Overcoming the Curse of Missing and Noisy Data in Computational Drug Design

Fanwang Meng

Department of Chemistry and Chemical Biology
McMaster University

This dissertation is submitted for the degree of
Doctor of Philosophy

DOCTOR OF PHILOSOPHY (2021) McMaster University, Chemistry, Hamilton, Ontario, Canada

- TITLE: Overcoming the Curse of Missing and Noisy Data in Computational Drug Design
- AUTHOR: Fanwang Meng
- SUPERVISOR: Professor Paul W. Ayers
- Number of Pages: xxxvi, 266

To people I love.

Declaration

I declare that this thesis is an original scientific summary of my studies in McMaster University and I composed all the chapters in this thesis. All the work was done by me and outcomes of collaborations are explicitly specified in the Acknowledgement section and in the main text. All the scientific references are listed clearly in the text. This thesis has not been submitted for any other degree or professional qualification.

Fanwang Meng
January 2022

Acknowledgements

Eighteen years ago, I was sitting in front of a white-black TV staring at a documentary named *World Famous Universities* in a small village. I was inspired by the scholars who reshaped the world with science and technology.

Thirteen years ago, I was studying Chinese medicine (focusing on natural products) in Nanjing.

Now, I am finishing up my Ph.D. thesis, in a Hamilton, Canada.

By looking back, it is such a long way and I have so many stories to tell. I would like to express my greatest gratitude to a lot of people, some of whom I know very well, some others I don't.

Special thanks go to Dr. Paul W. Ayers. This gentle, nice and handsome guy helped me out by admitting me as his Ph.D. student when I was frustrated and a little desperate in my third year of my master study. I came to this lab without any mathematical education background (just high school level calculus more precisely). Paul will walk me through all the equations one by one, helping me to get a sense of how to formulate a real-world problem into a mathematical problem. I am very grateful to Paul's support of helping me to explore scattered research topics, which I enjoyed a lot. He is a very intelligent, hard-working, but humble scholar. He is more than a supervisor to me, more like a good friend and a big brother. His devotion to science also has inspired me to enjoy doing science.

I also would like to express my thanks to Dr. Giuseppe Melacini, Dr. Randy Dumont for being my committee meeting members. They are very supportive and very tolerable to my slow progress. I took Dr. Melacini's NMR course in my first year, I have many questions because the lack of practical experimental experiences. He sacrificed his valuable time, sitting down with me to helping me to enjoy the beauty of NMR. The way he elaborate scientific problems gave me an impression and the allostery prediction project is an outcome of discussions with him. Dr. Dumont is always supportive and encouraging. I was a research assistant with Dr. Dumont for 3 times and I was impressed by his logic of delivering scientific concepts by dividing them into smaller pieces. When we are discussing new ideas or new research directions, he is always listening attentively.

I would also like to thank the collaborators. Dr. Farnaz Heidar-Zadeh (Queens University) introduced the topic of matrix completion, which inspired to write a review (Chapter 1) on it highlighting its applications in drug design. The allostery prediction (Chapter 2) was initiated after I had a discussion with Dr. Giuseppe Melacini and all the NMR data were generously provided by his lab. I also get helpful suggestions from Hebatallah Mohamed, Jinfeng Huang from his lab. Valerii Chuiko helped with clustering section of allostery prediction.

The Δ-learning based solubility prediction (Chapter 3) benefited from the teamwork with two undergraduate MITACS scholars, Hanwen Zhang from Sun Yat-sen University, China and Juan Samuel Collins-Ramirez from Tecnologico de Monterrey (Mexico) and they helped with the model constructions. `Procrustes` is a long run and I get a lot help from: Dr. Farnaz Heidar-Zadeh, Dr. Taewon David Kim, Michael Richer, Alireza Tehrani and Jonathan La. This project is also a starting point of my long, slow journey of learning programming. They not only help with programming the numerical algorithms, but also also taught me how to build a software/package, which includes version control, documentation, re-factorization *etc.* The BBB prediction project started a year ago. Jinfeng Huang, Yang Xi helped with molecule structure validation. Juan Samuel Collins-Ramirez provided a lot of help with running the classification models and performance evaluations.

Our department is not a giant one, but I enjoy working and talking with people here. Just to list a few people that helped me move froward, Dr. Gillian Goward, Dr. David Emslie, Dr. Peter Kruse, Dr. Jose Moran-Mirabal, Dr. Linda Davis, and, Christine Cosgrove the main office. In addition, Dr. Doug Welch pointed my way to McMaster and I do appreciate his help.

My friends helped me survived my Ph.D. studies. They are good listeners and they can always give you a hand when needed. These nice people include Jinfeg Huang, Shuo Feng, Ri Chen, Xueli Zhao, Qiulin Ma, Pengxiao Zhou, Dr. Yiguo Sun, Chunhua Wu and her lovely kids, Huawei Zhu, Xiang Song, Mingjun Liu, Chengsheng Wang, Fenglei Zhou, Jinlong Li, Wei Gu, Dr. Feng Tan, Dr. Cheng Luo and Dr. Wencong Lu, Dr. Zhongjie Liang, Dr. Jihui Zhao, Mr. Liang Liu, Dr. Shasha Han, Ms Jinxia Liu and many others. I would also like to thank all my friends from MERGE program, Denise Newman, Drew Smith, Julia Verhaeghe, Sofia Cementina, *et al.* and all the classmates. This is such an unforgettable experience and I enjoyed the atmosphere of having different opinions but still respecting each other. Dr. Yuping Tang is always being supportive and his practical suggestions helped a lot.

Last but not least, credit should go to my family who helped me grow up and help me walk out of the small village, to Nanjing, Shanghai and then fly to Hamilton, Canada across the Pacific ocean.

Abstract

Machine learning (ML) has enjoyed great success in chemistry and drug design, from designing synthetic pathways, to drug screening, to biomolecular property predictions, *etc.*. However, ML models generalizability and robustness requires the high-quality training data, which is often difficult to obtain, especially when the training data is acquired from experimental measurements. While one can always discard all data associated with noisy and/or missing values, this often results to discarding invaluable data.

This thesis presents and applies mathematical techniques to solve this problem, and applies them to problems in molecular medicinal chemistry. In chapter 1, we indicate that the missing-data problem can be expressed as a matrix completion problem, and we point out how frequently matrix completion problems arise in (bio)chemical problems. Next, we use matrix completion to impute the missing values in protein-NMR data, and use this as a stepping-stone for understanding protein allostery in Chapter 2. This chapter also used several other techniques from statistical data analysis and machine learning, including denoising (from robust principle component analysis), latent feature identification from singular-value decomposition, and residue clustering by a Gaussian mixture model.

In chapter 3, Δ-learning was used to predict free energies of hydration (ΔG). The aim of this study is to correct estimated hydration energies from low-level quantum chemistry calculations using continuum solvation models without significant additional computation. Extensive feature engineering, with 8 different regression algorithms and with Gaussian process regression (38 different kernels) were used to construct the predictive models. The optimal model gives us MAE of 0.6249 kcal/mol and RMSE of 1.0164 kcal/mol. Chapter 4 provides an open-source computational tool `Procrustes` to find the maximum similarities between metrics. Some examples are also given to show how to use `Procrustes` for chemical and biological problems. Finally, in Chapters 5 and 6, a database for permeability of the blood-brain barrier (BBB) was curated, and combined with resampling strategies to form predictive models. The resulting models have promising performance, and are released along with a computational tool `B3clf` for its evaluation.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Acronyms / Abbreviations

ADASYN adaptive synthetic

ADM alternating direction method

ALM augmented Lagrangian multiplier

ALS alternating least squares

ATC anatomical therapeutic chemical

AUC area under the curve

BACC balanced accuracy score

BBB blood-brain barrier

BCV Bi-cross-validation

BMPM biased minimax probability machine

BNNR bounded nuclear norm regularization

BP basis pursuit

CAR-T chimeric antigen receptor–modified T

CART classification and regression trees

CCLE Cell Line Encyclopedia

CD circular dichroism

CF collaborative filtering

CHESCA chemical shift covariance analysis

CID compound identifier

CMF Collective matrix factorization

CNS central nervous system

DBSMOTE density-based SMOTEE

DFT density-functional theory

DLGRMC dual Laplacian graph regularized matrix completion

DMCCDA double matrix completion for predicting the circRNA-disease association

DREAM dialogue on reverse engineering assessment and methods

DRRS drug repositioning recommendation system

DTI decision tree induction

EDA exploratory data analysis

EDM Euclidian distance matrix

EPAC exchange proteins directly activated by cAMP

FEP free energy perturbation

FFDNN feed-forward artificial neural network

FGRMF feature-derived graph regularized matrix factorization method

FID free induction decay

FN false negative

FRET fluorescence resonance energy transfer

gaff Generalized Amber Force Field

GAN generative adversarial networks

GDSC genomics of drug sensitivity in cancer

GEOM geometric mean score

GLM generalized linear model

GO gene ontology

GPA generalized Procrustes analysis

GPR Gaussian process regression

GR1BMC graph regularized 1-bit matrix compeltion

GWAS genome-wide association studies

HCN hyperpolarization-activated cyclic nucleotide-gated

HCS high content screening

HGIMC heterogeneous graph inference with matrix completion

HTS high-throughput screening

HVMC harmonic variety-based matrix completion

IALM inexact augmented Lagrange multiplier

ILRMR improved low-rank matrix recovery

IMCMDA inductive matrix completion for miRNA-disease association prediction

IMDAILM inferring miRNA-disease association by integrating lncRNA and miRNA data

IMDN improved prediction of miRNA-disease associations

InChI International Chemical Identifier

IR imbalance ratio

KL Kullback-Leibler

KNN k-nearest neighbors

KNR k-nearest neighbors regression

KPCA kernel PCA

KRR kernel ridge regression

Lasso linear regression with L_1 prior as regularizer

LightGBM Light Gradient Boosting Machine

LMC linear matrix completion

LR linear regression

MAE mean absolute error

MAR missing at random

MCAR missing completely at random

MCC Matthews correlation coefficient

MD Molecular dynamics

MDFP molecular dynamics fingerprints

MDL minimum description length

MF matrix factorization

ML machine learning

MLPCM machine-learning polarizable continuum solvation model

MLR multiple linear regression

MLRE multi-wiew low rank embedding

MNAR missing not at random

MRMF manifold regularized matrix factorization

MRMR minimum redundancy maximum relevance

MSM Markov state model

NCMC neighborhood constraint matrix completion

NIMCGCN neural inductive matrix completion with graph convolutional network

NLMC non-linear matrix completion

NMF negative matrix factorization

NMR negative matrix factorization

NUS Non-uniform sampling

OMC overlap matrix completion

PCA principal component analysis

PCP principal component pursuit

PDB Protein Data Bank

PDF portable document format

PES potential energy surfaces

PMF probabilistic matrix factorization

PPIs protein-protein interactions

ProWL prediction method with weak-label learning

QSAR quantitative structure activity relationship

RBM restricted Boltzmann machine

RDMC robust discrete matrix completion

RF random forests

RIP restricted isometry property

RMSD root-mean-square deviation

RMSE root mean squared error

RNN recurrent neural network

ROC receiver operating characteristic

RPCA robust principal component analysis

ScaledASD scaled alternating steepest descent

SGD stochastic gradient descent

SGIMC sparse-group inductive matrix completion

SHAP shapley additive explanations

SL synthetic lethality

SMBO sequential model-based global optimization

SMD solvation model

SMILES simplified molecular-input line-entry system

SMOTE synthetic minority oversampling technique

SNF similarity network fusion

SPARSE single point array reconstruction by spatial encoding

SPLR subspace pursuit low-rank

std standard deviation

SVD singular value decomposition

SVM support vector machine

SVT singular value thresholding

t-SNE t-distributed stochastic neighbor embedding

TN true negative

TP true positive

TPE tree of Parzen estimators

tPSA topological polar surface area

TSPP two-sided permutation Procrustes

Chapter 1

Matrix Completion for Computational Drug Design: A Review

1.1 Introduction

Recommender systems (also known as recommendation systems) provide personalized recommendations given the explicit or implicit information collected from the user side [1, 2], and have enjoyed great success in e-commerce [3, 4], online learning[4], social network prediction [5], healthcare [6], and molecular science [7–10]. As defined by Resnick and Varian: *In a typical recommender system people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients. In some cases the primary transformation is in the aggregation; in others the system's value lies in its ability to make good matches between the recommenders and those seeking recommendations.* [11].

A recommender system deals with two types of data, attribute information about users and items (such as movie genres and book contents/topics) and user-item interactions (such as movie ratings or shopping records). Content-based recommender methods use attribute data while collaborative filtering methods use interaction data; knowledge-based recommender systems are built to explicitly meet the requirements of the user. Hybrid recommender systems combine information and structure from all these methods. More information about different types of recommender systems can be found in [12]. Here we focus on matrix completion, a subset of collaborative filtering.

A fundamental assumption of collaborative filtering (CF) is that if two users A and B share similar rating patterns or similar behaviours over n items, their ratings/behaviours toward other items will be similar too [13, 14], which makes it possible to predict a user's response to a new item. There are two main categories of models in CF: the neighbourhood approach [15, 16] and latent factor models [17–20]. Neighborhood models highlight the similarities between users or items [21, 22] which can further be categorized into user-oriented methods [23] and item-oriented methods [24]. Latent factor models attempt to identify holistic latent features that explain observed ratings [21]. Latent factor models have been widely used in recommender systems [17, 18], and gained prominence after their success in the NetFlix Prize competition in 2006 [25]. The main idea of latent factor models is to factorize the ratings matrix into a user-latent-matrix (the user embedding) and an item-latent-matrix (the item embedding). Predictions are obtained from the product of these latent-factor matrices [25]. Hybrid models combine latent-factor matrices with side-information about the similarity of users/items [26].

A prototypical example of CF is the Netflix Prize competition. The goal of this competition was to build a recommender system using a dataset of 100,480,507 movie ratings from 480,189 thousand subscribers on 17,770 movies; the ratings matrix therefore had a sparsity of 98.82% [27]. This highlights the challenge inherent in recommender systems: one wishes to predict a full matrix given only a few elements thereof. Matrix factorization techniques for imputing the missing values stood out for their performance on the Netflix challenge[28, 29] establishing the utility of the matrix completion perspective for CF. Since then, matrix completion algorithms have received increasing attention from both academia and industry.

While matrix completion has had huge successes in e-commerce [30], link prediction of social networks [31], signal processing [32], and image processing [33], the pharmaceutical community has not fully adopted matrix completion algorithms. The goal of this review is to explain why matrix completion is a useful tool for drug discovery/design and computational chemistry. Specifically, we will (1) introduce matrix completion for computational drug design; (2) review algorithms for matrix completion; (3) discuss how matrix completion can be used for pharmaceutical applications, with the aim of inspiring computer scientists to design bespoke algorithms for the drug design domain.

We start this review with a fundamental introduction to *matrix completion methods*, including mathematical notation, key concepts and ideas, and different categories of matrix-completion algorithms. (Cf. section 1.2.) We will also summarize the leading software packages for matrix completion problems. Then

we address *matrix completion in drug design* in section 1.3, which highlights the power and potential of matrix completion algorithms in the pharmaceutical sciences. To help the pharmaceutical community to use matrix completion more broadly, the formulation of drug-design-related tasks as matrix completion problems is presented in section 1.4. A brief discussion of how to perform model selection is given in section 1.5. Finally, some analysis of the current state of the field and prospects for future research and development is given in section 1.6.

1.2 Matrix Completion Methods

1.2.1 Definition of Matrix Completion

The Netflix challenge was to recommend movies to a user based on the movie-watching history of that user and other users. To express this problem mathematically, a rating matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ is constructed, with element $\mathbf{X}_{ij}$ denoting the rating of movie j by user i. The rating matrix $\mathbf{X}$ is obviously incomplete and sparse as even the most avid cinephile can only watch a tiny fraction of Netflix's offerings. The Netflix challenge is thus posed as a matrix completion problem, where the goal is to approximate the full rating matrix $\mathbf{X}$ for m users and n movies based on the limited data available.

One approach to the matrix completion problem is to find a low rank matrix $\mathbf{M}$ with minimum Frobenius distance to the (incomplete) matrix $mathbf X$:

$$\begin{aligned} &\min && \|\mathbf{M} - \mathbf{X}\|_F^2 \\ &\text{subject to} && \text{rank}\,(\mathbf{M}) \leq c \end{aligned} \tag{1.1}$$

The Frobenius distance between matrices with missing elements includes only those elements for which the values are known,

$$\|\mathbf{M} - \mathbf{X}\|_F^2 = \sum_{\{i,j\} \in \Omega} \left(m_{ij} - x_{ij} \right)^2 \tag{1.2}$$

where Ω denotes the indices for which elements of both $\mathbf{M}$ and $\mathbf{X}$ are known.

1.2.2 Compressed Sensing, Matrix Recovery and Matrix Completion

Matrix completion is a natural extension of compressed sensing [34] which seeks to reconstruct the signal vectors by assuming sparsity and incoherence [35, 36]. In this

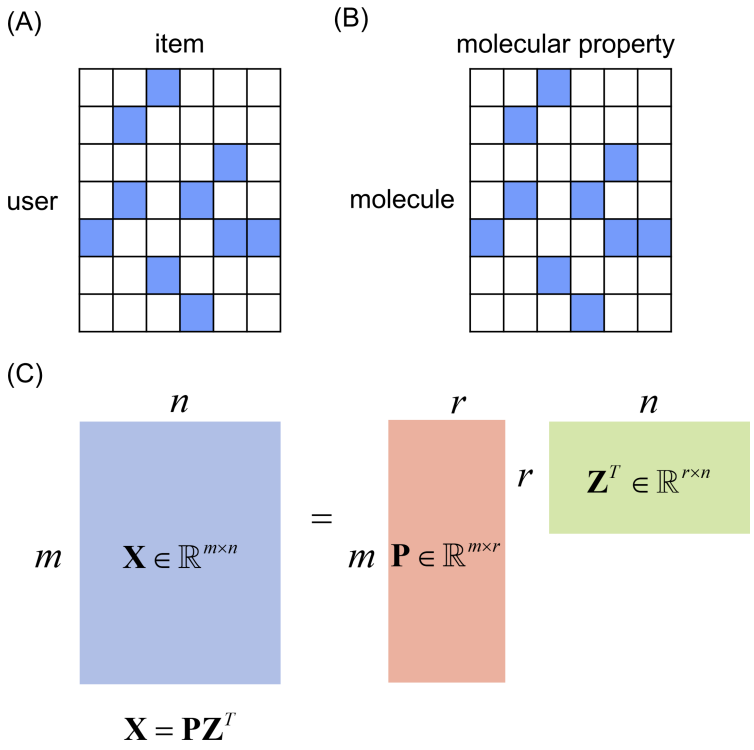


Fig. 1.1 Matrix completion for movie recommendations and molecular property predictions along with matrix factorization techniques.

context, sparsity captures the fact that most real-word signals are underdetermined linear systems, so only some specific domains are significant [37]. This is analogous to the low-rank assumption in subsection 1.2.4. The other hypothesis, incoherence, indicates that the singular vectors cannot be too spiky [38–40].

To express this problem mathematically, one wishes to recover the original signal x_0 from the observed data y given $y = \mathbf{A}x$ [41, 42]. Here $\mathbf{A} \in \mathbf{R}^{m \times n}$ is the (known) coding matrix ($m \ll n$). The problem now is formulated as

$$\begin{aligned} \min_{x} \quad & \|x\|_0 \\ \text{subject to} \quad & y = \mathbf{A}x \end{aligned} \tag{1.3}$$

where $\|x\|_0$ is ℓ_0 norm. However, this is a NP-hard problem [43–45]. Fortunately, Candès, Romberg, Tao and *et al.* proved that under reasonable conditions, the restricted isometry property (RIP) in compressed sensing lets one rephrase this intractable non-convex ℓ_0 optimization problem with a tractable convex ℓ_1 optimization problem [41, 46]. Specifically, Equation 1.3 can be rewritten as

$$\begin{aligned} \min_{x} \quad & \|x\|_1 \\ \text{subject to} \quad & y = \mathbf{A}x \end{aligned} \tag{1.4}$$

This is a basis pursuit (BP) problem and can be solved easily with linear programming [47].

Given the great success of RIP in solving compressed sensing problems, it is very natural to extend vector recovery in compressed sensing to low-rank *matrix recovery* with limited linear measurements [45, 48]. The special case of matrix recovery where specific elements of the incomplete matrix are measured is called *matrix completion*, as shown in Figure 1.2(A).

Mathematically, the matrix recovery problem is to decompose a sparse and corrupted matrix $\mathbf{D}$ as the sum of a low-rank matrix $\mathbf{L}$ and a sparse and noisy matrix $\mathbf{E}$.

$$\begin{aligned} \min_{\mathbf{A},\mathbf{E}} \quad & \text{rank}(\mathbf{L}) + \lambda \|\mathbf{E}\|_0 \\ \text{subject to} \quad & \mathbf{D} = \mathbf{L} + \mathbf{E} \end{aligned} \tag{1.5}$$

where $\mathbf{D}$ is the observed data matrix, $\mathbf{L}$ is the low-rank matrix, $\mathbf{E}$ is the sparse matrix including noise, and λ is the regularizer [42, 49]. This problem is hard to solve because it is non-linear and non-convex. However, in analogy to the use of the RIP for convex relaxation in compressed sensing, matrix recovery can be solved using principal component pursuit (PCP): [49]

$$\begin{aligned} \min_{\mathbf{L},\mathbf{E}} \quad & \text{rank}(\mathbf{L}) + \lambda \|\mathbf{E}\|_1 \\ \text{subject to} \quad & \mathbf{D} = \mathbf{L} + \mathbf{E} \end{aligned} \tag{1.6}$$

where $\|\mathbf{E}\|_1$ denotes ℓ_1 norm. For related algorithms, see [50, 51].

Given the close relationship between compressed sensing, matrix recovery, and matrix completion, it is not surprising that similar mathematical and numerical strategies are used to solve these problems. For example, a nuclear-norm relaxation is used for all three problems [52].

1.2.3 A Different Perspective: Matrix Completion and Missing Value Imputation in Statistical Analysis

Missing data is common problem in data-oriented disciplines, but the primary target of this review is method for adapting the mathematical framework of matrix completion to pharmaceutical research [53]. For example, certain properties (e.g. IC_{50}) of certain compounds can be missing from a dataset, perhaps because the compound was unavailable or because there were insufficient screening assays. Similarly,

participants drop out of clinical trials with such frequency that the FDA has issued guidelines on how to address the associated missing-value problem [54, 54, 55].

The simplest strategy is to remove the missing data by deleting the entire row or column containing it, but this discards potentially valuable information and can bias statistical predictions [56]. The second strategy is single imputation, where the missing data is replaced by the group means, medians or modes, or even by the results from a regression model. Most sophisticated is multiple imputation, where a statistical model for the missing values is constructed, then used to impute the missing values n times. This is clearly preferable to deletion or single imputation. For more statistical analysis and algorithms for missing value imputation, see [57–60].

To build a statistical model for the missing values, it is important to understand whether the unspecified values are missing at random (MAR), missing completely at random (MCAR), or missing not at random (MNAR) [59–62]. MAR means that the missingness arises from the observed data, so that there can be some correlation between which elements are observed and which are missing. In such cases, it is plausible that missing data values can be predicted by other variables in the data. MCAR means that the missing values arise as a pure random sample of the complete data. This is a more restrictive condition than MAR. The soft-impute algorithm for low-rank matrix completion [63] achieves similar accuracy for both MCAR and MAR [64, 65], even though the uniform sampling assumption assumes that observed matrix entries are independently sampled, and thus MCAR[64].

All other cases are categorized as MNAR. In MNAR, the probability of having missing data is related to the missing data, which makes MNAR very difficult to impute. MNAR is ubiquitous when data is missing because determination of values was too difficult to be feasible experimentally/computationally in some cases, or where data is missing because its value falls outside the dynamic range of the measuring apparatus. However, matrix completion methods can be used to fill missing values even in cases where data is MNAR [66–68].

1.2.4 Fundamental Assumptions of Matrix Completion

When mathematically developing methods for the matrix completion problem, it is useful to assume that the matrix is incoherent, low rank, and uniformly sampled. With these assumptions, one can build a robust mathematical framework for matrix completion that is valid even when the number of known values in the matrix is asymptotically small[69–71]. Elaborating on these three features:

(i) Uniform sampling of observed entries. It is often assumed that the elements of $\mathbf{X}$ are sampled uniformly and randomly. Quantitatively, Bernoulli sampling is often used, where each x_{ij} is included in $\mathbf{X}$ with probability p [34]. Another simplification is that sampling with replacement [72].

(ii) Low-rank. The target matrix $\mathbf{M}$ has rank r, and this rank does not depend on the data provided. By assuming the matrix has low rank, the degrees of freedom are restricted to be smaller than the amount of data available. Missing elements in low-rank matrices can be predicted with high confidence when the sparsity of the data and the rank of the target matrix are low enough, and the data sampling scheme is appropriate [34, 73–76].

The theory of matrix completion suggests that a matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ with rank r where $m \leq n$ can be recovered if and only if the number of the observed samples (with uniformly sampling at random) is greater than $Crm\log^2(n)$ where C is a constant factor [69]. Moreover, $rm\log m$ sampled entries with uniformly distribution are required to ensure that it is highly probable that there is at least one entry in each row and column [72].

(iii) Incoherence. The assumption that both the row and column space of the target matrix $\mathbf{M}$ are incoherent is inspired by the link to compressed sensing. Incoherence ensures that the (non-vanishing) singular values of matrix $\mathbf{M}$ have similar magnitude. Joint incoherence is not required [77].

1.2.5 Algorithms for Matrix Completion

There are two major types of linear matrix completion methods: matrix factorization methods and rank minimization methods [73]. Though we will focus on the basic principles and state-of-the-art for these two methods, we will also overview non-linear matrix completion algorithms.

1.2.5.1 Rank Minimization Methods

Rank Minimization Given a matrix $\mathbf{X} \in \mathbb{R}^{n\times n}$ with rank r, we aim to recover the missing elements in $\mathbf{X}$. The number of degrees of freedom is $n^2-(n-r)^2 = (2n-r)r$, which is much less than n^2 for a low-rank matrix. Therefore we can recover the matrix perfectly by sampling less than n^2 elements [74]. More explicitly, the $2nr-r^2$ degrees of freedom can be determined by constructing the singular value decomposition (SVD) of $\mathbf{X}$.

Candes showed that the elements of a low-rank matrix can be determined by optimizing:[74]

$$\begin{aligned} \min_{\mathbf{X}} \quad & \operatorname{rank}(\mathbf{X}) \\ \text{subject to} \quad & \mathbf{X}_{ij} = \mathbf{M}_{ij}, \ (i,j) \in \Omega, \end{aligned} \tag{1.7}$$

where Ω denotes the set of $\{i, j\}$ values for which the element $M_{i,j}$ is known. While the minimizing the rank of a matrix subject to convex constraints is NP-hard [78], there is a popular heuristic algorithm based on minimizing the trace of a positive-definite decision variable [79].

Nuclear Norm Minimization Fazel [80, 81] generalized this strategy to rectangular matrices by using the nuclear norm. Suppose we have a rank r matrix $\mathbf{X}$; $\mathbf{X}$ therefore has r positive singular values, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. The nuclear norm of $\mathbf{X}$ is defined as the sum of its singular values,

$$\|\mathbf{X}\|_* = \sum_{i=1}^{r} \sigma_i(\mathbf{X}) \tag{1.8}$$

Similarly, operator/spectral norm can be defined as the largest singular value of $\mathbf{X}$

$$\|\mathbf{X}\|_2 = \sigma_1(\mathbf{X}) \tag{1.9}$$

The matrix completion problem can then be formulated as

$$\begin{aligned} \min \quad & \|\mathbf{X}\|_* \\ \text{subject to} \quad & \mathbf{X}_{ij} = \mathbf{M}_{ij}, \ (i,j) \in \mathbf{\Omega}, \end{aligned} \tag{1.10}$$

whenever the number of entries in the partially-observed matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$ is greater than or equal to $Cn^{\frac{5}{6}} r \log(n)$, this problems has a unique solution. Here C is a positive constant, and it is assumed that the entries in $\mathbf{M}$ are randomly selected and that $m \leq n$.

The nuclear norm, alternatively named the Ky Fan norm, Schatten 1-norm and the trace class norm [82], is convex.[73] Therefore problem in Equation 1.10 can be efficiently numerically solved by standard methods, including semidefinite programming.

The parallelism between rank minimization and sparse approximation arises because the nuclear norm equals the ℓ_1 norm of the vector of singular values. Therefore, many mathematical properties and practical benefits of ℓ_1 (vector) optimization (in, e.g., compressed sensing) extend to nuclear-norm optimization in matrix completion.[34, 69, 82, 82–87].

When only a few singular values are believed to contain important information, a truncated nuclear norm can be used [88].

$$\begin{aligned} \min \quad & \|\mathbf{X}\|_r \\ \text{subject to} \quad & \mathbf{X}_{ij} = \mathbf{M}_{ij},\ (i,j) \in \mathbf{\Omega} \end{aligned} \tag{1.11}$$

The goal of using the truncated nuclear norm, $\|\mathbf{X}\|_r = \|\mathbf{X}\|_* - \sum_{i=1}^{r} \sigma_i(\mathbf{X}) = \sum_{i=r+1}^{\min(m,n)} \sigma_i(\mathbf{X})$, is to minimize the importance of all but the r-largest singular vectors. Numerical experiments show that the truncated nuclear norm is preferable to non-smooth formulations of the rank minimization problem.

Missing matrix entries can be reconstructed by nuclear-norm minimization if the RIP is appropriately satisfied [89, 90]. In this case, the (easy) convex nuclear norm optimization problem is equivalent to the (hard) rank minimization problem, as the two objective functions have the same unique solution [34]. This strategy supersedes earlier attempts at convex relaxations of Equation 1.7 [34, 69, 72].

Schatten p-Norm Minimization Improved numerical performance can be obtained by generalizing the nuclear-norm minimization approach to the Schatten p-norm [91]

$$\|\mathbf{X}\|_{S_p} = \left(\sum_{i=1}^{\min\{n,m\}} \sigma_i^p \right)^{\frac{1}{p}} \tag{1.12}$$

where σ_i is the i^{th}-largest singular value of matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ and $0 < p < \infty$. When $p = 1$, this is the nuclear norm (cf. Equation 1.8) and when $p \to \infty$ this is the spectral norm (cf. Equation 1.9). Using the Schatten p-norm, the matrix recovery problem can be stated as:[92]

$$\begin{aligned} \min \quad & \|\mathbf{X}\|_{S_p}^p \\ \text{subject to} \quad & \mathbf{X}_{ij} = \mathbf{M}_{ij},\ (i,j) \in \mathbf{\Omega}, \end{aligned} \tag{1.13}$$

This algorithm converges if $p \in (0, 2]$, and the numerical performance is better than nuclear norm minimization one when $0 < p < 1$ [91]. Better approximations result when p is closer to zero. Chen *et al.* explored the relationship between the restricted isometry constant [41, 93], δ_{4r} and p, and proposed that sufficient conditions for exact matrix recovery are $\frac{\sqrt{3}}{2} \leq \delta_{4r} < 1$ and $0 < p < 2 \times (1 - \delta_{4r})$ [94].

Schatten p-norm-optimization (and other related norms) are frequently used in matrix completion [92, 95]. For example, the Schatten p-norm can be adapted to

problems where features have different relevance by weighting the singular values. This gives rise to the weighted Schatten p-norm [96],

$$\|\mathbf{X}\|_{w,S_p} = \left(\sum_{i=1}^{\min(m,n)} w_i {\sigma_i}^p \right)^{\frac{1}{p}} \tag{1.14}$$

where $0 \leq w_i$ is the weight of the i-th singular value. Numerical experiments suggest that weighted Schatten p-norms are especially useful when $\mathbf{M}$ contains noisy data.

After the matrix completion problem has been recast as a Schatten p-norm optimization problem, it can be solved with augmented Lagrangian multiplier (ALM) methods [97] and the majorization minimization algorithm [90]. The the ℓ_1-norm ($p = 1$) is usually a robust choice for the loss function.

Singular Value Thresholding Algorithm Several algorithms for matrix recovery via nuclear norm minimization have been developed, including singular value thresholding (SVT) [98], the alternating direction method (ADM) [99], and the inexact augmented Lagrange multiplier (IALM) method. Interior point solvers like SDTP3 and SeDuMi are advantageous because one can rigorously prove that they converge to the solution of the matrix recovery problem [83, 100, 101]. However, their cost grows rapidly with the size of the matrix, because the number of linear constraints is equal to the number of available elements of $\mathbf{M}$.

The SVT algorithm proposed by Cai *et. al* [98] has high computational efficiency and is applicable to huge matrices. The basic idea of SVT is to use a shrinkage operator $\mathcal{D}_\tau$ which shrinks singular values below a threshold to zero. More details can be found in [98, 102].

Because computing the SVD can be very slow when the rank of the matrix is not much smaller than its dimension (i.e., $\mathrm{r}(M) \ll \dim(M)$. Cai proposed using Newton iteration instead of SVD to calculate $D_\tau(Y)$ [103]. Using similar matrix shrinkage strategy along with a matrix factorization strategy we shall introduce in the next section, a fixed-point continuation with approximate SVD (FPCA) algorithm was proposed to solve the matrix completion problem [104]. According to their numerical experiments, FPCA achieves better accuracy and takes less computational time than SVT. A fast iterative shrinkage thresholding algorithm (FISTA) using partial SVD has also been proposed for large datasets [105].

Because of their high computational costs, significant effort has been devoted to designing and analyzing the non-convex algorithms [106–108]. Because the objective function is not convex, good initialization schemes are extremely important.

Sun pointed that if the initialization is sufficiently accurate, the problem is effectively strongly convex [109]. This means that even if the problem of interest does not satisfy the mathematical conditions that are required for the convex relaxation to be exact, the convex relaxation of the problem can be used as a (hopefully accurate) initial guess for the (exact) non-convex formulation of the matrix recovery problem.

1.2.5.2 Matrix Factorization Based Methods

Low-rank Matrix Factorization The disadvantage of nuclear-norm algorithms is their high computational cost ($O(n^3)$ for flops and $O(n^2)$ for memory, where $\mathbf{X} \in \mathbb{R}^{m\times n}$), even using the fastest algorithms [110]. When the matrix factorization methods can dramatically reduce this cost[111]. Therefore, matrix factorization methods are preferred for for matrix completion problems using large datasets.

The idea of matrix factorization (MF) based methods is that an matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ with rank r can be factorized into two smaller matrices $\mathbf{P} \in \mathbb{R}^{m\times r}$ and $\mathbf{Z} \in \mathbb{R}^{r\times n}$; when $r << \min(m, n)$ this factorization can be constructed with far less computational cost than methods based on (partial) SVD [112–114]. MF can be interpreted as the rank constrained problem for matrix completion [115]:

$$\begin{aligned} \min_{\mathbf{X},\mathbf{P},\mathbf{Z}} \quad & \|\mathbf{P}\|_F^2 + \|\mathbf{Z}\|_F^2 \\ \text{subject to} \quad & \mathbf{X} = \mathbf{P}\mathbf{Z}^T, \\ & \mathbf{X}_{ij} = \mathbf{M}_{ij}, (i, j) \in \mathbf{\Omega}, \end{aligned} \tag{1.15}$$

where $\mathbf{P} \in \mathbb{R}^{m\times r}$, $\mathbf{Z} \in \mathbb{R}^{r\times n}$ and r is the rank of the matrix $\mathbf{X}$. The computation efficiency of this approach arises because QR factorization is much cheaper than SVD for low-rank problems, though the two methods give equivalent results:

$$\begin{aligned} \mathbf{X} &= \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \\ &= (\mathbf{U}\mathbf{\Sigma}^{\frac{1}{2}})(\mathbf{\Sigma}^{\frac{1}{2}}\mathbf{V}^T) \\ &= \mathbf{P}\mathbf{Z}^T \end{aligned} \tag{1.16}$$

$\mathbf{U}$ and $\mathbf{V}$ matrices are the latent factors that describe row and column spaces, respectively; $\mathbf{\Sigma}$ matrix represents the interaction between the factors.

Low-rank factorization compresses the n features (number of columns) to r new features. Knowing the rank, r, beforehand is, however, is frequently impossible. The nonlinear successive over-relaxation method developed in [104] adjusts r dynamically, can be shown to recover low-rank matrices with high probability, and works reliably in practice [104]. The theoretical proof for matrix reconstruction of weighted low rank

matrix approximations was reported by Y Li *et al.* using alternating minimization methods [116].

MF transforms matrix completion into a non-convex optimization problem for $\mathbf{P} \in \mathbb{R}^{m\times r}$ and $\mathbf{Z} \in \mathbb{R}^{r\times n}$. Nonetheless, the full matrix $\mathbf{M}$ is recoverable by local optimization methods like gradient-descent and alternating least squares provided the target matrix meets the basic assumptions we discussed previously (i.e., uniform sampling, low-rank, and incoherence) [108, 108, 109, 117–119]. Ge and his colleagues proved the local minimum is the global optimum when $\mathbf{M}$ is a positive-semidefinite symmetric matrix[110].

Alternating Least Squares Algorithms Alternating minimization is a popular strategy for MF-based matrix recovery, famously playing an essential role in the winning entry in the Netflix Prize competition [120–122].

