, in: 6th International Conference on Service-Oriented Computing, Proceedings, ICSOC'08, Springer-Verlag, Durban, South Africa, 2008, pp. 378–393.

[7] J. Cardoso, A. Sheth, J. Miller, J. Arnold, K. Kochut, Quality of service for workflows and web service processes, Journal of Web Semantics 1 (2004) 281–308.

[8] S. Rosario, A. Benveniste, S. Haar, C. Jard, Probabilistic qos and soft contracts for transaction-based web services orchestrations, IEEE Transactions on Services Computing 1 (2008) 187–200.

[9] P. Leitner, A. Michlmayr, F. Rosenberg, S. Dustdar, Monitoring, prediction and prevention of sla violations in composite services, in: IEEE International Conference on Web Services, Proceedings, ICWS'10, IEEE Computer Society, Miami, Florida, USA, 2010, pp. 369 –376.

[10] L. Zeng, B. Benatallah, A. H.H. Ngu, M. Dumas, J. Kalagnanam, H. Chang, Qos-aware middleware for web services composition, IEEE Transactions on Software Engineering 30 (2004) 311–327.

[11] R. Aggarwal, K. Verma, J. Miller, W. Milnor, Constraint driven web service composition in meteor-s, in: IEEE International Conference on Services Computing, Proceedings, SCC'04, IEEE Computer Society, Shanghai, China, 2004, pp. 23–30.

[12] J. Harney, P. Doshi, Speeding up adaptation of web service compositions using expiration times, in: 16th International Conference on World Wide Web, Proceedings, WWW'07, ACM, Banff, Alberta, Canada, 2007, pp. 1023–1032.

[13] T. Yu, Y. Zhang, K.-J. Lin, Efficient algorithms for web services selection with end-to-end qos constraints, ACM Transactions on the Web 1 (2007).

[14] V. N. Vapnik, Statistical learning theory, John Wiley & Sons, New York, 1998.

[15] J.A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, J. Vandewalle, Least squares support vector machines, World Scientific Pub. Co., Singapore, 2002.

[16] C.-L. Liu, K. Nakashima, H. Sako, H. Fujisawa, Handwritten digit recognition: benchmarking of state-of-the-art techniques, Pattern Recognition 36 (2003) 2271–2285.

[17] B.-J. Chen, M.-W. Chang, C.-J. lin, Load forecasting using support vector machines: a study on eunite competition 2001, IEEE Transactions on Power Systems 19 (2004) 1821–1830.

[18] A. Christmann, I. Steinwart, Consistency of kernel-based quantile regression, Applied Stochastic Models in Business and Industry 24 (2008) 171–183.

[19] M. C. Jaeger, G. Mühl, S. Golze, Qos-aware composition of web services: an evaluation of selection algorithms, in: Confederated international Conference on On the Move to Meaningful Internet Systems, Proceedings, OTM'05, Springer Verlag, Berlin, Germany, 2005, pp. 646–661.

[20] H. Xianglan, L. Yangguang, X. Bin, Z. Gang, A survey on qos-aware dynamic web service selection, in: 7th International Conference on

Wireless Communications, Networking and Mobile Computing, Proceedings, WiCOM'11, Wuhan, China, pp. 1–5.

[21] A. Strunk, Qos-aware service composition: A survey, in: IEEE 8th European Conference on Web Services, ECOWS'10, IEEE Computer Society, Ayia Napa, Syprus, 2010, pp. 67–74.

[22] W. M. P. Van Der Aalst, A. H. M. Ter Hofstede, B. Kiepuszewski, A. P. Barros, Workflow patterns, Distributed and Parallel Databases 14 (2003) 5–51.

[23] W. Wiesemann, R. Hochreiter, D. Kuhn, A stochastic programming approach for qos-aware service composition, in: 8th IEEE International Symposium on Cluster Computing and the Grid, Proceedings, CCGRID'08, IEEE Computer Society, Lyon, France, 2008, pp. 226–233.

[24] B. Kiepuszewski, A. H. M. t. Hofstede, C. Bussler, On structured workflow modelling, in: 12th International Conference on Advanced Information Systems Engineering, Proceedings, CAiSE'00, Springer-Verlag, Gdansk, Poland, 2000, pp. 431–445.

[25] K. Salimifard, M. Wright, Petri net-based modelling of workflow systems: an overview, European Journal of Operational Research 134 (2001) 664–676.

[26] S. Narayanan, S. A. McIlraith, Simulation, verification and automated composition of web services, in: 11th International Conference on World Wide Web, Proceedings, WWW'02, ACM, Honolulu, Hawaii, USA, 2002, pp. 77–88.

[27] P. Puschner, A. Burns, Guest editorial: A review of worst-case execution-timeanalysis, Real-Time Systems 18 (2000) 115–128.

[28] R. Koenker, J. Bassett, Gilbert, Regression quantiles, Econometrica 46 (1978) 33–50.

[29] I. Takeuchi, Q. V. Le, T. D. Sears, A. J. Smola, Nonparametric quantile estimation, Journal of Machine Learning Research 7 (2006) 1231–1264.

[30] K. Geebelen, E. Kulikowski, E. Truyen, W. Joosen, A mvc framework for policy-based adaptation of workflow processes: A case study on confidentiality, in: IEEE International Conference on Web Services, Proceedings, ICWS'10, Santa Clara Marriott, CA, USA, pp. 401–408.

[31] E. Al-Masri, Q. H. Mahmoud, Investigating web services on the world wide web, in: 17th International Conference on World Wide Web, Proceedings, WWW'08, ACM, Beijing, China, 2008, pp. 795–804.

Appendix

3.A Reformulation of the optimization problem

Theorem 34. *Optimization problem*

$$\min_{f_\tau \in \mathcal{H}} P(\textit{Type I error}|f_\tau)(1-\tau) + P(\textit{Type II error}|f_\tau)\tau \tag{3.35}$$

$$\textit{s.t. } P(\textit{Type II error}|\textit{accepted}, f_\tau) \leq 1-\tau \tag{3.36}$$

has the same solutions as optimization problem

$$\min_{f_\tau \in \mathcal{H}} P(\textit{Type II error}|\textit{accepted}, f_\tau)$$

$$+ P(\textit{rejected}|f_\tau)\left((1-\tau) - P(\textit{Type II error}|\textit{accepted}, f_\tau)\right) \tag{3.37}$$

$$\textit{s.t. } (1-\tau) - P(\textit{Type II error}|\textit{accepted}, f_\tau) \geq 0. \tag{3.38}$$

Proof. For simplicity we do the proof for the optimization problem without constraint (3.36), nevertheless it still holds with the constraint. First we rewrite equation (3.35) as

$$\min_{f_\tau \in \mathcal{H}} \left(P(\text{Type I error}|f_\tau) - P(\text{Type II error}|f_\tau)\right)(1-\tau) + P(\text{Type II error}|f_\tau).$$

As can be derived from the confusion matrix in Table 3.A.1, the following statement holds

$$P(\text{Type I error}|f_\tau) + P(\text{Violation}|f_\tau)$$

$$= P(\text{Rejected}|f_\tau) + P(\text{Type II error}|f_\tau). \tag{3.39}$$

The number of violations is a constant in the sense that it does not depend on whether we accept the request or not, therefore

$$P(\text{Violation}|f_\tau) = P(\text{Violation}). \tag{3.40}$$

Table 3.A.1: Confusion Matrix

	Accepted	**Rejected**
No Violation	True Positive	Type I Error
Violation	Type II Error	True Negative

After applying equation (3.39) and (3.40), the optimization problem becomes

$$\min_{f_\tau \in \mathcal{H}} \left(P(\text{Rejected}|f_\tau) - P(\text{Violation})\right)(1-\tau) + P(\text{Type II error}|f_\tau).$$

The term $P(\text{Violation})(1-\tau)$ is a constant for a given data set and can therefore be omitted from the optimization problem without affecting the optimum

$$\min_{f_\tau \in \mathcal{H}} P(\text{Rejected}|f_\tau)(1-\tau) + P(\text{Type II error}|f_\tau).$$

If we replace

$$P(\text{Type II error}|f_\tau) = P(\text{Type II error}|\mathit{Accepted}, f_\tau)(1 - P(\mathit{Rejected}|f_\tau))$$

in the optimization problem, we get the statement we had to proof: optimization problem (3.37). ■

3.B Obtaining data of real web services

A current problem for experiments regarding QoS of real web services is the lack of available datasets. For our analyses, we found no usable time series on Quality of Web Service attributes. Service providers usually only publish average values for their services. The QWS dataset[3] of Al-Masri et al. [31] includes measurements of 9 QoS attributes for 2500 real web services.