The optimization optimum is achieved by fixing only $\mathbf{P}$, while minimizing $\mathbf{Z}$, *vice versa*. The least squared loss function is typically used. Therefore, the problem becomes if we fix $\mathbf{Z}$ with the step index ℓ

$$\mathbf{P}_\ell = \arg\min_{\mathbf{P}} \sum_{(i,j)\in\mathbf{\Omega}} \left[\mathbf{X}_{ij} - (\mathbf{P}\mathbf{Z}_{\ell-1}^T)_{ij}\right]^2 \tag{1.17}$$

This step is repeated with $\mathbf{P}_\ell$ fixed to get the optimum value of $\mathbf{P}_\ell$.

$$\mathbf{Z}_\ell = \arg\min_{\mathbf{Z}} \sum_{(i,j)\in\mathbf{\Omega}} \left[\mathbf{X}_{ij} - (\mathbf{P}_\ell^T\mathbf{Z})_{ij}\right]^2 \tag{1.18}$$

The chief advantages of alternating least squares (ALS) approaches is its computational efficiency for large datasets, which arises because missing data is directly emerge when large datasets are considered: ALS is easy to parallelize, each iteration is computationally inexpensive, and memory costs are low, just $2r$ [106].

1.2.5.3 Non-Linear Matrix Completion

The aforementioned linear matrix completion (LMC) algorithms assume that the matrix values can be represented by linear transformations in a lower-dimension latent subspace [73]. In this case, observations can be represented as linear transformations of the latent features. However, nonlinearity is more prevalent in real-world applications. For example, one can consider a nonlinear mapping from the r-dimensional latent space to the observations, $f : \mathbb{R}^r \to \mathbb{R}^m$. Popular methods for encoding nonlinear data structures include kernel PCA (KPCA) [123] and t-distributed stochastic neighbor embedding (t-SNE) [124]. Inductive matrix completion includes nonlinear-

ity by using side-information about the relationships between the rows and columns. For example, a movie recommender system can achieve a better performance using side information about (dis)similarities between viewers (e.g., their age) or movies (e.g., movie genre). Si and his colleagues proposed a goal-directed inductive matrix completion method assuming that incomplete matrix M can be estimated as [125]

$$\mathbf{M} = \Phi(\mathbf{A})\mathbf{C}\Phi(\mathbf{B})^T \tag{1.19}$$

where $\mathbf{C}$ is the unknown low rank matrix and $\Phi(\mathbf{A})$ and $\Phi(\mathbf{B})$ are nonlinear mapped features of the side information $\mathbf{A}$ and $\mathbf{B}$, respectively.

Nonetheless, nonlinear matrix completion methods are comparatively undeveloped. Most current methods are inapplicable when side information is incomplete or unavailable. Second, some of the best methods for classification problems are inapplicable to more general matrix completion problems. Finally, most nonlinear matrix completion methods *a priori* knowledge of the kernel hyperparameters, regularization parameters, and the number of low rank components; this information is rarely available in practical applications [92]; these problems can be partly circumvented using various "kernel tricks" [73].

1.2.5.4 Relationship between Various Matrix Completion Methods

As depicted in Figure 1.2(B), Schatten p-norm minimization methods for matrix completion evolves into nuclear norm minimization based methods based on Equation 1.8 and Equation 1.12. When a truncation operation is imposed, the algorithm becomes truncated nuclear norm minimization method.

Recently, a new study has revealed the relationship between Schatten p-norm based minimization and matrix factorization based methods. The equivalence between these two optimization methods can be proved if we define p, p_1, p_2 as the (quasi-)norm from Schatten p-norm minimization and norms of the low-rank matrices in the matrix factorization are measured with the Schatten p_1- and p_2-norms. Here,

$$\frac{1}{p} = \frac{1}{p_1} + \frac{1}{p_2} \tag{1.20}$$

where $0 < p \leq 1$ and $p_1, p_2 > 0$ [126]. Most nonlinear matrix completion methods are embellishments of linear matrix completion models with kernel tricks [73, 127, 128].

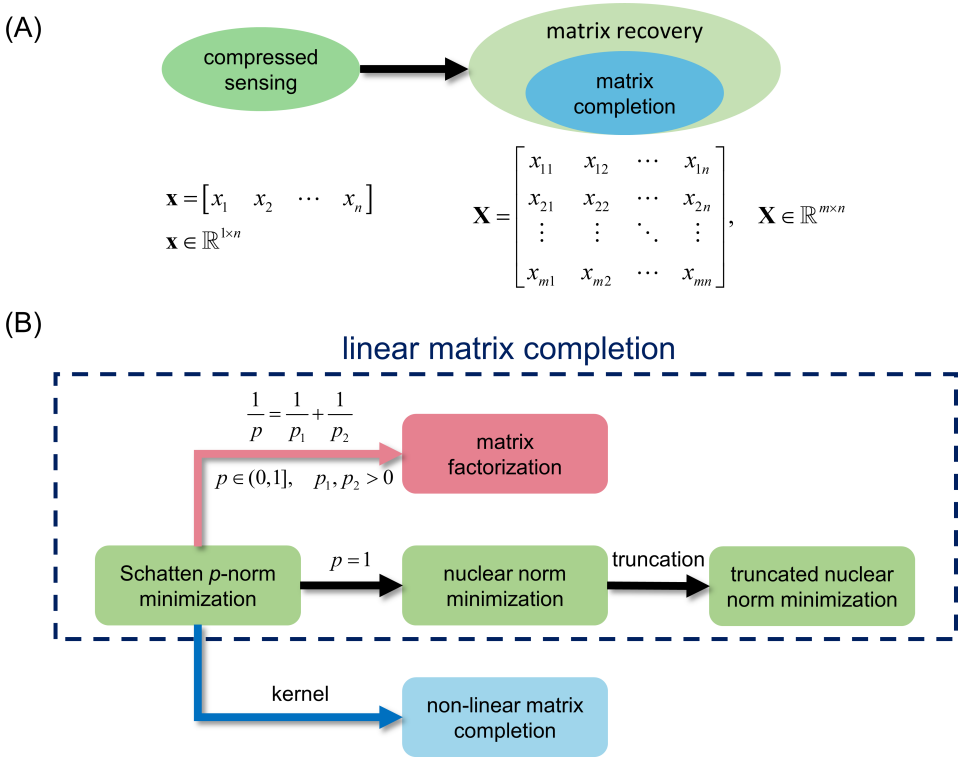


Fig. 1.2 Summary of matrix completion related theories and algorithms. (A). Relationship between compressed sensing, matrix recovery and matrix completion. (B). Relationship between matrix factorization, Schatten p-norm minimization, nuclear norm minimization, truncated nuclear norm minimization and non-linear matrix completion [73, 92].

1.2.6 Computational Packages for Matrix Completion

Due to their broad utility, many computational packages for matrix completion have been developed. This makes it relatively easy for users to apply matrix completion methods, even without a deep knowledge of the underlying mathematical theorems and numerical algorithms. As shown in Table 1.1, software for matrix completion is available in a wide variety of programming languages, including Java, Matlab, Python, C++, R and Scala. All the packages listed in Table 1.1 are free for academic use, though `mctc4mi` and `lrslibrary` require a Matlab license.

Different packages support various aspects of matrix completion. For example, both `mctc4bmi` and `SMURFF` support tensor completion. Some packages, like `H2O-3` and `LowRankModels.jl`, emphasize low-rank matrix recovery models; `lrslibrary` focus on signal/noise separation [129]. `lightfm` incorporates metadata from items and users into more conventional matrix factorization algorithms [130]. The `softimpute` and `filling` packages are mostly concerned with missing value imputation. The `implicit` package is specifically designed to solve

implicit feedback data in matrix completion, which just suggest whether a user has an interaction with one item (e.g. binary data about whether an item was purchased or liked), but not user-ratings. In contrast, `fancyimpute` answers questions related to explicit ratings. It is also noticed that some general purpose recommender system engines, for example `Microsoftrecommenders` and `ApacheSpark`, support not only matrix completion methods, but other problems too.

The package summary table also featured some new trends in matrix completion field. For example, `ICMC` and `gcmc` highlight a new trend in matrix completion: the close relationship between matrix completion and graph theory. `SMURFF` is notable because it explicitly supports side-information. `Collie` is the only package in Table 1.1 that can supports non-linear matrix completion.

Table 1.1 Computational packages for matrix completion. ⊛

Package name	Source codes	Version	LICENSE	Programming language	Reference
librec	https://github.com/guoguibing/librec	3.0.0 beta	GPL-3	Java	131
MatrixCompletion.jl	https://github.com/bethandtownes/MatrixCompletion.jl	0.1.0	MIT	Julia	
LowRankModels.jl	https://github.com/madeleineudell/LowRankModels.jl	1.1.1	MIT	Julia	132
mctc4bmi *	https://github.com/andrewssobral/mctc4bmi			Matlab	133
lrslibrary	https://github.com/andrewssobral/lrslibrary	1.0.10	⋇	Matlab	129
fancyimpute	https://github.com/iskandr/fancyimpute	0.5.5	Apache License 2.0	Python	134
Surprise	http://surpriselib.com	1.1.1	BSD 3-Clause	Python	135
lightfm	https://github.com/lyst/lightfm	1.16	Apache License 2.0	Python	130
RecBole	https://github.com/RUCAIBox/RecBole	0.2.1	MIT	Python	136
matrix-completion	https://github.com/tonyduan/matrix-completion		Eclipse Public License	Python	137
IGMC §	https://github.com/muhanzhang/IGMC		MIT	Python	138
bufflo	https://github.com/kakao/buffalo	1.2.1	Apache License 2.0	Python	
LibRecommender	https://github.com/massquantity/LibRecommender	0.6.4	MIT	Python	
collie_recs	https://github.com/ShopRunner/collie_recs	2	BSD 3-Clause	Python	
Microsoft recommenders	https://github.com/microsoft/recommenders	0.5.0	MIT	Python	139, 140
implicit	https://github.com/benfred/implicit	0.4.4	MIT	Python	
gcmc §	https://github.com/tanimutomo/gcmc			Python	31
SMURFF * ‡	https://github.com/ExaScience/smurff	0.16.0	MIT	Python, C++	141
H2O-3	https://docs.h2o.ai/	3.32.1.2	Apache License 2.0	Python, R	142
cmfrec	https://github.com/david-cortes/cmfrec/		MIT	Python, R	143
Apache Spark	https://spark.apache.org	3.3.1	Apache License 2.0	Python, Scala, Java	144
rsparse	https://cran.r-project.org/web/packages/rsparse	0.4.0	GPL-2 | GPL-3	R	
softimpute	https://cran.r-project.org/web/packages/softImpute	1.4	GPL-2	R	63
filling	https://cran.r-project.org/web/packages/filling	0.2.2	MIT	R	

⊛ All the data were collected as of May 4th, 2021.
* `mctc4bmi` and `SMURFF` supprt tensor completion.
§ `ICMC` and `gcmc` support graph matrix completion.
⋇ `lrslibrary` is free and open source for academia (non-commercial), but no license is specified.
‡ `SMURFF` supports side information incorporation.

1.3 Matrix Completion in Drug Design

The advance of high-throughput screening (HTS) [145], high content screening (HCS) [146], biological sequencing technologies [147], *etc.*, together with advances in computer hardware and numerical algorithms, has greatly increased the availability and usability of biological data in pharmaceutical science. "Big Data" is now an integral part of drug design and development. Nonetheless, missing data is common in pharmaceutical research. For example, micro-array data are plagued with missing measurements [58, 148]. Rather than deleting entire rows and columns of data due to missing entries therein, matrix completion allows us to use the data we have to not only predict the data we are missing, but also to learn the underlying structure of the original (complete) data matrix.

In this section, we will start with an overview of the potential of missing value imputation in the pharmaceutical sciences, discuss key mathematical theorems that undergird such applications, and review key applications.

1.3.1 Why Matrix Completion Can Be Used for Drug Design

In traditional recommender systems, we have incomplete ratings of different users on various items, which we represent mathematically as a rating matrix, where the rows are indexed by the users and columns are indexed by items, as shown in Figure 1.1(A). If we replace users with molecules and items with different molecular properties, this is directly analogous to a large number of problems that arise in the (bio)physical sciences. The molecule-property matrix is frequently very sparse due to experimental protocol limitations, compound (un)availability, or other reasons. For example, experimental measurements of $\log(P)$ values and biological activities can be cast as matrix completion problems Figure 1.1(B).

Matrix completion algorithms can be used in recommender systems because the strong correlation of similar users will demonstrate similar preferences over items. Analogously, structurally similar molecules often demonstrate similar physicochemical properties which is a fundamental assumption of modern quantitative structure-activity and structure property relationship (QSAR/QSPR) studies [149–151]. The similarity principle also undergirds ligand-based drug design [152, 153] and scaffold hopping [154]. When some property values are not measured or missing, we can argue that the missing value can be inferred from observed entries on the basis of the similarity assumption. This fits into uniform sampling assumption or MAR mechanism in statistical theory perfectly.

Moreover, it is a known phenomena that biological systems are composed of interacting components, such as genes, proteins, RNAs and so on, which also share similarities among themselves. For example, members of a family of proteins often share high sequence similarities, resulting in similar structure, similar dynamics patterns and thus similar functions. These correlations and associations reveal that biological data is often low-rank.

It should be pointed out that structurally similar or analogous molecules may demonstrate potency difference on the same target, which knows are activity cliffs [155, 156]. Meanwhile, attempts have been made to predict activity cliffs behaviors using similarity measurements [157, 158], still highlighting that biological activity is dependent on chemical similarity.

1.3.2 Why Big Matrices are Approximately Low-Rank

Most matrix completion methods are based on the assumption that large matrices are approximately low rank. Why are most big matrices approximately low-rank, and are the matrices that arise in pharmaceutical applications also low rank? Udell and Townsend performed a comprehensive mathematical analysis to rationalize the plausability of the low-rank assumption [159]. Their proof uses the Johnson-Lindenstrauss lemma, which states that, with high probability, any data points in a high dimensional Euclidean space can be mapped to low dimensional Euclidean space while approximately preserving the distance between two data points [160]. Even if the given matrix is full rank, they argue that the matrix can be embedded into a lower Euclidean space with a relative error bound [159]. Therefore, any sufficiently large matrix is approximately low-rank. For more detailed mathematical analysis, please refer to original paper [159]. Importantly, Udell-Townsend analysis is fully generic: it works for any matrix as long as its elements are more-or-less random, as opposed to highly structured. It applies in the biological context because measuring the properties of a molecule is equivalent to randomly selecting a point from the underlying distribution of molecular properties. However, if the molecules are selected to explicitly probe different constellations of molecular properties (e.g., molecules selected to ensure high diversity of training data), the rank of the large matrix may be larger than expected. Conversely, if the molecules are targeted (e.g., a library consisting only of closely-related compounds), the rank of the large matrix may be less than expected.

1.3.3 Applications of Matrix Completion in Pharmaceutical Studies

Matrix completion has been widely used in bioinformatics, but until recently people failed to notice its wide applicability in the broader context of drug design and discovery. The next sections overview how matrix completion can be used to address diverse problems in the drug development pipeline, see Figure 1.3.

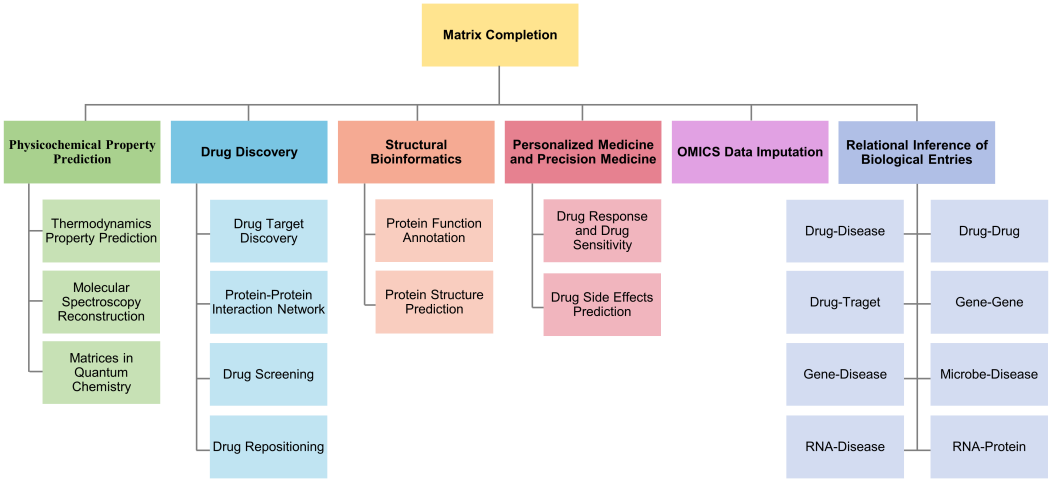


Fig. 1.3 Applications of matrix completion in pharmaceutical related fields. A hierarchical diagram showing the applicability of matrix completion in computational chemistry and drug design related sub-disciplines where application domain are highlighted in darker boxes and the usages of matrix completion algorithms are denoted in lighter boxes.

1.3.3.1 Matrix Completion for Physicochemical Property Predictions

Thermodynamics Property Prediction Matrix completion has demonstrated its potential in thermodynamics property prediction [161]. In this study, the authors formulated a sparse matrix $\mathbf{X} \in \mathbb{R}^{240\times250}$ for the activity coefficient for binary mixtures with 240 solutes and 250 solvents and employed a probabilistic matrix factorization model to predict the missing values. (The activity coefficient measures the deviations from ideal behavior in a chemical mixture system, and thereby describes the intermolecular interactions between solute and solvent molecules.) The proposed model achieved an absolute error less than 0.1 for 48.1% of the data even though the density of the matrix was just 6.82%. A follow-up study by the

same research team proposed a hybridizing approach by combining physical and data-driven prediction methods [162]. A knowledge distillation was conducted on top of the activity coefficient matrix with physics-based computational method, the modified UNIFAC, which provides some fitted parameters. Then informative priors were well-designed for the latent matrices. Both steps rely on probabilistic matrix completion and the model was significantly more accurate than alternative methods using UNIFAC, data-driven MCM, badding and boosting. [162]

Molecular Spectroscopy Reconstruction Molecular Spectroscopy can reveal important structural and dynamical information about molecular systems, but it is often time-consuming and technically difficult to obtain a full, precise, spectrum. For example, NMR has been widely applied to fragment-based lead discovery, structure-based drug design, protein-ligand interactions, hit compound validation and optimization, *etc.* [163–165]. Moreover, recent advances in in-cell NMR, on-cell NMR enables high-resolution studies of macromolecule structure, dynamics, and stability in solution, providing drug discovery with a very powerful tool [166–168]. However, obtaining high-resolution NMR is challenging.

Non-uniform sampling (NUS) provides a partial solution to this problem where matrix completion can be used to reconstruct a spectrum efficiently [169]. The free induction decay (FID) from NMR $\mathbf{x}$, was converted to a Hankel matrix with a operator $\mathbf{P}$,

$$\mathbf{X} = \mathbf{P}\mathbf{x} \tag{1.21}$$

where $\mathbf{X}$ is low rank. Qu *et al* proposed to use alternating direction minimization method [169], a matrix completion algorithms, to reconstruct the NMR spectra.

They successfully reconstructed a 2D ^{1}H-^{15}N HSQC spectrum with high quality by using just 35% of the complete acquired spectrum. Later a weighted version of Hankel matrix completion method, WLRHM, was developed by the same group, providing even better results [170]. In addition to singular-value weighting as in WLRHM, singular value thresholding was employed to reconstruct 2D non-linear spectroscopy, which is useful to uncover the dynamics of molecular complexes [171]. Similarly, a low-rank Hankel matrix model was developed for spatiotemporally encoded ultrafast NMR to faithfully reconstruct the spectrum by using an enhanced Hankel matrix from 1D inverse Fourier transformation [172]. The single point array reconstruction by spatial encoding (SPARSE) spectroscopy, which can be used to map multidimensional electronic structure and ultrafast dynamics, also benefits from reduced acquisition time by using matrix completion for spectra reconstruction [173].

Another important application is spectrum denoising. A regularization parameter λ was introduced to denoise the NMR spectroscopy, leading to a model named convex Hankel low-rank matrix approximation for denoising exponential signals (CHORD) [174]. Moreover, matrix completion has been shown to improve the resolution of multidimensional NMR as well [175]. For more examples of denoising and spectra reconstruction of NMR, please refer to [176–178].

There are application beyond NMR also. For example, compressed sensing can be used to replace the Fourier transformation algorithm in spectroscopy, and has been shown to be an effective strategy for the calculations of vibrational, optical absorption, and circular dichroism (CD) spectra [179].

Matrices in Quantum Chemistry Quantum chemical calculations provide accurate and fundamental information about the electron distribution and chemical bonding in chemical systems, but require significant computational resources, especially for large biomolecules. A particular problem is that the most accurate methods are only practical for smaller molecules. Matrix completion offers a solution to this problem. The main idea is to reconstruct a matrix from higher level of theory with a matrix from a lower level of theory.

For example, the hessian matrix, the second derivative of the energy with respect to nuclear displacements, is needed to compute vibrational frequencies and their normal modes. An recent study proposed a clever trick: evaluate energy second derivatives for some randomly-selected modes using molecular mechanics, then try to recover the full Hessian matrix [180]. Information about vibrational modes and frequencies can be obtained after projecting into the atomic coordinates basis and diagonalizing. It was found that 99% accuracy can be achieved by only sampling 30% of the modes, reducing the computation time by 70%. While this study used molecular mechanics to evaluate the Hessian, the authors provide an implementation using the `Q-Chem` quantum chemistry software [181].

Matrix completion algorithms be used to model the molecular potential energy surfaces (PES), which is a central topic in computational chemistry. A popular way to constructing a PES is to sample different atomic conformations, compute the conformations' energy (and possibly energy derivatives), then fit the results using splines [182, 183]. Based on the sparsity property of the PES, matrix completion can be used to construct a sparse-tensor polynomial basis with only a few PES evaluations [184]. A matrix completion variant, harmonic variety-based matrix completion (HVMC), is proposed to solve the minimum energy path finding problem in chemical

reactions where all the eigenvalues can be recovered successfully with just 30% random sampling. [185].

1.3.3.2 Matrix Completion for Drug Discovery

Drug Target Discovery Synthetic lethality (SL) is an emerging approach for cancer drug target discovery [186, 187]. The synthetic lethal interaction denotes a pair of two genes where perturbation (such as loss or inhibition with small molecules) of either one is viable, but loss of both will be lethal to the cell [186, 187]; this has been proved to be a useful for cancer drug target identification. However, experimental methos for assessing of SL, using methods like CRISP-CAS gene editing, are time-consuming and labor-intensive. The computational prediction of SL has gained increased attention [187].

Unlike models which do not consider prior knowledge about macromolecules and network topological structure, logistic matrix factorization based matrix completion was used for SL predictions by adding gene ontology (GO) semantic similarities as side-information [188]. Collective matrix factorization (CMF) was proposed as a unified framework to get jointly low-rank factorization of arbitrary set of matrices of pairwise relational data and experimental results revealed that mixed information of different relational data types help improve CMF performance [189]. Inspired by this, Liany *et al.* proposed a variant of CMF for SL prediction with heterogeneous data sources [190]. In constrast to CMF, the new model can deal with matrices with the same row and column entity types, extending the applicability of CMF by (1) learning a transformed representation (eigenvectors projected by principal component analysis algorithm); (2) transforming matrices with graph features; (3) assigning matrix-specific weights. The newly proposed model outperforms state-of-the-art methods for SL prediction, such as MNMC, MCA, MetaSL, and Mashup.

Besides identifying lethal gene pairs of SL, matrix completion can also help anti-viral drug target identification. Because influenza viruses undergoes rapid mutation to escape from population immunity, predicting its antigenicity becomes a crucial task. A H3N2 influenza dataset with 253 viruses (antigens) and 79 vaccine (antisera) was used to formulate a matrix completion problem [191].

As a natural extension and enhancement of matrix completion, tensor completion/ tensor factorization can also facilitate target discovery; this allows heterogeneous data sources to be incorporated, and often fits experimental data (where one may multiple measurements of a single molecular properties, with each measurement occurring under different conditions) The Rosalind method uses a tensor factorization model to identify disease-related genes by completing a heterogeneous knowledge graph tensor

which incorporates information about disease, gene-protein, compound, mechanism, and pathway [192]. This model demonstrated 18%-50% improvement compared with 5 state-of-art methods. Experimental validations *in vitro* for rheumatoid arthritis confirmed the utility of Rosalind predictions, as a new gene was identified.

Protein-Protein Interaction Networks Protein-protein interactions (PPIs) serve as the physical foundations of intermolecular communications in biological systems and are regarded as an important class of drug targets [193, 194]. PPI networks are often represented as a graph where each node denotes a protein and each edge represents a PPI. Therefore, the adjacency matrix of the PPI graph is a good match for matrix completion approaches; this type of matrix completion problem is often refereed to as link prediction in literature.

Subspace pursuit low-rank (SPLR) completion was an early attempt to use matrix completion for PPI predictions; it was applied to PPI network data for *Saccharomyces cerevisiae* [195] with 13.5% data points observed for 1200×1200 protein pairs, which became a standard dataset for studies of this type. A new algorithm for PPI predictions, robust discrete matrix completion (RDMC), was developed by using the augmented Lagrangian method to deal with integer programming problem that is inherent in binary matrix completion problems [196]. Additionally, inspired by the work of logistic matrix factorization for SL predictions [188], a symmetric logistic matrix factorization method was recently reported for PPI network predictions [197].

None of these approaches included any side information, like protein sequence or secondary structure. Later, a study using the same *Saccharomyces cerevisiae* PPI dataset exploited protein sequence as side information, where the sequence information was encoded with amino acid composition method [198, 199]. Another method for PPI reconstruction, non-negative matrix tri-factorization, enriches these tools by using information about protein sequences and structure and gene expressions, together with manifold regularization techniques [200].

Drug Screening One of the most straightforward applications of matrix completion is in drug screening, where each molecule may be tested against multiple (but not all) targets, and each target may have been probed by multiple (but not all) molecules. An automated high throughput screening (HTS) pipeline was developed by incorporating active learning and categorical matrix completion, the latter of which efficiently completes the phenotype matrix in HTS [201]. One innovation in this study was that not only were predictions provided, the reliability of the predictions were quantified by uncertainty analysis.

Another study used a computational model for antiviral drug discovery which is built on the top of sparse-group inductive matrix completion (SGIMC) algorithms [8]. The training data used a sparse matrix with 247994 compounds as rows, 158 viral species as columns, and 400,281 interaction (antiviral activity) values. SGIMC takes side information into consideration including chemical descriptors of the compounds and virus features; the raw descriptors were then processed using feature selection methods. The model achieved a moderate performance on an external dataset with receiver operating characteristic (ROC) score greater than 0.9.

Drug Repositioning Although drug repositioning (also known as drug repurposing) is not a brand new concept, it has been gained considerable momentum in the last decades which enables us to find new usages for known drugs [202, 203]. Drug repositioning allows an accelerated drug development process because more prior information is available to prove the lead compound's efficiency and safety [204]. Another factor that makes drug repurposing so attractive is that the drug development process is more economical compared with *de novo* drug design and development [205]. Many computational approaches have been used to suggest new therapeutic applications for approved drugs, such as signature matching, molecular docking, genetic association, pathway/connectivity mapping, drug-target interaction predictions, and retrospective clinical data analysis [205–207].

Matrix completion algorithms has proved useful for drug repurposing, especially for identifying new drug-target interactions and new drug-disease associations. It is generally assumed that there are some latent factors that contribute to highly-correlated drug-disease associations, which ensures that the drug-disease matrix is low-rank. Luo *et al.* proposed a drug repositioning recommendation system (DRRS) [208] to recover missing values for a a heterogeneous drug–disease interaction network which incorporates drug–drug, disease–disease and drug–disease networks. This study followed a conventional way of performing matrix completion for drug-disease predictions by representing heterogeneous network as a large drug–disease adjacency matrix with 1933 validated drug–disease associations for 593 drugs and 313 diseases. The proposed matrix model used SVT-R^{4}SVD method and achieved very promising results after comprehensive analysis. A novel matrix completion algorithm by projecting the association matrix onto convex sets for drug repositioning [209].

Attempts were given to incorporate side information to build computation models on top of matrix completion algorithms. The first approach is to compute similarities between small molecules and/or macromolecules. For example, drug-drug and

disease-disease similarities were used in BNNR model to improve model accuracy and robustness [210]. To build robust models against noise in drug-drug and disease-disease similarities, a bounded nuclear norm regularization (BNNR) was introduced to matrix completion for drug repositioning purposes [210]. A particularly successful application of matrix completion has been antiviral drug repurposing using the drug-virus association matrix [211, 212].

The second approach uses matrix completion methods that were explicitly designed to use side-information, cf. section 1.3.3.2. Word embedding can be used to include the abundant side information available about drugs and diseases [213]. Each drug and each disease is represented by a vector, which is updated with similarity measurements of drug-disease association. An inductive matrix completion method with regularization was then used to learn the projection matrix that maps drug-vector space to disease-vector space [213]. Similar representations were used in an independent study that incorporates drug-drug similarity and disease-disease similarity using similarity network fusion (SNF) and k-nearest neighbour methods [214]. A Bayesian inductive matrix completion was then applied to the drug-disease association matrix along with drug features matrix and disease feature matrix. Systematic analysis suggested that this method outperforms methods like DisDrugPred, SCMFDD, and DRRS MBiRW, with AUC values greater than 0.954 and AUPR values greater than 0.161 on 3 independent datasets [214]. Targeting miRNA is proposed to be an effective strategy for drug repurposing [215]; this motivated Deepthi and Jereesh to propose a bilateral-inductive matrix completion strategy for drug repositioning [216]. Two rounds of inductive matrix completion were performed on drug-miRNA and miRNA-disease association matrices; the product of these matrices then provides the drug-disease matrix. Side information are included by computing drug-drug similarities, miRNA-miRNA similarities, and disease-disease similarities [216]. Last but not least, multi-objective optimization techniques were used for formulate a matrix completion problem to accommodate multiple information sources, including drug response, gene expression, drug chemical structure, and target protein sequence [217].

Drug repurposing is closely entwined with drug side effects. Specifically, it is argued that clinical side-effects reflect information about human phenotype on a given drug and therefore implies additional clinical indications for a given drug [218, 219]. This topic is discussed in section 1.3.3.4.

1.3.3.3 Matrix Completion for Structural Bioinformatics

Matrix completion approaches are emerging as useful tools in structural bioinformatics, a field which addresses problems of structural biology using computational methods. These examples include structure determination [220], protein function annotation [221], and protein classification [222]. Studies employing non-negative matrix factorization (NMR) for structural bioinformatics [223, 224] are beyond the scope of this article and will not be discussed here.

Protein Function Annotation Kin *et al.* made a very early application of matrix completion to predict protein function [222]. The authors defined a structure-kernel matrix $\mathbf{D}$ and a sequence-kernel matrix $\mathbf{M}$; then an iterative *EM-project* algorithm was applied until the objective function, the Kullback-Leibler (KL) divergence, converged. The kernel trick captures similarities of input matrix entries between the structure and sequence matrices, thereby filling in the missing values in structure kernel matrix $\mathbf{D}$, where support vector machine (SVM) was used for protein classification. Similarly, a kernel technique is also employed in Protein Function Prediction method with Weak-label Learning (ProWL) [225] for imputing incomplete function annotations. Most computational models with matrix completion use protein sequence as side information. However, a recent study found that text information extracted from biological literature can outperform sequence information [221].

Protein Structure Prediction Nuclear magnetic resonance (NMR) has been widely used for protein structure determination. However, the Euclidean distance matrix (EDM) deduced from NMR data is incomplete. A rank-minimization approach was used to fill the incomplete EDM, followed by a mapping to the Cartesian coordinates [220]. A key point in this method is constructing initial guess values for missing entries in EDM that satisfy the triangle inequality. Scaled alternating steepest descent (ScaledASD) method has also been used for completing the EDM from NMR experiments, resulting to reasonable structures [226, 227].

In addition, matrix completion can also be applied to impute the incomplete EDM in X-ray crystallographic structures by embedding data from NMR, leaning to a higher resolution structures [228].

1.3.3.4 Matrix Completion for Personalized Medicine and Precision Medicine

Personalized medicine aims to guide medical practices by using an individual's susceptibility to a particular disease or therapeutic intervention (such as drugs) to

improve health outcomes [229, 230]. This endeavor spans a diverse range of scientific research and medical practices, including drug discovery, genome/molecular profiling for diagnosis purposes, and decision making on therapeutic interventions [230–232]. With personalized medicine, patients have a better prognosis, lower probability of unwanted side effects, and decreased medical costs. In addition, personalized medicine can lower the barrier to developing a safe clinical drug by reducing the financial and temporal costs associated with identifying novel drug targets [233, 234]. As a new paradigm of drug discovery and health care, many problems remain to be addressed by pharmaceutical industry and scientific community including drug response profiling with phenotyping of patients, predicting side effects of drugs, and drug sensitivity [229, 233–235]. These are main topics in pharmacogenomics, which studies how drugs affect a individual's gene phenotyping to maximize therapeutic interventions and minimize side effects. We do not distinguish between personalized medicine and precision medicine here, and will use the terms interchangeably.

Most of the aforementioned problems have a relational data structure, so matrix completion is a good fit for precision medicine research. These problems include, but are not limited to, drug response prediction, drug sensitivity analysis, and drug side effects forecasting. Precisely, drug response phenotype denotes adverse drug effects or treatment efficacy upon drug administration and drug sensitivity indicates the (in)tolerance of side effects caused by drugs at a therapeutic dose or concentration. Herein, we will discuss how matrix completion has been applied to drug response, drug sensitivity, and drug side-effects. The datasets used here are mostly at cancer cell line level given the heterogeneous nature of human tissues. Given the close relationship between drug response and drug sensitivity, the corresponding predictive models often use the same datasets, so we shall discuss computational approaches for drug response and drug sensitivity together.

Drug Response and Drug Sensitivity Some big data projects have measured drug responses of a large and diverse set of molecules on different cell lines, providing an unprecedented chance to use matrix completion for drug response predictions. For example, the PRISM Repurposing dataset contains measured drug responses for 4686 molecules on 578 cell lines covering 23 disease types [236]. As a convention, a relational matrix is formulated with cell lines as rows and small molecules as columns where some compounds have not been measured against some cell lines. Again, it is assumed that compounds with similar scaffolds will display similar response in the same or inherently close cells, making the matrix low rank. This problem can also

be viewed as a a bipartite graph matching problem where drugs and cell lines serve as two subsets.

One of the most well-known matrix completion algorithm, *SoftImpute*, was used to predict the missing values in compound-cell line matrix [237]. The cell line-drug response matrix elements can either be the IC_{50} value of a certain compound (in the CCLE database) or the AUC (area under dose-response curve) value (in the GDSC dataset). Using the same benchmark datasets, CCLE and GDSC, a regularization term was introduced to prevent overfitting. Moreover, the authors built an ensemble learning model by integrating matrix completion for the cell-line/drug-response matrix and ridge regression for cell-line/gene-expression matrix, showing that a linear combination of models achieved better performance than either individual model [238]. This indicates that ridge regression can capture the relationship between drug response and gene expression. Similarity-regularization can improve model performance for drug response predictions [239]. Models built with similarity regularization have lower RMSE and higher PCC and AUC thank competing approaches like pairwiseMKL and HNMDRP. [240].

Another approach is to express drug/cell-line relational data as a tensor [241]. For example, a dataset from the Dialogue on Reverse Engineering Assessment and Methods (DREAM) project with 52 breast cancer cell lines and 26 drug molecules can be expressed as a 3-dimensional tensor encoding cell lines, drugs, and doses. The problem is then solved with BaTFLED, a generative probabilistic model on top of Tucker decomposition, achieving better performance than LASSO, random forest and neural networks. Similarly, a tensor completion tool, DRIM, was developed by integration of muti-omics data for drug response predictions [242]. What's more, DRIM incorporates the time-series drug response data benefiting from tensor representation. This reformulates drug response process into a dynamic process, being more realistic to physical systems. Using tensor structure to include temporal dimension of time-course gene expression data was also validated on two other different datasets, one sample with 53 multiple sclerosis patients and another sample with 25 patients [243]. In most cases of real-world applications, the side information is noisy and can lead to misleading predictions for tensor completion algorithms. Dimitris and Colin proposed a new computational framework termed as `TensorGenomic` to leverage the noisy side information [244]. `TensorGenomic` achieves surprising good performance on the Genomics of Drug Sensitivity in Cancer (GDSC; $R^2 = 0.552$) and the Cancer Cell Line Encyclopedia (CCLE; $R^2 = 0.524$) datasets $R^2 = 0.552$ even with 80% missing data.

Applying manifold learning to the binary response matrix can help build a better classification model of drug sensitivity on cell lines [245]. Kernelized Bayesian matrix completion has been used to build an integrative quantitative structure-activity relationship (QSAR) for drug response profiles, extending traditional QSAR models from a single compound to multiple compounds [246]. The kernel technique used here enables the inclusion of various types of chemical features.

Drug Side Effects Prediction Most predictive models for predicting adverse drug reactions, or side effects, fail to provide accurate predictions for less-characterized drugs. This suggests using inductive matrix completion was applied for adverse drug reaction prediction [247], which assumes the association matrix heavily relies on the feature vectors (side information) [248]. The side information used in this study include drug features (chemical structure similarity, cosine similarity, Jaccard similarity coefficient, and integrated similarities), adverse drug reaction feature (MedDRA taxonomy similarities) and target features (Smith–Waterman scores). A more general purpose computational approach, REMAP, was developed for off-target identification with dual regularized one-class collaborative filtering; REMAP is fast, scalable and more accurate than competing methods [249]. Note that the off-target predictive framework can not only be used to prevent undesirable side effects, but to predict drug repositioning.

Attempts have been made to make full use of (possibly noisy) side information like drug structure, target sequence, and side effect similarities. When the side-information is noiseless (e.g., computed Tanimoto molecular similarities) the model performance is signficantly enhanced, but noisy protein-protein similarity data degrades the accuracy [249]. A model named feature-derived graph regularized matrix factorization method (FGRMF) was developed by using PubChem feature vector of drugs and imposing an graph regularization term [250]. Information about drug-drug interactions can facilitate drug side-effect prediction as well. A recent study demonstrated one strategy for doing this, using an non-negative matrix factorization (NMF) based matrix completion, followed by a heat diffusion on the resulting undirected weighted drug-drug semantic network to determine side-effects [251].

1.3.3.5 Matrix Completion for OMICS Data Imputation

OMICS technologies, such as genomics, proteomics, epigenomics, transcriptomics and Microbiomics, have become powerful tools in disease biology and have reshaped modern drug discovery [252–254]. For example, approximately 30 computational

platforms have been developed for target discovery: DrugBank, ChEMBL, the Comparative Toxicogenomics Database, *etc.* [255]. However, the problem of missing OMICS data prevents systematic data analysis because most computational methods require complete data structures (without missing entries), but it is rare for complete genomics and proteomics data to be available.[256, 257] Matrix completion provides a powerful toolbox of methods for missing data imputation in OMICS studies.

In particular, the gene expression matrix that encodes interactions between genes is low rank, meaning that matrix completion is appropriate for genomics and, specifically, genotype imputation in genome-wide association studies (GWAS). A computational model by minimizing the nuclear norm with Nesterov algorithm as the solver was used for genotype imputation and systematic studies on diverse datasets (real data of HapMap 3, synthetic pedigree data, and simulated low-coverage data from the 1000 Genomes Project) suggests that it achieves comparable performance to explicit models, but has better computational performance [258]. Other matrix-completion-based imputation methods include McImpute [259], scImpute [260], drImpute [261], DSNN [262] and CMF-impute [263]. Moreover, recent progress of machine learning concepts are also useful for imputation of OMICS matrices. For example, an autocoder is used in AutoImpute [264]; Generative Adversarial Networks (GAN) was used in VIGAN [265]; Integrative Bayesian Analysis (iBAG) of genomics data was used in FBM [266].

1.3.3.6 Matrix Completion for Relational Inference of Biological Entries

Matrix completion has been widely used for to predict drug-disease, drug-drug, drug-target, gene-gene, gene-disease, microbe-disease, RNA-disease, and RNA-protein associations; see Table 1.2. We will present two different viewpoints on these relational/association prediction studies with matrix completion, and give a few examples of pharmaceutical relevance.

One way to encode associations between different biological entities is using a bipartite graph, as in a drug-target interaction network. I.e., we assign drugs and targets to graph nodes, and add an unweighted edge between known drug-target pairs. The edge can be weighted if the strength of association is available (e.g., an IC_{50} or K_d value). The problem of identifying interactions is thus expressed as a link-prediction problem in a bipartite graph, which explains why new innovations in network science and graph theory accelerate biological research and drug discovery. The graph is often represented with a adjacency matrix where matrix elements indicate whether a pair of vertices are adjacent or not. In most biological relational inference problems,

the graph is undirected and unweighted, so the matrix is symmetric and binary (all entries are zero or one).

Associations between biological entities can also be encoded as a relational inference problem [189]. If we store information about biological entities in a structured relational database, the application studies in Table 1.2 can be formulated as relational inference problems. The advantage of using such structured representation is that information can be extracted from data, which further enables knowledge discovery. The main focus of relational inference or association prediction are link prediction and link regression, the former of which determines whether there exists an association relationship and the latter of which determines the value or strength of the association [189].

Table 1.2 Summary of matrix completion for relational inference of biological entries

Objective	Algorithm/Model	Reference
drug-disease	heterogeneous graph inference with matrix completion (HGIMC)	267
drug-disease	overlap matrix completion (OMC)	268
drug-disease	graph regularized 1-bit matrix compeltion (GR1BMC)	269
drug-drug	manifold regularized matrix factorization (MRMF)	270
drug-drug	multitask dyadic prediction method	271
drug-target	imputation conditional consistency (GICC)	272
drug-target	probabilistic matrix factorization (PMF)	273
drug-target	Uzawas algorithm for matrix completion	274
drug-target	coupled matrix completion and coupled tensor-matrix completion	275
drug-target	multi-wiew low rank embedding (MLRE)	276
drug-target	deep latent factor model	277
drug-target	dual Laplacian graph regularized matrix completion (DLGRMC)	278
drug-target	neighborhood constraint matrix completion (NCMC)	279
drug-target	cross-network embedding	280
drug-target	multiple matrix completion algorithms used	211
gene-gene	integrative matrix trifactorization for GRN inference (iMTF-GRN)	281
gene-disease	an improved PU learning formulation	282
microbe-disease	mHMDA	283
microbe-disease	BMCMDA	284
RNA-protein	orthogonality-regularized nonnegative matrix factorization (iONMF)	285
RNA-disease	neural inductive matrix completion with graph convolutional network (NIMCGCN)	286
RNA-disease	SIMCLDA	287
RNA-disease	arbitrarily-order proximity network embedding	288
RNA-disease	matrix completion for miRNA-disease association (MDMDA)	289
RNA-disease	inductive matrix completion for miRNA-disease association prediction (IMCMDA)	290
RNA-disease	inferring miRNA-disease association by integrating lncRNA and miRNA data (IMDAILM)	291
RNA-disease	double matrix completion for predicting the circRNA-disease association (DMCCDA)	292
RNA-disease	improved prediction of miRNA-disease associations (IMDN)	293
RNA-disease	matrix completion model with dual Laplacian regularization (DLRMC)	294
RNA-disease	improved low-rank matrix recovery (ILRMR)	295

1.4 How to Formulate a Matrix Completion Problem

We have discussed examples of using matrix completion algorithms solving computational chemistry and pharmaceutical related problems, but no guidance about how to formulated a matrix completion problem. While every problem is different, there are nonetheless general strategies that are useful, and in this section we shall discuss different scenarios, as summarized in Figure 1.4. It is important to remember that different strategies may overlap; for example, relational inference and edge-prediction in graphs are different expressions of the same problem.

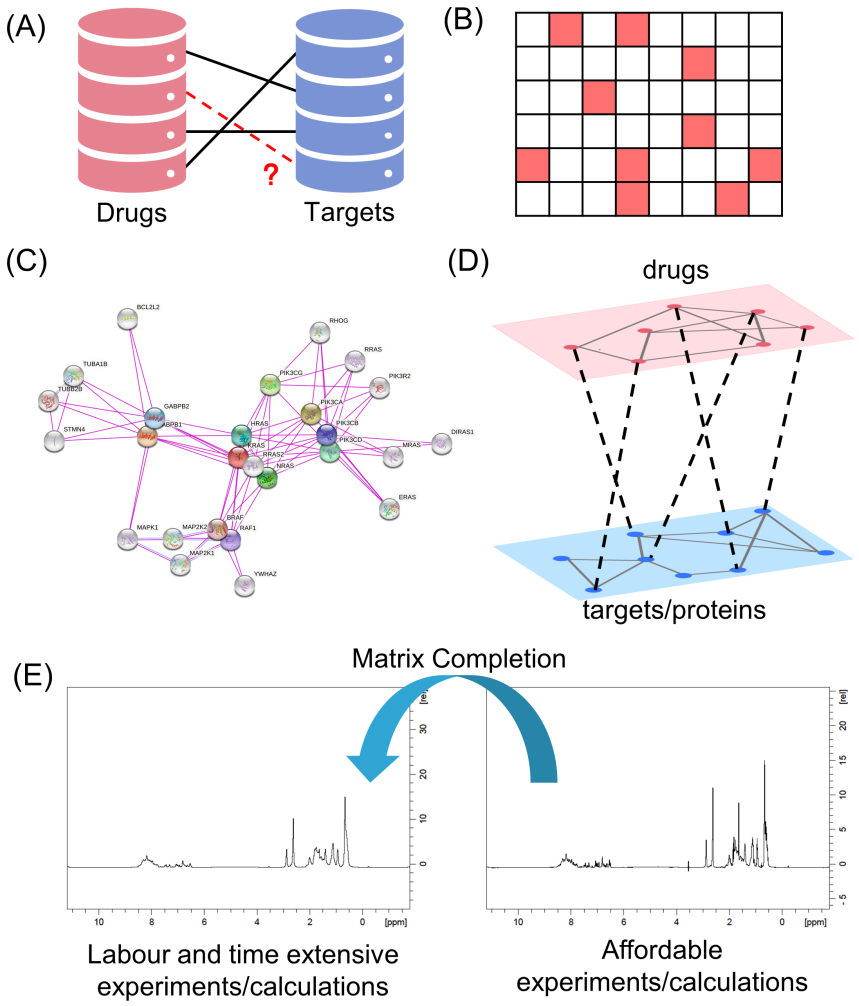


Fig. 1.4 A summary of various ways to formulate a matrix completion problems. (A). Biological relational inference and association. (B). Matrix completion representation for (A), (C), (D) and (E). (C). Protein-protein network of KRAS, which is generated with STRING web server [296]. (D). Drug-target interaction prediction with side information. (E). High quality signal reconstruction for NMR spectrum.

1.4.1 Relation Inference and Association Predictions

A relation can be viewed as a table in relational database where rows and columns denote different entries, e.g., drugs and proteins; values in the table indicate whether there exists a interaction (binary data) or a quantitative measurements of the interaction (numerical data), Figure 1.4(A). These quantitative values can be IC_{50}, K_d, *etc.*. The goal of relational inference is to compute or predict new relations or associations based on partially observed data.

For example, suppose we are given a set of drugs and proteins and some experiments have provided data on which drugs can provide to which proteins. Not all drug-protein pairs have been measured, however, due to compound availability and difficulty of developing screening assays. This table of relationships can be easily converted into a matrix completion problem by representing the table as a matrix. If the available experiments provide numerical data, this is regression problem for matrix completion. Otherwise, this becomes a binary matrix completion problem, also known as a Boolean matrix completion problem.

1.4.2 Link Predictions

In the settings of graph theory and network science, matrix completion methods arise naturally as a powerful tools for link prediction, Figure 1.4(C). Link prediction in medicinal chemistry arises in the prediction of protein-protein, drug-target, RNA-disease, gene-gene, and other interactions; see subsubsection 1.3.3.6. Link predictions is closely related to relational inference and association prediction.

A common example is the prediction of protein pair interactions (PPI). Suppose we have observed the interactions of some protein pairs. The PPI network can be written as an adjacency matrix, with entries only for confirmed PPIs. Predicting additional PPIs becomes the problem of link prediction, recall section 1.3.3.2. When side information is available, Figure 1.4(D), it can help improve the performance of matrix completion algorithms. Take drug-target interaction predictions discussed in section 1.3.3.2. When incorporating drug-drug similarities, model accuracy is improved in (DRRS) [208]. Similarly, information about structure/function similarity of proteins can be useful side information for PPI prediction.

1.4.3 Missing Value Imputations

Missing data imputation is arguably the most direct applications of matrix completion methods in pharmaceutical science, Figure 1.4(B). An enormous amount of time, talent, and money is invested in the collection of high quality biological data, and

these costs encourage approaches that require less data. Moreover, missing data is inevitable in most cases because of resource insufficiency and technogical limitations. For example, it is estimated that the genome sequencing cost for per cancer case is 6841 and 7050 per rare disease case [297]. By representing the whole data as a matrix and filling in the observed values, this is exactly a (low-rank) matrix completion problem. Further examples are given in subsubsection 1.3.3.5 with details.

1.4.4 High Quality Data Reconstruction with Partially Measured/Computed Values

Whenever getting high quality data is labor and time extensive, matrix completion can be useful, Figure 1.4(E). For example, studies have used matrix completion to reconstruct NMR spectroscopy with reduced acquisition time as discussed in section 1.3.3.1. This promising approach is also highlighted for its power to reduce the noise to obtain high quality spectroscopy.

This strategy is not restricted in experiments, but can also be applied to expensive and/or technically challenging calculations like those of molecular quantum chemistry. The Hessian matrix can be recovered by only sampling selected random modes with matrix completion [180]. Harmonic variety-based matrix completion (HVMC) can find all the eigenvalues correctly for chemical reaction path with only 30% random sampling [185]. More information can be found in section 1.3.3.1.

1.5 Cross-Validation Techniques for Matrix Completion

1.5.1 Data Splitting

A key problem in matrix completion methods is determining the optimal rank r. A large value of r can leads to better fitting of the training data, but risks overfitting when the data is noisy.[298].

As one of the most powerful resampling technique, cross-validation has been widely used to measure the prediction error on testing data so as to optimize the model hyperparameters both for regression and classification problems [299, 300]. The main idea is to split the data into training, testing (and validation) datasets and then to estimate the average generalization error when the trained model is applied

to the testing data [301]. Take linear regression for example, Figure 1.5(A) ∗. The linear regression problem is

$$\min_{\mathbf{x}} \quad \|\mathbf{A}\mathbf{x}^T - \mathbf{b}\|_F^2 \tag{1.22}$$

where $\mathbf{A}$ denotes the feature matrix, $\mathbf{x}$ denotes a vector of parameters to learn and $\mathbf{b}$ denotes the target values. A very simple, but widely used way to performing cross-validation is to hold some data out. For example, a mask vector is generated to mask the selected rows in the feature matrix $\mathbf{A}$ and the corresponding values in $\mathbf{b}$, which are highlighted in light blue in Figure 1.5(A). More mathematically, Equation 1.22 is reformulated as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{m}\mathbf{I} \circ \mathbf{A}\mathbf{x}^T - \mathbf{m} \circ \mathbf{b}\|_F^2 \\ \text{subject to} \quad & \mathbf{m}_i \in \{0, 1\} \end{aligned} \tag{1.23}$$

$\mathbf{m}$ is a random binary vector composed of 0 or 1 that serves as a mask to select rows/data. $\mathbf{I}$ is an identity matrix with the same shape of $\mathbf{A}$, which is used to broadcast the mask vector m into a mask matrix in alignment of the shape of $\mathbf{A}$. The $\circ$ denotes the Hadamard product that performs element-wise product of two matrices. The holdout rows correspond to the hold out values in $\mathbf{b}$ as highlighted in light blue in Figure 1.5(A).

Such a divide-and-conquer strategy is not directly applicable to matrix completion problems because the training and testing datasets are not two independent matrices. If take some rows columns out of original incomplete matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$, Figure 1.5(E), that suggests we will hold out a row in $\mathbf{P} \in \mathbb{R}^{m \times r}$ which results to a situation where we cannot estimate all the hyperparameters and overfitting will be observed [302]. The same problem occurs for removing entire columns of data, as shown in Figure 1.5(F). One may propose to combine the data splitting of leaving out rows and columns sequentially to avoid overfitting. But this involves the hold-out matrix elements inmodel training, violating the underlying precepts of cross-validation [303].

An ideal sampling method for cross-validation removes matrix entries out randomly while keep missing values untouched. As depicted in Figure 1.5(B), we have an incomplete matrix $\mathbf{X}$ with missing values denotes in light purple blocks and we can generate a mask matrix $\mathbf{M} \in \mathbf{R}^{m \times n}$ that hold out observed values only (light magenta blocks in Figure 1.5(B)) which is a Boolean matrix. By doing this, we get training data $\mathbf{M} \circ \mathbf{X} \in \mathbf{R}^{m \times n}$ and testing data $(\neg\mathbf{M}) \circ \mathbf{X} \in \mathbf{R}^{m \times n}$ where $\neg$

∗Inspired by alexhwilliams.info/itsneuronalblog/2018/02/26/crossval/ by Alex Williams

denotes the logical *not* operation and $\circ$ denotes the Hadamard product. This sets up an optimization problem for training data

$$\begin{aligned} \min_{\mathbf{P},\mathbf{Z}} \quad & \| \mathbf{M} \circ (\mathbf{X} - \mathbf{P}\mathbf{Z}^T) \|_F^2 \\ \text{subject to} \quad & \mathbf{M}_{ij} \in \{0, 1\} \end{aligned} \tag{1.24}$$

and the optimal rank can be estimated with grid search or Bayesian optimization techniques. Model performance can be evaluated applying the estimated rank on testing data.

Under some circumstances, one may wish to remove a fixed number of elements in each row of input matrix, as pictured in Figure 1.5(C). From a mathematical perspective, the only difference is we add another constraint for the optimization to ensure the sum of elements selected in each row is a constant,

$$\begin{aligned} \min_{\mathbf{P},\mathbf{Z}} \quad & \| \mathbf{M} \circ (\mathbf{X} - \mathbf{P}\mathbf{Z}^T) \|_F^2 \\ \text{subject to} \quad & \mathbf{M}_{ij} \in \{0, 1\} \\ & \textstyle\sum_{j=0}^{j=k} \mathbf{M}_{ij} = const \end{aligned} \tag{1.25}$$

This is useful when sparsity or density in all rows are approximately equal where we can take fixed percent of data out. A further step is to force the number of missing elements in each row to be a constant. But this can be risky because frequently some rows have only very limited number of observed values.

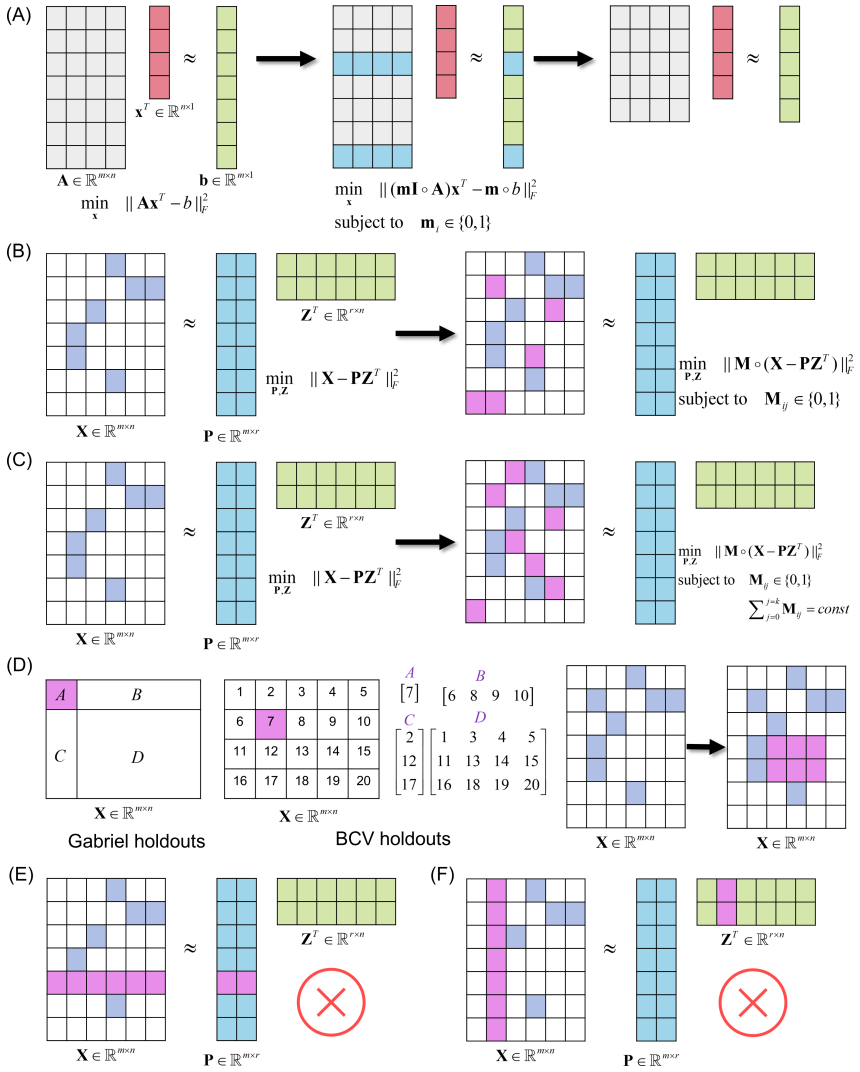


Fig. 1.5 Cross-validation schemes for linear regression and matrix completion. (A) Cross-validation for linear regression. Selected testing data are in light blue blocks. (B) Cross-validation for matrix completion by random sampling observed values. (C) Cross-validation for matrix completion by random sampling fixed number of observed values in each row. (D) Gabriel holdouts and Bi-cross-validation (BCV) holdouts along with an illustration of BCV for matrix completion. (E) and (F) Leaving rows and columns out as testing data is not the right way for cross-validation of matrix completion. For (B)-(F), all the selected data points for testing dataset are highlighted in light magenta blocks.

However, these two cross-validation methods can computationally expensive when the input matrix $\mathbf{X} \in \mathbf{R}^{m \times n}$ becomes big because of the possible combinations of selected indices grows with a binomial coefficient. If we are given a matrix 1000×800 matrix with 15% of data missing and it is asked to take 10% out of the original data as the testing data, we have to deal with $\binom{1000\times800\times(1-0.15)}{1000\times800\times0.10}$ operations, which is far too large. Instead of considering all possible samples, it is useful to impose an early-stopping mechanism that counts how many times the observed values have been sampled, denoted as a vector h. When the minimum value in h is greater than pre-defined threshold t (e.g., $t = 30$), one assumes that every matrix element has been adequately sampled and the loop is terminated. Another option is to loosen this early-stopping criteria by determining if certain percentage of data has been sampled. For example, the cross-validation can be terminated if 80% of observed data have been sampled at least 100 times; this is especially helpful for very large datasets.

Bi-cross-validation (BCV) provides an alternative, and nicer, solution to rank selection for matrix completion [298]. BCV is a natural generation of Gabriel's matrix partition method [304]. Gabriel partitions the matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$ into 4 submatrices (Figure 1.5(D)), $\mathbf{A} \in \mathbb{R}^{1\times1}$, $\mathbf{B} \in \mathbb{R}^{1\times(n-1)}$, $\mathbf{C} \in \mathbb{R}^{(m-1)\times1}$ and $\mathbf{D} \in \mathbb{R}^{(m-1)\times(n-1)}$. A matrix completion model can be trained with $\mathbf{D}$ and an estimation of $\mathbf{A}$ is obtained by using

$$\hat{\mathbf{A}} = \mathbf{B}\hat{\mathbf{D}}\mathbf{C} \tag{1.26}$$

where $\hat{\mathbf{D}}$ denotes the fitted matrix from SVD. But Gabriel's method is too expensive for large matrices because it only holds out one entry at a time. BCV is achieved by dividing the matrix rows into h groups and splitting matrix columns into l groups, leading to number of folds $h \times l$, where the holdout submatrix (testing data) is defined by selected the row and column groups simultaneously. For instance, we have a matrix $\mathbf{X} \in \mathbf{R}^{4\times5}$ with rows divided into 4 groups and columns divided into 5 groups, and one possible holdout is to select the second row block and column block as highlighted in magenta in Figure 1.5, leading to four submatrices $\mathbf{A} \in \mathbb{R}^{1\times1}$, $\mathbf{B} \in \mathbb{R}^{1\times3}$, $\mathbf{C} \in \mathbb{R}^{3\times1}$ and $\mathbf{D} \in \mathbb{R}^{3\times4}$ after rearrangement. Similar to Gabriel's method, submatrix $\mathbf{D}$ serves as the training data and an estimate for the holdout testing data $\hat{\mathbf{A}}$ is constructed with 1.26.

Performance and robustness improves when using BCV [305]. It is also shown that error of BCV is U-shaped with respect to rank [298], making BCV a nice tool for rank selection. Smaller holdouts tend to risk overfitting and large holdouts risk underfitting, so one typically chooses 2×2 holdout or 3×3 holdouts. For other related studies, please refer [306–308].

1.5.2 Choice of Algorithm for Computing Mean and Variance

Computing not only the predicted values (means) but their variances (error estimates) is important, as it helps the user understand the reliability of the predictions and select the best model. If the matrix is small, approximately 200×100, we can save the prediction matrix each time, and then compute the mean and variance at the end. But this direct approach takes too much memory for large matrices.

Online algorithms solve this problem by processing the predictions sequentially [309, 310]. Specifically, Welford's online algorithm provides us an flexible solution to compute mean and variance for hyperparameter optimizations of matrix completion. Welford's is numerically stable and easy to implement (cf. the pseudocode in section 4.2.2 of Ref. [311].) An extension of Welford's online to determine the variance of fixed-length contiguous subsequences has recently been reported [312].

1.6 Conclusions and Future Perspectives

The missing-value problem is pervasive, appearing not only throughout the experimental sciences, but in the computational, social, and applied sciences also. The traditional approach—discarding the entire subset of data that contains the missing value—often results in the loss of valuable data. Matrix completion allows one to impute the missing values from the values that are present, so no data need be discarded.

Matrix completion is an especially important in recommender systems, where missing values are ubiquitous. After it first entered mainstream research through the Netflix prize competition, matrix completion has been widely used in e-commerce [30], link prediction of social networks [31], signal processing [32], and image processing [33]. By contrast, we feel that matrix completion methods are underappreciated in the pharmaceutical sciences, especially in molecular medicine, where the missing data problem is also ubiquitous. This review aims increase awareness about the utility of matrix completion algorithms by overviewing their mathematical foundation (see section 1.2) and explaining their utility and applications to drug design with explicit examples. We also overviewed software packages for matrix completion (cf. Table 1.1) and discussed practical aspects, including formulate a drug design problem as a matrix completion problem (section 1.4) and considerations (e.g., cross-validation) needed to build robust models (section 1.5).

A key element of this review is reasonably comprehensive list of key applications of matrix completion to problems related to pharmaceutical chemistry, including thermodynamic property prediction [162], molecular spectroscopy reconstruction

[171, 175], drug target identification [189, 191, 192], drug screening [8], drug repositioning [208], protein function prediction [220], association inference of biological entries, and more subsection 1.3.3.

Matrix completion algorithms are an active area of research, and we reviewed a few emerging trends. One important embellishment of "ordinary" matrix completion is the incorporation of side information to make more accurate and robust predictions. For example, if each row of a matrix corresponds to a different molecule, then including side-information about chemical structure similarity can increase the accuracy of the matrix completion algorithm [248]. Similar considerations hold for protein sequence similarity [199]. Similarity-regularization [239], graph regularization [250], and similarity network fusion [214] are all ways to include side-information. Side information is often noisy, but methods like `TensorGenomic`, which is used for drug sensitivity predictions, are robust to this noise [244].

A imperfection of matrix completion algorithms is its transductive nature, which means that if we are given a new matrix, the original built model has to be retrained because the learned latent factors won't apply to the new matrix [248]. Inductive matrix completion models can be easily generalized to unseen data by incorporating the side information about users and items [125, 248, 313]. This makes inductive matrix completion a good match for association predictions, such as RNA-disease association predictions [287, 290, 314]. More examples can be found at section 1.3.3.2 and section 1.3.3.4.

In the same way that compressed-sensing (missing entries in a vector) is generalized to matrix completion, tensor completion is the natural higher-order generalization of matrix completion. By using the higher order of representations, we can not only easily capture similarity between molecules, targets and many other items, but also use additional spatial- and spatiotemporal-information. Various algorithms have been proposed within tensor completion [315–319], but some of them are computationally demanding (e.g., some require solving large semidefinite programming problems). Recently, a new algorithm for tensor completion was developed which can finish in almost linear time for exact completion and can easily generalized to thousands of dimensions [320]. Tensor completion methods have also been applied to drug-target interaction predictions [275, 321], drug repositioning [322], pharmacogenomic multi-relation predictions [323] and molecular spectroscopy reconstruction [176] so on.

In summary, drug design problems can often be expressed as low-rank matrix completion problems, which allows the powerful mathematical framework of ma-

trix completion methods (e.g., statistically robust predictions [64]) and practical computational models/algorithms to be used.

References

[1] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-Based Systems*, 46:109–132, 2013.

[2] Prem Melville and Vikas Sindhwani. Recommender systems. *Encyclopedia of Machine Learning*, 1:829–838, 2010.

[3] Robin Burke, Alexander Felfernig, and Mehmet H Göker. Recommender systems: An overview. *AI Magazine*, 32(3):13–18, 2011.

[4] Kangning Wei, Jinghua Huang, and Shaohong Fu. A survey of e-commerce recommender systems. In *2007 international conference on service systems and service management*, pages 1–5. IEEE, 2007.

[5] Guandong Xu, Zhiang Wu, Yanchun Zhang, and Jie Cao. Social networking meets recommender systems: survey. *International Journal of Social Network Mining*, 2(1):64–100, 2015.

[6] Thi Ngoc Trang Tran, Alexander Felfernig, Christoph Trattner, and Andreas Holzinger. Recommender systems in the healthcare domain: state-of-the-art and research issues. *Journal of Intelligent Information Systems*, pages 1–31, 2020.

[7] Márcia Barros, André Moitinho, and Francisco M Couto. Hybrid semantic recommender system for chemical compounds. In *European Conference on Information Retrieval*, pages 94–101. Springer, 2020.

[8] Ekaterina A Sosnina, Sergey Sosnin, Anastasia A Nikitina, Ivan Nazarov, Dmitry I Osolodkin, and Maxim V Fedorov. Recommender systems in antiviral drug discovery. *ACS Omega*, 5(25):15039–15051, 2020.

[9] John Savage, Akihiro Kishimoto, Beat Buesser, Ernesto Diaz-Aviles, and Carlos Alzate. Chemical reactant recommendation using a network of organic chemistry. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, pages 210–214, 2017.

[10] Kota Suzuki, Kosei Ohura, Atsuto Seko, Yudai Iwamizu, Guowei Zhao, Masaaki Hirayama, Isao Tanaka, and Ryoji Kanno. Fast material search of lithium ion conducting oxides using a recommender system. *Journal of Materials Chemistry A*, 8(23):11582–11588, 2020.

[11] Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[12] Charu C Aggarwal et al. *Recommender systems*, volume 1. Springer, 2016.

[13] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.

[14] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.

[15] Daniel Valcarce, Javier Parapar, and Álvaro Barreiro. Finding and analysing good neighbourhoods to improve collaborative filtering. *Knowledge-Based Systems*, 159:193–202, 2018.

[16] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[17] Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 19–28, 2009.

[18] Yelong Shen and Ruoming Jin. Learning personal+ social latent factor model for social recommendation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1303–1311, 2012.

[19] Aanchal Mongia, Neha Jhamb, Emilie Chouzenoux, and Angshul Majumdar. Deep latent factor model for collaborative filtering. *Signal Processing*, 169: 107366, 2020.

[20] Jiansheng Fang, Xiaoqing Zhang, Yan Hu, Yanwu Xu, Ming Yang, and Jiang Liu. Probabilistic latent factor model for collaborative filtering with bayesian inference. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 73–80. IEEE, 2021.

[21] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. Ieee, 2008.

[22] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434, 2008.

[23] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 230–237, 1999.

[24] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[25] Robert M Bell and Yehuda Koren. Lessons from the netflix prize challenge. *Acm Sigkdd Explorations Newsletter*, 9(2):75–79, 2007.

[26] Dheeraj Bokde, Sheetal Girase, and Debajyoti Mukhopadhyay. Matrix factorization model in collaborative filtering algorithms: A survey. *Procedia Computer Science*, 49:136–146, 2015.

[27] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD Cup and Workshop*, volume 2007, page 35. Citeseer, 2007.

[28] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[29] Dietmar Jannach, Paul Resnick, Alexander Tuzhilin, and Markus Zanker. Recommender systems—beyond matrix completion. *Communications of the ACM*, 59(11):94–102, 2016.

[30] Andy Ramlatchan, Mengyun Yang, Quan Liu, Min Li, Jianxin Wang, and Yaohang Li. A survey of matrix completion methods for recommendation systems. *Big Data Mining and Analytics*, 1(4):308–323, 2018.

[31] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[32] Simon Foucart and Holger Rauhut. An invitation to compressive sensing. In *A mathematical introduction to compressive sensing*, pages 1–39. Springer, 2013.

[33] Hongyan Zhang, Wei He, Liangpei Zhang, Huanfeng Shen, and Qiangqiang Yuan. Hyperspectral image restoration using low-rank matrix recovery. *IEEE transactions on geoscience and remote sensing*, 52(8):4729–4743, 2013.

[34] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

[35] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[36] Irena Orović, Vladan Papić, Cornel Ioana, Xiumei Li, and Srdjan Stanković. Compressive sensing in signal processing: algorithms and transform domain formulations. *Mathematical Problems in Engineering*, 2016, 2016.

[37] David L Donoho. For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(6):797–829, 2006.

[38] Emmanuel Candes and Justin Romberg. Sparsity and incoherence in compressive sampling. *Inverse problems*, 23(3):969, 2007.

[39] Angshul Majumdar. *Compressed sensing for engineers*. CRC Press, 2018.

[40] Kurt Bryan and Tanya Leise. Making do with less: An introduction to compressed sensing. *Siam Review*, 55(3):547–566, 2013.

[41] Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005. ISSN 0018-9448.

[42] Peng Yi-Gang, Suo Jin-Li, DAI Qiong-Hai, and XU Wen-Li. From compressed sensing to low-rank matrix recovery: theory and applications. *Acta Automatica Sinica*, 39(7):981–994, 2013.

[43] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Mathematical Sciences Series. W. H. Freeman, 1979. ISBN 9780716710448. URL https://books.google.ca/books?id=fjxGAQAAIAAJ.

[44] Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[45] Mathukumalli Vidyasagar. *An introduction to compressed sensing*. SIAM, 2019.

[46] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

[47] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.

[48] Elad Romanov and Matan Gavish. Near-optimal matrix recovery from random linear measurements. *Proceedings of the National Academy of Sciences*, 115 (28):7200–7205, 2018.

[49] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[50] Thierry Bouwmans and El Hadi Zahzah. Robust pca via principal component pursuit: A review for a comparative evaluation in video surveillance. *Computer Vision and Image Understanding*, 122:22–34, 2014.

[51] T. Bouwmans, N.S. Aybat, and E. Zahzah. *Handbook of Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing*. CRC Press, 2016. ISBN 9781315353531. URL https://books.google.ca/books?id=j5OKDQAAQBAJ.

[52] Benjamin Recht, Maryam Fazel, and Pablo A Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

[53] Timothy J Grigsby and Justin McLawhorn. Missing data techniques and the statistical conclusion validity of survey-based alcohol and drug use research studies: A review and comment on reproducibility. *Journal of Drug Issues*, 49(1):44–56, 2019.

[54] Roderick J Little, Ralph D'Agostino, Michael L Cohen, Kay Dickersin, Scott S Emerson, John T Farrar, Constantine Frangakis, Joseph W Hogan, Geert Molenberghs, Susan A Murphy, et al. The prevention and treatment of missing data in clinical trials. *New England Journal of Medicine*, 367(14):1355–1360, 2012.

[55] N.R. Council, D.B.S.S. Education, C.N. Statistics, and P.H.M.D.C. Trials. *The Prevention and Treatment of Missing Data in Clinical Trials*. National Academies Press, 2010. ISBN 9780309186513. URL https://books.google.ca/books?id=_CSF1v2c8jQC.

[56] Judi Scheffer. *Dealing with missing data*. Massey University, 2002.

[57] Wei-Chao Lin and Chih-Fong Tsai. Missing value imputation: a review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, 53(2): 1487–1509, 2020.

[58] Alan Wee-Chung Liew, Ngai-Fong Law, and Hong Yan. Missing value imputation for gene expression data: computational techniques to recover missing data from available information. *Briefings in bioinformatics*, 12(5): 498–513, 2011.

[59] Craig K Enders. *Applied missing data analysis*. Guilford Press, 2010.

[60] Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

[61] John W Graham. Missing data analysis: Making it work in the real world. *Annual review of psychology*, 60:549–576, 2009.

[62] Todd D Little, Kyle M Lang, Wei Wu, and Mijke Rhemtulla. Missing data. *Developmental psychopathology*, pages 1–37, 2016.

[63] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322, 2010.

[64] Eric C Chi and Tianxi Li. Matrix completion from a computational statistics perspective. *Wiley Interdisciplinary Reviews: Computational Statistics*, 11(5): e1469, 2019.

[65] Tianxi Li. A note on the statistical view of matrix completion. *arXiv preprint arXiv:1605.03040*, 2016.

[66] Aude Sportisse, Claire Boyer, and Julie Josse. Imputation and low-rank estimation with missing not at random data. *Statistics and Computing*, 30(6): 1629–1643, 2020.

[67] Xiaofeng Liu, Xue Wang, Lang Zou, Jing Xia, and Wei Pang. Spatial imputation for air pollutants data sets via low rank matrix completion algorithm. *Environment international*, 139:105713, 2020.

[68] Wei Ma and George H Chen. Missing not at random in matrix completion: The effectiveness of estimating missingness probabilities under a low nuclear norm assumption. *arXiv preprint arXiv:1910.12774*, 2019.

[69] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[70] Irit Dinur and Igor Shinkar. On the conditional hardness of coloring a 4-colorable graph with super-constant number of colors. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 138–151. Springer, 2010.

[71] Moritz Hardt, Raghu Meka, Prasad Raghavendra, and Benjamin Weitz. Computational limits for matrix completion. In *Conference on Learning Theory*, pages 703–725, 2014.

[72] Benjamin Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12(12), 2011.

[73] Jicong Fan and Tommy WS Chow. Non-linear matrix completion. *Pattern Recognition*, 77:378–394, 2018.

[74] Emmanuel J Candes and Benjamin Recht. Exact low-rank matrix completion via convex optimization. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 806–812. IEEE, 2008.

[75] Guangcan Liu and Ping Li. Low-rank matrix completion in the presence of high coherence. *IEEE Transactions on Signal Processing*, 64(21):5623–5633, 2016.

[76] Xavier Alameda-Pineda, Elisa Ricci, Yan Yan, and Nicu Sebe. Recognizing emotions from abstract paintings using non-linear matrix completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5240–5248, 2016.

[77] Yudong Chen. Incoherence-optimal matrix completion. *IEEE Transactions on Information Theory*, 61(5):2909–2923, 2015.

[78] Raghu Meka, Prateek Jain, Constantine Caramanis, and Inderjit S Dhillon. Rank minimization via online learning. In *Proceedings of the 25th International Conference on Machine learning*, pages 656–663, 2008.