Each service was tested over a ten-minute period for three consecutive days. However, only the average QoS values are publicly available. Another public

[3]http: //www.uoguelph.ca/~qmahmoud/qws/index.html

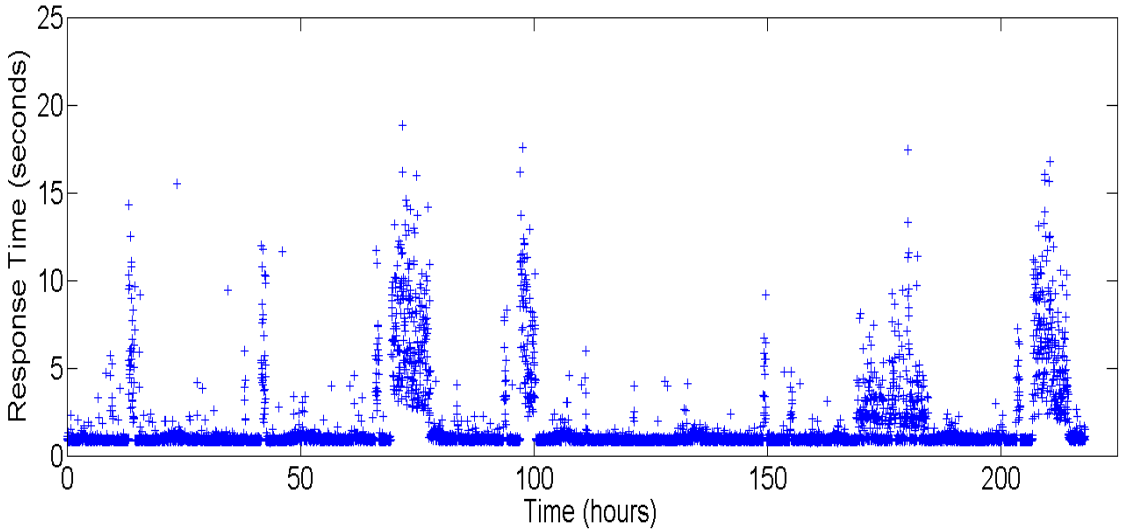


Figure 3.B.1: Time-varying RTs of online web service

dataset is WS-DREAM[4] which offers real invocation info on 100 web services by using 150 distributed computer nodes located all over the world. The dataset contains data on consecutive invocations of the services but is limited to 100 time series datapoints per service, which is not sufficient for our experiments. There is also no labelling on the time span of the different invocations of a service. Both datasets are restricted to short-running automated web services.

To cope with the data problem, we have collected real time series data for short-running online services ourselves. We used Web Inject[5], a free client-side monitoring tool for automated testing of web applications and web services. The tool allows to send soap requests to web services to analyse their response time and fault counts. We monitored a set of 8 popular online web services over a two-minute period for 7 consecutive days. Figure 3.B.1 illustrates the response time of a web service that allows a client to retrieve information on movies and theaters in the US. We can observe that a QoS attribute like response time can be very dynamic in time.

[4]http://www.wsdream.net/wsdream/
[5]http://www.webinject.org/

Chapter 4

Reducing the Run Time Complexity of SVM Classifiers

Reducing the Number of Support Vectors of SVM Classifiers using Smoothed Separable Case Approximation

Dries Geebelen, Johan A.K. Suykens, Joos Vandewalle

Published in *IEEE Transactions On Neural Networks And Learning Systems*, vol. 23, pp 682-688, April 2012.

Author contributions: D.G. developed the methods, performed the experiments and wrote the paper. The remaining authors provided feedback.

Abstract

In this brief, we propose a new method to reduce the number of support vectors of support vector machine (SVM) classifiers. We formulate the approximation of an SVM solution as a classification problem that is separable in the feature space. Due to the separability, the hard-margin SVM can be used to solve it. This approach, which we call the separable case approximation (SCA), is very similar to the cross-training algorithm explained in [1], which is inspired by editing algorithms [7]. The norm of the weight vector achieved by SCA can, however, become arbitrarily large. For that reason, we propose an algorithm, called the smoothed SCA (SSCA), that additionally upper-bounds the weight vector of the pruned solution and, for the commonly used kernels, reduces the number of support vectors even more. The lower the chosen upper bound, the larger this extra reduction becomes. Upper-bounding the weight vector is important because it ensures numerical stability, reduces the time to find the pruned solution, and avoids overfitting during the approximation phase. On the examined datasets, SSCA drastically reduces the number of support vectors.

4.1 Introduction

Classifying data is a common task in machine learning. Given a set of input vectors along with corresponding class labels, the task is to find a function that defines the relation between input vectors and their class labels. Vapnik's support vector machine (SVM) classifier [5, 15] is a classifier that obtains good accuracy on real-life datasets. Compared to other classification algorithms, the SVM classifier has an important drawback: the run time complexity, as explained in [3] and [12], can be considerably higher because of the large number of support vectors.

Several methods have been proposed for pruning the solutions of SVMs. In [3] and [13], an algorithm is proposed that approximates the weight vector such that the distance to the original weight vector is minimized. Downs et al. [8] present an algorithm that eliminates support vectors linearly dependent on the other support vectors. Osuna and Girosi [12] used SVM itself as a regression tool to approximate the decision surface. Keerthi et al. [10] present an effective greedy method SVM that uses a basis selection criterion which is directly related to the training cost. In fixed-size (FS) least square (LS)-SVM [14], [6] an active subset selection method based on quadratic Renyi entropy is used to select basis functions after which a training cost similar to that of LS-SVM is optimized. Bakir et al.[1] propose to selectively remove examples from the training set using probabilistic estimates related to editing algorithms.

In this brief, an algorithm, called the separable case approximation (SCA), is discussed that improves the sparsity of SVM classifiers. SCA is strongly related

to the cross-training algorithm discussed in [1]. SCA first makes the training set separable by removing misclassified examples or by swapping their labels, and then separates the dataset using Vapnik's SVM classifier for the separable case. Additionally, we propose an extension called the smoothed SCA (SSCA) that upper-bounds the weight vector of the pruned solution. In many cases, SSCA results in even fewer support vectors than SCA when using a radial basis function (RBF) kernel or a polynomial (Pol) kernel.

The remainder of this brief is organized as follows. In Section 4.2, the separable and nonseparable cases are briefly reviewed. Section 4.3 explains how approximating an SVM classifier can be formulated as a classification problem, and presents two algorithms, namely, SCA and SSCA, to solve this classification problem. Existing pruning methods are reviewed in Section 4.4, and their performances are compared to that of SSCA in Section 4.5. Finally, Section 4.6 makes some concluding remarks.