[79] Mehran Mesbahi and George P Papavassilopoulos. On the rank minimization problem over a positive semidefinite linear matrix inequality. *IEEE Transactions on Automatic Control*, 42(2):239–243, 1997.

[80] Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, PhD thesis, Stanford University, 2002.

[81] Maryam Fazel, Haitham Hindi, and Stephen P Boyd. A rank minimization heuristic with application to minimum order system approximation. In *American Control Conference, 2001. Proceedings of the 2001*, volume 6, pages 4734–4739. IEEE, 2001. ISBN 0780364953.

[82] Benjamin Recht, Weiyu Xu, and Babak Hassibi. Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 3065–3070. IEEE, 2008. ISBN 1424431239.

[83] Zhang Liu and Lieven Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1235–1256, 2010.

[84] Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

[85] Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[86] Joel A Tropp. Just relax: Convex programming methods for identifying sparse signals in noise. *IEEE transactions on information theory*, 52(3):1030–1051, 2006.

[87] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996. ISSN 0035-9246.

[88] Debing Zhang, Yao Hu, Jieping Ye, Xuelong Li, and Xiaofei He. Matrix completion by truncated nuclear norm regularization. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2192–2199. IEEE, 2012.

[89] Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(3):471–501, 2010. ISSN 0036-1445 1095-7200. doi: 10.1137/070697835.

[90] Lu Liu, Wei Huang, and Di-Rong Chen. Exact minimum rank approximation via schatten p-norm minimization. *Journal of Computational and Applied Mathematics*, 267:218–227, 2014. ISSN 0377-0427.

[91] Feiping Nie, Heng Huang, and Chris Ding. Low-rank matrix recovery via efficient schatten p-norm minimization. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, pages 655–661. AAAI Press, 2012.

[92] Feiping Nie, Hua Wang, Xiao Cai, Heng Huang, and Chris Ding. Robust matrix completion via joint schatten p-norm and lp-norm minimization. In *2012 IEEE 12th International Conference on Data Mining*, pages 566–574. IEEE, 2012.

[93] Amir Kalev, Robert L Kosut, and Ivan H Deutsch. Quantum tomography protocols with positivity are compressed sensing protocols. *Npj Quantum Information*, 1:15018, 2015. ISSN 2056-6387.

[94] WenGu Chen and YaLing Li. Stable recovery of low-rank matrix via nonconvex schatten p-minimization. *Science China Mathematics*, 58(12):2643–2654, 2015. ISSN 1674-7283 1869-1862. doi: 10.1007/s11425-015-5081-6.

[95] Alessandro Benfenati, Emilie Chouzenoux, and J-C Pesquet. A proximal approach for a class of matrix optimization problems. *arXiv preprint arXiv:1801.07452*, 2018.

[96] Yuan Xie, Shuhang Gu, Yan Liu, Wangmeng Zuo, Wensheng Zhang, and Lei Zhang. Weighted schatten p-norm minimization for image denoising and background subtraction. *IEEE Transactions on Image Processing*, 25(10): 4842–4857, 2016. ISSN 1057-7149.

[97] Feiping Nie, Hua Wang, Heng Huang, and Chris Ding. Joint schatten p-norm and l p-norm robust matrix completion for missing value recovery. *Knowledge and Information Systems*, 42(3):525–544, 2015.

[98] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20

(4):1956–1982, 2010. ISSN 1052-6234 1095-7189. doi: 10.1137/080738970. URL http://epubs.siam.org/doi/abs/10.1137/080738970.

[99] Caihua Chen, Bingsheng He, and Xiaoming Yuan. Matrix completion via an alternating direction method. *IMA Journal of Numerical Analysis*, 32(1): 227–245, 2012. ISSN 0272-4979.

[100] Jos F Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999. ISSN 1055-6788.

[101] Reha H Tütüncü, Kim-Chuan Toh, and Michael J Todd. Solving semidefinite-quadratic-linear programs using sdpt3. *Mathematical Programming*, 95(2): 189–217, 2003.

[102] Jian-Feng Cai and Stanley Osher. Fast singular value thresholding without singular value decomposition. *Methods and Applications of Analysis*, 20(4): 335–352, 2013.

[103] Jian-Feng Cai and Stanley Osher. Fast singular value thresholding without singular value decomposition. *Methods and Applications of Analysis*, 20(4): 335–352, 2013. ISSN 1945-0001.

[104] Zaiwen Wen, Wotao Yin, and Yin Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4(4):333–361, 2012. ISSN 1867-2949.

[105] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2 (1):183–202, 2009. ISSN 1936-4954.

[106] Moritz Hardt. Understanding alternating minimization for matrix completion. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 651–660. IEEE, 2014. ISBN 1479965170.

[107] Po-Ling Loh and Martin J Wainwright. Regularized m-estimators with nonconvexity: Statistical and algorithmic theory for local optima. In *Advances in Neural Information Processing Systems*, pages 476–484, 2013.

[108] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth*

Annual ACM Symposium on Theory of Computing, pages 665–674. ACM, 2013. ISBN 1450320295.

[109] Ruoyu Sun and Zhi-Quan Luo. Guaranteed matrix completion via non-convex factorization. *IEEE Transactions on Information Theory*, 62(11):6535–6579, 2016. ISSN 0018-9448.

[110] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016.

[111] Samuel Burer and Renato DC Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003. ISSN 0025-5610.

[112] Benjamin Recht and Christopher Ré. Parallel stochastic gradient algorithms for large-scale matrix completion. *Mathematical Programming Computation*, 5(2):201–226, 2013. ISSN 1867-2949.

[113] Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 713–719. ACM, 2005. ISBN 1595931805.

[114] Aleksandr Aravkin, Rajiv Kumar, Hassan Mansour, Ben Recht, and Felix J Herrmann. Fast methods for denoising matrix completion formulations, with applications to robust seismic data interpolation. *SIAM Journal on Scientific Computing*, 36(5):S237–S266, 2014. ISSN 1064-8275.

[115] Stephen Boyd, Reza Zadeh, Corinne Horn, and Madeleine Udell. Generalized low rank models. *Foundations and Trends® in Machine Learning*, 9(1):1–118, 2016. ISSN 1935-8237 1935-8245. doi: 10.1561/2200000055.

[116] Yuanzhi Li, Yingyu Liang, and Andrej Risteski. Recovery guarantee of weighted low-rank approximation via alternating minimization. In *International Conference on Machine Learning*, pages 2358–2367, 2016.

[117] Raghunandan H. Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56 (6):2980–2998, 2010. ISSN 0018-9448. doi: 10.1109/tit.2010.2046205.

[118] Yudong Chen and Martin J Wainwright. Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025*, 2015.

[119] Moritz Hardt and Mary Wootters. Fast matrix completion without the condition number. In *Conference on Learning Theory*, pages 638–678, 2014.

[120] Robert M Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 43–52. IEEE, 2007. ISBN 0769530184.

[121] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009. ISSN 0018-9162. doi: 10.1109/mc.2009.263.

[122] Justin P Haldar and Diego Hernando. Rank-constrained solutions to linear matrix equations using powerfactorization. *IEEE Signal Processing Letters*, 16(7):584–587, 2009. ISSN 1070-9908.

[123] Sebastian Mika, Bernhard Schölkopf, Alexander J Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel pca and de-noising in feature spaces. In *NIPS*, volume 11, pages 536–542, 1998.

[124] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(11), 2008.

[125] Si Si, Kai-Yang Chiang, Cho-Jui Hsieh, Nikhil Rao, and Inderjit S Dhillon. Goal-directed inductive matrix completion. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1165–1174, 2016.

[126] Fanhua Shang, Yuanyuan Liu, Fanjie Shang, Hongying Liu, Lin Kong, and Licheng Jiao. A unified scalable equivalent formulation for schatten quasi-norms. *Mathematics*, 8(8):1325, 2020.

[127] Xing Xu, Li He, Huimin Lu, Atsushi Shimada, and Rin-Ichiro Taniguchi. Non-linear matrix completion for social image tagging. *IEEE Access*, 5: 6688–6696, 2016.

[128] Zhaoliang Chen, Wei Zhao, and Shiping Wang. Kernel meets recommender systems: A multi-kernel interpolation for matrix completion. *Expert Systems with Applications*, 168:114436, 2021.

[129] Andrews Sobral, Thierry Bouwmans, and El-hadi Zahzah. Lrslibrary: Low-rank and sparse tools for background modeling and subtraction in videos. In *Robust Low-Rank and Sparse Matrix Decomposition: Applications in Image and Video Processing*. CRC Press, Taylor and Francis Group, 2015.

[130] Maciej Kula. Metadata embeddings for user and item cold-start recommendations. In Toine Bogers and Marijn Koolen, editors, *Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015.*, volume 1448 of *CEUR Workshop Proceedings*, pages 14–21. CEUR-WS.org, 2015. URL http://ceur-ws.org/Vol-1448/paper4.pdf.

[131] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. Librec: A java library for recommender systems. In *UMAP Workshops*, volume 4, pages 1–4, 2015.

[132] Madeleine Udell, Corinne Horn, Reza Zadeh, and Stephen Boyd. Generalized low rank models. *Foundations and Trends in Machine Learning*, 9(1), 2016. ISSN 1935-8237. doi: 10.1561/2200000055. URL http://dx.doi.org/10.1561/2200000055.

[133] A. Sobral and E. Zahzah. Matrix and tensor completion algorithms for background model initialization: A comparative evaluation. *Pattern Recognition Letters*, 2016. ISSN 01678655. doi: 10.1016/j.patrec.2016.12.019.

[134] Alex Rubinsteyn and Sergey Feldman. fancyimpute: An imputation library for python, 2016. URL https://github.com/iskandr/fancyimpute.

[135] Nicolas Hug. Surprise: A python library for recommender systems. *Journal of Open Source Software*, 5(52):2174, 2020.

[136] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Kaiyuan Li, Yushuo Chen, Yujie Lu, Hui Wang, Changxin Tian, Xingyu Pan, et al. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. *arXiv preprint arXiv:2011.01731*, 2020.

[137] Tony Duan. Lightweight python library for in-memory matrix completion., 2020. URL https://github.com/tonyduan/matrix-completion.

[138] Muhan Zhang and Yixin Chen. Inductive matrix completion based on graph neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ByxxgCEYDS.

[139] Scott Graham, Jun-Ki Min, and Tao Wu. Microsoft recommenders: tools to accelerate developing recommender systems. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 542–543, 2019.

[140] Andreas Argyriou, Miguel González-Fierro, and Le Zhang. Microsoft recommenders: Best practices for production-ready recommendation systems. In *Companion Proceedings of the Web Conference 2020*, pages 50–51, 2020.

[141] Tom Vander Aa, Imen Chakroun, and Thomas J Ashby. Smurff: a high-performance framework for matrix factorization. In *2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, pages 304–308. IEEE, 2019.

[142] T. Nykodym, T. Kraljevic, A. Wang, and Wong W. *Generalized Linear Modeling with H2O*, March 2020. URL http://docs.h2o.ai/h2o/latest-stable/h2o-docs/booklets/GLMBooklet.pdf.

[143] David Cortes. Cold-start recommendations in collective matrix factorization. *arXiv preprint arXiv:1809.00366*, 2018.

[144] Matei Zaharia, Reynold S Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J Franklin, et al. Apache spark: a unified engine for big data processing. *Communications of the ACM*, 59(11):56–65, 2016.

[145] Julie A Frearson and Iain T Collie. Hts and hit finding in academia–from chemical genomics to drug discovery. *Drug discovery today*, 14(23-24): 1150–1158, 2009.

[146] Fabian Zanella, James B Lorens, and Wolfgang Link. High content screening: seeing is believing. *Trends in biotechnology*, 28(5):237–245, 2010.

[147] Sophia Yohe and Bharat Thyagarajan. Review of clinical next-generation sequencing. *Archives of pathology & laboratory medicine*, 141(11):1544–1557, 2017.

[148] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[149] Mati Karelson, Victor S Lobanov, and Alan R Katritzky. Quantum-chemical descriptors in qsar/qspr studies. *Chemical Reviews*, 96(3):1027–1044, 1996.

[150] Nina Nikolova and Joanna Jaworska. Approaches to measure chemical similarity–a review. *QSAR & Combinatorial Science*, 22(9-10):1006–1026, 2003.

[151] Artem Cherkasov, Eugene N Muratov, Denis Fourches, Alexandre Varnek, Igor I Baskin, Mark Cronin, John Dearden, Paola Gramatica, Yvonne C Martin, Roberto Todeschini, et al. Qsar modeling: where have you been? where are you going to? *Journal of Medicinal Chemistry*, 57(12):4977–5010, 2014.

[152] Jérôme Hert, Peter Willett, David J Wilton, Pierre Acklin, Kamal Azzaoui, Edgar Jacoby, and Ansgar Schuffenhauer. New methods for ligand-based virtual screening: use of data fusion and machine learning to enhance the effectiveness of similarity searching. *Journal of Chemical Information and Modeling*, 46(2):462–470, 2006.

[153] Kenneth M Merz Jr, Dagmar Ringe, and Charles H Reynolds. *Drug design: structure-and ligand-based approaches*. Cambridge University Press, 2010.

[154] Steffen Renner and Gisbert Schneider. Scaffold-hopping potential of ligand-based similarity concepts. *ChemMedChem: Chemistry Enabling Drug Discovery*, 1(2):181–185, 2006.

[155] Dagmar Stumpfe, Huabin Hu, and Jürgen Bajorath. Advances in exploring activity cliffs. *Journal of Computer-Aided Molecular Design*, 34(9):929–942, 2020.

[156] Jürgen Bajorath. Duality of activity cliffs in drug discovery. *Expert Opinion on Drug Discovery*, 14(6):517–520, 2019.

[157] Dagmar Stumpfe, Huabin Hu, and Jürgenn Bajorath. Evolving concept of activity cliffs. *ACS Omega*, 4(11):14360–14368, 2019.

[158] Huabin Hu and Jürgen Bajorath. Introducing a new category of activity cliffs combining different compound similarity criteria. *RSC Medicinal Chemistry*, 11(1):132–141, 2020.

[159] Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

[160] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.

[161] Fabian Jirasek, Rodrigo AS Alves, Julie Damay, Robert A Vandermeulen, Robert Bamler, Michael Bortz, Stephan Mandt, Marius Kloft, and Hans Hasse. Machine learning in thermodynamics: Prediction of activity coefficients by matrix completion. *The Journal of Physical Chemistry Letters*, 11(3):981–985, 2020.

[162] Fabian Jirasek, Robert Bamler, and Stephan Mandt. Hybridizing physical and data-driven prediction methods for physicochemical properties. *Chemical Communications*, 56(82):12407–12410, 2020.

[163] Maurizio Pellecchia, Ivano Bertini, David Cowburn, Claudio Dalvit, Ernest Giralt, Wolfgang Jahnke, Thomas L James, Steve W Homans, Horst Kessler, Claudio Luchinat, et al. Perspectives on nmr in drug discovery: a technique comes of age. *Nature Reviews Drug Discovery*, 7(9):738–745, 2008.

[164] Serena Monaco, Louise E Tailford, Nathalie Juge, and Jesus Angulo. Differential epitope mapping by std nmr spectroscopy to reveal the nature of protein–ligand contacts. *Angewandte Chemie International Edition*, 56(48): 15289–15293, 2017.

[165] Toshihiko Sugiki, Kyoko Furuita, Toshimichi Fujiwara, and Chojiro Kojima. Current nmr techniques for structure-based drug discovery. *Molecules*, 23(1): 148, 2018.

[166] Darón I Freedberg and Philipp Selenko. Live cell nmr. *Annual Review of Biophysics*, 43:171–192, 2014.

[167] Gregg Siegal and Philipp Selenko. Cells, drugs and nmr. *Journal of Magnetic Resonance*, 306:202–212, 2019.

[168] Enrico Luchinat, Letizia Barbieri, Matteo Cremonini, Alessio Nocentini, Claudiu T Supuran, and Lucia Banci. Drug screening in human cells by nmr spectroscopy allows the early assessment of drug potency. *Angewandte Chemie International Edition*, 59(16):6535–6539, 2020.

[169] Xiaobo Qu, Maxim Mayzel, Jian-Feng Cai, Zhong Chen, and Vladislav Orekhov. Accelerated nmr spectroscopy with low-rank reconstruction.

Angewandte Chemie International Edition, 54(3):852–854, 2015. doi: https://doi.org/10.1002/anie.201409291. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/anie.201409291.

[170] Di Guo and Xiaobo Qu. Improved reconstruction of low intensity magnetic resonance spectroscopy with weighted low rank hankel matrix completion. *IEEE Access*, 6:4933–4940, 2018.

[171] J Almeida, J Prior, and Martin B Plenio. Computation of two-dimensional spectra assisted by compressed sampling. *The Journal of Physical Chemistry Letters*, 3(18):2692–2696, 2012.

[172] Hengfa Lu, Xinlin Zhang, Tianyu Qiu, Jian Yang, Di Guo, Zhong Chen, and Xiaobo Qu. A low rank hankel matrix reconstruction method for ultrafast magnetic resonance spectroscopy. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3269–3272. IEEE, 2017.

[173] Austin P Spencer, Boris Spokoyny, Supratim Ray, Fahad Sarvari, and Elad Harel. Mapping multidimensional electronic structure and ultrafast dynamics with single-element detection and compressive sensing. *Nature Communications*, 7(1):1–6, 2016.

[174] Tianyu Qiu, Wenjing Liao, Di Guo, Dongbao Liu, Xin Wang, Jian-Feng Cai, and Xiaobo Qu. An auto-parameter denoising method for nuclear magnetic resonance spectroscopy based on low-rank hankel matrix. *arXiv preprint arXiv:2001.11815*, 2020.

[175] MJ Bostock, DJ Holland, and Daniel Nietlispach. Improving resolution in multidimensional nmr using random quadrature detection with compressed sensing reconstruction. *Journal of Biomolecular NMR*, 68(2):67–77, 2017.

[176] Tianyu Qiu, Zi Wang, Huiting Liu, Di Guo, and Xiaobo Qu. Review and prospect: Nmr spectroscopy denoising and reconstruction with low-rank hankel matrices and tensors. *Magnetic Resonance in Chemistry*, 59(3): 324–345, 2021.

[177] Cancan Yi, Yong Lv, Han Xiao, and Shan Tu. Laser induced breakdown spectroscopy for quantitative analysis based on low-rank matrix approximations. *Journal of Analytical Atomic Spectrometry*, 32(11):2164–2172, 2017.

[178] Daniel J Holland and Lynn F Gladden. Less is more: how compressed sensing is transforming metrology in chemistry. *Angewandte Chemie International Edition*, 53(49):13330–13340, 2014.

[179] Xavier Andrade, Jacob N Sanders, and Alán Aspuru-Guzik. Application of compressed sensing to the simulation of atomic systems. *Proceedings of the National Academy of Sciences*, 109(35):13928–13933, 2012.

[180] Jacob N Sanders, Xavier Andrade, and Alán Aspuru-Guzik. Compressed sensing for the fast computation of matrices: application to molecular vibrations. *ACS Central Science*, 1(1):24–32, 2015.

[181] Jacob N. Sanders. *Compressed Sensing for Chemistry*. PhD thesis, Graduate School of Arts & Sciences, Harvard University, 2016.

[182] Adam S Abbott, Justin M Turney, Boyi Zhang, Daniel GA Smith, Doaa Altarawy, and Henry F Schaefer III. Pes-learn: An open-source software package for the automated generation of machine learning models of molecular potential energy surfaces. *Journal of Chemical Theory and Computation*, 15 (8):4386–4398, 2019.

[183] Tibor Győri and Gábor Czakó. Automating the development of high-dimensional reactive potential energy surfaces with the robosurfer program system. *Journal of Chemical Theory and Computation*, 16(1):51–66, 2020. doi: 10.1021/acs.jctc.9b01006.

[184] Prashant Rai, Khachik Sargsyan, Habib Najm, and So Hirata. Sparse low rank approximation of potential energy surfaces with applications in estimation of anharmonic zero point energies and frequencies. *Journal of Mathematical Chemistry*, 57(7):1732–1754, 2019.

[185] Stephen Jon Quiton, Urbashi Mitra, and Shaama Mallikarjun Sharada. A matrix completion algorithm to recover modes orthogonal to the minimum energy path in chemical reactions. *The Journal of Chemical Physics*, 153(5): 054122, 2020.

[186] Alan Huang, Levi A Garraway, Alan Ashworth, and Barbara Weber. Synthetic lethality as an engine for cancer drug target discovery. *Nature Reviews Drug Discovery*, 19(1):23–38, 2020.

[187] Nigel J O'Neil, Melanie L Bailey, and Philip Hieter. Synthetic lethality and cancer. *Nature Reviews Genetics*, 18(10):613–623, 2017.

[188] Yong Liu, Min Wu, Chenghao Liu, Xiao-Li Li, and Jie Zheng. Sl^2mf: Predicting synthetic lethality in human cancers via logistic matrix factorization. *IEEE/ACM transactions on computational biology and bioinformatics*, 17(3): 748–757, 2019.

[189] Ajit P Singh and Geoffrey J Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 650–658, 2008.

[190] Herty Liany, Anand Jeyasekharan, and Vaibhav Rajan. Predicting synthetic lethal interactions using heterogeneous data sources. *Bioinformatics*, 36(7): 2209–2216, 2020.

[191] Peng Wang, Wen Zhu, Bo Liao, Lijun Cai, Lihong Peng, and Jialiang Yang. Predicting influenza antigenicity by matrix completion with antigen and antiserum similarity. *Frontiers in Microbiology*, 9:2500, 2018.

[192] Saee Paliwal, Alex de Giorgio, Daniel Neil, Jean-Baptiste Michel, and Alix MB Lacoste. Preclinical validation of therapeutic targets predicted by tensor factorization on heterogeneous graphs. *Scientific Reports*, 10(1):1–19, 2020.

[193] Duncan E Scott, Andrew R Bayly, Chris Abell, and John Skidmore. Small molecules, big targets: drug discovery faces the protein–protein interaction challenge. *Nature Reviews Drug Discovery*, 15(8):533–550, 2016.

[194] Ulrich Stelzl, Uwe Worm, Maciej Lalowski, Christian Haenig, Felix H Brembeck, Heike Goehler, Martin Stroedicke, Martina Zenkner, Anke Schoenherr, Susanne Koeppen, et al. A human protein-protein interaction network: a resource for annotating the proteome. *Cell*, 122(6):957–968, 2005.

[195] Olgica Milenkovic Wei Dai and Narayanan Santhanam Prasad. Low-rank matrix completion for inference of protein-protein interaction networks. *AIP Conference Proceedings*, 1281(1):1531–1534, 2010. doi: 10.1063/1.3498084. URL https://aip.scitation.org/doi/abs/10.1063/1.3498084.

[196] Jin Huang, Feiping Nie, and Heng Huang. *Robust discrete matrix completion*. AAAI Publications, 2013.

[197] Fen Pei, Qingya Shi, Haotian Zhang, and Ivet Bahar. Predicting protein–protein interactions using symmetric logistic matrix factorization. *Journal of Chemical Information and Modeling*, 61(4):1670–1682, 2021.

[198] Kuo-Chen Chou. Prediction of protein cellular attributes using pseudo-amino acid composition. *Proteins: Structure, Function, and Bioinformatics*, 43(3): 246–255, 2001.

[199] Ya-Nan Zhang, Xiao-Yong Pan, Yan Huang, and Hong-Bin Shen. Adaptive compressive learning for prediction of protein–protein interactions from primary sequence. *Journal of theoretical biology*, 283(1):44–52, 2011.

[200] Hua Wang, Heng Huang, Chris Ding, and Feiping Nie. Predicting protein–protein interactions from multimodal biological data sources via nonnegative matrix tri-factorization. *Journal of Computational Biology*, 20(4):344–358, 2013.

[201] Junyi Chen, Junhui Hou, and Ka-Chun Wong. Categorical matrix completion with active learning for high-throughput screening. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2020.

[202] Alan Talevi and Carolina L Bellera. Challenges and opportunities with drug repurposing: finding strategies to find alternative uses of therapeutics. *Expert Opinion on Drug Discovery*, 15(4):397–401, 2020.

[203] Yves A Lussier and James L Chen. The emergence of genome-based drug repositioning. *Science translational medicine*, 3(96):96ps35–96ps35, 2011.

[204] Maya A Farha and Eric D Brown. Drug repurposing for antimicrobial discovery. *Nature Microbiology*, 4(4):565–577, 2019.

[205] Sudeep Pushpakom, Francesco Iorio, Patrick A Eyers, K Jane Escott, Shirley Hopper, Andrew Wells, Andrew Doig, Tim Guilliams, Joanna Latimer, Christine McNamee, et al. Drug repurposing: progress, challenges and recommendations. *Nature Reviews Drug Discovery*, 18(1):41–58, 2019.

[206] Y Cha, T Erez, IJ Reynolds, D Kumar, J Ross, G Koytiger, R Kusko, B Zeskind, S Risso, E Kagan, et al. Drug repurposing from the perspective of pharmaceutical companies. *British journal of pharmacology*, 175(2):168–180, 2018.

[207] Rachel A Hodos, Brian A Kidd, Khader Shameer, Ben P Readhead, and Joel T Dudley. In silico methods for drug repurposing and pharmacology. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 8(3):186–210, 2016.

[208] Huimin Luo, Min Li, Shaokai Wang, Quan Liu, Yaohang Li, and Jianxin Wang. Computational drug repositioning using low-rank matrix approximation and randomized algorithms. *Bioinformatics*, 34(11):1904–1912, 2018.

[209] Yin-Ying Wang, Chunfeng Cui, Liqun Qi, Hong Yan, and Xing-Ming Zhao. Drpocs: drug repositioning based on projection onto convex sets. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(1):154–162, 2018.

[210] Mengyun Yang, Huimin Luo, Yaohang Li, and Jianxin Wang. Drug repositioning based on bounded nuclear norm regularization. *Bioinformatics*, 35 (14):i455–i463, 2019.

[211] Aanchal Mongia, Sanjay Kr Saha, Emilie Chouzenoux, and Angshul Majumdar. A computational approach to aid clinicians in selecting anti-viral drugs for covid-19 trials. *Scientific Reports*, 11(1):1–12, 2021.

[212] Aanchal Mongia, Stuti Jain, Emilie Chouzenoux, and Angshul Majumda. Deepvir–graphical deep matrix factorization for" in silico" antiviral repositioning: Application to covid-19. *arXiv preprint arXiv:2009.10333*, 2020.

[213] Sahil Manchanda and Ashish Anand. Representation learning of drug and disease terms for drug repositioning. In *2017 3rd IEEE International Conference on Cybernetics (CYBCONF)*, pages 1–6. IEEE, 2017.

[214] Wenjuan Zhang, Hunan Xu, Xiaozhong Li, Qiang Gao, and Lin Wang. Drimc: an improved drug repositioning approach using bayesian inductive matrix completion. *Bioinformatics*, 36(9):2839–2847, 2020.

[215] Xu Zhou, Enyu Dai, Qian Song, Xueyan Ma, Qianqian Meng, Yongshuai Jiang, and Wei Jiang. In silico drug repositioning based on drug-mirna associations. *Briefings in Bioinformatics*, 21(2):498–510, 2020.

[216] K Deepthi and AS Jereesh. Drug repositioning based on the target micrornas using bilateral-inductive matrix completion. *Molecular Genetics and Genomics*, 295(5):1305–1314, 2020.

[217] Yunda Hao, Menglan Cai, and Limin Li. Drug repositioning via matrix completion with multi-view side information. *IET Systems Biology*, 13(5): 267–275, 2019.

[218] Lun Yang and Pankaj Agarwal. Systematic drug repositioning based on clinical side-effects. *PLoS ONE*, 6(12):e28025, 2011.

[219] Hao Ye, Qi Liu, and Jia Wei. Construction of drug network based on side effects and its application for drug repositioning. *PLoS ONE*, 9(2):e87864, 2014.

[220] Zhicheng Li, Yang Li, Qiang Lei, and Qing Zhao. Protein structure estimation from nmr data by matrix completion. *European Biophysics Journal*, 46(6): 525–532, 2017.

[221] Jiansheng Wu, Qin Yin, Chengxin Zhang, Jingjing Geng, Hongjie Wu, Haifeng Hu, Xiaoyan Ke, and Yang Zhang. Function prediction for g protein-coupled receptors through text mining and induction matrix completion. *ACS Omega*, 4(2):3045–3054, 2019.

[222] Taishin Kin, Tsuyoshi Kato, Koji Tsuda, and Kiyoshi Asai. Protein classification via kernel matrix completion. *Genome Informatics*, 14:516–517, 2003.

[223] Kazi Lutful Kabir, Gopinath Chennupati, Raviteja Vangara, Hristo Djidjev, Boian S Alexandrov, and Amarda Shehu. Decoy selection in protein structure determination via symmetric non-negative matrix factorization. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 23–28. IEEE, 2020.

[224] Oleg Okun and Helen Priisalu. Fast nonnegative matrix factorization and its application for protein fold recognition. *EURASIP Journal on Advances in Signal Processing*, 2006:1–8, 2006.

[225] Guoxian Yu, Huzefa Rangwala, Carlotta Domeniconi, Guoji Zhang, and Zhiwen Yu. Protein function prediction with incomplete annotations. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 11(3):579–591, 2013.

[226] Xian Wei, Zhi-Cheng Li, Shi-Jian Li, Xu-Biao Peng, and Qing Zhao. Protein structure determination using a riemannian approach. *FEBS Letters*, 594(6): 1036–1051, 2020.

[227] Zhicheng Li, Shijian Li, Xian Wei, and Qing Zhao. Scaled alternating steepest descent algorithm applied for protein structure determination from

nuclear magnetic resonance data. *Journal of Computational Biology*, 26(9): 1020–1029, 2019.

[228] Zhicheng Li, Shijian Li, Xian Wei, Xubiao Peng, and Qing Zhao. Recovering the missing regions in crystal structures from the nuclear magnetic resonance measurement data using matrix completion method. *Journal of Computational Biology*, 27(5):709–717, 2020.

[229] Geoffrey S Ginsburg and Kathryn A Phillips. Precision medicine: from science to value. *Health Affairs*, 37(5):694–701, 2018.

[230] Michael R Kosorok and Eric B Laber. Precision medicine. *Annual Review of Statistics and Its Application*, 6:263–286, 2019.

[231] Margaret A Hamburg and Francis S Collins. The path to personalized medicine. *New England Journal of Medicine*, 363(4):301–304, 2010.

[232] Reza Mirnezami, Jeremy Nicholson, and Ara Darzi. Preparing for precision medicine. *New England Journal of Medicine*, 366(6):489–491, 2012.

[233] Euan A Ashley. Towards precision medicine. *Nature Reviews Genetics*, 17(9): 507–522, 2016.

[234] Geoffrey S Ginsburg and Jeanette J McCarthy. Personalized medicine: revolutionizing drug discovery and patient care. *TRENDS in Biotechnology*, 19(12): 491–496, 2001.

[235] Inke R Koenig, Oliver Fuchs, Gesine Hansen, Erika von Mutius, and Matthias V Kopp. What is precision medicine? *European Respiratory Journal*, 50(4), 2017.

[236] Steven M Corsello, Rohith T Nagari, Ryan D Spangler, Jordan Rossen, Mustafa Kocak, Jordan G Bryan, Ranad Humeidi, David Peck, Xiaoyun Wu, Andrew A Tang, et al. Discovering the anticancer potential of non-oncology drugs by systematic viability profiling. *Nature Cancer*, 1(2):235–248, 2020.

[237] Giang TT Nguyen and Duc-Hau Le. A matrix completion method for drug response prediction in personalized medicine. In *Proceedings of the Ninth International Symposium on Information and Communication Technology*, pages 410–415, 2018.

[238] Chuanying Liu, Dong Wei, Ju Xiang, Fuquan Ren, Li Huang, Jidong Lang, Geng Tian, Yushuang Li, and Jialiang Yang. An improved anticancer drug-response prediction based on an ensemble method integrating matrix completion and ridge regression. *Molecular Therapy-Nucleic Acids*, 21:676–686, 2020.

[239] Lin Wang, Xiaozhong Li, Louxin Zhang, and Qiang Gao. Improved anticancer drug response prediction in cell lines using matrix factorization with similarity regularization. *BMC cancer*, 17(1):1–12, 2017.

[240] Betül Güvenç Paltun, Hiroshi Mamitsuka, and Samuel Kaski. Improving drug response prediction by integrating multiple data sources: matrix factorization, kernel and network-based approaches. *Briefings in bioinformatics*, 22(1): 346–359, 2021.

[241] Nathan H Lazar, Mehmet Gönen, and Kemal Sönmez. Batfled: Bayesian tensor factorization linked to external data. *arXiv preprint arXiv:1612.02965*, 2016.

[242] Minsik Oh, Sungjoon Park, Sangseon Lee, Dohoon Lee, Sangsoo Lim, Dabin Jeong, Kyuri Jo, Inuk Jung, and Sun Kim. Drim: A web-based system for investigating drug response at the molecular level by condition-specific multi-omics data integration. *Frontiers in Genetics*, 11, 2020.

[243] Cheng Qian, Amin Emad, and Nicholas D Sidiropoulos. A recursive framework for predicting the time-course of drug sensitivity. *Scientific Reports*, 10(1): 1–12, 2020.

[244] Colin Pawlowski. *Machine learning for problems with missing and uncertain data with applications to personalized medicine*. PhD thesis, Massachusetts Institute of Technology, 2019.

[245] Fatemeh Ahmadi Moughari and Changiz Eslahchi. A computational method for drug sensitivity prediction of cancer cell lines based on various molecular information. *PLoS ONE*, 16(4):1–31, 04 2021. doi: 10.1371/journal.pone.0250620.

[246] Muhammad Ammad-ud din, Elisabeth Georgii, Mehmet Gönen, Tuomo Laitinen, Olli Kallioniemi, Krister Wennerberg, Antti Poso, and Samuel Kaski. Integrative and personalized qsar analysis in cancer by kernelized bayesian matrix factorization. *Journal of Chemical Information and Modeling*, 54(8): 2347–2359, 2014. doi: 10.1021/ci500152b.

[247] Rong Li, Yongcheng Dong, Qifan Kuang, Yiming Wu, Yizhou Li, Min Zhu, and Menglong Li. Inductive matrix completion for predicting adverse drug reactions (adrs) integrating drug–target interactions. *Chemometrics and Intelligent Laboratory Systems*, 144:71–79, 2015.

[248] Prateek Jain and Inderjit S Dhillon. Provable inductive matrix completion. *arXiv preprint arXiv:1306.0626*, 2013.

[249] Hansaim Lim, Aleksandar Poleksic, Yuan Yao, Hanghang Tong, Di He, Luke Zhuang, Patrick Meng, and Lei Xie. Large-scale off-target identification using fast and accurate dual regularized one-class collaborative filtering and its application to drug repurposing. *PLoS Computational Biology*, 12(10): e1005135, 2016.

[250] Wen Zhang, Xinrui Liu, Yanlin Chen, Wenjian Wu, Wei Wang, and Xiaohong Li. Feature-derived graph regularized matrix factorization for predicting drug side effects. *Neurocomputing*, 287:154–162, 2018.

[251] Mohan Timilsina, Meera Tandan, Mathieu d'Aquin, and Haixuan Yang. Discovering links between side effects and drugs using a diffusion based method. *Scientific Reports*, 9(1):1–10, 2019.

[252] Jihyeob Mun, Gildon Choi, and Byungho Lim. A guide for bioinformaticians:'omics-based drug discovery for precision oncology. *Drug Discovery Today*, 2020.

[253] Yehudit Hasin, Marcus Seldin, and Aldons Lusis. Multi-omics approaches to disease. *Genome Biology*, 18(1):1–15, 2017.

[254] Sean Eddy, Laura H Mariani, and Matthias Kretzler. Integrated multi-omics approaches to improve classification of chronic kidney disease. *Nature Reviews Nephrology*, 16(11):657–668, 2020.

[255] Jussi Paananen and Vittorio Fortino. An omics perspective on drug target discovery platforms. *Briefings in Bioinformatics*, 21(6):1937–1953, 2020.

[256] Parampreet Kaur, Ashima Singh, and Inderveer Chana. Computational techniques and tools for omics data analysis: State-of-the-art, challenges, and future directions. *Archives of Computational Methods in Engineering*, pages 1–37, 2021.

[257] Anjun Ma, Adam McDermaid, Jennifer Xu, Yuzhou Chang, and Qin Ma. Integrative methods and practical challenges for single-cell multi-omics. *Trends in Biotechnology*, 2020.

[258] Eric C Chi, Hua Zhou, Gary K Chen, Diego Ortega Del Vecchyo, and Kenneth Lange. Genotype imputation via matrix completion. *Genome Research*, 23 (3):509–518, 2013.

[259] Aanchal Mongia, Debarka Sengupta, and Angshul Majumdar. Mcimpute: matrix completion based imputation for single cell rna-seq data. *Frontiers in Genetics*, 10:9, 2019.

[260] Wei Vivian Li and Jingyi Jessica Li. An accurate and robust imputation method scimpute for single-cell rna-seq data. *Nature Communications*, 9(1): 1–9, 2018.

[261] Wuming Gong, Il-Youp Kwak, Pruthvi Pota, Naoko Koyano-Nakagawa, and Daniel J Garry. Drimpute: imputing dropout events in single cell rna sequencing data. *BMC Bioinformatics*, 19(1):1–10, 2018.

[262] Akanksha Farswan, Anubha Gupta, Ritu Gupta, and Gurvinder Kaur. Imputation of gene expression data in blood cancer and its significance in inferring biological pathways. *Frontiers in Oncology*, 9:1442, 2020.

[263] Junlin Xu, Lijun Cai, Bo Liao, Wen Zhu, and JiaLiang Yang. Cmf-impute: an accurate imputation tool for single-cell rna-seq data. *Bioinformatics*, 36(10): 3139–3147, 2020.

[264] Divyanshu Talwar, Aanchal Mongia, Debarka Sengupta, and Angshul Majumdar. Autoimpute: Autoencoder based imputation of single-cell rna-seq data. *Scientific Reports*, 8(1):1–11, 2018.

[265] Chao Shang, Aaron Palmer, Jiangwen Sun, Ko-Shin Chen, Jin Lu, and Jinbo Bi. Vigan: Missing view imputation with generative adversarial networks. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 766–775. IEEE, 2017.

[266] Zhou Fang, Tianzhou Ma, Gong Tang, Li Zhu, Qi Yan, Ting Wang, Juan C Celedón, Wei Chen, and George C Tseng. Bayesian integrative model for multi-omics data with missingness. *Bioinformatics*, 34(22):3801–3808, 2018.

[267] Mengyun Yang, Lan Huang, Yunpei Xu, Chengqian Lu, and Jianxin Wang. Heterogeneous graph inference with matrix completion for computational drug repositioning. *Bioinformatics*, 36(22-23):5456–5464, 2020.

[268] Mengyun Yang, Huimin Luo, Yaohang Li, Fang-Xiang Wu, and Jianxin Wang. Overlap matrix completion for predicting drug-associated indications. *PLoS Computational Biology*, 15(12):e1007541, 2019.

[269] Aanchal Mongia, Emilie Chouzenoux, and Angshul Majumdar. Computational prediction of drug-disease association based on graph-regularized one bit matrix completion. *bioRxiv*, 2020.

[270] Wen Zhang, Yanlin Chen, Dingfang Li, and Xiang Yue. Manifold regularized matrix factorization for drug-drug interaction prediction. *Journal of Biomedical Informatics*, 88:90–97, 2018.

[271] Bo Jin, Haoyu Yang, Cao Xiao, Ping Zhang, Xiaopeng Wei, and Fei Wang. Multitask dyadic prediction and its application in prediction of adverse drug-drug interaction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[272] Mingqi Wu, Ye Luo, and Faming Liang. Accelerate training of restricted boltzmann machines via iterative conditional maximum likelihood estimation. *Statistics and Its Interface*, 12(3):377, 2019.

[273] Murat Can Cobanoglu, Chang Liu, Feizhuo Hu, Zoltán N Oltvai, and Ivet Bahar. Predicting drug–target interactions using probabilistic matrix factorization. *Journal of Chemical Information and Modeling*, 53(12):3399–3409, 2013.

[274] Qing Liao, Naiyang Guan, Chengkun Wu, and Qian Zhang. Predicting unknown interactions between known drugs and targets via matrix completion. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 591–604. Springer, 2016.

[275] Maryam Bagherian, Renaid B Kim, Cheng Jiang, Maureen A Sartor, Harm Derksen, and Kayvan Najarian. Coupled matrix–matrix and coupled tensor–matrix completion methods for predicting drug–target interactions. *Briefings in Bioinformatics*, 22(2):2161–2171, 2021.

[276] Limin Li and Menglan Cai. Drug target prediction by multi-view low rank embedding. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 16(5):1712–1721, 2017.

[277] Aanchal Mongia, Vidit Jain, Emilie Chouzenoux, and Angshul Majumdar. Deep latent factor model for predicting drug target interactions. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1254–1258. IEEE, 2019.

[278] Minhui Wang, Chang Tang, and Jiajia Chen. Drug-target interaction prediction via dual laplacian graph regularized matrix completion. *BioMed Research International*, 2018, 2018.

[279] Xin Fan, Yuxiang Hong, Xiaohu Liu, Yaogong Zhang, and Maoqiang Xie. Neighborhood constraint matrix completion for drug-target interaction prediction. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 348–360. Springer, 2018.

[280] Huiyuan Chen, Feixiong Cheng, and Jing Li. idrug: Integration of drug repositioning and drug-target prediction via cross-network embedding. *PLoS Computational Biology*, 16(7):e1008040, 2020.

[281] Nisar Wani and Khalid Raza. imtf-grn: Integrative matrix tri-factorization for inference of gene regulatory networks. *IEEE Access*, 7:126154–126163, 2019.

[282] Nagarajan Natarajan, Nikhil Rao, and Inderjit Dhillon. Pu matrix completion with graph information. In *2015 IEEE 6th international workshop on computational advances in multi-sensor adaptive processing (CAMSAP)*, pages 37–40. IEEE, 2015.

[283] Chuanyan Wu, Rui Gao, and Yusen Zhang. mhmda: human microbe-disease association prediction by matrix completion and multi-source information. *IEEE Access*, 7:106687–106693, 2019.

[284] Jian-Yu Shi, Hua Huang, Yan-Ning Zhang, Jiang-Bo Cao, and Siu-Ming Yiu. Bmcmda: a novel model for predicting human microbe-disease associations via binary matrix completion. *BMC Bioinformatics*, 19(9):85–92, 2018.

[285] Martin Stražar, Marinka Žitnik, Blaž Zupan, Jernej Ule, and Tomaž Curk. Orthogonal matrix factorization enables integrative analysis of multiple rna binding proteins. *Bioinformatics*, 32(10):1527–1535, 2016.

[286] Jin Li, Sai Zhang, Tao Liu, Chenxi Ning, Zhuoxuan Zhang, and Wei Zhou. Neural inductive matrix completion with graph convolutional networks for mirna-disease association prediction. *Bioinformatics*, 36(8):2538–2546, 2020.

[287] Chengqian Lu, Mengyun Yang, Feng Luo, Fang-Xiang Wu, Min Li, Yi Pan, Yaohang Li, and Jianxin Wang. Prediction of lncrna–disease associations based on inductive matrix completion. *Bioinformatics*, 34(19):3357–3364, 2018.

[288] Xiujin Wu, Wenhua Zeng, Yuxiu Xu, Beizhan Wang, Xiangrong Liu, Fan Lin, and Gil Alterovitz. Predicting of associations between microrna and human diseases based on multiple similarities and arbitrarily-order proximity network embedding. *IEEE Access*, 7:86625–86634, 2019.

[289] Jian-Qiang Li, Zhi-Hao Rong, Xing Chen, Gui-Ying Yan, and Zhu-Hong You. Mcmda: Matrix completion for mirna-disease association prediction. *Oncotarget*, 8(13):21187, 2017.

[290] Xing Chen, Lei Wang, Jia Qu, Na-Na Guan, and Jian-Qiang Li. Predicting mirna–disease association based on inductive matrix completion. *Bioinformatics*, 34(24):4256–4265, 2018.

[291] Yuhua Yao, Binbin Ji, Sihong Shi, Junlin Xu, Xiaofang Xiao, Enchao Yu, Bo Liao, and Jialiang Yang. Imdailm: inferring mirna-disease association by integrating lncrna and mirna data. *IEEE Access*, 8:16517–16527, 2019.

[292] Zong-Lan Zuo, Rui-Fen Cao, Pi-Jing Wei, Jun-Feng Xia, and Chun-Hou Zheng. Double matrix completion for circrna-disease association prediction. *BMC Bioinformatics*, 22(1):1–15, 2021.

[293] Jihwan Ha, Chihyun Park, Chanyoung Park, and Sanghyun Park. Improved prediction of mirna-disease associations based on matrix completion with network regularization. *Cells*, 9(4):881, 2020.

[294] Chang Tang, Hua Zhou, Xiao Zheng, Yanming Zhang, and Xiaofeng Sha. Dual laplacian regularized matrix completion for microrna-disease associations prediction. *RNA Biology*, 16(5):601–611, 2019.

[295] Li Peng, Manman Peng, Bo Liao, Guohua Huang, Wei Liang, and Keqin Li. Improved low-rank matrix recovery method for predicting mirna-disease association. *Scientific Reports*, 7(1):1–10, 2017.

[296] Damian Szklarczyk, Annika L Gable, Katerina C Nastou, David Lyon, Rebecca Kirsch, Sampo Pyysalo, Nadezhda T Doncheva, Marc Legeay, Tao Fang, Peer Bork, et al. The string database in 2021: customizable protein–protein

networks, and functional characterization of user-uploaded gene/measurement sets. *Nucleic Acids Research*, 49(D1):D605–D612, 2021.

[297] Katharina Schwarze, James Buchanan, Jilles M Fermont, Helene Dreau, Mark W Tilley, John M Taylor, Pavlos Antoniou, Samantha JL Knight, Carme Camps, Melissa M Pentony, et al. The complete costs of genome sequencing: a microcosting study in cancer and rare diseases from a single center in the united kingdom. *Genetics in Medicine*, 22(1):85–94, 2020.

[298] Art B Owen and Patrick O Perry. Bi-cross-validation of the svd and the nonnegative matrix factorization. *The annals of applied statistics*, 3(2): 564–594, 2009.

[299] Richard R Picard and R Dennis Cook. Cross-validation of regression models. *Journal of the American Statistical Association*, 79(387):575–583, 1984.

[300] Sylvain Arlot, Alain Celisse, et al. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010.

[301] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer New York, 2013. ISBN 9780387216065. URL https://books.google.ca/books?id=yPfZBwAAQBAJ.

[302] Bhargav Kanagal and Vikas Sindhwani. Rank selection in low-rank matrix approximations: A study of cross-validation for nmfs. In *Proc Conf Adv Neural Inf Process*, volume 1, pages 10–15, 2010.

[303] Rasmus Bro, Karin Kjeldahl, Age K Smilde, and HAL Kiers. Cross-validation of component models: a critical look at current methods. *Analytical and Bioanalytical Chemistry*, 390(5):1241–1251, 2008.

[304] K Ruben Gabriel. Le biplot-outil d'exploration de données multidimensionnelles. *Journal de la société française de statistique*, 143(3-4):5–55, 2002.

[305] Quan Gu and Kirill Veselkov. Bi-clustering of metabolic data using matrix factorization tools. *Methods*, 151:12–20, 2018.

[306] Patrick O Perry. *Cross-validation for unsupervised learning*. PhD thesis, Stanford University, 2009.

[307] Wei Fu and Patrick O Perry. Estimating the number of clusters using cross-validation. *Journal of Computational and Graphical Statistics*, 29(1):162–173, 2020.

[308] Svante Wold. Cross-validatory estimation of the number of components in factor and principal components models. *Technometrics*, 20(4):397–405, 1978.

[309] Richard M Karp. On-line algorithms versus off-line algorithms: How much is it worth to know the future? In *IFIP congress (1)*, volume 12, pages 416–429, 1992.

[310] Joan Boyar, Lene M Favrholdt, Christian Kudahl, Kim S Larsen, and Jesper W Mikkelsen. Online algorithms with advice: A survey. *ACM Computing Surveys (CSUR)*, 50(2):1–34, 2017.

[311] Donald E Knuth. *Art of computer programming, volume 2: Seminumerical algorithms*. Addison-Wesley Professional, 2014.

[312] Andrzej Chmielowiec. Algorithm for error-free determination of the variance of all contiguous subsequences and fixed-length contiguous subsequences for a sequence of industrial measurement data. *Computational Statistics*, pages 1–28, 2021.

[313] Muhan Zhang and Yixin Chen. Inductive matrix completion based on graph neural networks. *arXiv preprint arXiv:1904.12058*, 2019.

[314] Menglu Li, Mengya Liu, Yannan Bin, and Junfeng Xia. Prediction of circrna-disease associations based on inductive matrix completion. *BMC Medical Genomics*, 13(5):1–13, 2020.

[315] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2012.

[316] Silvia Gandy, Benjamin Recht, and Isao Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2): 025010, 2011.

[317] Zemin Zhang and Shuchin Aeron. Exact tensor completion using t-svd. *IEEE Transactions on Signal Processing*, 65(6):1511–1526, 2016.

[318] Tatsuya Yokota, Qibin Zhao, and Andrzej Cichocki. Smooth parafac decomposition for tensor completion. *IEEE Transactions on Signal Processing*, 64 (20):5423–5436, 2016.

[319] Qingquan Song, Hancheng Ge, James Caverlee, and Xia Hu. Tensor completion algorithms in big data analytics. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(1):1–48, 2019.

[320] Allen Liu and Ankur Moitra. Tensor completion made practical. *arXiv preprint arXiv:2006.03134*, 2020.

[321] Vasanth Siruvallur Murali, Didem Agac Cobanoglu, Michael Hsieh, Meyer Zinn, Venkat Sai Malladi, Jonathan Gesell, Noelle S Williams, Erik S Welf, Ganesh Venkataraman Raj, and Murat Can Cobanoglu. Cancer drug discovery as a low rank tensor completion problem. *bioRxiv*, 2021.

[322] Ran Wang, Shuai Li, Lixin Cheng, Man Hon Wong, and Kwong Sak Leung. Predicting associations among drugs, targets and diseases by tensor decomposition for drug repositioning. *BMC Bioinformatics*, 20(26):1–19, 2019.

[323] Zhuliu Li, Wei Zhang, R Stephanie Huang, and Rui Kuang. Learning a low-rank tensor of pharmacogenomic multi-relations from biomedical networks. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 409–418. IEEE, 2019.

Chapter 2

A New Framework for Protein Allostery Prediction from NMR Data

2.1 Introduction

Known as the second secret of life [1], protein allostery was originally proposed by Monod *et al.* to explain the inhibitory effect that binding a ligand has on a distant active site in multimeric proteins [2–4]. This concept was later expanded to monomeric proteins. Allostery is a very special way of regulating protein functions, especially for enzymes: allosteric enzymes are regulated by the binding of an effector at a site that's topologically distant from the active site[1]. The effector can be a small molecule, another protein, DNA, or another macromolecule [5]. Allostery is a universal phenomena in biological systems and is not restricted to proteins: allostery also serves a vital switch for DNA [6–8] and RNA [9–13]. Studies have revealed that the allostery intermediates the ligand binding at a distal site and macromolecule function regulation through conformational or dynamic changes. Biological consequences include activating downstream signaling cascades [14], molecular cooperativity, and cell-recognition specificity [15].

Targeting allosteric sites with drugs is an important and effective strategy because: (1) Allosteric compounds display better binding selectivity profiles by targeting the non-conserved allosteric sites while bypassing the endogenous co-factor binding site, which is often highly conserved within a protein family (or subfamily) due to evolutionary pressure [16]. (2) Reduced side effects can be achieved with higher compound selectivity with less off-target effects [17]. (3) Allosteric drugs pave a new way to overcome drug resistance, either targeting the allosteric site [18, 19] or in combination with orthosteric drugs which target the functional site [20]. The allostery related mutants recently reported [21, 22] are beyond the scope of this study.

(4) Targeting allosteric sites allows us to design and develop drugs targeting otherwise "undruggable" targets. For example, experimental studies using cultured cells have identified a highly-selective allosteric antagonist that does not impair the functioning of HIF-1, while disrupting HIF-2[23]. Several allosteric compounds targeting phosphatase, a "undruggable" target for cancers, are in clinical trial development [24, 25]. (5) Recently, covalent allosteric drugs have received intense attention [15, 26] because they combine the beneficial pharmacological properties of allosteric drugs and covalent drugs [27, 28].

Given the biological significance of protein allostery and success of allosteric drugs, numerous efforts have been put to identify allosteric site(s). As the most often used structural biology technique, X-ray crystallography have been used extensively to elucidate the structural basis of allostery in a target protein [29–32]. However, because X-ray crystallography is incapable of capturing the structural and dynamical changes of protein allostery in a time-dependent manner, nuclear magnetic resonance (NMR) provides valuable information about allostery at atomic resolution in the solution environment [33–36]. Other experimental methods to uncover allosteric effects include hydrogen-deuterium exchange mass spectrometry [37, 38], patch-clamp fluorometry [39], fluorescence resonance energy transfer (FRET) [40], and atomic force microscopy [41].

Computational models for protein allostery are attractive because it is expensive to conduct experiments and some research facilities/tools/methods, like protein crystallography facilities, are not available to every research group. There are many different computational approaches to allostery, including communication network theory [5], thermodynamics, free energy landscape of population/conformational ensembles' shift, molecular dynamics change, evolutionary analysis [14, 42–45]. Molecular dynamics (MD) remains one of the most used computational methods for protein allostery, but other methods including Markov state model (MSM), normal mode analysis, elasticity-based methods, statistical coupling analysis, evolutionary analysis, are also used [41, 46, 47].

Chemical shift covariance analysis (CHESCA) decodes chemical shift data from NMR experiments to reveal allosteric and structural information [48]. The main idea of CHESCA is to use singular value decomposition (SVD) to perform (linear) dimensionality reduction on integrated NMR data. The resulting principle components define a linear manifold, mapping chemical shifts to structural information. CHESCA has been successfully applied to study allostery in protein kinase A [49] and hyperpolarization-activated cyclic nucleotide-gated (HCN) [50], eukaryotic protein kinase [51], *etc.*.

While CHESCA is very powerful, there is the potential for further improvements. First, incomplete NMR data is not uncommon and traditional approaches (disregarding entire residues if their NMR data is incomplete) leads to significant information loss. Secondly, the original CHESCA method uses only used chemical shifts, but NMR experiments also provide information about peak heigh and line width, which also encode information about protein dynamics. Finally, a linear combination of the Nitrogen and Hydrogen chemical shifts is used to define a single composite variable. The definition of the combined chemical shift variable relies on human intervention based on physical intuitions: different people use different coefficients.

In this study, we report a new computational framework for allostery prediction that resolves these problems. A synthetic dataset was generated to facilitate our understanding of the model selection procedure for matrix completion. Then we used matrix completion based on iterative SVD for missing value imputation, taking line width into consideration. Feature engineering was conducted with robust principal component analysis (RPCA) [52] and SVD, leading to an decreased dimensionality of input data. The underlying assumption of CHESCA study is that the chemical shifts of allostery-related residues either increase or decrease due to perturbations associated with the binding. Qualitatively, similar changes occur to residues in the active site upon binding. We use standard derivation (std) to measure residues' response to different perturbations, specifically different concentrations and different ligands. A Gaussian mixture model was used for clustering.

2.2 Methods and Materials

2.2.1 Synthetic Data Generation

In order to generate a set of incomplete matrices, we generate a set of random low-rank matrices and then hide some values as missing data points. Specifically, we define entries in the full matrix with:

$$\mathbf{X}_{m;p} = \sum_{\substack{k_1+k_2+k_3+\cdots+k_p\leq K;\\ k_1,k_2,k_3,\cdots,k_p>0}} \frac{c_{k_1,k_2,k_3,\cdots;p}}{(k_1+k_2+k_3+\cdots+k_p)!} \prod_{d=1}^{D} P_{k_d}(y_{md}) \tag{2.1}$$

where $P_k(y)$ is the Legendre polynomial, $c_{k_1,k_2,k_3,\cdots;p} \in [-1,1]$ is a set of random selected numbers within the interval of $[-1,1]$, m is number of molecules/proteins, and p is the number of protein/molecular properties. D can be cast as the affiliated

dimensionality and k is the measurement of the degree of Legendre polynomial. The matrix elements can be efficiently evaluated using the recurrence relation

$$P_{k+1}(x) = \frac{2k+1}{k+1} x P_k(x) - \frac{k}{k+1} P_{k-1}(x) \tag{2.2}$$

with $P_{-1}(x) = 0$ and $P_0(x) = 1$. Examples of Legendre polynomials are shown below (up to $n = 5$):

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ P_2(x) &= \frac{1}{2}(3x^2 - 1) \\ P_3(x) &= \frac{1}{2}(5x^3 - 3x) \\ P_4(x) &= \frac{1}{8}(35x^4 - 30x^2 + 3) \\ P_5(x) &= \frac{1}{8}(63x^5 - 70x^3 + 15x) \end{aligned} \tag{2.3}$$

A matrix $\mathbf{X}_0 \in \mathbb{R}^{500\times100}$ was first generated with `numpy` and `Scipy` using Equation 2.1, where we set $m = 500$, $p = 100$, $D = 5$ and $k = 3$ for Legendre polynomials. To quantify the effect of noise on prediction accuracy of matrix completion, random noise matrices were added to $\mathbf{X}$ by implementing

$$\mathbf{X} = \mathbf{X}_0 + z\mathbf{X}_{noise} \tag{2.4}$$

where z denotes the noise level of $[0.00, 0.01, 0.05, 0.10, 0.50, 1.00]$ and $\mathbf{X}_{noise} \in [-1, 1]$ denotes random noise drawn from a uniform distribution.

2.2.2 NMR Data Preprocessing

The exchange proteins directly activated by cAMP (EPAC) is used as the sample model in this study [48] The NMR data for EPAC was provided by Dr. Giuseppe Melacini in the Department of Chemistry and Chemical Biology at McMaster University. The experimental data includes several perturbations of EPAC, i.e., NMR data for the *apo* form of EPAC and its conformers when different ligands are bound. (The *apo* form denotes a protein wit nothing bound to it.) A concerted response to non-covalent perturbations in a protein indicates an allosteric regulation network. All the chemical structures are shown in Figure 2.1.

More specifically, there are 5 states of EPAC in this study: the *apo* form and EPAC bound to cAMP, Sp-cAMPS, Rp-cAMPS, 2'-OMe-AMP. EPAC in its *apo*

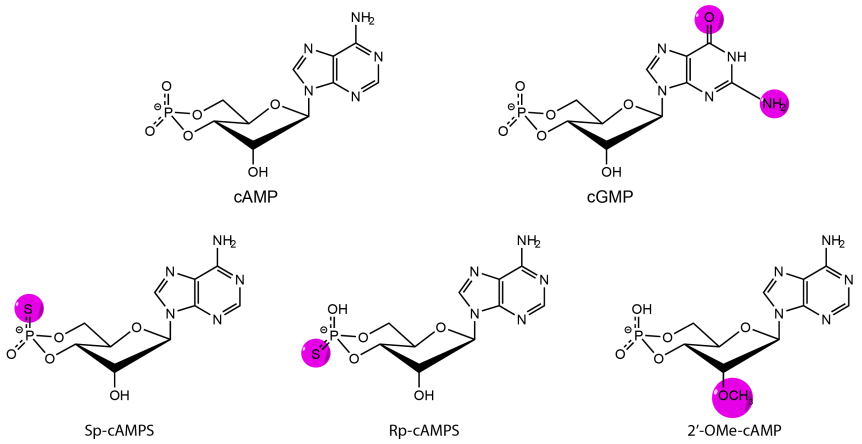


Fig. 2.1 Chemical structures that are used to build the perturbation library of EPAC. The atoms or groups that are different from cAMP are highlighted in pink bubble. The cGMP is also listed.

form and bound to Rp-cAMPS are inactive; EPAC is activated in its other forms. For the selected *apo* and bound states, there is data at varying concentrations, so that combining all the NMR data, we get a matrix $\mathbf{X} \in \mathbb{R}^{145\times52}$ with 145 residues and 52 NMR measurements (chemical shifts and line widths) after removing rows with all the data points missing.

Exploratory data analysis (EDA) was conducted to better understand the data. First, the data density were computed for columns in the matrix, which is the fraction of observed values in all the data

$$\rho_i = \frac{counting(\mathbf{X}_i \neq NaN)}{counting(\mathbf{X}_i)} \tag{2.5}$$

where ρ_i denotes the density for column i in $\mathbf{X}$, $counting$ denotes a function to count number of elements in data and $\mathbf{X}_i$ denotes i-th column in $\mathbf{X}$. Then we can easily define sparsity, which is the proportion of missing values for each data type,

$$s_i = 1 - \rho_i \tag{2.6}$$

Moreover, box plotting and violin plotting were done for chemical shifts and line widths of N and H atoms respectively to understand the data distribution as show in Figure 2.3.

To mitigate the problem of very different scales and mean-values for the chemical shifts and line widths of N and H, we applied a logarithmic transformation (Figure A.2), followed by data standardization using `Scikit-Learn`. The stan-

dardization makes the data (nearly) normally distributed with zero mean and unit variance.

2.2.3 Cross-validation and Hyperparameter Optimization for Matrix Completion

We used iterative SVD for matrix completion [53, 54] as implemented in *fancyimpute* [55]. After experimenting with synthetic data, we found that *convergence_threshold, svd_algorithm, init_fill_method* do not change the performance; neither does the maximum iteration number, *max_iters* as long as it is large enough (we chose 250, which suffices). The only hyperparameter that the results were sensitive to was the matrix's *rank*. We attempted to find the optimal rank by grid searching using the same cross-validation procedures for both the synthetic data and the experimental EPAC NMR data.

For the EPAC data, we hold $x\%$ of original data points out as testing data and the remaining observed elements serve as the training data where
$x = 2, 5, 10, 15, 20, 25, 30, 40, 50, 60, 70$, Figure 2.4. Unlike conventional regression problems in machine learning, where the training/testing splitting results to entirely independent samples, in the context of matrix completion we must preserve the matrix's shape. Sampling over all possible ways to remove a given number of entries from the matrix is computationally intractable. Therefore, a counting threshold value is imposed, and the sampling procedure was terminated once all the observed elements have been sample for certain times, 150 times for EPAC and 100 times for the synthetic datasets.

Within each cross-validation, Welford's online algorithm is used to to compute the mean and the sample standard derivation of the predictions[56, 57]. Both population standard derivation and sample standard derivation were implemented following Knuth's recipe [56]. The problem of missing values in the matrix is also taken into account, which achieves close performance with the standard *numpy* function *nanmean()* and *nanstd()*.

2.2.4 Denoising Line Width NMR Data with Robust PCA and Singular Value Decomposition

There are 13 measured states for EPAC and 4 NMR measurements for each residue each state (chemical shifts and line widths for the N and H atoms in each residue), so the experimental data can be represented as a matrix $\mathbf{X}_0 \in \mathbf{R}^{145\times52}$, Table A.1. The chemical shifts highly accuracy with very low noise to signal ratio, but the

line width measurements are noisy. Therefore, RPCA was employed to denoise the line width data by reshaping $\mathbf{X}_{lw} \in \mathbb{R}^{145\times 26}$ to $\mathbf{X}_{lw} \in \mathbb{R}^{1885\times 2}$, the columns of the latter are line width for N and H atoms respectively. The denoising is done with https://github.com/dganguli/robust-pca.

Examining the denoised line width, $\mathbf{X}_{lw}^{denoised} \in \mathbb{R}^{1885\times 2}$, with SVD, we noticed that there is only one numerically significant singular value. (Specifically, the largest singular value is 1.88×10^{3} and the second is merely 1.33×10^{-13}. This means that the denoised line-width data has effective rank 1. The coefficients of the right-singular vectors share almost the same magnitude, implying that the numerically significant feature is the sum of line widths for N and H, $\mathbf{X}_{lw}^{comb} \in \mathbb{R}^{1885\times 1}$ to replace the original $\mathbf{X} \in \mathbb{R}^{1885\times 2}$. By augmenting with chemical shift data, we are given $\mathbf{X}_{new} \in \mathbb{R}^{1885\times 3}$. We confirmed that similar conclusions (that the key features of the denoised NMR measurement matrix are determined by the chemical shifts and a single composite linewidth feature) could be obtained in different ways.

2.2.5 Clustering with Gaussian Mixture Model and a Distance Matrix

We used the data of *apo* structure as reference and all the data points were processing by subtracting off the data for this reference, which reduces the data into a 1740×36 matrix. The data was standardized to have mean zero and interquartile range one; the interquartile range was chosen due to its robustness to outliers. The rescaling allows N chemical shifts, H chemical shifts, and the line-width-sum to be considered on equal footing, despite their widely different scales.

A fundamental assumption in this study is that the NMR data for binding-related residues correlate with each other well and this holds for the allosteric residues as well. This inspires us to use the standard derivation (std) as a distance measurement for the 12×3 NMR values for one residues. The root-mean-square deviation of each residue's chemical shifts and line-width-sum from the *apo* reference is considered a measure of how sensitive the residue is to ligand binding; this gives a manageable feature matrix $\mathbf{D} \in \mathbb{R}^{145\times 3}$. A Gaussian mixture model was developed to divide the residues into two clusters using `Scikit-Learn` [58], with parameters n_init of 50, covariance_type of "full" [58].

2.3 Results and Discussions

2.3.1 Missing Value Imputation of NMR Data

2.3.1.1 Formulation of Matrix Completion Problem for NMR Data

Matrix completion computational methods, especially the matrix factorization algorithms on the basis of low-rank assumption, have gained popularity after its success in NetFlix prize competition [59–62]. The data structure of NetFlix problem resembles NMR data that we use in this study, as shown in Figure 2.2 (A) and (B) where residues resemble user and NMR measurements (such as chemical shift and line width of NMR spectroscopy) resemble items. Moreover, the underlying assumption of the matrix completion algorithms holds for the NMR data in this study. The basic assumption of using PCA in CHESCA is the intrinsic low-rank structure of NMR data, making it possible to project NMR data into two latent factors to understand protein allostery. This low-rank hypothesis has also been used for NMR spectroscopy reconstruction [63–66], which further rationalizes the usage of matrix completion to impute the missing values.

Take the NMR data for EPAC as an example, there are 145 residues but full data for every residue is not available in every EPAC variant. The sparsity as defined in Equation 2.5 is approximately 13.67% on average, Figure 2.3 (A). Our goal is to get an estimation of missing values given the observations. Mathematically speaking, the objective function is defined as

$$\begin{aligned} &\min && \operatorname{rank}(\mathbf{Z}) \\ &\text{subject to} && \sum_{(i,j)\in\mathbf{\Omega}} \left(\mathbf{X}_{ij} - \mathbf{Z}_{ij}\right)^2 \leq \delta \end{aligned} \tag{2.7}$$

where $\mathbf{X} \in \mathbb{R}^{145\times 52}$ denotes the whole NMR matrix for EPAC, Ω denotes the indices of observed values in NMR matrix, $\mathbf{Z} \in \mathbb{R}^{145\times 52}$ denotes the estimation of original matrix $\mathbf{X}$ and δ is the threshold value to determine the convergence.

We employed iterative SVD to solve this problem [53] as implemented in `fancyimpute` [55]. The main idea is to initialize the matrix by filling the missing values with zeros and decompose the filled matrix with SVD. By taking the biggest k eigenvalues and the corresponding eigenvectors, we can construct a matrix as an estimation and compute the error of estimation. The algorithm achieves the optimization in an iterative manner, where k is a hyperparameter that must beoptimized.

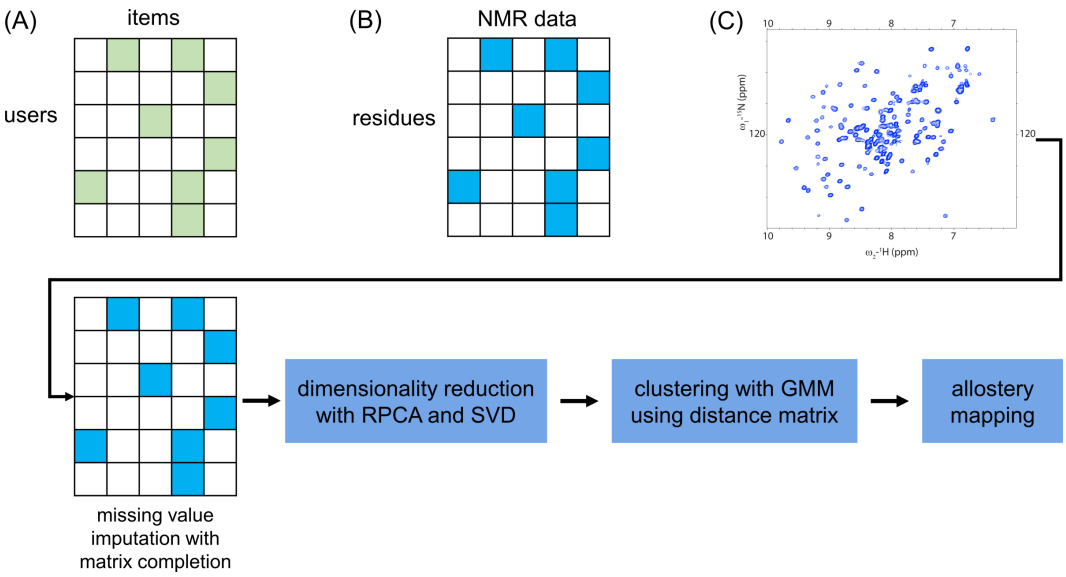


Fig. 2.2 Problem formulation and general workflow of the computational model to map the protein allostery. (A). Matrix completion for recommendation systems where each row of the matrix represents users and each column represents items (such as commercial products or movies). The missing values are denotes in light blue. (B). Matrix completion for NMR data imputation where each row denotes residue name and each column denotes NMR measurements namely chemical shift and line width of N and H atoms respectively. (C). General workflow of the computational model which includes missing value imputation, data denoising with RPCA (for line width only), dimensional reduction with SVD, clustering with GMM using a newly defined distance matrix.

2.3.1.2 Cross-validation and Hyperparameter Optimization for Synthetic Data

To get a better understanding of the behavior of the learning curve, we constructed a matrix $\mathbf{X} \in \mathbb{R}^{500\times100}$ with random noise ratio *null* (noise ratio=0.00), 0.01, 0.05, 0.10, 0.50 and 1.00 where all the values are within the interval $[-1, 1]$. In this study, 15% of data points were masked as missing values for all the synthetic data in accordance with the sparsity of EPAC data. The concept of synthetic data has been widely used in the machine learning field because synthetic data can be designed to mimic real data, whilst nonetheless remaining controllable, with (exact) tests available for predicted values. In this study, we held out 15% of the synthetic data to characterize model performance, and the ranks of all the synthetic matrices considered were 31. More details of generating the synthetic data can be found in subsection 2.2.1.

For each matrix with different levels of noise, different percents of data points (2%, 5%, 10%, 15%, 20%, 30%, 40%, 50%, 60%, 70%) were hold out as testing data to exam the effect of different sampling sizes of testing data. For a specific size of testing data, a list of ranks were tested, $k \in [1, 3, 5, \cdots, 97, 99]$ and resampling-based

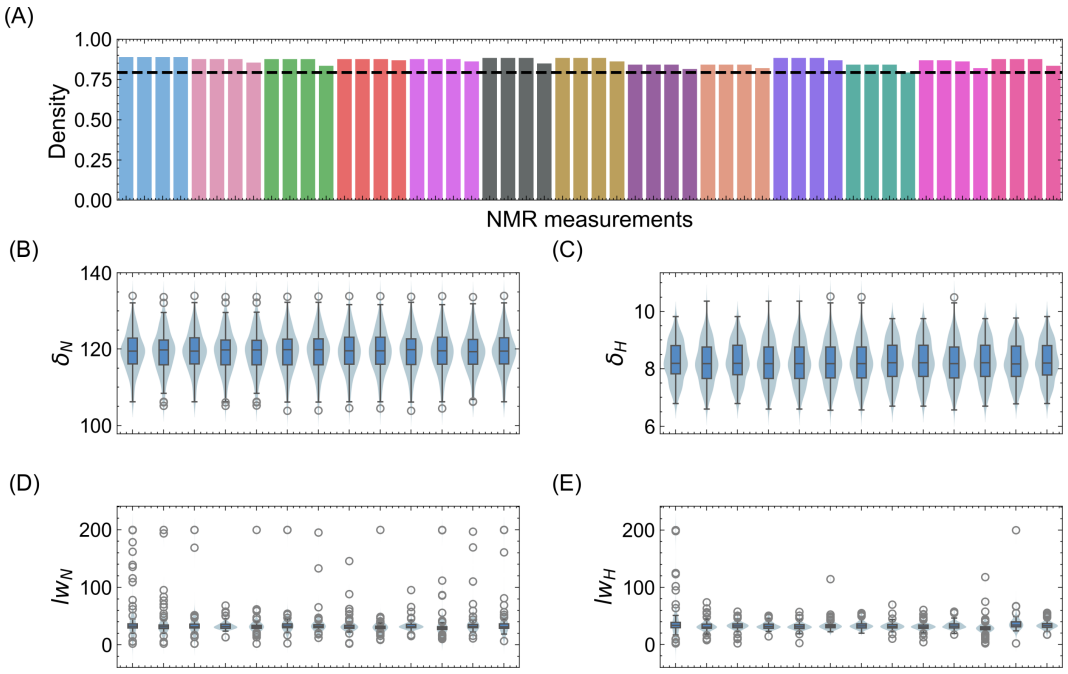


Fig. 2.3 Exploratory data analysis for EPAC NMR data. (A) Column density NMR data where for each residue, there are 4 different measurements, chemical shift and line width of N and H respectively. The complete NMR measurements are listed in Table A.1. (B) Box and violin plots for data distribution of chemical shift of N atoms.(C) Box and violin plots for data distribution of chemical shift of H atoms. (D) Box and violin plots for data distribution of line width of N atoms. (E) Box and violin plots for data distribution of line width of H atoms.

strategy was used for each selected rank k to ensure that each observed data point has been sampled for 100 times, Figure 2.4. In order to compute the predicted mean value and std, Welford's online algorithm [56, 57] was used. Our implementation supports computation of sparse matrices. The mean values returned by Welford's algorithm can be used to compute mean absolute error (MAE), which gives us a learning curve. This provides us information on the optimal rank. For more details of Welford's algorithm, please refer to section A.2.