4.2 Support Vector Machine Classifier

Given a set of N training examples with input vectors $\{x_k \in \mathbb{R}^b\}_{k=1}^N$ and corresponding labels $\{y_k \in \mathbb{R}^b\}_{k=1}^N$, Vapnik's SVM classification algorithm builds a linear classifier in the feature space

$$\hat{y}(x) = \text{sign}(w^T \varphi(x) + b) \tag{4.1}$$

where the feature map $\varphi(x)$ maps the input data into a feature space. It solves the following primal problem:

$$\min_{w,b,\xi} J_P(w) = \frac{1}{2} w^T w + C \sum_{k=1}^{N} \xi_k$$

$$\text{such that } \begin{cases} y_k[w^T \varphi(x_k) + b] \geq 1 - \xi_k, & k = 1, \ldots, N, \\ \xi_k \geq 0, & k = 1, \ldots, N, \end{cases} \tag{4.2}$$

where C is a positive real constant. The hyperparameter C determines the tradeoff between the margin and the training error. In the nonseparable case ($C < \infty$), Vapnik's SVM classifier tolerates misclassifications in the training set due to the slack variables (ξ_k) in the problem formulation. In the separable case ($C = \infty$), all ξ_k equal 0, which implies misclassifications are not tolerated. The dual, after applying the kernel trick, becomes

$$\max_{\alpha} J_D \alpha = -\frac{1}{2} \sum_{k,l=1}^{N} y_k y_l K(x_k, x_l) \alpha_k \alpha_l + \sum_{k=1}^{N} \alpha_k$$

$$\text{such that } \begin{cases} \sum_{k=1}^{N} \alpha_k y_k = 0 \\ 0 \leq \alpha_k \leq C, \qquad k = 1, \ldots, N. \end{cases} \tag{4.3}$$

According to Mercer's theorem [11], any positive semidefinite kernel function can be expressed as

$$K(x, z) = \varphi(x)^T \varphi(z). \tag{4.4}$$

Due to the kernel trick, we can now work in a huge dimensional feature space without having to perform calculations explicitly in this space. Popular nonlinear kernel functions are the RBF kernel ($K(x, z) = exp\left(-\|x - z\|_2^2/\sigma^2\right)$) and the Pol kernel ($K(x, z) = (x^T z + \tau)^d$ with $\tau \geq 0$). The predicted label of a new data point becomes

$$\hat{y}(x) = \text{sign}\left(\sum_{k=1}^{N} \alpha_k y_k K(x, x_k) + b\right) \tag{4.5}$$

The run time complexity scales linearly with the number of support vectors. For the nonseparable case, the support vectors are characterized by

$$y_k[w^T \varphi(x_k) + b] \leq 1. \tag{4.6}$$

This means that the support vectors are misclassified examples ($y_k[w^T \varphi(x_k) + b] \leq 0$) or examples close to the decision boundary in the feature space ($0 < y_k[w^T \varphi(x_k) + b] \leq 1$). For the separable case, all support vectors satisfy $y_k[w^T \varphi(x_k) + b] = 1$. This means that all the support vectors are on the hyperplane where $w^T \varphi(x) + b = 1$ or the hyperplane where $w^T \varphi(x) + b = -1$.

4.3 SCA

4.3.1 Approximating the decision Boundary

This section explains the approximation criterion SCA and SSCA try to minimize and shows how minimizing this approximation error can be formulated as a classification problem. The goal of sparse approximation is to approximate the SVM classifier defined by w and b by a sparser SVM classifier defined by w' and b' such that the classification on unseen data is as similar as possible. We define $\hat{y}(x)$ and $\hat{y}'(x)$ as the label of a data point according to the original and the approximating classifier, respectively. The first approximation error measure we propose equals the probability that a data point is classified differently by $\hat{y}(x)$ and $\hat{y}'(x)$:

$$E_1 = P(\hat{y}(x) \neq \hat{y}'(x)). \tag{4.7}$$

If we modify the training set by flipping the labels of training examples misclassified by $\hat{y}(x)$, then the approximation problem according to (4.7) becomes a classification problem over the modified training set. If it is only important to classify correctly classified data points similarly, then the approximation error becomes

$$E_2 = P(\hat{y}(x) \neq \hat{y}'(x) \mid \hat{y}(x) = y(x)). \tag{4.8}$$

If we modify the training set by removing training examples misclassified by $\hat{y}(x)$, then the approximation problem according to (4.8) becomes a classification problem over the modified training set.

In both cases, the modified training dataset is separable in the feature space because the nonpruned solution separates it. This separability holds only when the SVM classifier used to solve the approximation problem has the same kernel as the original classifier. We exploit this separability to gain sparsity.

4.3.2 SCA

The SCA algorithm (See Algorithm 5) realizes sparsity by using the SVM classifier for the separable case to solve the approximation problem. The final support vectors are training examples close to the original separating hyperplane. Vapnik's SVM classifier for the separable case will, in general, not cause more support vectors than $p + 1$, where p is the dimensionality of the feature space. This upper bound exists because all support vectors lie on one of two parallel hyperplanes in the feature space and the degrees of freedom for choosing two parallel hyperplanes in a p-dimensional space equals $p + 1$. Under certain nongeneric circumstances, such as linear dependency between p or less training data points, this upper bound can be exceeded. This algorithm is very similar to the cross-training algorithm [1]. It differs in the sense that for step 1 the crosstraining algorithm divides the training set in subsets, trains an SVM on each subset, and then combines these different SVM classifiers into a global classifier. The flipping of labels is also not discussed in [1].

The SCA has an important drawback: $\|w\|_2$ can become arbitrarily large because the hyperplane where $w^T\varphi(x)+b = 1$ and the hyperplane where $w^T\varphi(x)+b = -1$ can lie arbitrarily close to each other. We want to upper-bound $\|w\|_2$ for the following reasons:

- We want to remove the ill-conditioning of the approximation problem. This ill-conditioning makes the solution numerically unstable and significantly increases the time that the sequential minimal optimization algorithm needs to find the solution.

Algorithm 5 SCA

1. Find the non-pruned solution usinf Vapnik's SVM for the non-separable case.
2. Modify the training set such that it becomes separable by applying rules a) or b) to misclassified examples ($y_k[w^T\varphi(x_k) + b] \leq 0$).
   (a) Flip their labels.
   (b) Remove them.
3. Given the modified training set, find the SVM solution for the separable case.

Table 4.1: SCA applied to the Ripely dataset. SV is the number of support vectors, ACC_{TR} is the training set accuracy, and ACC_{TEST} is the test set accuracy.

	SV	$\text{ACC}_{\text{TR}}(\%)$	$\text{ACC}_{\text{TEST}}(\%)$
Nonpruned	78	89.6	89.7
Remove	7	90.0	89.0
Flip	8	89.6	89.0

Algorithm 6 SSCA

1. Find the nonpruned solution using Vapnik's SVM for the nonseparable case.
2. Modify the training set such that it becomes separable by applying rules a) and b) to misclassified examples ($y_k[w^T\varphi(x_k) + b] \leq 0$),
   (a) Flip their labels
   (b) Remove them

   and additionally remove training examples for which $y_k[w^T\varphi(x_k) + b] \leq D$ with $D > 0$.
3. Given the modified training set, find the SVM solution for the separable case.

Table 4.2: SSCA with removal of misclassified examples applied to the Ripley dataset. D is a parameter of the SSCA algorithm, SV is the number of support vectors, $\mathrm{ACC_{TR}}$ is the training set accuracy, $\mathrm{ACC_{TEST}}$ is the test set accuracy, and $\|w'\|_2$ is the 2-norm of the weight vector.

D	SV	$\mathrm{ACC_{TR}}(\%)$	$\mathrm{ACC_{TEST}}(\%)$	$\|w'\|_2$
Nonpruned	78	89.6	89.7	25
0	7	90.0	89.0	236
0.3	9	89.6	89.9	47
0.9	8	88.8	90.1	21
1.3	5	87.2	90.2	11

- We want to avoid overfitting (in the sense that training data points can be classified similarly by the pruned and nonpruned classifiers while the unseen data points are classified differently) by imposing smoothness.
- Ideally, upper-bounding $\|w\|_2$ decreases the number of support vectors.