By examining the learning curves of data at different noise levels, we can learn that the fitting of training data converges very quickly but the prediction of testing subsets behave quite differently (cf. Figure A.3, Figure A.4, Figure A.5, Figure A.6, Figure A.7 and Figure A.8). As seen in the learning curves of data without any noise, Figure A.3, the more data we take out from the original data as the testing data, the model becomes less robust. Take the case of holding 50% testing data out for example (Figure A.3), starting from rank 1 to 20, model performance is increased for both the training and testing data. But when it goes over rank of 20, it quickly climbs up with an increased MAE. This is caused by the number of data points is

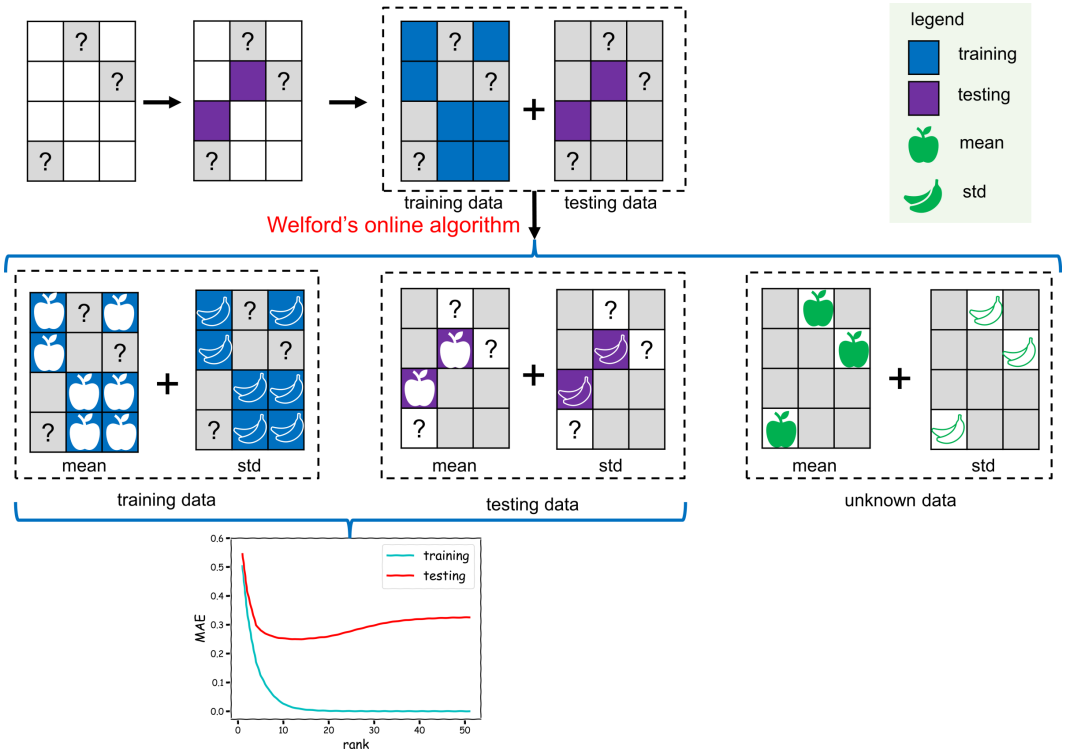


Fig. 2.4 Cross-validation of matrix completion for missing value imputation. We perform data splitting of the sparse input matrix with a fixed ratio for multiple times until each data point of observed values has been sampled at least 150 times. Welford's online algorithm is used to compute the mean and std. The question mark denotes the missing values in the original NMR matrix. The blue squared box implies the training data and the purple squared box represents testing data. Mean value of fitting or predictions are shown with apples and std of fitting or predictions are shown with bananas. Grey boxes denote missing values in the matrix.

not sufficient to fit the models with higher ranks, suggesting the model becomes under-determined. In addition, when holding less data points out, we have more data to fit or characterize the model, making the model less biased. In the case of no noise, selecting 2% data as testing subsets is the optimal choice.

Another interesting finding is that iterative SVD is somewhat insensitive to noise. The MAE values are less than 0.05 with a consistent pattern of different hold-out percents for data with noise level less than 0.10. When the noise goes up to 0.50 (Figure A.7), using 2% of testing data is not the optimal because the selected data points become underrepresented. This is more obvious for data with noise level of 1.00, Figure A.8. Therefore, the optimal rank should be selected based on different hold-outs, but not a single one because the noise level of data is unknown in real-world applications.

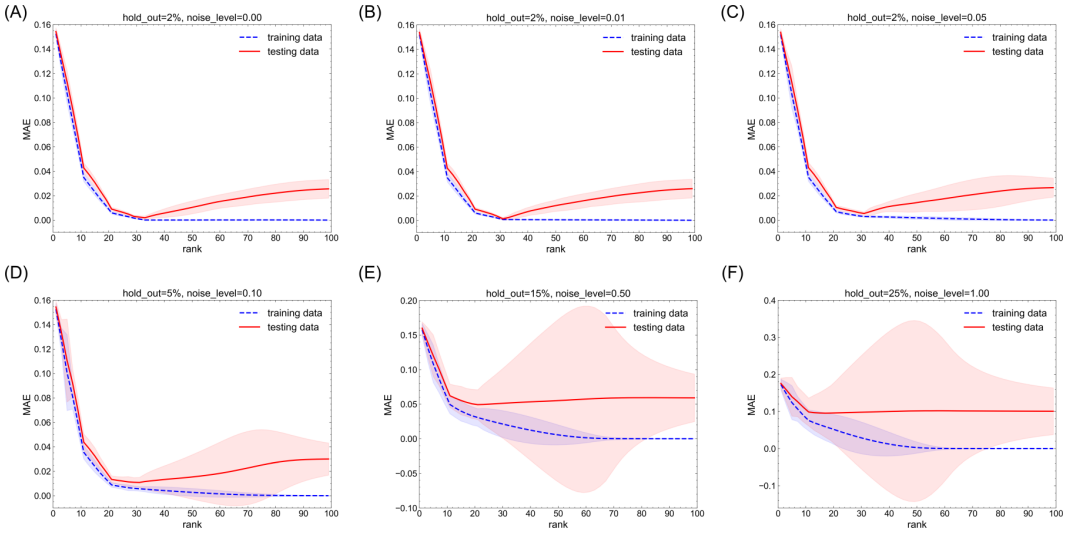


Fig. 2.5 Selected learning curves for various synthetic dataset at different noise levels. The dashed blue line denotes MAE of training data with filled area for std of averaged std. The solid red line is for MAE of testing data with filled area for std of averaged std.

Therefore, we selected the optimal hold-outs as testing data for data at various noise levels. More specifically, we select 2%, 2%, 2%, 5%, 15% and 25% for data with noise level of 0.00, 0.01, 0.05, 0.10, 0.50 and 1.00 respectively, as shown in Figure 2.4. The trend is clear that the testing error increases together with noise levels. When noise is within 0.05 of the original data, the MAE and the corresponding standard deviation are acceptable.

Further analysis of the predictions of missing values suggest the selected iterative SVD method is capable of recovering the incomplete matrices (cf. Figure A.9, Figure A.10, Figure A.11, Figure A.12, Figure A.13, and Figure A.14). It can be found that the predicted values get closer to actual values when ranks climbs up from 1 to 27 and the prediction errors increase for data with zero, 0.01 and 0.05 noise when we have bigger ranks. This suggests that the optimal rank for the synthetic data is 27 which comes with smaller std values than that of 29. This is consistent with results of previous discussions. It is noticed that the predictions can be affected by the noise in the data when the noise level goes over 0.50.

2.3.1.3 Cross-validation and Hyperparameter Optimization for NMR Data

All the results of synthetic data rationalize our hypothesis that matrix completion can be used for missing values imputation of NMR data because they are both low-rank and small-size. The same computational experiments for cross-validation and

hyperparameter optimization were conducted as that for synthetic data by sampling a list of ranks, $[1, 2, 3, \cdots, 51, 52]$.

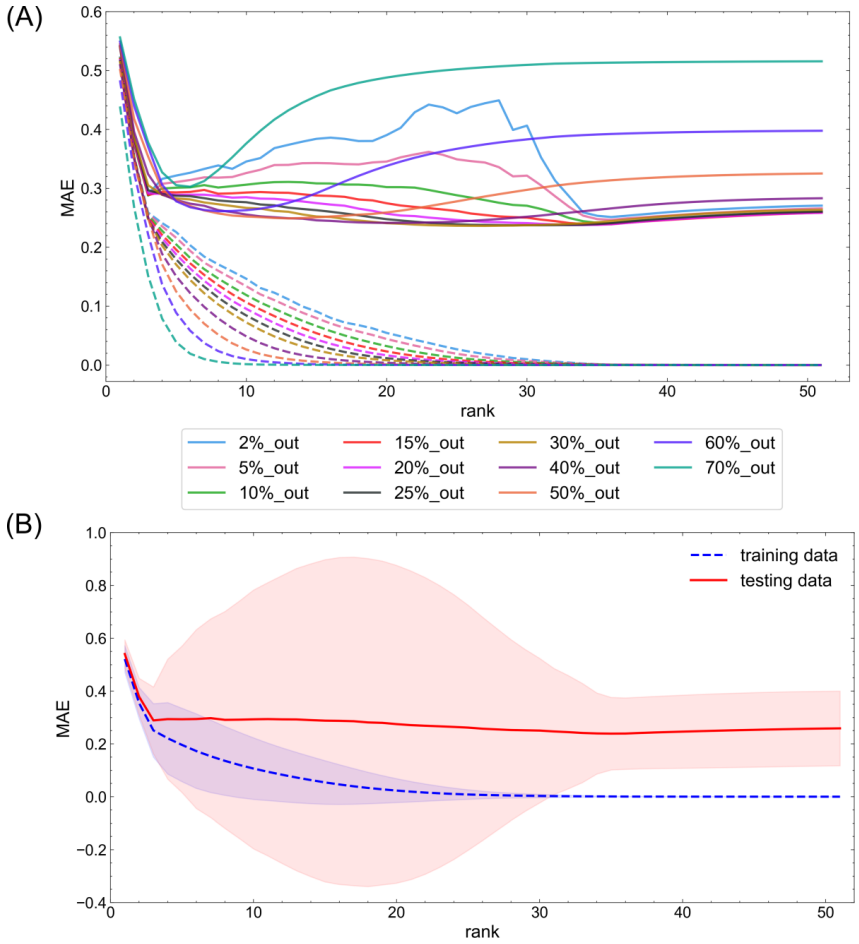


Fig. 2.6 Learning curve with different ranks for EPAC NMR data. (A). All the learning curves for EPAC. (B). Learning curve with 15% data as testing data. The dashed lines on the bottom denote the MAE for training data and the solid lines on the top denote the MAE for testing data. The std are represented in the filled areas.

As shown in Figure 2.6, all the curves for training data look smooth which implies that the more data we throw away, the lower rank we need to converge to zero error. This is because we will need fewer latent features to represent our data when there is more missing data. For the learning curve of leaving $2\%, 5\%, 10\%$ of the data out, the test data may be unrepresentative and the model is not robust. When the more thant 40% of the data is thrown away, there is not enough training data for higher-rank models. Holding out about 15% percent of the entries seems optimal.

Based on the learning curves of EPAC and our understanding of model performance based on the synthetic data, 15% data were selected as testing data and the optimal rank should between 34 and 40 as in Figure 2.6 (A). Further analysis

shows that rank 35 is the optimal choice as it provides lowest mean absolute error of 0.2387 and a standard deviation of 0.1373. With rank less than 35, the standard deviation values of the testing data fluctuate significantly, but after that, the standard deviation values of the testing the standard deviation values of testing data become small and vary smoothly, and the standard deviations fluctuate dramatically; the standard deviation values become much smaller after the optimal rank. Moreover, the fluctuations in the predicted values of the training data are negligible beyond rank 35.

The Figure 2.6 (B) not only provides optimal rank, but also provides the information of how confident we are about the predictions. The prediction of missing values gives an approximate mean absolute error of 0.30 with a standard deviation of 0.10. We used rank 35 to impute the missing values, then copied the imputed values and the known measured data into a new full matrix of (measured and imputed) NMR data.

2.3.2 Allostery Predicted with Gaussian Mixture Model

Using the Gaussian mixture model described earlier on the variation in residues' properties relative to their values in the *apo* form, we were able to divide residues into binding-related residues (red dots; clustered together in the corner) and allostery-related residues (scattered); see Figure 2.7 (A). Our model can identify the residues that involved with the binding of the endogenous ligand cAMP (its analog S1P is show in Figure 2.7 (B)), which include Gly269, Leu 271, Ala272, Arg279 and Ala280. These residues form the binding pocket and they form hydrogen bonds with the cAMP except Leu271.

Comparing the predicted residues for allostery with that of CHESCA, it is found that these two models match well with the overlapped residues highlighted in Figure 2.7 (C). Our prediction shows that Leu207, Val218 and Leu219 are allosteric residues as the α_4 helix lies between them. More studies are needed to better understand the role of the residues that this approach newly predicts to contribute to allostery.

2.4 Conclusions

NMR has been widely used to study protein allostery. One of the key tools in such studies is chemical shift covariance analysis (CHESCA), where principle component analysis to process NMR data. However, the original CHESCA method

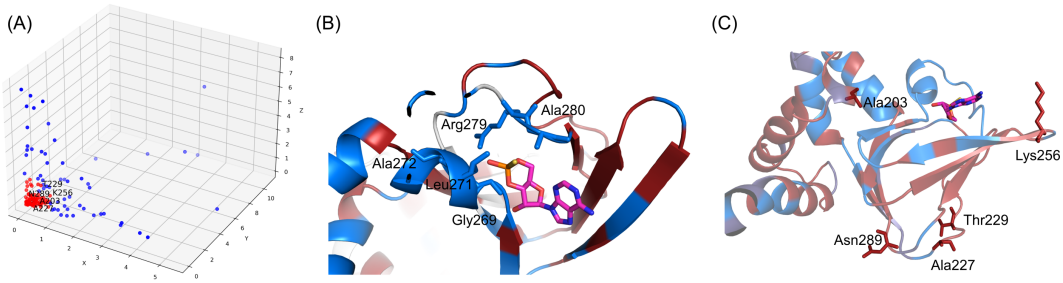


Fig. 2.7 Allostery predictions of EPAC. (A). Residue clustering of EPAC with GMM. The blue dots denote binding related residues and the red dots denote the allostery related residues. X, Y and Z axis represent chemical shift of N, H atoms and combine line width. (B). Residues related to the binding of endogenous ligands. (C). Residues related to allostery. The analog of cGMP, S1P, is shown in magenta in (B) and (C). These residues were selected based on the structural analysis in original CHESCA study [48]. The cartoon representation of EPAC uses red to denote the allostery related residues and blue for binding related.

was incapable of dealing with missing values, and missing data is ubiquitous in real NMR experiments. Moreover, CHESCA dos not use line-width data from NMR, and only uses a (user-chosen) linear combination of chemical shifts. To partially resolve these problems, we propose a new workflow to characterize protein allostery. First, we use matrix completion, with cross-validation used to select the optimal matrix rank, to impute the missing values for the nitrogen and hydrogen chemical shifts and linewidths for each residue of interest. Next, using the (full) data matrix of line widths, we used robust PCA to identify that there is only one key linewidth feature, name the sum of the nitrogen and hydrogen line widths. Finally, we constructed a feature matrix by using the root-mean-square deviation in nitrogen chemical shift, hydrogen chemical shift, and line-width-sum for each residue from their values in the *apo* form. Using a Gaussian mixture model, two clusters were found, which are identified as allosteric and binding-related residues.

A few problems remain to be further explored. The optimal rank for matrix completion is determined by analyzing the prediction errors, but the differences are very small. One promising way of dealing with this is to use the minimum description length (MDL) for model selection, the main idea behind which is that the optimal model is provided by that can compress the model to the greatest extent [67–69]. MDL has successfully applied to generalized linear regression, section A.3. Several studies have employed the informatic-theory MDL method for model selection in the matrix completion problems [70–72].

Another interesting direction is how to incorporate side information to the computational framework, either to improve the accuracy of missing value imputation

or improve the clustering. One potential solution is inductive matrix completion, which has been used for predictions of gene-disease associations [73], RNA-disease associations [74, 75] and microbe–disease associations [76].

References

[1] Aron W Fenton. Allostery: an illustrated definition for the 'second secret of life'. *Trends in Biochemical Sciences*, 33(9):420–425, 2008.

[2] Jacques Monod and François Jacob. General conclusions: teleonomic mechanisms in cellular metabolism, growth, and differentiation. In *Cold Spring Harbor symposia on quantitative biology*, volume 26, pages 389–401. Cold Spring Harbor Laboratory Press, 1961.

[3] Jean-Pierre Changeux. The feedback control mechanism of biosynthetic l-threonine deaminase by l-isoleucine. In *Cold Spring Harbor symposia on quantitative biology*, volume 26, pages 313–318. Cold Spring Harbor Laboratory Press, 1961.

[4] Jacques Monod, Jean-Pierre Changeux, and Francois Jacob. Allosteric proteins and cellular control systems. *Journal of molecular biology*, 6(4):306–329, 1963.

[5] Nina M Goodey and Stephen J Benkovic. Allosteric regulation and catalysis emerge via a common route. *Nature Chemical Biology*, 4(8):474–482, 2008.

[6] Jonathan B. Chaires. Allostery: Dna does it, too. *ACS Chemical Biology*, 3 (4):207–209, 2008. doi: 10.1021/cb800070s. URL https://doi.org/10.1021/cb800070s.

[7] Sangjin Kim, Erik Broströmer, Dong Xing, Jianshi Jin, Shasha Chong, Hao Ge, Siyuan Wang, Chan Gu, Lijiang Yang, Yi Qin Gao, et al. Probing allostery through dna. *Science*, 339(6121):816–819, 2013.

[8] Xinliang Xu, Hao Ge, Chan Gu, Yi Qin Gao, Siyuan S Wang, Beng Joo Reginald Thio, James T Hynes, X Sunney Xie, and Jianshu Cao. Modeling spatial correlation of dna deformation: Dna allostery in protein binding. *The Journal of Physical Chemistry B*, 117(42):13378–13387, 2013.

[9] David Rueda, Gregory Bokinsky, Maria M Rhodes, Michael J Rust, Xiaowei Zhuang, and Nils G Walter. Single-molecule enzymology of rna: essential functional groups impact catalysis from a distance. *Proceedings of the National Academy of Sciences*, 101(27):10066–10071, 2004.

[10] Alexey G Vitreschak, Dimitry A Rodionov, Andrey A Mironov, and Mikhail S Gelfand. Riboswitches: the oldest mechanism for the regulation of gene expression? *TRENDS in Genetics*, 20(1):44–50, 2004.

[11] Inés Li de la Sierra-Gallay, Olivier Pellegrini, and Ciarán Condon. Structural basis for substrate binding, cleavage and allostery in the trna maturase rnase z. *Nature*, 433(7026):657–661, 2005.

[12] Stéphane Thore, Marc Leibundgut, and Nenad Ban. Structure of the eukaryotic thiamine pyrophosphate riboswitch with its regulatory ligand. *Science*, 312 (5777):1208–1211, 2006.

[13] Alexander Serganov, Anna Polonskaia, Anh Tuân Phan, Ronald R Breaker, and Dinshaw J Patel. Structural basis for gene regulation by a thiamine pyrophosphate-sensing riboswitch. *Nature*, 441(7097):1167–1171, 2006.

[14] Nikolay V Dokholyan. Controlling allosteric networks in proteins. *Chemical Reviews*, 116(11):6463–6487, 2016.

[15] Ruth Nussinov and Chung-Jung Tsai. Allostery in disease and in drug discovery. *Cell*, 153(2):293–305, 2013.

[16] Ruth Nussinov, Chung-Jung Tsai, and Peter Csermely. Allo-network drugs: harnessing allostery in cellular networks. *Trends in pharmacological sciences*, 32(12):686–693, 2011.

[17] Ashok Kumar Grover. Use of allosteric targets in the discovery of safer drugs. *Medical Principles and Practice*, 22(5):418–426, 2013.

[18] Yong Jia, Cai-Hong Yun, Eunyoung Park, Dalia Ercan, Mari Manuia, Jose Juarez, Chunxiao Xu, Kevin Rhee, Ting Chen, Haikuo Zhang, et al. Overcoming egfr (t790m) and egfr (c797s) resistance with mutant-selective allosteric inhibitors. *Nature*, 534(7605):129–132, 2016.

[19] Jeffrey R Simard, Sabine Klüter, Christian Grütter, Matthäus Getlik, Matthias Rabiller, Haridas B Rode, and Daniel Rauh. A new screening assay for allosteric inhibitors of csrc. *Nature Chemical Biology*, 5(6):394–396, 2009.

[20] Duan Ni, Yun Li, Yuran Qiu, Jun Pu, Shaoyong Lu, and Jian Zhang. Combining allosteric and orthosteric drugs to overcome drug resistance. *Trends in Pharmacological Sciences*, 2020.

[21] Rafal M Pielak, Jason R Schnell, and James J Chou. Mechanism of drug inhibition and drug resistance of influenza a m2 channel. *Proceedings of the National Academy of Sciences*, 106(18):7379–7384, 2009.

[22] Shaoyong Lu, Yuran Qiu, Duan Ni, Xinheng He, Jun Pu, and Jian Zhang. Emergence of allosteric drug-resistance mutations: new challenges for allosteric drug discovery. *Drug Discovery Today*, 25(1):177–184, 2020.

[23] Thomas H Scheuermann, Qiming Li, He-Wen Ma, Jason Key, Lei Zhang, Rui Chen, Joseph A Garcia, Jacinth Naidoo, Jamie Longgood, Doug E Frantz, et al. Allosteric inhibition of hypoxia inducible factor-2 with small molecules. *Nature Chemical Biology*, 9(4):271, 2013.

[24] Asher Mullard. Phosphatases start shedding their stigma of undruggability. *Nature Reviews Drug Discovery*, 17(12):847–850, 2018.

[25] Ying-Nan P Chen, Matthew J LaMarche, Ho Man Chan, Peter Fekkes, Jorge Garcia-Fortanet, Michael G Acker, Brandon Antonakos, Christine Hiu-Tung Chen, Zhouliang Chen, Vesselina G Cooke, et al. Allosteric inhibition of shp2 phosphatase inhibits cancers driven by receptor tyrosine kinases. *Nature*, 535 (7610):148–152, 2016.

[26] Jörn Weisner, Ina Landel, Christoph Reintjes, Niklas Uhlenbrock, Marija Trajkovic-Arsic, Niklas Dienstbier, Julia Hardick, Swetlana Ladigan, Marius Lindemann, Steven Smith, et al. Preclinical efficacy of covalent-allosteric akt inhibitor borussertib in combination with trametinib in kras-mutant pancreatic and colorectal cancer. *Cancer Research*, 79(9):2367–2378, 2019.

[27] Ruth Nussinov and Chung-Jung Tsai. The design of covalent allosteric drugs. *Annual Review of Pharmacology and Toxicology*, 55:249–267, 2015.

[28] Jörn Weisner, Rajesh Gontla, Leandi van der Westhuizen, Sebastian Oeck, Julia Ketzer, Petra Janning, Andre Richters, Thomas Mühlenberg, Zhizhou Fang, Abu Taher, et al. Covalent-allosteric kinase inhibitors. *Angewandte Chemie International Edition*, 54(35):10313–10316, 2015.

[29] Stephen W Wright, Anthony A Carlo, Maynard D Carty, Dennis E Danley, David L Hageman, George A Karam, Carolyn B Levy, Mahmoud N Mansour, Alan M Mathiowetz, Lester D McClure, et al. Anilinoquinazoline inhibitors of fructose 1, 6-bisphosphatase bind at a novel allosteric site: synthesis, in vitro characterization, and x-ray crystallography. *Journal of Medicinal Chemistry*, 45(18):3865–3877, 2002.

[30] Joseph D Bauman, Disha Patel, Chhaya Dharia, Marc W Fromer, Sameer Ahmed, Yulia Frenkel, RSK Vijayan, J Thomas Eck, William C Ho, Kalyan Das, et al. Detecting allosteric sites of hiv-1 reverse transcriptase by x-ray crystallographic fragment screening. *Journal of Medicinal Chemistry*, 56(7): 2738–2746, 2013.

[31] Xiangyu Liu, Seungkirl Ahn, Alem W Kahsai, Kai-Cheng Meng, Naomi R Latorraca, Biswaranjan Pani, AJ Venkatakrishnan, Ali Masoudi, William I Weis, Ron O Dror, et al. Mechanism of intracellular allosteric β 2 ar antagonist revealed by x-ray crystal structure. *Nature*, 548(7668):480–484, 2017.

[32] Yuan Yao, Tong Huo, Yi-Lun Lin, Shenyou Nie, Fangrui Wu, Yuanda Hua, Jingyu Wu, Alexander R Kneubehl, Megan B Vogt, Rebecca Rico-Hesse, et al. Discovery, x-ray crystallography and antiviral activity of allosteric inhibitors of flavivirus ns2b-ns3 protease. *Journal of the American Chemical Society*, 141 (17):6832–6836, 2019.

[33] Gregory Manley and J Patrick Loria. Nmr insights into protein allostery. *Archives of Biochemistry and Biophysics*, 519(2):223–231, 2012.

[34] Charalampos G Kalodimos. Nmr reveals novel mechanisms of protein activity regulation. *Protein Science*, 20(5):773–782, 2011.

[35] Stephen Boulton and Giuseppe Melacini. Advances in nmr methods to map allosteric sites: from models to translation. *Chemical Reviews*, 116(11): 6267–6304, 2016.

[36] Bryan VanSchouwen and Giuseppe Melacini. Cracking the allosteric code of nmr chemical shifts. *Proceedings of the National Academy of Sciences*, 113 (34):9407–9409, 2016.

[37] Cesar A Ramirez-Sarmiento and Elizabeth A Komives. Hydrogen-deuterium exchange mass spectrometry reveals folding and allostery in protein-protein interactions. *Methods*, 144:43–52, 2018.

[38] Angela Patterson, Zhongchao Zhao, Elizabeth Waymire, Adam Zlotnick, and Brian Bothner. Dynamics of hepatitis b virus capsid protein dimer regulate assembly through an allosteric network. *ACS Chemical Biology*, 15(8):2273–2280, 2020.

[39] Solene N Lefebvre, Anais Menny, Marc Gielen, and Pierre-Jean Corringer. Allosteric transitions of pentameric ligand-gated ion channels studied by

fluorescence quenching to explore pathological mutations and pharmacological effectors. *Biophysical Journal*, 116(3):396a, 2019.

[40] Ryan J Durham, Nabina Paudyal, Elisa Carrillo, Nidhi Kaur Bhatia, David M Maclean, Vladimir Berka, Drew M Dolino, Alemayehu A Gorfe, and Vasanthi Jayaraman. Conformational spread and dynamics in allostery of nmda receptors. *Proceedings of the National Academy of Sciences*, 117(7):3839–3847, 2020.

[41] Galen Collier and Vanessa Ortiz. Emerging computational approaches for the study of protein allostery. *Archives of Biochemistry and Biophysics*, 538(1): 6–15, 2013.

[42] Chung-Jung Tsai and Ruth Nussinov. A unified view of "how allostery works". *PLoS Computational Biology*, 10(2):e1003394, 2014.

[43] Joanna F Swain and Lila M Gierasch. The changing landscape of protein allostery. *Current Opinion in Structural Biology*, 16(1):102–108, 2006.

[44] Shaoyong Lu, Shuai Li, and Jian Zhang. Harnessing allostery: a novel approach to drug discovery. *Medicinal Research Reviews*, 34(6):1242–1285, 2014.

[45] Qiang Cui and Martin Karplus. Allostery and cooperativity revisited. *Protein Science*, 17(8):1295–1307, 2008.

[46] Andre AST Ribeiro and Vanessa Ortiz. A chemical perspective on allostery. *Chemical Reviews*, 116(11):6488–6502, 2016.

[47] Ora Schueler-Furman and Shoshana J Wodak. Computational approaches to investigating allostery. *Current Opinion in Structural Biology*, 41:159–171, 2016.

[48] Rajeevan Selvaratnam, Somenath Chowdhury, Bryan VanSchouwen, and Giuseppe Melacini. Mapping allostery through the covariance analysis of nmr chemical shifts. *Proceedings of the National Academy of Sciences*, 108 (15):6133–6138, 2011.

[49] Madoka Akimoto, Rajeevan Selvaratnam, E Tyler McNicholl, Geeta Verma, Susan S Taylor, and Giuseppe Melacini. Signaling through dynamic linkers as revealed by pka. *Proceedings of the National Academy of Sciences*, 110(35): 14231–14236, 2013.

[50] Madoka Akimoto, Zaiyong Zhang, Stephen Boulton, Rajeevan Selvaratnam, Bryan VanSchouwen, Melanie Gloyd, Eric A Accili, Oliver F Lange, and Giuseppe Melacini. A mechanism for the auto-inhibition of hyperpolarization-activated cyclic nucleotide-gated (hcn) channel opening and its relief by camp. *Journal of Biological Chemistry*, 289(32):22205–22220, 2014.

[51] Jonggul Kim, Lalima G Ahuja, Fa-An Chao, Youlin Xia, Christopher L McClendon, Alexandr P Kornev, Susan S Taylor, and Gianluigi Veglia. A dynamic hydrophobic core orchestrates allostery in protein kinases. *Science advances*, 3(4):e1600663, 2017.

[52] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):1–37, 2011.

[53] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[54] Samuel F Buck. A method of estimation of missing values in multivariate data suitable for use with an electronic computer. *Journal of the Royal Statistical Society: Series B (Methodological)*, 22(2):302–306, 1960.

[55] Apache Software Foundation. fancyimpute. URL https://github.com/iskandr/fancyimpute.

[56] Donald E Knuth. *Art of computer programming, volume 2: Seminumerical algorithms*. Addison-Wesley Professional, 2014.

[57] BP Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[58] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

[59] Dietmar Jannach, Paul Resnick, Alexander Tuzhilin, and Markus Zanker. Recommender systems—beyond matrix completion. *Communications of the ACM*, 59(11):94–102, 2016.

[60] Luong Trung Nguyen, Junhan Kim, and Byonghyo Shim. Low-rank matrix completion: A contemporary survey. *IEEE Access*, 7:94215–94237, 2019.

[61] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013.

[62] Eric C Chi and Tianxi Li. Matrix completion from a computational statistics perspective. *Wiley Interdisciplinary Reviews: Computational Statistics*, 11(5): e1469, 2019.

[63] Jun Lang and Da Cheng. A fast low rank vandermonde factorization reconstruction method for non-uniformly sampled 2d nmr spectroscopy. *Digital Signal Processing*, 118:103249, 2021.

[64] Tianyu Qiu, Zi Wang, Huiting Liu, Di Guo, and Xiaobo Qu. Review and prospect: Nmr spectroscopy denoising and reconstruction with low-rank hankel matrices and tensors. *Magnetic Resonance in Chemistry*, 59(3):324–345, 2021.

[65] Di Guo, Hengfa Lu, and Xiaobo Qu. A fast low rank hankel matrix factorization reconstruction method for non-uniformly sampled magnetic resonance spectroscopy. *IEEE Access*, 5:16033–16039, 2017.

[66] Xiaobo Qu, Maxim Mayzel, Jian-Feng Cai, Zhong Chen, and Vladislav Orekhov. Accelerated nmr spectroscopy with low-rank reconstruction. *Angewandte Chemie International Edition*, 54(3):852–854, 2015.

[67] Peter D Grünwald. *The minimum description length principle*. MIT press, 2007.

[68] Mark H Hansen and Bin Yu. Minimum description length model selection criteria for generalized linear models. *Lecture Notes-Monograph Series*, pages 145–163, 2003.

[69] Mark H Hansen and Bin Yu. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454): 746–774, 2001.

[70] Mojtaba Abolfazli, Anders Host-Madsen, June Zhang, and Andras Bratincsak. Graph compression with application to model selection. *arXiv preprint arXiv:2110.00701*, 2021.

[71] Ignacio Ramírez and Guillermo Sapiro. Low-rank data modeling via the minimum description length principle. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2165–2168. IEEE, 2012.

[72] Ignacio Ramirez and Guillermo Sapiro. An mdl framework for sparse coding and dictionary learning. *IEEE Transactions on Signal Processing*, 60(6): 2913–2927, 2012.

[73] Nagarajan Natarajan and Inderjit S Dhillon. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics*, 30(12):i60–i68, 2014.

[74] Chengqian Lu, Mengyun Yang, Feng Luo, Fang-Xiang Wu, Min Li, Yi Pan, Yaohang Li, and Jianxin Wang. Prediction of lncrna–disease associations based on inductive matrix completion. *Bioinformatics*, 34(19):3357–3364, 2018.

[75] Jin Li, Sai Zhang, Tao Liu, Chenxi Ning, Zhuoxuan Zhang, and Wei Zhou. Neural inductive matrix completion with graph convolutional networks for mirna-disease association prediction. *Bioinformatics*, 36(8):2538–2546, 2020.

[76] Yahui Long, Jiawei Luo, Yu Zhang, and Yan Xia. Predicting human microbe–disease associations via graph attention networks with inductive matrix completion. *Briefings in Bioinformatics*, 22(3):bbaa146, 2021.

Chapter 3

Improved Solvation Free Energy Prediction with Δ-Learning

3.1 Introduction

Oral drug delivery is widely accepted and preferred because of it has greater convenient, more flexible dosing, and fewer side-effects than intravenous injections and other drug delivery methods[1]. Consequently, molecular solubility is among the key properties that determine the practical performance of a drug candidate [2, 3]. It is estimated that 40% of approved drugs, and 70% of candidate drugs in clinical trials, suffer from poor aqueous solubility [4]. When a compound's solubility is poor, it often indicates inadequate bioavailability, meaning that a drug cannot be administered orally [5]. Molecular solubility is thus a key component of *drug-likeness*, a qualitative measurement of how a candidate drug molecule resembles previously approved drugs [6–8]. Moreover, many chemical reactions happen in aqueous environments, so molecular solubility is a fundamental problem in chemical synthesis too.

Although there are ways to modify the drug formulation molecular structure to enhance solubility [9, 10], such modifications require tedious and expensive trial-and-error experimental verification [11]. It is preferable to screen for sufficient solubility early in the drug development pipeline.

Two types of solubility data are often reported: Gibbs free energies for solvation (often in kcal/mol or kJ/mol) and intrinsic solubility (often the common logarithm of the experimental solubility value in mol/L, $\log S_0$. The relationship of Gibbs free energies for solvation and the intrinsic solubility is [12]

$$\Delta G_{sol} = \Delta G_{sub} + \Delta G_{hyd} = -RT \ln (S_0 V_m) \tag{3.1}$$

where ΔG_{sol} denotes the solvation free energy, ΔG_{sub} denotes the free energy of sublimation, ΔG_{hyd} denotes the free energy of hydration (i.e., transferring molecules from the gas phase to the aqueous state), R is the molar gas constant, T is the temperature (298 K as the standard state), V_m is the molar volume of the crystal, and S_0 is the intrinsic solubility (in moles per liter).

In this study, we are focusing on the hydration free energy, which is an important physicochemical property for fundamental chemical problems, environmental engineering, mineral mining [13], *etc.*. It can be used to compute the Henry's law constant of aqueous solutions [14], soil sorption coefficients [15], and phase equilibrium constants [16, 17]. The hydration free energy is also involved with molecular recognition [18], ligand binding [19], pharmacokinetics [20], protein folding and aggregation [21, 22], protein stability [23], *etc.*. Its close relation to activity coefficients is also highlighted [24, 25].

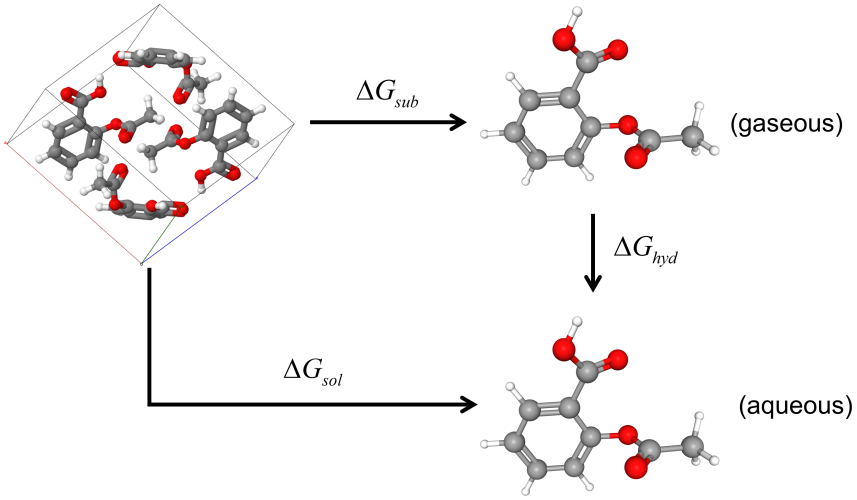


Fig. 3.1 Thermodynamic cycle for dissolving organic crystal structures in aqueous solution. Adapted from [12].

Measuring the molecular solubility experimentally is time-consuming and expensive, and small research groups may have very limited access to the facilities needed for high-throughput solubility measurements. This is one reason that computational methods for predicting molecular solubility, especially those based on machine-learning (ML) methods, are increasingly popular. Reported ML models include message passing neural networks [26], graph neural networks [27], transfer learning based models [28], Delfos [29], and MLSolvA [30].

In addition to emerging ML approaches for solubility prediction, there are traditional computational approaches for predicting solvation free energies including Monte Carlo and molecular dynamics (MD) simulations [31–36], molecular mechan-

ics based models [37, 38], quantum mechanics based models [39], and QM/MM based methods [40–42]. Unlike ML approaches, these traditional methods directly model the underlying physical laws that govern molecular solubility and can be categorized as explicit and implicit solvent models [43], depending on whether solvent molecules are explicitly including in the calculation (explicit) or not (implicit). For example, one of the most widely used implicit solvation models, the solvation model based on density (SMD), provides robust solvation free energy predictions based on continuum electrostatics and thermodynamics. Unfortunately, SMD sometimes gives errors of 10 $kcal/mol$ or more for neutral molecules, see Figure 3.3.

Attempts have been given to combine ML and QM or MD to improve the model accuracy. For example, a hybrid model of free energy perturbation (FEP) and ML was proposed to predict the hydration free energies of the FreeSolv dataset [44]. The recently-proposed machine-learning polarizable continuum solvation model (MLPCM) combines a neural network with data from the Minnesota solvation database [45]. New features engineered from molecular dynamics have been incorporated into ML algorithms, such as molecular dynamics fingerprints (MDFP) [46] and 3D-RISM hydration thermodynamic descriptors [47].