To conclude this section we apply SCA to the Ripley dataset using an SVM classifier with an RBF kernel. The resulting classifier is visualized in Figure 4.1 and the resulting accuracies are shown in Table 4.1. The number of support vectors reduces drastically. As expected, the training set accuracy does not decrease after the pruning. Figure 4.1 shows that the support vectors of the pruned solution lie close to the decision boundary of the nonpruned solution.

4.3.3 SSCA

The SSCA algorithm upper bounds $\|w'\|_2$ as shown in Algorithm 6. SSCA makes a tradeoff between the approximation accuracy on the training set and the norm of the weight vector. Increasing D decreases the lower bound on the approximation accuracy of the training set and decreases the upper bound on $\|w'\|_2$ because:

- all training examples at a distance higher than $D/\|w\|_2$ from the separating hyperplane in the feature space are classified similarly by $\hat{y}$ and $\hat{y}'$
- $\|w'\|_2$ is upper bounded by $\|w\|_2/D$ as we explain next.

In the modified training set, all examples satisfy

$$y_k \left[\left(\frac{w}{D} \right)^T \varphi(x_k) + \frac{b}{D} \right] \geq 1 \tag{4.9}$$

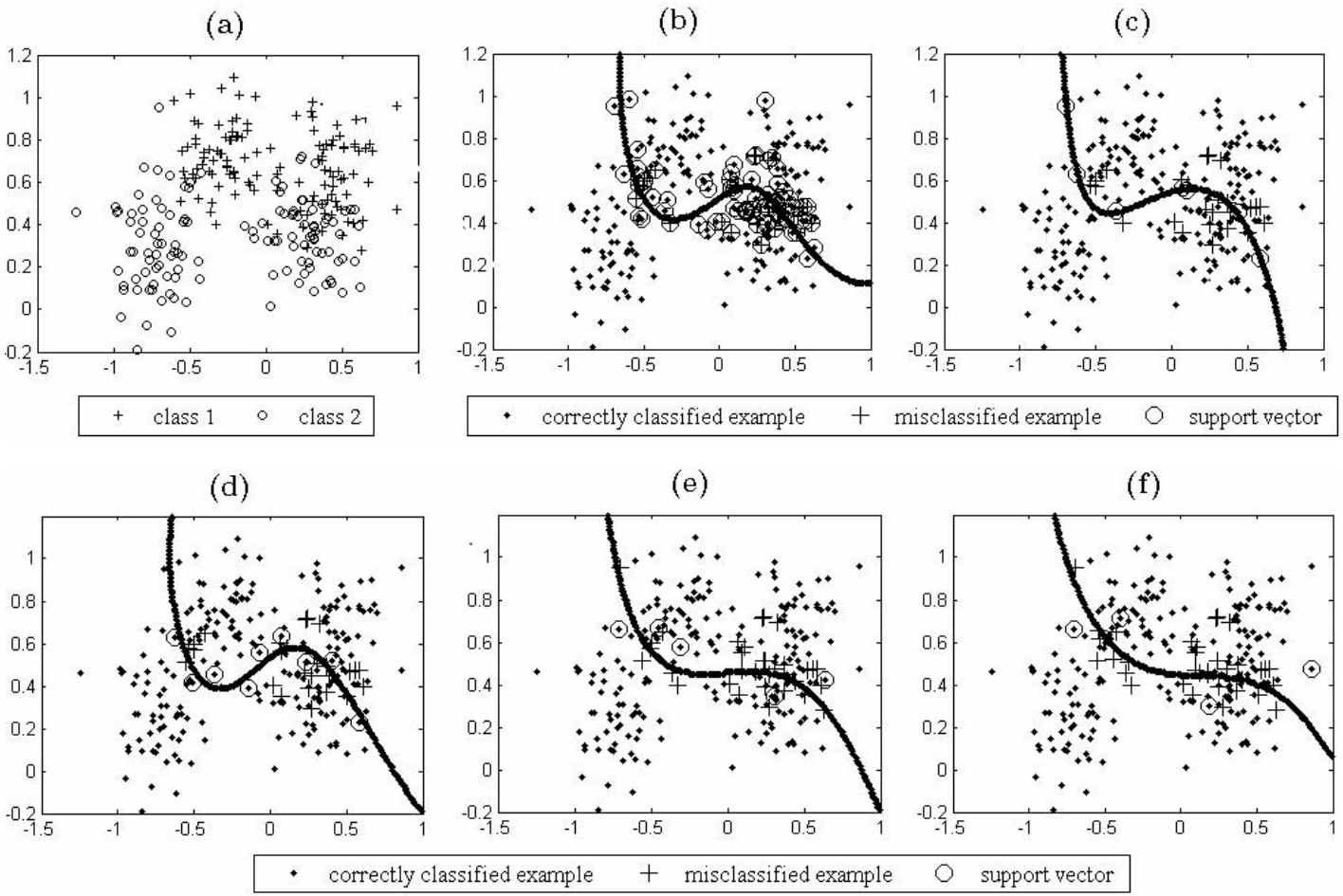


Figure 4.1: Ripley dataset containing 250 training data points and 1000 test data points. (a) Training examples of the Ripley dataset with corresponding labels. (b) Nonpruned solution using Vapnik's SVM classifier for the nonseparable case with $\sigma = 1$ and $C = 100$. (c) Pruned solution using SCA with removal of misclassified examples. The remaining figures illustrate SSCA with removal of misclassified examples. Parameter D differs per figure: (d) $D = 0.3$, (e) $D = 1.3$, (f) $D = 2.0$.

Table 4.3: NSCA with removal of misclassified examples applied to the ripley dataset. C is a parameter of the NSCA algorithm, SV is the number of support vectors, ACC_{TR} is the training set accuracy, ACC_{TEST} is the test set accuracy, and $\|w'\|_2$ is the 2-norm of the weight vector.

C	SV	$\text{ACC}_{\text{TR}}(\%)$	$\text{ACC}_{\text{TEST}}(\%)$	$\|w'\|_2$
Nonpruned	78	89.6	89.7	25
∞	7	90.0	89.0	236
1000	10	89.6	89.6	67
100	22	88.8	89.4	32
10	43	87.2	89.8	14

because w', the weight vector of the pruned solution, is found according to the maximal margin criterion, its 2-norm is the smallest of all weight vectors satisfying

$$y_k[w^T\varphi(x_k) + b] \geq 1, k = 1, \ldots, N \tag{4.10}$$

for the modified training set. Hence, the following holds:

$$\|w'\|_2 \leq \frac{\|w\|_2}{D}. \tag{4.11}$$

Using SSCA, we can upper-bound $\|w'\|_2$. In many cases, increasing D reduces the number of support vectors even more when using an RBF or Pol kernel, as empirically shown in Section 4.5. The exact reason for this extra reduction is unclear. The best value for D can be found using (cross) validation. The parameter D can be optimized independently from the other hyperparameters. The overhead generated to determine D is thus relatively small. The results for the Ripley dataset, when removing misclassified examples, are shown in Table 4.2 and visualized in Figure 4.1. Increasing D decreases the training set accuracy, decreases $\|w'\|_2$, and does not increase the number of support vectors.

4.3.4 NSCA

The most straightforward way of upper-bounding $\|w'\|$ is using the SVM for the nonseparable case to solve the approximation problem. This implies an additional hyperparameter (C'). The results of applying NSCA on the Ripley dataset are shown in Table 4.3: decreasing C' decreases the training set accuracy, decreases $\|w'\|_2$, and increases the number of support vectors. The increase in the number of support vectors for upper-bounding $\|w'\|_2$ is a very big disadvantage, which makes us prefer SSCA over this algorithm.

4.4 Other pruning methods

This section briefly explains other pruning algorithms.