In this study, we propose a Δ-learning strategy using Gaussian process regression (GPR) to correct results from SMD calculations. Δ-learning is a very promising strategy to improve the accuracy of quantum chemical calculations with negligible computational cost [48]. Δ-learning can be seen as a composite technique where ML is used to correct the error from a computationally inexpensive first-principles calculation. The goal is to mimic the accuracy of demanding high-level calculations or experiments whilst retaining the tractability of low-level approximate computational models. Delta-learning has been successfully invoked to predict thermochemical properties [48–50], NMR chemical shifts [51], coupled-cluster energies from DFT densities [52], and so on. In our application, meticulously measured experimental free energies of solvation are modelled starting from their predicted value from inexpensive SMD calculations, ΔG_{smd}, and a ML-based error correction term, $\Delta\Delta G$:

$$\Delta G_{solv} = \Delta G_{smd} + \Delta\Delta G \tag{3.2}$$

This can easily turned into

$$\Delta\Delta G = \Delta G_{solv} - \Delta G_{smd} \tag{3.3}$$

where $\Delta\Delta G$ is the target value in our machine learning models.

Because the quantity of available data is low, we choose to learn $\Delta\Delta G$ using Gaussian process regression (GPR), an interpolation and regression algorithm that has been widely used in chemical (bio)physics [53–56]. An especially appealing feature of GPR is that it not only predicts $\Delta\Delta G$, but predicts the error in its prediction.

This chapter is organized as follows, Figure 3.2 (A). We will address how the molecular features were generated and how we select features for our model. There are 3 types of machine learning algorithms that are used in this study: GPR [57], k-nearest neighbors (KNN) [58], and decision trees [59]. All the hyperparameters were optimized and the performance of various models were evaluated. Last but not least, we will show how confident we are about our improved model. Our contribution are two-fold: we built improved predictive models for hydration free energy and a command line tool (in progress). This chapter is joint work with Juan Samuel Collins and Hanwen Zhang, two MITACS Globalink undergraduate researchers in our group.

3.2 Methods and Materials

3.2.1 Dataset Preparation

The initial dataset used in this study is CombiSolv-Exp [28] which contains 10145 experimental solvation energies for 291 solvents and 1368 solutes. The CombiSolv-Exp dataset is a curated data from MNSol (2275 records) [60], FreeSolv (560 records) [25], Compsol (3548 records) [61] and Abraham (6091 records) [62]. The data was downloaded from the supporting information of Ref. 28. Because of license issues, MNSol is not included in the downloadable file. We obtained MNSol from https://comp.chem.umn.edu/mnsol/ and converted UniChemXYZ files of MNSol to standard XYZ format with `IOData` [63] because UniChemXYZ files are not recognizable in commonly used cheminformatics packages. The XYZ files were further converted into SDF with `OpenBabel 3.1.1` [64] and International Chemical Identifier (InChI) formats with `RDKit 2021.03.1` [65].

Only records with binary mixtures and molecules with zero charges were selected in MNSol dataset. The records with xylene as solvent were also eliminated because it is a molecular mixture. We merged MNSol and the downloaded CombiSolv-Exp into one dataset, then removed NaN values in solute or solvents. All the duplicated records were removed as well. Self-solvation related records were also dropped. There are 1253 records left after filtering by setting water molecule as the solvent and selecting uncharged solute molecules. The dataset contains information of Simplified

Molecular Input Line Entry System (SMILES), InChI of solutes, and the average and standard deviation (std) of hydration free energies.

Data cleaning was performed on the curated dataset as described in Figure 3.2. Molecules with SMILES that were not recognizable by `RDKit` were removed. Molecules containing metal atoms like Na, Mg, and Fe were eliminated. Salts within the molecules were stripped and the molecules were neutralized. A further validation was also performed with `RDKit`. To avoid issues with protonation states, all carboxylic acids were deleted with SMARTS matching using `RDKit`.

3.2.2 3D Coordinates Generation and Conformer Searching

The 3D coordinates were generated with `RDKit` using the molecular SMILES as input, then the UFF force field to minimize the energy. Initially, 400 iterations were allowed for the energy minimization; when optimization failed to converge, the number of iterations was doubled. When UFF did not have parameters for the molecule in question, MMFF94s [66] was used instead.

Conformer searching was conducted with the Confab method [67] as implemented in `OpenBabel`; Confab systematically generates a diverse set of conformers. Inspired by the adaptive way of determining the number of conformers based on the number of rotatable bonds, n_{rtb}, the number of conformers is defined as[68]

$$n_{conf} = \begin{cases} 2, & \text{if } n_{rtb} \in \{0, 1\} \\ 2\left\lceil n_{rtb}^{1.8} \right\rceil, & \text{if } 1 < n_{rtb} \leq 12 \\ 200, & \text{if } n_{rtb} > 12 \end{cases} \tag{3.4}$$

where n_{conf} denotes the number of conformers, n_{rtb} denotes the number of rotatable bonds in a molecule, $\lceil\ \rceil$ denotes the ceiling operator. The n_{conf} is then multiplied by 4×10^6 followed by a conformer searching using MMFF94s force field without change the default of cut-off energy (50 $kcal/mol$) and RMSD values 0.5Å. All the energies of conformers were evaluated with Generalized Amber Force Field (gaff) [69] due to its good performance in optimizing small organic molecules. The top 5 conformers were save if available and only the conformer with the lowest energy is used.

3.2.3 Geometry Optimization and Computations of Hydration Free Energies

The top conformers were further optimized with density-functional theory (DFT) using the ωB97XD [70] functional with the def2-SVP basis set, using `Gaussian 16.C01` [71]. The hydration free energy was computed with SCRF keyword in using water as the solvent.

The hydration free energies for the molecule in the dataset was calculated with [72]

$$\Delta G_{smd} = (E_{scrf} - E_{opt}) \times 627.5 \frac{\text{kcal/mol}}{\text{a.u.}} \tag{3.5}$$

where ΔG_{smd} denotes the hydration free energy, E_{scrf} denotes the energy from SCRF, and E_{opt} denotes the energy from the geometry optimization using `Gaussian`. Only the solutes where $-6 \frac{\text{kcal}}{\text{mol}} \leq \Delta\Delta G \leq 6 \frac{\text{kcal}}{\text{mol}}$ were retained, resulting in the removal of 75 molecules whose solubility seems difficult to describe from the available data. 1137 records remain in the dataset.

3.2.4 Molecular Feature Generation

Three types of molecular features are used in this study: molecular fingerprints, chemical descriptors, and quantum-chemical descriptors taken from the `Gaussian` geometry optimization and SMD results. Top conformers with lowest energies were used for fingerprint generation and chemical descriptor calculations.

QM descriptors were extracted from the `Gaussian` log files. A complete list of selected QM descriptors were listed in Table C.1. Rotational constants, which are related to molecular spectroscopy, were engineered to include its projections on X, Y and Z axis either by taking sum, product, or square root of sum of products.

We computed 1826 chemical descriptors with `DeepChem` 2.4.0 [73] which uses `mordred` [74] as its backend. A list of 208 RDKit descriptors were also generated with `DeepChem` with H_2 removed because its problem with the *FpDensityMorgan1* descriptor. These two sets of descriptors were merged, forming an input matrix with 2032 features. Besides, RDKit fragment features were also calculated, which fragment descriptors for functional groups in molecules.

We gathered 5 different fingerprints using `RDKit`, namely SECFP6 [75], ECFP6, Morgan, MACCSkeys and RDK fingerprints. All the fingerprints are have 2048 bits except for MACCSkeys, which has 167 bits.

3.2.5 Feature Selection

To further improve the quality of input features, feature selection was performed to reduce redundant information for the augmented chemical descriptors and QM descriptors respectively, Figure 3.2.

In our exploratory data analysis, we noticed that there are many NaN values in QM descriptors for a few molecules: we therefore removed H_2 (CSE_810), SO_2 (CSE_1155), I_2 (CSE_1471), N_2 CSE_927, O_2 (CSE_928), C_2H_2 (CSE_647), N_2O (CSE_912) and COS (CSE_1048) from our database (the molecular indices are listed in parentheses). This list is termed the *special molecular list* for future reference. Any columns that have constant values were removed. A variance threshold of 0.01 was used to filter out constant features with `Scikit-Learn` 0.23.0 [76]. Features with high Pearson correlation, greater than 0.90, were disregarded after standardization; 15 features remain for further model construction. The original quantum chemical descriptor matrix was standardized without any feature filtering for reference.

To align with quantum chemical features, molecules in *special molecular list* were eliminated for the augmented descriptors as well. Together with exploratory data analysis, the first 100 features with biggest values were removed, which can spanning from zero to 1×10^8. The first 25 features sorted by minimum values were deleted as we noticed obvious outliers followed by eliminating the features with spanning range greater than 50. We followed the same procedure as of QM descriptors, which includes the removal of constant features, filtering with variance 0.01, and removing (near) duplicate features. There are 602 features left for the augmented descriptor matrix for now.

The chemical descriptors (except RDKit fragment feature) were further processed with minimum redundancy maximum relevance (MRMR) [77] using `scikit-feature` package, which automates feature selection by picking up relevant features for target values while minimizing the redudancy of selected feature. As the number of features to take has to be defined by the user, we generated subset of the pre-processed features with size of 20, 30, 50, 100, 150, 200, 250, 300, 350, 400, 500, 602. MRMR was not used for fingerprints and QM descriptors.

3.2.6 Model Construction and Hyper-Parameter Optimization

We consider 3 types of different models and 3 categories of molecular features, which in a diverse set of models. For all the models, a scheme of data splitting 0.85:0.15 was used.

Some classical linear regression models were constructed with `Scikit-Learn` using all the 3 types of molecular features, which include ordinary least squares linear regression (LR), linear regression with L_1 regularizer (Lasso), linear least squares with l2 regularization (Ridge), kernel ridge regression (KRR), linear regression with Elastic net regularization (Elastic; this combines L_1 and L_2 penalties), linear model using stochastic gradient descent (SGD) to minimize the regularized empirical loss (SGDReg), k-nearest neighbors regression (KNR) and regression using decision tree (DTrees).

`GPFlow` [78] 2.2.1 was used to build the Gaussian process regression (GPR) models with a diverse set of kernel functions. The target values $\Delta\Delta G$ were standardized to improve numerical stability and designed for a better initial guess of kernel parameters. $\Delta\Delta G$ with Yeo-Johnson transformation [79] and unchanged target values were both used as well. The Yeo-Johnson transformation is defined as

$$\psi(\lambda, x) = \begin{cases} \left\{(x+1)^{\lambda} - 1\right\}/\lambda & (x \geqslant 0, \lambda \neq 0) \\ \log(x+1) & (x \geqslant 0, \lambda = 0) \\ -\left\{(-x+1)^{2-\lambda} - 1\right\}/(2-\lambda) & (x < 0, \lambda \neq 2) \\ -\log(-x+1) & (x < 0, \lambda = 2) \end{cases} \tag{3.6}$$

where x denotes the raw data and λ is a parameter that can be estimated with maximum likelihood inference. Zero mean was used when training the model with `GPFlow`.

All the hyper-parameters of the selected kernel functions were optimized with the quasi-Newton algorithm, L-BFGS-B, as implemented in `Scipy` optimizer; up to 20000 optimization steps were done. The mean function was always set as the mean of the target values.

3.2.7 Model Performance Evaluation

To characterize the model performance, mean absolute error (MAE), root mean squared error (RMSE), and coefficient of determination (R^2) were used:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.7}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.8}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{3.9}$$

where y_i denotes the i-th value of our target vector y, $\hat{y}_i$ denotes the estimation or prediction of y_i, $\bar{y}$ denotes the mean value of y. The target values were scaled back when using Yeo-Johnson transformation in GPR model.

3.2.8 Software and Packages

To make all the results reproducible, we used a consistent virtual environment using `Python` 3.7.4, `tensorflow` 2.3.0, `GPFlow` 2.2.1, `numpy` 1.18.4, `pandas` 1.0.3 and `Scikit-Learn` 0.23.0. All the figures were generated with `matplotlib` 3.4.2 and `seaborn` 0.11.2. All the calculations were done on Compute Canada.

3.3 Results and Discussions

3.3.1 Dataset Preparation

CombiSolv-Exp [28], a curated dataset from MNSol (2275 records) [60], FreeSolv (560 records) [25], Compsol (3548 records) [61] and Abraham (6091 records) [62] was used for this project. The downloaded CombiSolv-Exp dataset is a subset of MNSol due to license limitations. Therefore, a complete CombiSolv-Exp dataset was built after merging MNSol as described in subsection 3.2.1. Only binary mixtures and neutral molecules were selected. As we are interested in aqueous free energies of solvation, the only solvent we considered was water. We also eliminated the self-solvation free energy for water. There 1263 records selected had SMILES, InChI for the solutes (input) and mean and standard deviation values for the hydration free energy (output).

A well-designed pre-processing procedure was performed to clean up the dataset, Figure 3.2 (B), which can further improve model performance. Molecules that are unrecognizable by `RDKit` were removed and this can caused by invalid bond types, or not being able to Kekulize or sanitize molecules. This can help avoid problems when computing molecular descriptors. The salts were stripped out and all the carboxylic acids were left out as well. This is due to the limitations of the universal solvation model, SMD, for not capable of dealing with the increased atomic charge of the hydroxyl hydrogen in carboxylic acids [45, 80].

Molecule 3D coordinates generation, conformer searching and geometry optimization were performed afterwards. We employed an adaptive method to determine the number of conformers to be sampled [68] with Equation 3.4, which can efficiently balance the sampling diversity of conformers and computation efficiency. Geometry and SMD jobs were run to get the $\Delta\Delta G$ values with Eq.

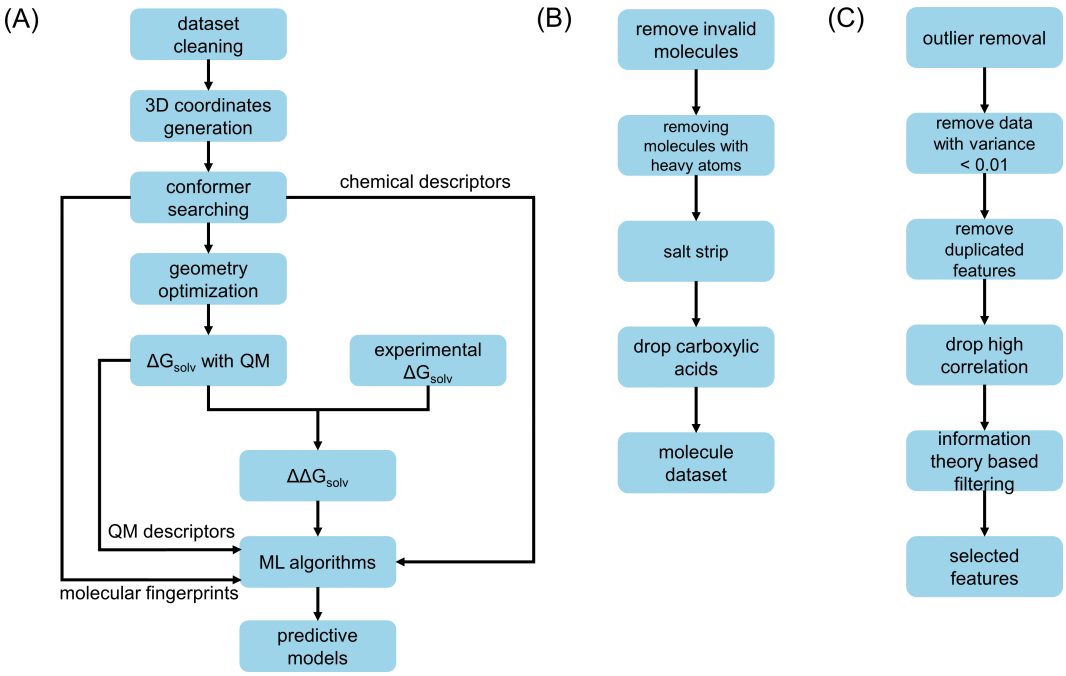


Fig. 3.2 The framework of constructing Δ-learning models with various machine learning algorithms (A) and design of feature selection (B). RDKit fragment feature belong to the chemical descriptor sub-category and is not depicted explicitly here.

Equation 3.5. By examining the error of SMD calculations with experimental measurements, we noticed that there are 13 molecules that can be seen as outliers with $\Delta\Delta G < -6 kcal/mol$ or $\Delta\Delta G > 6 kcal/mol$ while the others lie in $\Delta\Delta G \in [-6, 6]\ kcal/mol$, Figure 3.3 (A). Poor performance of SMD can be caused by (1) special geometries, such as crown ethers (CSE_1428 and CSE_1430); (2) flexible molecules with many rotatable bonds or rings, making conformer searching a challenging task, namely CSE_1775, CSE_1382, CSE_1457, CSE_930; (3) increased atomic charges, CSE_1410, CSE_1869, CSE_1421, CSE_1196, CSE_1448, CSE_1447 and CSE_1844. Removing these outliers make the distribution of target values resemble to a normal distribution, Figure Figure 3.3 (B).

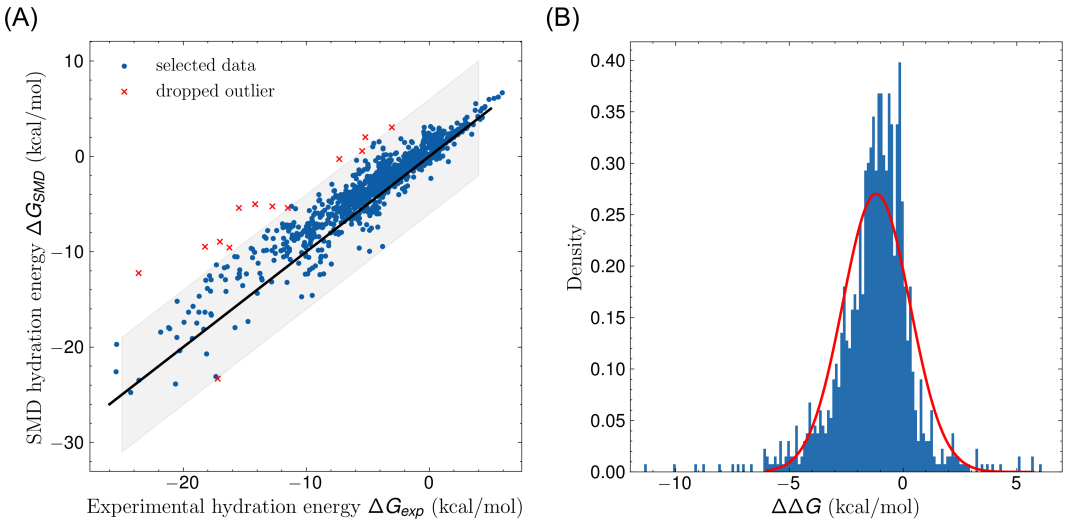


Fig. 3.3 Hydration free energies computed with SMD method. (A). The computed hydration energies of organic molecules *vs.* the experimental hydration energies. The grey box highlights molecules (blue circular dots) with smaller errors, less than 6 $kcal/mol$, between hydration energy from SMD and experiments. The red crossed dots represent molecules with error greater than 6 $kcal/mol$. The black line ($y = x$) denotes cases where $\Delta\Delta G = 0$. (B) Histogram of $\Delta\Delta G$ values computed with Equation 3.3 with fitted normal distribution highlighted in red line (mean $\mu = -1.18$ and standard deviation $\sigma = 1.48$). The histogram is for all the records, but the fitted normal distribution is based on selected molecules. All units of ΔG_{SMD}, ΔG_{exp} and $\Delta\Delta G$ are in $kcal/mol$.

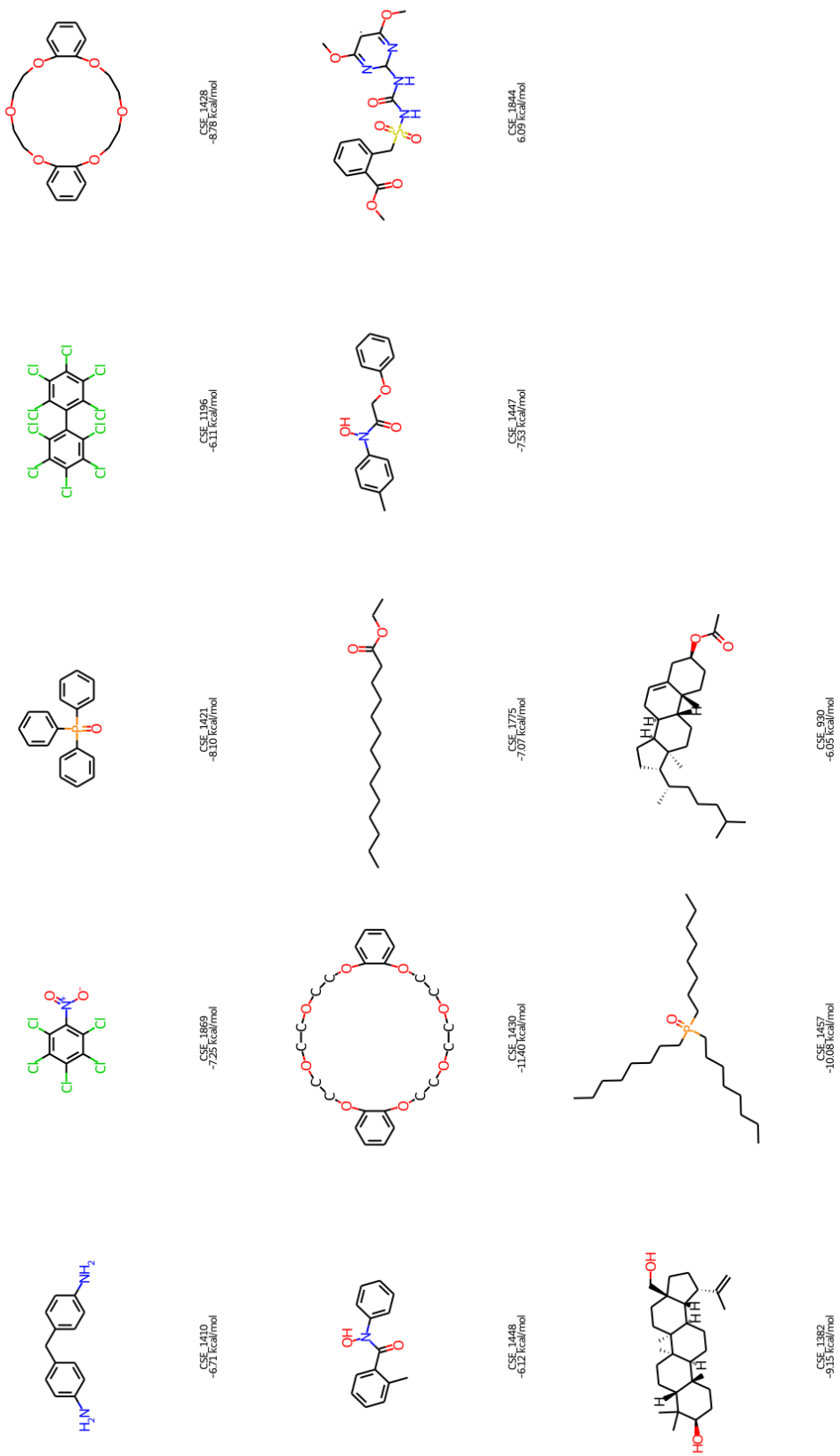


Fig. 3.4 Dropped molecules with molecule ID and the error between SMD calculations and experimental hydration free energy, $\Delta\Delta G$ in $kcal/mol$.

3.3.2 Feature Generation and Feature Selection

There are 3 different features that are used in this study for 3 types of ML algorithms, Figure 3.2 (A), namely molecular fingerprints, chemical descriptors and QM descriptors. Chemical and QM descriptors were standardized before dropping highly correlated features. There was no standardization step for RDKit fragment features.

Molecular fingerprint provides an efficient way of encoding molecules in a binary way, which has been widely used in computational chemistry and drug design [81, 82]. Chemical descriptors often come with physical interpretations, such as number of hydrogen bond donors, polar surface area. We also include RDKit fragment features, which count functional groups in a molecule.

QM descriptors, as listed in Table C.1, was employed to improve SMD accuracy without introducing extra computing, but only text processing of Gaussian log files. It comes to our attention that some molecules have NaN values for some QM descriptors, which will lead to problems of building computational models. These molecules were removed from all kind of molecule features when performing feature filtering and model constructions for consistency.

Feature selection was done following procedures described in Figure 3.2 (C) and only MRMR was applied to augmented chemical descriptors. MRMR, a supervised filter-based feature selection method, seeks maximal relevance with the target values while minimizes the correlations between selected features using mutual information as the measurement of pair-wise similarities [77, 83] and it has been widely used in image processing [84], bioinformatics [85], marketing platform with ML [86], *et al*. Given the 602 features by dropping high correlated features (Pearson correlation threshold 0.90), a list of different number of chemical descriptors were selected, 20, 30, 50, 100, 150, 200, 250, 300, 350, 400, 500, 602 (full feature without MRMR).

3.3.3 Performance Evaluation of Commonly Used Algorithms

Three different types of models were built for commonly used regression algorithms in `Scikit-Learn` [76] and Gaussian process regression (GPR). Grid search was employed to optimize the hyperparameters of commonly used regression algorithms and L-BFGS-B method [87] was used to tune the hyperparameters of GPR in `GPFlow` [78].

We experimented with 8 different regression algorithms as listed in subsection 3.2.6 in combination with engineered features as in subsection 3.3.2; and 10-fold cross-validation was used to split data into training and testing subsets for model selection purposes along with grid searching for optimal hyperparameters. For

simplicity, we only select top 10 models for further analysis based on mean absolute error (MAE), root mean squared error (RMSE) and coefficient of determination (R^2) for testing data, Figure 3.5.

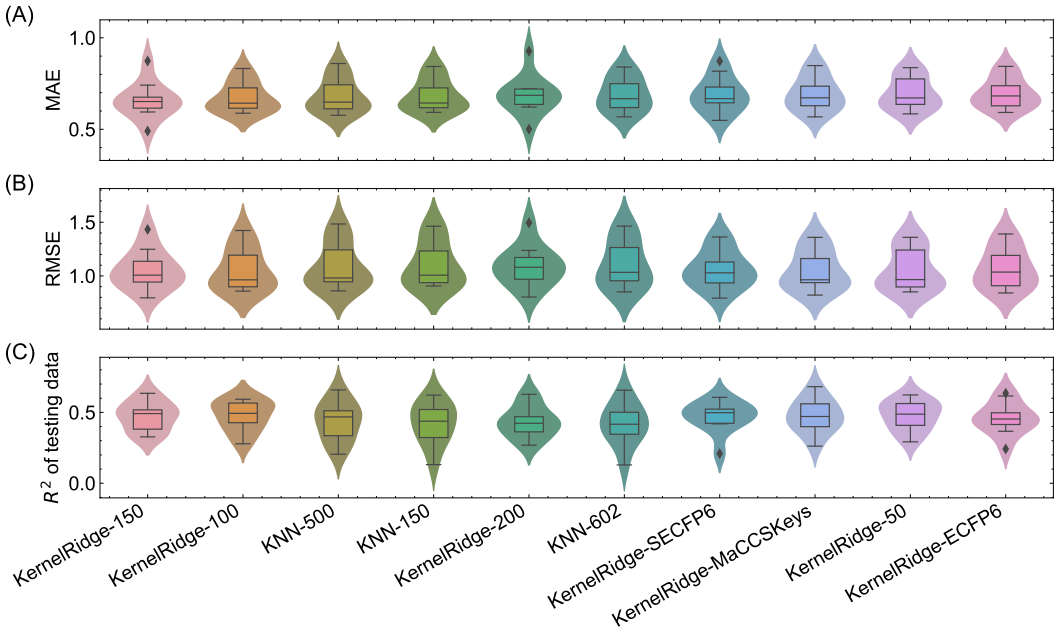


Fig. 3.5 Box plots and violin plots for top 10 regression models with commonly used algorithms.

All the selected top models display have similar mean absolute errors (MAE) of 0.66 ~ 0.70, root-mean-square errors (RMSE) of 1.06 ~ 1.07 and R^2 of 0.42 0.48, Table C.2. 7 of the 10 best models use kernel-ridge regression (KRR), suggesting that KRR is preferred for this study. The kernel ridge together with 150 selected chemical descriptors (KernelRidge-150) gives smallest error with MAE of 0.6581, RMSE of 1.0568 and R^2 of 0.4717. It should be noted that some of the predictions from 10-fold cross-validation demonstrate bigger errors than others, such as the dots in KernelRidge-200 of Figure 3.5, implying that the hyperparameters can be further optimized. 3 kNN models appear in the top 10, and all of them use chemical descriptors, suggesting that kNN performs better with chemical descriptors than molecular fingerprints.

We also explored models built on the top of QM related descriptors which are parsed from `Gaussian` log files. Similar to the top 10 models, kNN and kernel ridge based models dominate the top 10 QM-based models with MAE of 0.80 0.84, RMSE of 1.24 1.28, R^2 of 0.23 0.28 (Table 3.1). In addition, decision tree based models demonstrate close performances. Using more sophisticated analysis based

Table 3.1 Top 10 QM based models from commonly used algorithms.

Regressor	**feature_type**	**MAE_test**	**RMSE_test**	R^2**_test**
KNN	qc_features_rot_sum_full	0.7984±0.1023	1.2424±0.2257	0.2767±0.1285
KNN	qc_features_rot_product_full	0.7993±0.0990	1.2453±0.2200	0.2734±0.1225
KNN	qc_features_rot_spread_full	0.8027±0.1010	1.2521±0.2218	0.2650±0.1277
KNN	qc_features_full	0.8048±0.0993	1.2508±0.2207	0.2669±0.1235
KernelRidge	qc_features_rot_product_full	0.8397±0.1015	1.2796±0.2338	0.2352±0.1135
KernelRidge	qc_features_rot_spread_full	0.8405±0.1009	1.2802±0.2333	0.2345±0.1131
KernelRidge	qc_features_rot_sum_full	0.8407±0.1007	1.2804±0.2332	0.2343±0.1129
KernelRidge	qc_features_full	0.8413±0.1012	1.2808±0.2331	0.2337±0.1128
DTrees	qc_features_full	0.8568±0.0999	1.2933±0.2002	0.2151±0.0958
DTrees	qc_features_rot_sum_full	0.8585±0.0986	1.3041±0.1949	0.2009±0.0978

on invariants of the rotational eigenvalues does not seem to improve performance. Figure C.1.

3.3.4 Performance Evaluation of Gaussian Process Regression

Unlike other conventional machine learning algorithms, Gaussian process regression (GPR) provides uncertainty estimation of the predictions. We built GPR based models with different kernels and different features. Inspired by the practice of `FlowMO` [88], a package for training GPR, our target values $\Delta\Delta G$ was standardized to improve numerical stability and model performance. We also scaled $\Delta\Delta G$ to attempt to ensure the data were normally distributed by using the Yeo-Johnson transformation [79], along with standardization of input features to ensure zero-mean and unit-variance.

As show in Table C.3, we find that Matern-class kernels dominate the top 10 GPR models, which show very close performances (MAE of 0.63, RMSE of 1.02 and R^2 of 0.51 approximately). They also share similar behavior in 10-fold cross-validation, Figure 3.6. In addition, all the top GPR models share the same feature, 150 descriptors (Table C.3).

As for the top 10 QM based models, they share almost identical performances with MAE of 0.79, RMSE of 1.24 and R^2 of 0.29 (Table 3.2).

3.4 Conclusions

Molecular solubility represents a important property of molecules and has great implications for chemical reactions, and chemical engineering. Moreover, solubility is a central topic in drug design and development, especially for drug delivery and formulation. Therefore, predictions of molecular solubility are of great significance.

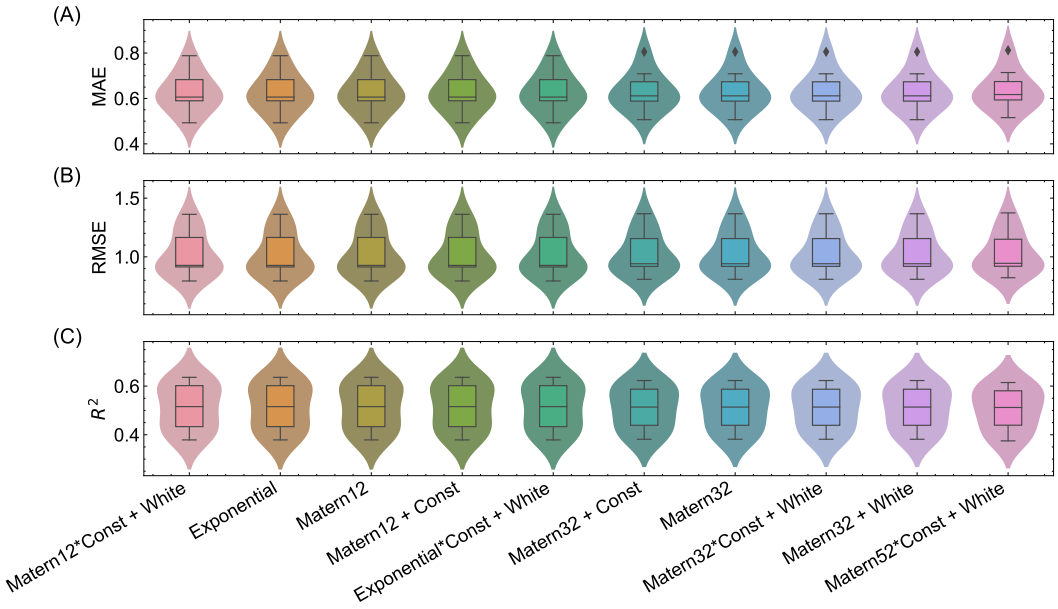


Fig. 3.6 Box plots and violin plots for top 10 GPR models.

Table 3.2 Top 10 QM based models from GPR.

kernel	**feature_type**	**MAE_test**	**RMSE_test**	R^2**2_test**
Exponential	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Exponential + Constant	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Exponential + White	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Matern12 + Constant	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Matern12	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Matern12*Constant + White	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Exponential*Constant + White	qc_full_rot_product	0.7940±0.1015	1.2363±0.2492	0.2864±0.1362
Matern12 + Constant	qc_full_rot_spread	0.7950±0.1009	1.2366±0.2490	0.2860±0.1359
Matern12	qc_full_rot_spread	0.7950±0.1009	1.2366±0.2490	0.2860±0.1359
Matern12 + White	qc_full_rot_spread	0.7950±0.1009	1.2366±0.2490	0.2860±0.1359

To address this challenging task, we constructed a Δ-learning strategy to correct the error of widely used solvation model based on density (SMD). The CombiSolv-Exp database was used for training/testing and, after clean the data, we had 1253 records. Extensive feature engineering was employed, and then a wide variety of regression models including Gaussian processes. We find that we can improve the (raw) performance of SMD by building a machine-learning model for the correction term.

It does appear that further hyperparameter optimization could be beneficial. It is possible that better features would lead to better results, but it may also be that we are limited by the lack of experimental data and the measurement errors from the experiment. We believe the general strategy of adding extremely inexpensive machine-learning-based corrections for quantum chemistry calculations (not only for

the free energy of hydration, but also for other quantities) is powerful, and we plan to release a free and open-source command-line tool that automates the workflow and predictions described here.

References

[1] Ketan T Savjani, Anuradha K Gajjar, and Jignasa K Savjani. Drug solubility: importance and enhancement techniques. *International Scholarly Research Notices*, 2012, 2012.

[2] Shobha N Bhattachar, Laura A Deschenes, and James A Wesley. Solubility: it's not just for physical chemists. *Drug Discovery Today*, 11(21-22):1012–1018, 2006.

[3] William L Jorgensen and Erin M Duffy. Prediction of drug solubility from structure. *Advanced Drug Delivery Reviews*, 54(3):355–366, 2002.

[4] Hywel D Williams, Natalie L Trevaskis, Susan A Charman, Ravi M Shanker, William N Charman, Colin W Pouton, and Christopher JH Porter. Strategies to address low drug solubility in discovery and development. *Pharmacological Reviews*, 65(1):315–499, 2013.

[5] William Curatolo. Physical chemical properties of oral drug candidates in the discovery and exploratory development settings. *Pharmaceutical Science & Technology Today*, 1(9):387–393, 1998.

[6] Oleg Ursu, Anwar Rayan, Amiram Goldblum, and Tudor I Oprea. Understanding drug-likeness. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1(5):760–781, 2011.

[7] Giulio Vistoli, Alessandro Pedretti, and Bernard Testa. Assessing drug-likeness–what are we missing? *Drug Discovery Today*, 13(7-8):285–294, 2008.

[8] Wenxiu Wei, Srinivasulu Cherukupalli, Lanlan Jing, Xinyong Liu, and Peng Zhan. Fsp^3: A new parameter for drug-likeness. *Drug Discovery Today*, 2020.

[9] Daniela Hartmann Jornada, Guilherme Felipe dos Santos Fernandes, Diego Eidy Chiba, Thais Regina Ferreira De Melo, Jean Leandro Dos Santos, and Man Chin Chung. The prodrug approach: A successful tool for improving drug solubility. *Molecules*, 21(1):42, 2016.

[10] Michael A Walker. Improvement in aqueous solubility achieved via small molecular changes. *Bioorganic & Medicinal Chemistry Letters*, 27(23):5100–5108, 2017.

[11] Li Di, Paul V Fish, and Takashi Mano. Bridging solubility between drug discovery and development. *Drug Discovery Today*, 17(9-10):486–495, 2012.

[12] David S Palmer, James L McDonagh, John BO Mitchell, Tanja van Mourik, and Maxim V Fedorov. First-principles calculation of the intrinsic aqueous solubility of crystalline druglike molecules. *Journal of Chemical Theory and Computation*, 8(9):3322–3337, 2012.