4.4.1 Cross-Training

Cross-training begins with partitioning the training set randomly into subsets and training-independent SVMs on each subset. The decision functions of these SVMs are then used to discard two types of training examples: namely, those that are confidently recognized, and those that are misclassified. A final SVM is trained using the remaining examples. The training of the final SVM can be done using SCA. The innovation of our algorithm compared to cross-training is the smoothing used in SSCA.

4.4.2 Greedy iterative methods

Greedy iterative methods, such as Keerthi's method [10], incrementally choose basis functions according to a criterion that is directly related to the training cost function. In most cases, the basis functions correspond to training data points. In Keerthi's method, the training cost function to minimize, given a chosen set of basis functions ($\mathcal{J}$), becomes

$$min_{\beta_J} f(\beta_j) = \frac{1}{2}\beta_J^T K_{JJ}\beta_J + \frac{C}{2}\sum_{i=1}^{n}\max(0, 1 - y_i o_i) \tag{4.12}$$

where J contains the indexes of the basis functions belonging to $\mathcal{J}$, $K_{ij} = k(x_i, x_j)$, $o_i = K_{iJ}\beta_j$, and K_{IJ} corresponds to the submatrix of the kernel matrix made of the rows indexed by I and columns indexed by J.

4.4.3 FS LS-SVM

In FS LS-SVM[14], the basis functions are first selected according to the quadratic Renyi entropy. Then, a cost function similar to that of LS-SVM is minimized in the primal space. FS LS-SVM is not an iterative method; the number of basis functions has to be chosen in advance or needs to be tuned. The bandwidth of the kernel used for the selection of the basis functions is, in our experiments, chosen according to the ISNR [9].

4.5 Experimental Results

We will compare SSCA with Keerthi's method [10] and FS LS-SVM [14]. The datasets used are Adult (*adu*) and Protein (*pro*). The dataset *adu* has been obtained from the UCI benchmark repository [2], and *pro* has been obtained from LIBSVM [4] and is used in [16]. The properties of the datasets are shown in Table 4.1. The dataset *pro* contains three classes. We modify this into a two-class problem by merging the classes labeled as "1" and "2". For the pruned solution, we approximate the separable case by taking a very large, but finite, value of C. This reduces the impact of the ill-conditioning. LIBSVM [4] is the software used to find both the nonpruned and pruned solution. The experiments ran in MATLAB on a Dell OptiPlex 780 with Windows 7 Professional, 4.0 GB RAM, and a Intel(R) Core(TM)2 Quad Processor Q9550 (2.83 GHz).

In Table 4.2, we can see that flipping labels does not cause a significantly better tradeoff between the test set. accuracy and the number of support vectors compared to removing misclassified examples for *adu*(rbf) and *pro*(rbf). In further experiments, we will always apply SSCA with removal of misclassified labels. Figures 4.1 and 4.2 show that SSCA drastically reduces the number of support vectors. On *adu*, Keerthi's method has mostly the best tradeoff between test set accuracy and the number of support vectors. On *pro*(pol), SSCA has the better tradeoff for 300 support vectors. The differences in performance can be partly caused by the differences in loss function. To compare the times required to find the solution, we have to take the time needed to tune the hyperparameters into account. In FS LS-SVM, the basis functions are chosen independently of the hyperparameters. This selection has to be done only once before tuning, and the cost is thus not proportional to the number of tried hyperparameter values. Assuming that the number of values of hyperparameters to be tried exceeds 20, we can conclude from Table 4.3 that FS LS-SVM is the fastest method, followed by Keerthi's method. SSCA is almost as fast as SVM because the tuning of D can be done independently of the other hyperparameters.

4.6 Conclusion

In this brief, we succeeded in drastically reducing the number of support vectors of SVM classifiers using the SSCA. Instead of approximating the underlying weight vector, we directly approximate the way the data points are classified. Compared to the SCA, SSCA has the following benefits: 1) it ensures numerical stability; 2) reduces the time to find the pruned solution; 3) avoids overfitting during the approximation phase; and 4) reduces the number of support vectors even more in certain cases. SSCA is slower than some other pruning algorithms, but almost as fast as finding the nonpruned solution.

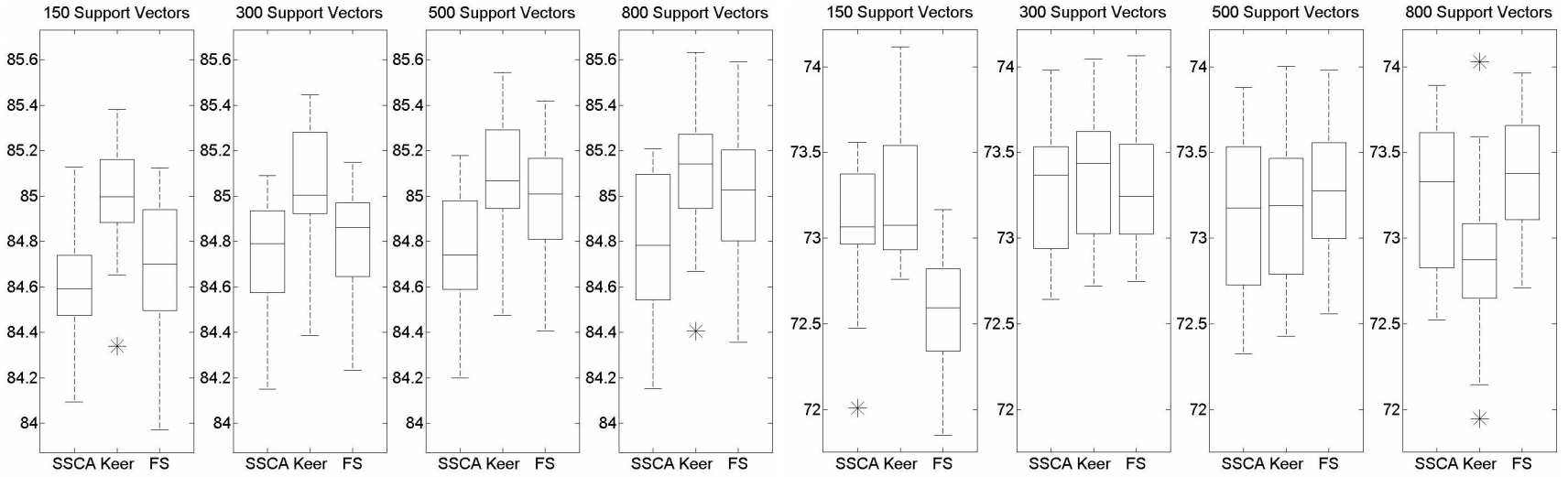


Figure 4.1: SSCA, Keerthi's method (Keer), and FS LS-SVM reducing the number of support vectors of SVM classifiers using a RBF kernel ($K(x,z) = exp\left(-\|x-z\|_2^2/\sigma^2\right)$). The bandwidth of the kernel used for the selection of the basis functions in FS LS-SVM is chosen according to the improved Silverman's normal rule of thumb (ISNR). For each randomization, the data points that belong to the training set are randomly selected. The remaining data points belong to the test set. $\mathrm{ACC}_{\mathrm{TEST,npr}}$ is the average test set accuracy of the nonpruned SVM solution, and $\mathrm{SV}_{\mathrm{npr}}$ is the average number of support vectors of the nonpruned SVM solution. In the SSCA method, we always remove misclassified data points instead of flipping their labels. The hyperparameters are optimized using 10-fold cross-validation. For FS LS-SVM, the hyperparameters depend on the number of basis functions and differ from the hyperparameters of the nonpruned SVM solution. Hyperparameters of the SVM solution have subscript "svm". Hyperparameters of Keerthi's method have subscript "kee". Given a certain dataset, kernel, and number of support vectors, the figures show one boxplot for each method: the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered as outliers, and the outliers are plotted individually. (a) *adu*(rbf); randomizations: 20; hyperparameters: $\sigma_{svm} = 8$, $C_{svm} = 5$, $\sigma_{kee} = 8$, $C_{kee} = 3$; $\mathrm{ACC}_{\mathrm{TEST,npr}} = 84.71\%$; and $\mathrm{SV}_{\mathrm{npr}} = 11604$. (b) *pro*(rbf); randomizations: 20; hyperparameters: $\sigma_{svm} = 7$, $C_{svm} = 20$, $\sigma_{kee} = 20$, $C_{kee} = 200$; $\mathrm{ACC}_{\mathrm{TEST,npr}} = 74.42\%$; and $\mathrm{SV}_{\mathrm{npr}} = 8116$.

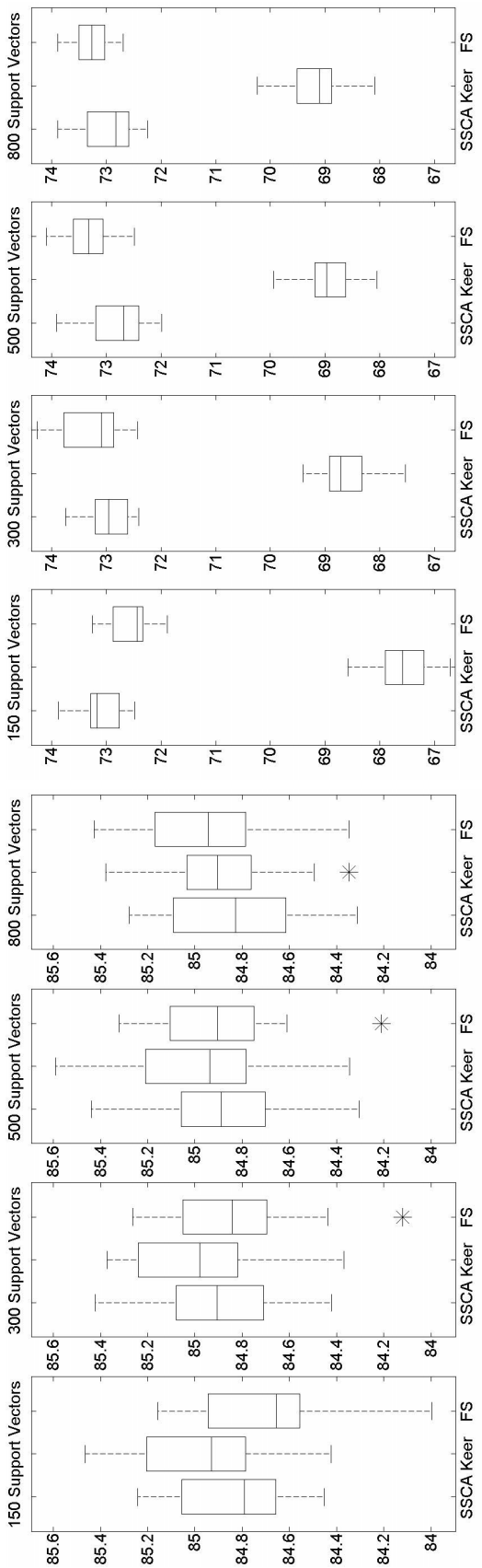


Figure 4.2: SSCA, Keerthi's method (Keer), and FS LS-SVM reducing the number of support vectors of SVM classifiers using a Pol kernel ($K(x,z) = (x^T z + \tau)^d$ with $\tau \geq 0$). (c) *adu*(pol); randomizations: 20; hyperparameters: $d_{svm} = 3$, $\tau_{svm} = 0.1$, $C_{svm} = 500$, $d_{kee} = 2$, $\tau_{kee} = 0.9$, $C_{kee} = 0.01$; $\mathrm{ACC}_{\mathrm{TEST,npr}} = 84.75\%$; and $\mathrm{SV}_{\mathrm{npr}} = 11583$. (d) *pro*(pol); randomizations: 20; hyperparameters: $d_{svm} = 3$, $\tau_{svm} = 1.1$, $C_{svm} = 500$, $d_{kee} = 3$, $\tau_{kee} = 0.9$, $C_{kee} = 2000$; $\mathrm{ACC}_{\mathrm{TEST,npr}} = 74.37\%$; and $\mathrm{SV}_{\mathrm{npr}} = 8002$.

Table 4.1: Properties of the benchmark datasets. N_{TEST} stands for the number of test data points, N_{TR} for number of the training data points, n_{INP} for the number of input dimensions, and n_{CLA} for the number of classes.

	N_{TEST}	N_{TR}	n_{INP}	n_{CLA}
adu	12222	30000	123	2
pro	5922	11844	357	3

Table 4.2: Comparison of SSCA with removal of misclassified examples and SSCA with flipping of labels of misclassified examples when using a RBF kernel. D is a parameter of the SSCA algorithm, REM(*adu*) is the test set accuracy and, in parentheses, the number of support vectors of SSCA applied on *adu* when removing misclassified examples. FLIP(*adu*), REM(*pro*), and FLIP(*pro*) are defined similarly. The nonpruned solution is denoted as npr.

D	rem(*adu*)	flip(*adu*)	rem(*pro*)	flip(*pro*)
npr	84.7	84.7	74.2	74.2
0	84.4(577)	84.4(771)	74.1(3678)	73.8(3782)
0.5	84.4(254)	84.4(271)	73.7(1740)	73.7(1741)
1	84.2(136)	84.2(140)	73.7(681)	73.7(682)
1.5	84.3(74)	84.3(76)	73.6(384)	73.6(387)
2	84.0(58)	84.0(58)	73.0(272)	73.1(273)

Table 4.3: Time required to find the solution containing 300 support vectors for *adu*(rbf) and 500 support vectors for *pro*(pol). The compared algorithms are SSCA, Keerthi's method, and FS LS-SVM. In the case of SSCA, The time required to find the pruned solution given the nonpruned solution is in parentheses and the time required to find the nonpruned solution is before the parentheses. In case of FS LS-SVM, the time required to find the basis functions is in parentheses, and the time required to find the solution, given the basis functions, is before the parentheses.

	adu(seconds)	*pro*(seconds)
Nonpruned SVM	88.3	1112
SSCA	88.3(5.8)	1112(3.1)
Keerthi's method	23	78.5
FS LS-SVM (ISNR)	0.9(397)	1.4(483)

4.7 Acknowledgement

The authors would like to thank an anonymous reviewer for the suggestion of flipping labels.

References

[1] Bakir, G. H., J. Weston, and L. Bottou, "Breaking SVM complexity with cross-training," in Proc. 17th Neural Inf. Process. Syst. Conf., vol. 17. 2005, pp. 81–88.

[2] Blake, C. L., and C. J. Merz. (1998). UCI Repository of Machine Learning Databases. Dept. Inf. Comput. Sci., Univ. California, Irvine [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[3] Burges, C. J. C. , "Simplified support vector decision rules," in Proc. 13th Int. Conf. Mach. Learn., 1996, pp. 71–77.

[4] Chang, C.-C., and C.-J. Lin. (2001). LIBSVM: A Library for Support Vector Machines [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm

[5] Cortes, C., and V. Vapnik, "Support vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, 1995.

[6] De Brabanter, K., J. De Brabanter, J. A. K. Suykens, and B. De Moor, "Optimized fixed-size least squares support vector machines for large data sets," Comput. Stat. Data Anal., vol. 54, no. 6, pp. 1484–1504, Jun. 2010.

[7] Devijver, P., and J. Kittler, Pattern Recogniton: A Statistical Approach. Englewood Cliffs, NJ: Prentice-Hall, 1982.

[8] Downs, T., K. E. Gates, and A. Masters, "Exact simplification of support vector solutions," J. Mach. Learn. Res., vol. 2, pp. 293–297, Dec. 2001.

[9] Hjort, N. L., and D. Pollard, "Better rules of thumb for choosing bandwith in density estimation," Dept. Math., Univ. Oslo, Oslo, Norway, Res. Rep., 1996.

[10] Keerthi, S. S., O. Chapelle, and D. DeCoste, "Building support vector machines with reduced classifier complexity," J. Mach. Learn. Res., vol. 7, pp. 1493–1515, Jul. 2006.

[11] Mercer, J., "Functions of positive and negative type, and their connection with the theory of integral equations," Philos. Trans. Royal Soc. London, vol. 209, nos. 441–458, pp. 415–446, 1909.

[12] Osuna, E., and F. Girosi, "Reducing the run-time complexity of support vector machines," in Proc. Int. Conf. Pattern Recognit., Brisbane, Australia, Aug. 1998, pp. 1–10.

[13] Schölkopf, B., S. Mika, C. J. C. Burges, P. Knirsch, K. Müller, G. Rätsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," IEEE Trans. Neural Netw., vol. 10, no. 5, pp. 1000–1017, Sep. 1999.

[14] Suykens, J. A. K., T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, Least Squares Support Vector Machines. Singapore: World Scientific, 2002.

[15] Vapnik, V., Statistical Learning Theory. New York: Wiley, 1998.

[16] Wang, J.-Y. "Application of support vector machines in bioinformatics," M.S. thesis, Dept. Comput. Sci. Inf. Eng., National Taiwan University, Taipei, Taiwan, 2002.

Chapter 5

Conclusion

5.1 Summary

In Chapter 2 we developed methods that (1) quantify diversity, richness and related properties in a similarity-based framework, (2) predict QoS attributes of workflow compositions, and (3) reduce the number of support vectors of SVM classifiers. All of these methods have in common that they use the kernel trick. First, we proposed the OLC diversity, which is a diversity index that reflects the trade-off between the approximation error and the description length when compressing the representation of a group of entities optimally, and the OLC richness, which equals the OLC diversity maximized in the abundances. Because we are not only interested in the value of the OLC diversity and the OLC richness but also in the underlying causes, we developed the following derived indices: (1) the OLC contribution vector quantifies the contribution of each type towards the OLC richness, (2) the OLC average contribution expresses how much of the richness is lost due to the similarity between the types, (3) the OLC evenness expresses how much of the diversity is lost due to not having the abundances that maximize the diversity, and (4) the decomposition in alpha and beta provides insight in the differences between subgroups. We show that these indices meet a number of important properties and overcome several issues of existing indices. An important benefit of our framework is the fact that all indices are defined within one framework and are linked to each other. For example, adding a type increases the Shannon OLC richness if and only if the type is to some extend unique (the uniqueness is linked to the richness here). In the literature there are many papers that propose only a few indices and therefore a lot of links are missing. Although our framework is suited for many applications, in this thesis we mainly used it for measuring biodiversity.

Second, in Chapter 3 we proposed kernel-based quantile estimators to predict QoS attributes of workflow compositions. Our estimators have the following favorable properties: (1) they perform time series prediction and thus incorporate time related information; (2) dependencies between the elementary services are implicitly taken into account; and (3) the number of violations converges to the agreed quantile value when the number of datapoints goes to infinity. We proposed two performance indicators in this context. The first performance index expresses how accurate the prediction algorithm is locally, i.e., it expresses how close the predicted QoS value is to the actual QoS value on average. The second performance index reflects whether the quantile value is satisfied globally, i.e., whether the number of violations in the entire test set corresponds to the number of violations agreed in the SLA. In both the simulated and the real-world experiments, our proposed methods outperform existing state-of-the-art algorithms with respect to both performance indices.

Finally, in Chapter 4 we propose SCA and SSCA, two algorithms that reduce the run time complexity of SVM classifiers. These algorithms approximate the non-pruned solution by first making the dataset separable and then applying the SVM classifier for the separable case. SSCA has the following advantages compared to SCA: (1) it ensures numerical stability; 2) it requires less the time to find the pruned solution; 3) it avoids overfitting during the approximation phase; and 4) it reduces the number of support vectors even more in certain cases. In the experiments, we drastically reduce the number of support vectors.

5.2 Limitations and future work

5.2.1 Measuring biodiversity

The similarity-based indices that we propose have the following limitations. To start with, it is not easy to understand our indices and our framework, especially for people that do not have a strong background in mathematics or machine learning. In order to understand our framework you need to be familiar with:

- Linear algebra: you need to know what positive semidefinite matrices are, what eigenvalues and eigenvectors are, and how the PCA algorithms works (our compression algorithm is similar to PCA). Moreover, you need to understand the Schur–Horn theorem as well as its reformulation from a polyhedral geometry perspective because it explains, on the one hand, how our algorithm optimizes the trade-off between the approximation error and the description length links and, on the other hand, how this trade-off can be linked to entropy.

- Kernels: we transform all algorithms into kernel methods by applying the kernel trick. The first important advantage of the kernel trick is that it can reduce the computational complexity of calculating the indices because no explicit mapping into the feature space is needed. The second advantage is that we use it to derive the properties that a meaningful similarity function should satisfy (e.g., the similarity of an object with itself always equals 1).

- Entropy: our Shannon entropy is mathematically equivalent to the Von Neumann entropy. We make this entropy kernel-based and, on top of that, we solve optimization problems in this entropy. The Von Neumann entropy itself is mathematically not easy to understand and our proposed diversity, richness and contribution vector are mathematically even more complicated. Also, alpha and beta diversity can be linked to conditional entropy and understanding conditional entropy therefore gives insight in the meaning of both alpha and beta diversity. The link with the Von Neumann entropy is important because a lot of non-trivial mathematical properties are proven to hold for the Von Neumann entropy and all of these properties also hold for the Shannon OLC entropy. People that are familiar with the Von Neumann entropy will thus have better insight in the behavior of the OLC diversity.

- Optimization: to calculate the richness and the contribution vector we need to solve an optimization problem. We show that this optimization problem is quasiconcave and, furthermore, that it can be transformed into a concave optimization problem. In the case of the Shannon OLC richness and the Shannon OLC contribution vector, this optimization problem most likely cannot be cast as a SemiDefinite Programming (SDP) problem, which is quite rare for concave optimization problems. This implies that we cannot use existing software - which mainly focuses on SDP problems - to solve the optimization problem and therefore we had to implement a gradient ascent algorithm ourselves. Note that due to the mathematical complexity of the indices, operations that seem trivial (e.g., taking the derivative of the Shannon OLC entropy with respect to the relative abundances) are often non-trivial. Understanding these optimization problems does not only give insight in how they can be solved, but also in how the diversity relates to the relative abundances (e.g., the OLC diversity is quasi-concave in the relative abundances).

- Ecology: before you can use our indices to measure biodiversity, you need to understand what biodiversity is and in which context it is used (e.g., it can be used for conservation prioritization). Designing the similarity function such that the resulting index behaves as expected is arguably the hardest part of using our indices. It requires extensive knowledge of both our indices and the field in which they are applied.

Applying an index that you do not fully grasp is a risk because you have to trust the authors that the index is decent and you might not apply the index correctly. Other indices are often simpler but at the cost of more shortcomings. In our opinion, it is better to meet the desired properties than to be easy to understand.

Another limitation is that we did not provide ready-to-use similarity functions. Developing similarity functions for specific applications is far from trivial which can discourage users from using our index. The impact of the choice of the similarity function on the values of the indices should not be underestimated. Most other indices, however, either chose the similarity for the user without explicitly stating which non-trivial assumptions this choice implies or do not properly define similarity in their framework in which case it is not possible to choose the similarity decently. In the latter case simple rules of thumb are often proposed (e.g., chose the similarity equal to the proportion of shared features) but it is poorly explained why these rules make sense. Many existing indices thus gives the user a false sense of confidence. The phylogenetic similarity that we used in the experiments depends on the position of the root; it doesn't make sense that the actual similarity between species would depend on this position. Therefore it is not recommended to use this similarity in real applications. We want to note here that many other indices in the literature also depend on the position of the root and they thus share this disadvantage. In future work, we are planning to develop a phylogenetic similarity function that is independent of this position. We are also planning to develop functional similarity functions.

This thesis does not explain in detail what uniqueness means (e.g., does a uniqueness of 0.5 mean that 50% of its features are unique or does it mean something else?). We only explained that a type with unique features has a non-zero uniqueness and that a type completely distinct from the other types has a uniqueness equal to 1. In future work, we will explain that uniqueness equals the proportion of unique information that a type contains - which is different from the proportion of unique features. For example, suppose that we have two types with the feature pairs 'AB' and 'AC', then adding a type with feature pair 'AD' decreases the uniqueness of the first type despite the fact that the proportion of unique features remains 50% (feature 'B' is the unique feature before and after adding the type). Intuitively, suppose that type 2 is an organism that has a certain characteristic which is caused by its genetic code, and the feature pair of a species represent its genetic code. Knowing whether or not type 3 has this characteristic as well decreases the uncertainty with respect to type 1 as follows. If type 3 does not have it, then feature 'A' is less likely to be the cause and type 1 is less likely to have it. On the other hand, if type 3 has it, then feature 'A' is more likely the cause and type 1 is more likely to have it. Type 3 thus provides additional information and therefore it further reduces the uniqueness.

A last limitation of our framework is that it lacks a similarity-based rarity index. The rarity of a species is high when its features are rare within the group. This rarity index can improve our framework in several ways. First, the rarity is interesting by itself. It can, for example, be used as an indicator of the conservation priority of a species. It can also be used to do kernel-based novelty detection, i.e., identifying data that is new in the sense that not much similar data occurs in the training set. For more information about novelty detection and its applications, we refer the reader to [1]. Second, it can give more insight in the behavior of the diversity. For example, it would be nice if adding an organism to a group increases the diversity if and only if the rarity of that organism exceeds the average rarity. Note that in a similarity-based framework equally rare is different from equally abundant.

A technical challenge that we would like to address in the future, is replacing the gradient ascent algorithm that we use to maximize the OLC entropy by a more scalable algorithm. Currently we can handle a few thousand datapoints on a normal PC and it would be nice if we could handle a few million datapoints instead. However, creating such an algorithm is out of our league at the moment because it requires specialised knowledge about optimization which we do not possess. Our hope is that a specialist in the field of optimization creates such an algorithm so that we can use it. In ecology not many data sets exceed a few thousand datapoints but in other applications (e.g., fraud detection of credit card holders) datasets containing millions of datapoints are not that uncommon.

Although we focus on biodiversity in this thesis, we want to emphasise that our framework is suited for any kind of diversity. It can, for example, also be used to quantify the ethnic, cultural, linguistic and religious diversity in different countries [2, 3]. Currently fractionalization is mostly used to quantify this kind of diversity. Fractionalization expresses the probability that two randomly drawn individuals have a different type. It has two important disadvantages. First, it does not take into account the similarity between the different types (e.g., some religions differ more than others or sometimes a person has a mixed race). Second, it gives relatively a lot of weight to common types. For instance, if one type accounts for 50% of the population, then it doesn't matter whether the remaining 50% consist of 50 or 50000 equally abundant types, the fractionalization remains approximately 0.25. The Shannon OLC diversity, on the other hand, assigns considerably different diversities to both cases. Also, our proposed alpha and beta diversity can be used to quantify how the different types are distributed over the subgroups. These subgroups can be cities within a county or neighbourhoods within a city in the case of ethnic diversity. A high beta diversity and a low alpha diversity means that the composition of different subgroups differs a lot but the diversity within each subgroup is low (e.g., one neighbourhood mainly consists of African-American people, another neighbourhood mainly consists of Hispanic people, etc.). Locally the diversity is thus low, but globally the diversity is high. A low beta and a

high alpha diversity means that there are a lot of different types within each subgroup but all subgroups contain more or less the same proportion of each type (e.g., each neighbourhood has more or less 15% Hispanic, 30% white, 35% African-American and 20% Asian people).

5.2.2 Predicting QoS

We introduced a two-step approach to estimate QoS values of composite services. These QoS prediction algorithms have the following disadvantages. First, when making small changes to the composition (e.g., when one individual service is changed), the kernel-based quantile estimator needs to be retrained completely. In the context of service selection this is a major disadvantage because the complexity of the prediction algorithm is the bottle neck and small changes are often made. It is therefore often preferable to use algorithms that are less accurate but easier to update. Nevertheless, as a final step we can still use our algorithm to get a more accurate view on whether the SLA is likely to be met or not. Second, the prediction algorithms perform poorly when the candidate services act close to random or when events occur that did not happen in the training data. Other prediction algorithms also perform poorly in these situations but often it is better to do no prediction at all than to do bad prediction - especially in terms of computational overhead. Third, we did not fully automate the conversion of the composition choice into the predicted QoS values. Some steps of the algorithm still involve human interaction (e.g., the choice of the inputs of the prediction algorithm) which is preferably not the case when used in actual service prediction software.

The following additions would improve the framework: (1) an algorithm that chooses the prediction algorithm based on the prediction task (i.e., either an accurate algorithm with a high computational overhead or less accurate algorithm with a low computational overhead is chosen); (2) a method that designs the inputs depending on the predictions task (e.g., which historical values do we use as an input in the case of time series prediction), and (3) support for other technologies and standards (e.g., it would be nice if the framework would also support the Business Process Model and Notation (BPMN) 2.0).

5.2.3 Reducing the run time complexity of SVM Classifiers

We presented two algorithms, SCA and SSCA, that reduce the run-time complexity of SVM classifiers. These methods have two important limitations: they don't reduce the training complexity and they can't be used for SVM regression. We did not find an obvious way to overcome these limitations. To decide which training datapoints need to be flipped or removed, we need an accurate solution to start from and therefore we need another algorithm to

provide this solution. SCA and SSCA are ineffective when applied to regressors because the position of a data point with respect to the separating hyperplane - which is what SCA and SSCA approximate - does not determine the value of the predicted label, it only determines its sign.

References

[1] Pimentel, M. A. F., D. A. Clifton, L. Clifton, and L. Tarassenko. "A review of novelty detection." Signal Processing 99, pp. 215-249, 2014.

[2] Alesina, A., E. La Ferrara. "Ethnic Diversity and Economic Performance". Journal of Economic Literature 43, pp. 762-800, 2004.

[3] Patsiurko, N., J. L. Campbell, and J.A. Hall. "Measuring cultural diversity: ethnic, linguistic and religious fractionalization in the OECD". Ethnic and Racial Studies 35(2), pp. 195-217, 2012.

List of publications

Published International Journal Articles

- Geebelen, Dries, Johan A.K. Suykens, and Joos Vandewalle. "Reducing the number of support vectors of SVM classifiers using the smoothed separable case approximation." IEEE Transactions on Neural Networks and Learning Systems, vol. 23, no. 4, Apr. 2012, pp. 682-688.
- Geebelen, Dries, Kristof Geebelen, Eddy Truyen, Sam Michiels, Johan A.K. Suykens, Joos Vandewalle, and Wouter Joosen. "QoS prediction for web service compositions using kernel-based quantile estimation with online adaptation of the constant offset." Information Sciences, vol. 268, 2014, pp. 397-424.

Unpublished International Journal Articles

- Geebelen, Dries, Luc De Meester, Johan A.K. Suykens, and Joos Vandewalle. "Extending true diversity, richness and derived indices to a similarity-based framework using optimal lossy compression", submitted to Ecological Monographs.

International Conference Articles

- Geebelen, Dries, Kim Batselier, Philippe Dreesen, Marco Signoretto, Johan Suykens, Bart De Moor, and Joos Vandewalle. "Joint regression and linear combination of time series for optimal prediction." European Symposium on Artificial Neural Networks (ESANN), Bruges, Belgium, Apr. 2012.