[13] Yves Tardy and Joelle Duplay. A method of estimateng the gibbs free energies of formation of hydrated and dehydrated clay minerals. *Geochimica et Cosmochimica Acta*, 56(8):3007–3029, 1992.

[14] Andreas Klamt, Frank Eckert, and Michael Diedenhofen. Prediction of the free energy of hydration of a challenging set of pesticide-like compounds. *The Journal of Physical Chemistry B*, 113(14):4508–4510, 2009.

[15] Drazen Petrov, Daniel Tunega, Martin H Gerzabek, and Chris Oostenbrink. Molecular modelling of sorption processes of a range of diverse small organic molecules in leonardite humic acid. *European Journal of Soil Science*, 71(5): 831–844, 2020.

[16] Alauddin Ahmed and Stanley I Sandler. Hydration free energies of multifunctional nitroaromatic compounds. *Journal of Chemical Theory and Computation*, 9(6):2774–2785, 2013.

[17] Jason D Thompson, Christopher J Cramer, and Donald G Truhlar. Predicting aqueous solubilities from aqueous free energies of solvation and experimental or calculated vapor pressures of pure substances. *The Journal of Chemical Physics*, 119(3):1661–1670, 2003.

[18] Philip Ball. Water—an enduring mystery. *Nature*, 452(7185):291–292, 2008.

[19] Julien Michel and Jonathan W Essex. Prediction of protein–ligand binding affinity by free energy simulations: assumptions, pitfalls and expectations. *Journal of Computer-Aided Molecular Design*, 24(8):639–658, 2010.

[20] Ayesha Zafar and Jóhannes Reynisson. Hydration free energy as a molecular descriptor in drug design: a feasibility study. *Molecular Informatics*, 35(5): 207–214, 2016.

[21] Philippa M Wiggins. Hydrophobic hydration, hydrophobic forces and protein folding. *Physica A: Statistical Mechanics and its Applications*, 238(1-4): 113–128, 1997.

[22] Valery Weber and D Asthagiri. Regularizing binding energy distributions and the hydration free energy of protein cytochrome c from all-atom simulations. *Journal of Chemical Theory and Computation*, 8(9):3409–3415, 2012.

[23] Gerhard König, Stefan Bruckner, and Stefan Boresch. Absolute hydration free energies of blocked amino acids: implications for protein solvation and stability. *Biophysical Journal*, 104(2):453–462, 2013.

[24] Themis Lazaridis and Michael E Paulaitis. Activity coefficients in dilute aqueous solutions from free energy simulations. *AIChE journal*, 39(6):1051–1060, 1993.

[25] Guilherme Duarte Ramos Matos, Daisy Y Kyu, Hannes H Loeffler, John D Chodera, Michael R Shirts, and David L Mobley. Approaches for calculating solvation free energies and enthalpies demonstrated with an update of the freesolv database. *Journal of Chemical & Engineering Data*, 62(5):1559–1569, 2017.

[26] Yashaswi Pathak, Siddhartha Laghuvarapu, Sarvesh Mehta, and U Deva Priyakumar. Chemically interpretable graph interaction network for prediction of pharmacokinetic properties of drug-like molecules. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 873–880, 2020.

[27] Yashaswi Pathak, Sarvesh Mehta, and U Deva Priyakumar. Learning atomic interactions through solvation free energy prediction using graph neural networks. *Journal of Chemical Information and Modeling*, 61(2):689–698, 2021.

[28] Florence H Vermeire and William H Green. Transfer learning for solvation free energies: From quantum chemistry to experiments. *Chemical Engineering Journal*, 418:129307, 2021.

[29] Hyuntae Lim and YounJoon Jung. Delfos: deep learning model for prediction of solvation free energies in generic organic solvents. *Chemical Science*, 10 (36):8306–8315, 2019.

[30] Hyuntae Lim and YounJoon Jung. Mlsolva: solvation free energy prediction from pairwise atomistic interactions by machine learning. *Journal of Cheminformatics*, 13(1):1–10, 2021.

[31] William L Jorgensen and C Ravimohan. Monte carlo simulation of differences in free energies of hydration. *The Journal of Chemical Physics*, 83(6):3050–3054, 1985.

[32] SJ Wierzchowski and PA Monson. Calculation of free energies and chemical potentials for gas hydrates using monte carlo simulations. *The Journal of Physical Chemistry B*, 111(25):7274–7282, 2007.

[33] David L Mobley, Christopher I Bayly, Matthew D Cooper, and Ken A Dill. Predictions of hydration free energies from all-atom molecular dynamics simulations. *The Journal of Physical Chemistry B*, 113(14):4533–4537, 2009.

[34] Pavel V Klimovich and David L Mobley. Predicting hydration free energies using all-atom molecular dynamics simulations and multiple starting conformations. *Journal of Computer-Aided Molecular Design*, 24(4):307–316, 2010.

[35] Michael R Shirts, Jed W Pitera, William C Swope, and Vijay S Pande. Extremely precise free energy calculations of amino acid side chain analogs: Comparison of common molecular mechanics force fields for proteins. *The Journal of Chemical Physics*, 119(11):5740–5761, 2003.

[36] Andrew W Milne and Miguel Jorge. Polarization corrections and the hydration free energy of water. *Journal of Chemical Theory and Computation*, 15(2): 1065–1078, 2018.

[37] Joakim PM Jämbeck and Alexander P Lyubartsev. Update to the general amber force field for small solutes with an emphasis on free energies of hydration. *The Journal of Physical Chemistry B*, 118(14):3793–3804, 2014.

[38] Christopher M Baker, Pedro EM Lopes, Xiao Zhu, Benoît Roux, and Alexander D MacKerell Jr. Accurate calculation of hydration free energies using pair-specific lennard-jones parameters in the charmm drude polarizable force field. *Journal of Chemical Theory and Computation*, 6(4):1181–1198, 2010.

[39] T Wesolowski and Arieh Warshel. Ab initio free energy perturbation calculations of solvation free energy using the frozen density functional approach. *The Journal of Physical Chemistry*, 98(20):5183–5187, 1994.

[40] Bryant Kim, Ryan Snyder, Mulpuri Nagaraju, Yan Zhou, Pedro Ojeda-May, Seth Keeton, Mellisa Hege, Yihan Shao, and Jingzhi Pu. Reaction path-force matching in collective variables: Determining ab initio qm/mm free energy profiles by fitting mean force. *Journal of Chemical Theory and Computation*, 17(8):4961–4980, 2021.

[41] Zhaoxi Sun. Bar-based multi-dimensional nonequilibrium pulling for indirect construction of qm/mm free energy landscapes: from semi-empirical to ab initio. *Physical Chemistry Chemical Physics*, 21(39):21942–21959, 2019.

[42] Pengfei Li, Xiangyu Jia, Xiaoliang Pan, Yihan Shao, and Ye Mei. Accelerated computation of free energy profile at *ab initio* quantum mechanical/molecular mechanics accuracy via a semi-empirical reference potential. i. weighted thermodynamics perturbation. *Journal of Chemical Theory and Computation*, 14(11):5583–5596, 2018.

[43] Jin Zhang, Haiyang Zhang, Tao Wu, Qi Wang, and David van der Spoel. Comparison of implicit and explicit solvent models for the calculation of solvation free energy in organic solvents. *Journal of Chemical Theory and Computation*, 13(3):1034–1043, 2017.

[44] Jenke Scheen, Wilson Wu, Antonia SJS Mey, Paolo Tosco, Mark Mackey, and Julien Michel. Hybrid alchemical free energy/machine-learning methodology for the computation of hydration free energies. *Journal of Chemical Information and Modeling*, 60(11):5331–5339, 2020.

[45] Aleksandr V Marenich, Christopher J Cramer, and Donald G Truhlar. Universal solvation model based on solute electron density and on a continuum model of the solvent defined by the bulk dielectric constant and atomic surface tensions. *The Journal of Physical Chemistry B*, 113(18):6378–6396, 2009.

[46] Sereina Riniker. Molecular dynamics fingerprints (mdfp): machine learning from md data to predict free-energy differences. *Journal of Chemical Information and Modeling*, 57(4):726–741, 2017.

[47] Vigneshwari Subramanian, Ekaterina Ratkova, David Palmer, Ola Engkvist, Maxim Fedorov, and Antonio Llinas. Multisolvent models for solvation free energy predictions using 3d-rism hydration thermodynamic descriptors. *Journal of Chemical Information and Modeling*, 60(6):2977–2988, 2020.

[48] Raghunathan Ramakrishnan, Pavlo O Dral, Matthias Rupp, and O Anatole von Lilienfeld. Big data meets quantum chemistry approximations: the δ-machine learning approach. *Journal of Chemical Theory and Computation*, 11(5): 2087–2096, 2015.

[49] LiHong Hu, XiuJun Wang, LaiHo Wong, and GuanHua Chen. Combined first-principles calculation and neural-network correction approach for heat of formation. *The Journal of Chemical Physics*, 119(22):11501–11507, 2003.

[50] XiuJung Wang, LiHong Hu, LaiHo Wong, and GuanHua Chen. A combined first-principles calculation and neural networks correction approach for evaluating gibbs energy of formation. *Molecular Simulation*, 30(1):9–15, 2004.

[51] Pablo A Unzueta, Chandler S Greenwell, and Gregory JO Beran. Predicting density functional theory-quality nuclear magnetic resonance chemical shifts via δ-machine learning. *Journal of Chemical Theory and Computation*, 17(2): 826–840, 2021.

[52] Mihail Bogojeski, Leslie Vogt-Maranto, Mark E Tuckerman, Klaus-Robert Müller, and Kieron Burke. Quantum chemical accuracy from density functional approximations via machine learning. *Nature Communications*, 11(1):1–11, 2020.

[53] Jie Cui and Roman V Krems. Gaussian process model for collision dynamics of complex molecules. *Physical review letters*, 115(7):073202, 2015.

[54] Daniel Born and Johannes Kätner. Geometry optimization in internal coordinates based on gaussian process regression: Comparison of two approaches. *Journal of Chemical Theory and Computation*, 17(9):5955–5967, 2021.

[55] Ralf Meyer and Andreas W Hauser. Geometry optimization using gaussian process regression in internal coordinate systems. *The Journal of chemical physics*, 152(8):084112, 2020.

[56] Volker L Deringer, Albert P Bartók, Noam Bernstein, David M Wilkins, Michele Ceriotti, and Gábor Csányi. Gaussian process regression for materials and molecules. *Chemical Reviews*, 121(16):10073–10141, 2021.

[57] C.E. Rasmussen, C.K.I. Williams, M.I.T. Press, F. Bach, and ProQuest (Firm). *Gaussian Processes for Machine Learning*. Adaptive computation and machine learning. MIT Press, 2006. ISBN 9780262182539. URL https://books.google.ca/books?id=Tr34DwAAQBAJ.

[58] Oren Anava and Kfir Levy. k*-nearest neighbors: From global to local. In *Advances in neural information processing systems*, pages 4916–4924, 2016.

[59] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3): 660–674, 1991.

[60] Aleksandr V Marenich, Casey P Kelly, Jason D Thompson, Gregory D Hawkins, Candee C Chambers, David J Giesen, Paul Winget, Christopher J Cramer, and Donald G Truhlar. Minnesota solvation database. *Minnesota Solvation Database version*, 20, 2012.

[61] Edouard Moine, Romain Privat, Baptiste Sirjean, and Jean-Noël Jaubert. Estimation of solvation quantities from experimental thermodynamic data: Development of the comprehensive compsol databank for pure and mixed solutes. *Journal of Physical and Chemical Reference Data*, 46(3):033102, 2017.

[62] Laura M Grubbs, Mariam Saifullah, E Nohelli, Shulin Ye, Sai S Achi, William E Acree Jr, and Michael H Abraham. Mathematical correlations for describing solute transfer into functionalized alkane solvents containing hydroxyl, ether, ester or ketone solvents. *Fluid Phase Equilibria*, 298(1):48–53, 2010.

[63] Toon Verstraelen, William Adams, Leila Pujal, Alireza Tehrani, Braden D Kelly, Luis Macaya, Fanwang Meng, Michael Richer, Raymundo Hernández-Esparza, Xiaotian Derrick Yang, et al. Iodata: A python library for reading, writing, and converting computational chemistry file formats and generating input files. *Journal of Computational Chemistry*, 42(6):458–464, 2021.

[64] Noel M O'Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3(1):1–14, 2011.

[65] Greg Landrum. Rdkit documentation. *Release*, 1(1-79):4, 2021.

[66] Paolo Tosco, Nikolaus Stiefl, and Gregory Landrum. Bringing the mmff force field to the rdkit: implementation and validation. *Journal of Cheminformatics*, 6(1):1–4, 2014.

[67] Noel M O'Boyle, Tim Vandermeersch, Christopher J Flynn, Anita R Maguire, and Geoffrey R Hutchison. Confab-systematic generation of diverse low-energy conformers. *Journal of Cheminformatics*, 3(1):1–9, 2011.

[68] Jean-Paul Ebejer, Garrett M Morris, and Charlotte M Deane. Freely available conformer generation methods: how good are they? *Journal of Chemical Information and Modeling*, 52(5):1146–1158, 2012.

[69] Junmei Wang, Romain M Wolf, James W Caldwell, Peter A Kollman, and David A Case. Development and testing of a general amber force field. *Journal of Computational Chemistry*, 25(9):1157–1174, 2004.

[70] Jeng-Da Chai and Martin Head-Gordon. Long-range corrected hybrid density functionals with damped atom–atom dispersion corrections. *Physical Chemistry Chemical Physics*, 10(44):6615–6620, 2008.

[71] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox. Gaussian~16 Revision C.01, 2016. Gaussian Inc. Wallingford CT.

[72] RE Skyner, JL McDonagh, CR Groom, T Van Mourik, and JBO Mitchell. A review of methods for the calculation of solution free energies and the modelling of systems in solution. *Physical Chemistry Chemical Physics*, 17(9):6174–6191, 2015.

[73] Bharath Ramsundar, Peter Eastman, Patrick Walters, Vijay Pande, Karl Leswing, and Zhenqin Wu. *Deep Learning for the Life Sciences*. O'Reilly Media, 2019.

[74] Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, and Tatsuya Takagi. Mordred: a molecular descriptor calculator. *Journal of Cheminformatics*, 10 (1):1–14, 2018.

[75] Daniel Probst and Jean-Louis Reymond. A probabilistic molecular fingerprint for big data settings. *Journal of Cheminformatics*, 10(1):1–12, 2018.

[76] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn:

Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

[77] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, 3(02):185–205, 2005.

[78] Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. Gpflow: A gaussian process library using tensorflow. *Journal of Machine Learning Research*, 18(40):1–6, 2017.

[79] In-Kwon Yeo and Richard A Johnson. A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959, 2000.

[80] Maximiliano Riquelme, Alejandro Lara, David L Mobley, Toon Verstraelen, Adelio R Matamala, and Esteban Vohringer-Martinez. Hydration free energies in the freesolv database calculated with polarized iterative hirshfeld charges. *Journal of Chemical Information and Modeling*, 58(9):1779–1797, 2018.

[81] Alice Capecchi, Daniel Probst, and Jean-Louis Reymond. One molecular fingerprint to rule them all: drugs, biomolecules, and the metabolome. *Journal of Cheminformatics*, 12(1):1–15, 2020.

[82] Adrià Cereto-Massagué, María José Ojeda, Cristina Valls, Miquel Mulero, Santiago Garcia-Vallvé, and Gerard Pujadas. Molecular fingerprint similarity search in virtual screening. *Methods*, 71:58–63, 2015.

[83] Peter Bugata and Peter Drotar. On some aspects of minimum redundancy maximum relevance feature selection. *Science China Information Sciences*, 63 (1):1–15, 2020.

[84] Benjamin Auffarth, Maite López, and Jesús Cerquides. Comparison of redundancy and relevance measures for feature selection in tissue classification of ct images. In *Industrial conference on data mining*, pages 248–262. Springer, 2010.

[85] Milos Radovic, Mohamed Ghalwash, Nenad Filipovic, and Zoran Obradovic. Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. *BMC Bioinformatics*, 18(1):1–14, 2017.

[86] Zhenyu Zhao, Radhika Anand, and Mallory Wang. Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform. In *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 442–452. IEEE, 2019.

[87] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[88] Henry B Moss and Ryan-Rhys Griffiths. Gaussian process molecule property prediction with flowmo. *arXiv preprint arXiv:2010.01118*, 2020.

Chapter 4

`Procrustes`: A Python Library to Find Transformations that Maximize the Similarity Between Matrices

4.1 Introduction

In 1962, Hurley and Catell posed the first Procrustes problem to find the orthogonal transformation that makes a matrix, $\mathbf{A}$, maximally resemble a target matrix $\mathbf{B}$, in a least-squares sense [1]. Four years later, Schönemann presented a solution to this problem using singular value decomposition [2]. Since then, many researchers have extended the Procrustes analysis to include a wide range of problems, which can all be represented by

$$\underbrace{\min}_{\mathbf{S},\mathbf{T}} \|\mathbf{SAT} - \mathbf{B}\|_F^2 \tag{4.1}$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ is the input matrix, $\mathbf{B} \in \mathbb{R}^{m\times n}$ is the reference (target) matrix, and $\|\cdot\|_F$ denotes the Frobenius norm defined as,

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2} = \sqrt{\mathrm{Tr}(\mathbf{A}^\dagger\mathbf{A})} \tag{4.2}$$

where a_{ij} and $\mathrm{Tr}(\mathbf{A})$ denote the element ij and trace of $\mathbf{A}$, respectively. Different Procrustes problems use different choices for the transformation matrices $\mathbf{S}$ and $\mathbf{T}$ which are commonly taken to be orthogonal/unitary matrices, rotation matrices, symmetric matrices, or permutation matrices. When $\mathbf{S}$ is an identity matrix,

Equation 4.1 is called a one-sided Procrustes problem [3], and when it is equal to $\mathbf{T}$, Equation 4.1 is called a two-sided Procrustes problem with one transformation.

The Procrustes problems implemented in our packaged are summarized in Table 4.1 and detailed in the Appendix. Except for two-sided permutation Procrustes, all these problems have explicit solutions. So, for the two-sided permutation problems our library implements several formulas and iterative methods for approximating the solution. For the two-sided permutation with two-transformations, these include the heuristic k-opt[4–6] and flip-flop [7] algorithms. For the two-sided permutation with one-transformation, we implement several explicit approximate solutions (normal1, normal2, umeyama, and umeyama-SVD) which optionally can be used as initial guesses for the iterative solutions (k-opt, soft-assign, and non-negative matrix factorization (NMF)). Among these, k-opt is a brute-force approach which iteratively checks all permutations of k-th order until no further improvements are possible. If k equals the dimensions of the permutation matrix, then k-opt amounts to an exhaustive search over all possible permutations. The general-purpose implementations of the heuristic k-opt, flip-flop, and softassign algorithms could be used independently.

Procrustes analysis has a wide-range of applications and, in fact, as we became more familiar with Procrustes algorithms, we realized that many problems can be formulated as a Procrustes problem. For example, when matrices $\mathbf{A}$ and $\mathbf{B}$ represent lists of the coordinates of multidimensional points, Procrustes analysis can be used to transform the coordinates of $\mathbf{A}$ to most closely resemble those in $\mathbf{B}$. By convention, the column of the matrix contains spatial coordinates or characteristic properties/features of each point, and each row represents a point. In addition, the one-sided rotational Procrustes analysis is widely applied in image recognition[25–27], computational biology [28–31], chemistry [31, 32], physics [33], signal processing [34, 35], data science and machine learning [36–40], geography [41], and other fields.

Motivated by the scientific significance of Procrustes problem and the absence of a dedicated package, we designed the `Procrustes` library: a flexible, easy-to-use, and easy-to-extend Python library for finding the optimal transformation that makes two matrices as close as possible to each other. As summarized in Table 4.1, the `Procrustes` library includes algorithms for orthogonal [9], rotational [9–11], symmetric [12–14], and permutation [16, 17] Procrustes problems and their two-sided counterparts [7, 19, 20]. Notably, the softassign algorithm [23, 42] is applied for solving the two-sided permutation Procrustes problem, which gives higher accuracy than other approaches, with commensurate computational costs. The most prevalent alternative for Procrustes analysis are the utilities in ‘scipy‘, which only supports

Table 4.1 Procrustes Library: Summary of Procrustes methods currently implemented for constrained minimization of the $\|\mathbf{SAT} - \mathbf{B}\|_F^2$ objective function.*

Procrustes Type	**S**	**T**	Constraints
Generic [3]	$\mathbf{I}$	$\mathbf{T}$	None
Orthogonal [2, 8, 9]	$\mathbf{I}$	$\mathbf{Q}$	$\mathbf{Q}^{-1} = \mathbf{Q}^{\dagger}$
Rotational [9–11]	$\mathbf{I}$	$\mathbf{R}$	$\begin{cases} \mathbf{R}^{-1} = \mathbf{R}^{\dagger} \\ \lvert\mathbf{R}\rvert = 1 \end{cases}$
Symmetric [12–14]	$\mathbf{I}$	$\mathbf{X}$	$\mathbf{X} = \mathbf{X}^{\dagger}$
Permutation [15–18]	$\mathbf{I}$	$\mathbf{P}$	$\begin{cases} [\mathbf{P}]_{ij} \in \{0,1\} \\ \sum_{i=1}^{n}[\mathbf{P}]_{ij} = \sum_{j=1}^{n}[\mathbf{P}]_{ij} = 1 \end{cases}$
Two-sided Orthogonal [19]	$\mathbf{Q}_1^{\dagger}$	$\mathbf{Q}_2$	$\begin{cases} \mathbf{Q}_1^{-1} = \mathbf{Q}_1^{\dagger} \\ \mathbf{Q}_2^{-1} = \mathbf{Q}_2^{\dagger} \end{cases}$
Two-sided Orthogonal with One Transformation [20]	$\mathbf{Q}^{\dagger}$	$\mathbf{Q}$	$\mathbf{Q}^{-1} = \mathbf{Q}^{\dagger}$
Two-sided Permutation [7]	$\mathbf{P}_1^{\dagger}$	$\mathbf{P}_2$	$\begin{cases} [\mathbf{P}_1]_{ij} \in \{0,1\} \\ [\mathbf{P}_2]_{ij} \in \{0,1\} \\ \sum_{i=1}^{n}[\mathbf{P}_1]_{ij} = \sum_{j=1}^{n}[\mathbf{P}_1]_{ij} = 1 \\ \sum_{i=1}^{n}[\mathbf{P}_2]_{ij} = \sum_{j=1}^{n}[\mathbf{P}_2]_{ij} = 1 \end{cases}$
Two-sided Permutation with One Transformation [2, 21]	$\mathbf{P}^{\dagger}$	$\mathbf{P}$	$\begin{cases} [\mathbf{P}]_{ij} \in \{0,1\} \\ \sum_{i=1}^{n}[\mathbf{P}]_{ij} = \sum_{j=1}^{n}[\mathbf{P}]_{ij} = 1 \end{cases}$

* The general-purpose implementations of the heuristic k-opt [4–6], flip-flop [7], and softassign [22–24] algorithms are also implemented in Procrustes library. The generalized Procrustes method is also being implemented in the future release.

one-sided orthogonal Procrustes and one-sided rotational Procrustes with centering and scaling [43].

This chapter demonstrates the design structure and usage of the free, open-source, and cross-platform `Procrustes` library written in Python 3. This library is well-documented and extensively tested on Microsoft Windows, macOS, and Linux operating systems. The section 4.2 shows how to install and use the `Procrustes` library, section 4.3 discusses its design, section 4.4 showcases various applications of its functionality, and section 4.5 summarizes our concluding remarks. A detailed description of various `Procrustes` algorithms implemented in our package is presented in the Appendix B.

4.2 Installation

4.2.1 Prerequisites

To install, test, and use the `Procrustes` library, one needs the following Python packages:

- Python ≥ 3.6: http://www.python.org/
- SciPy ≥ 1.5.0: http://www.scipy.org/
- NumPy ≥ 1.18.5: http://www.numpy.org/
- Pip ≥ 19.0: https://pip.pypa.io/
- PyTest ≥ 5.3.4: https://docs.pytest.org/
- PyTest-Cov ≥ 2.8.0: https://pypi.org/project/pytest-cov/
- Sphinx ≥ 2.3.0, if one wishes to build the documentation locally: https://www.sphinx-doc.org/

These packages can be installed individually or all at once using `pip` and `conda` package management systems, as described in the next section.

4.2.2 Installation

The stable release of the package can be easily installed through the `pip` and `conda` package management systems, which install the dependencies automatically, if not available. To use `pip`, simply run the following command:

```
pip install qc-procrustes
```

To use `conda`, one can either install the package through Anaconda Navigator or run the following command in a desired `conda` environment:

```
conda install -c theochem procrustes
```

Alternatively, the `Procrustes` source code can be download from GitHub (either the stable version or the development version) and then installed from source. For example, one can download the latest source code using *git* by:

```
# download source code
git clone git@github.com:theochem/procrustes.git
cd procrustes
```

From the parent directory, the dependencies can either be installed using `pip` by:

```
# install dependencies using pip
pip install -r requirements.txt
```

or, through `conda` by:

```
# create and activate myenv environment
conda create -n myenv python=3.6
conda activate myenv

# install dependencies using conda
conda install --yes --file requirements.txt
```

Finally, the `Procrustes` package can be installed (from source) by:

```
# install Procrustes from source
pip install .
```

4.2.3 Testing

To make sure that the package is installed properly, the `Procrustes` tests should be executed using `pytest` from the parent directory:

```
pytest -v .
```

In addition, to generate a coverage report alongside testing, one can use:

```
pytest --cov-config=.coveragerc --cov=procrustes procrustes/test
```

4.3 Structure of the Package

The source code and tests for the package are placed in *procrustes/procrustes* directory, which by itself includes the *utils* module and seven other modules for each flavour of the Procrustes analysis. The *utils* module contains some common utility functions such as: zero-padding columns/rows, removing zero-padded columns/rows, translating and scaling matrices and computing error/distance between matrices. For example, when the **A** and **B** matrices have different dimensions, the smaller matrix is padded with zeros rows and/or columns. It is also common to first translate and scale one matrix before transforming it to match the other. Additionally, the module includes the 'ProcrustesResult' class, instances of which are returned by all Procrustes functions, and which contains the results (e.g., transformation matrix, final errors, and translated/scaled/padded matrices) as attributes.

The remaining modules, including *orthogonal*, *rotational*, *permutation*, *symmetric*, *softassign*, *generic*, and *generalized*, implement various one-sided Procrustes algorithms and their two-sided variations, when applicable (see Table 4.1). For *permutation.py*, there are additional flavors compared to other modules due to its wider application range. The *test* directory contains units test for each module to prevent the inadvertent introduction of bugs; currently our test coverage is 94%.

Of note, k-opt heuristic algorithms are also implemented in *kopt* module with two functions, *kopt_heuristic_single* and *kopt_heuristic_double*. The main idea of k-opt is swap manipulations [4–6], which can be used to improve the result of two-sided permutation Procrustes problems. With the successful applications in traveling salesman problem [44] and quadratic assignment problems [45], the greedy k-opt search algorithm can also be used for softassign problem [46], which turned out to improve the accuracy.

The documentation and tutorials of the `Procrustes` library are contained in the *procrustes/doc* directory. The examples include Python scripts (and Jupyter notebooks) showing various applications of the Procrustes analysis. For further

information and the most up-to-date documentation and examples, please refer to the `Procrustes` website.

To ensure that our library is extendable, we adhered to many modern software development best practices including comprehensive documentation, extensive testing (current coverage=94%), strict or "uncompromising" code formatting (with `bandit`, `pylint`, `flake8`, and `PEP8`), and continuous integration (with GitHub Actions and Travis CI).

4.4 How to Use Procrustes

4.4.1 Quick Start of `Procrustes`

The code block below gives an example of the orthogonal Procrustes problem for random matrices **A** and **B**. Here, matrix **B** is constructed by shifting an orthogonal transformation of matrix **A**, so the matrices can be perfectly matched. As is the case with all Procrustes flavours, the user can specify whether the matrices should be translated (so that both are centered at origin) and/or scaled (so that both are normalized to unity with respect to the Frobenius norm). In addition, the other optional arguments (not appearing in the code-block below) specify whether the zero columns (on the right-hand side) and rows (at the bottom) should be removed prior to transformation.

```
import numpy as np
from scipy.stats import ortho_group
from procrustes import orthogonal

# random input 10x7 matrix A
a = np.random.rand(10, 7)

# random orthogonal 7x7 matrix T
t = ortho_group.rvs(7)

# target matrix B (which is a shifted AxT)
b = np.dot(a, t) + np.random.rand(1, 7)

# orthogonal Procrustes analysis with translation
result = orthogonal(a, b, scale=True, translate=True)

# display Procrustes results
print(result.error)     # error (expected to be zero)
```

```
print(result.t)         # transformation matrix (same as T)
print("Does the obtained transformation matches t matrix? ",
    np.allclose(t, result.t))
print(result.new_b)
print(result.new_a)
```

4.4.2 Chemical Structure Alignment

Molecular alignment is a fundamental problem in cheminformatics and can be used for structure determination [47], similarity based searching [48, 49], and ligand-based drug design [50] *et al.*. This problem can be solved by orthogonal Procrustes when given two matrices representing three-dimensional coordinates. The code block below shows the ease-of-use of the `Procrustes` library for protein structure alignment, one of the most fundamental problems in structural biophysics.

Here, we used `IOData` library [51] to load the Protein Data Bank (PDB) file format containing the X-ray crystal structure of the human deoxyhemoglobin (PDB ID: 2HHB). This selection of this protein was inspired by the `BiomolecularStructures` library* which contains the well-known Kabsch algorithm [52, 53] for structure alignment. This algorithm is the same as rotational Procrustes, which allows one to compare the accuracy and efficiency of our implementation to those of existing libraries. The structure of 2HHB has cyclic-C_2 global symmetry, where chain A and chain C (chain B and chain D) are hemoglobin deoxy-alpha (beta) chains as denoted in Figure 4.1 (a). Thus the rotational Procrustes can be used to align the C_α atoms in chain A and C of the protein to show that they are homologous. The root-mean-square deviation (RMSD) is traditionally used to assess the discrepancy between structures before and after the translation-rotation transformation (39.5 Å and 0.23 Å respectively).

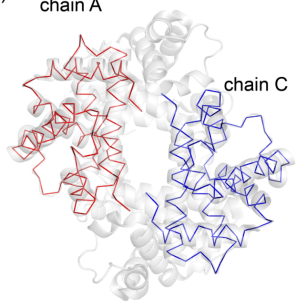
```
import numpy as np

from iodata import load_one
from iodata.utils import angstrom
from procrustes import rotational

# load PDB
pdb = load_one("2hhb.pdb")

```

*https://biomolecularstructures.readthedocs.io/en/latest/kabsch/

```
# get coordinates of C_alpha atoms in chains A & C (in angstrom)
chainid = pdb.extra['chainids']
attypes = pdb.atffparams['attypes']
ca_a = pdb.atcoords[(chainid == 'A') & (attypes == 'CA')] / angstrom
ca_c = pdb.atcoords[(chainid == 'C') & (attypes == 'CA')] / angstrom

print("RMSD of initial coordinates:")    # output: 39.47
print(np.sqrt(np.mean(np.sum((ca_a - ca_c)**2, axis=1))))

# rotational Procrustes analysis
result = rotational(ca_a, ca_c, translate=True)

# compute transformed (translated & rotated) coordinates of chain A
ca_at = np.dot(result.new_a, result.u)

print("RMSD of transformed coordinates:")    # output: 0.23
print(np.sqrt(np.mean(np.sum((ca_at - result.new_b)**2, axis=1))))
```

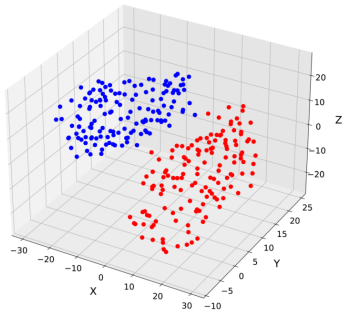
(A) Protein Structure Alignment: Rotational Procrustes

(i)



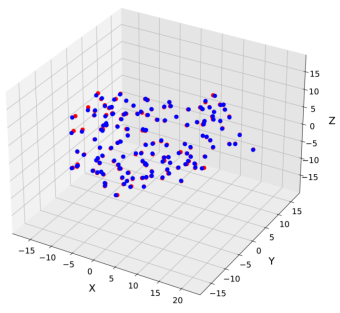
human deoxyhemoglobin
(PDB ID: 2HHB)

(ii)



before alignment
RMSD = 39.47 Å

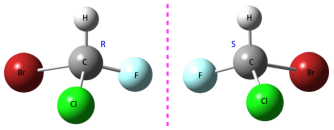
(iii)



after alignment
RMSD = 0.23 Å

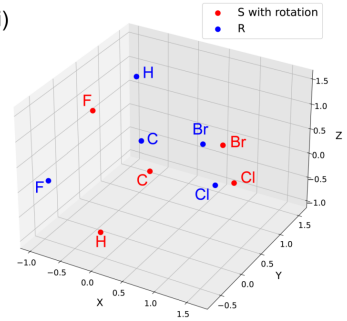
(B) Chirality Center Checking: Rotational and Orthogonal Procrustes

(i)



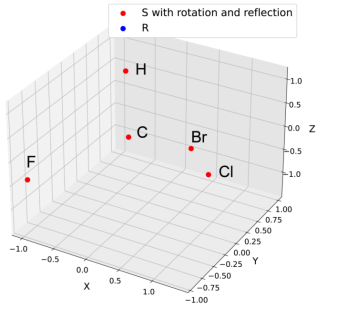
(*R*)-bromochloro-fluoromethane (*S*)-bromochloro-fluoromethane

(ii)



rotational Procrustes: rotation only

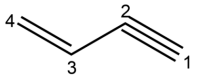
(iii)



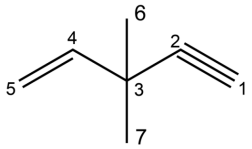
orthogonal Procrustes: rotation + reflection

(C) Atom-Atom Mapping: Two-sided Permutation Procrustes with One Transformation

(i)



but-1-en-3-yne



3,3-dimethylpent-1-en-4-yne

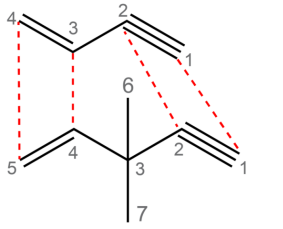
(ii)

$$\mathbf{A} = \begin{bmatrix} 6 & 3 & 0 & 0 & 0 & 0 & 0 \\ 3 & 6 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 6 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 6 & 3 & 0 & 0 & 0 & 0 & 0 \\ 3 & 6 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 6 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 6 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 6 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 6 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 6 \end{bmatrix}$$

(iii)

$$\mathbf{P}^{\mathrm{T}}\mathbf{A}\mathbf{P} = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{bmatrix} 6 & 3 & 0 & 0 & 0 & 0 & 0 \\ 3 & 6 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 6 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{matrix}$$



mapping atoms of molecules A and B

(D) Ranking by Reordering: Two-sided Permutation Procrustes with One Transformation

(i)

$$\mathbf{A} = \begin{array}{c} \begin{matrix} \textit{Duke} & \textit{Miami} & \textit{UNC} & \textit{UVA} & \textit{VT} \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 45 & 0 & 18 & 8 & 20 \\ 3 & 0 & 0 & 2 & 0 \\ 31 & 0 & 0 & 0 & 0 \\ 45 & 0 & 27 & 38 & 0 \end{bmatrix} \end{array} \begin{matrix} \textit{Duke} \\ \textit{Miami} \\ \textit{UNC} \\ \textit{UVA} \\ \textit{VT} \end{matrix}$$

score-differential matrix

(ii)

$$\mathbf{B} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

rank-differential matrix

(iii)

Team	Ranking
Duke	5
Miami	2
UNC	4
UVA	3
VT	1

Fig. 4.1 Examples of Procrustes analysis. (More figure captions on next page.)

(A). Protein structure alignment with rotational Procrustes. (i). The structure of human deoxyhemoglobin includes hemoglobin deoxy alpha chains (chain A in red and chain C in blue), and hemoglobin deoxy beta chains (both in grey). (ii). Coordinates of C_α atoms in chain A and chain C before orthogonal-rotational Procrustes alignment. (iii) Coordinates of C_α atoms in chain A and chain C after orthogonal-rotational Procrustes alignment.

(B). Chirality center checking with orthogonal-rotational Procrustes. (i). Chemical structure of (*R*)-CHFClBr and (*S*)-CHFClBr. (ii). Atomic coordinates of (*R*)-CHFClBr and (*S*)-CHFClBr before alignment. (iii). Atomic coordinates of (*R*)-CHFClBr and (*S*)-CHFClBr after alignment with rotational Procrustes.

(C). Atom-Atom mapping with two-sided permutation with one transformation Procrustes. (i). Chemical structure of molecule A and molecule B. (ii). Matrix representation of molecule A and molecule B. (iii). Atom-atom mapping of molecule A and molecule B where the transformed representation matrix is show above and the the assignment is show below. The number in grey are the assignment numbers. The zero-padded rows and columns are highlighted in blue.

(D). Ranking by reordering with two-sided permutation Procrustes with one transformation (i). The score-differential matrix based on the team by team score. (ii). The rank-differential matrix built from the given team by team score table. (iii). Ranking result of five American collegiate football teams.

4.4.3 Chirality Check

In chemistry, a molecule is chiral if it cannot be superimposed onto its mirror image by any combination of translation and rotation. These non-superposable mirror images are called enantiomers which share identical chemical and physical properties, but have distinct chemical reactivity and optical rotation properties.

The code block below shows how easily the `Procrustes` library can be used to check whether two geometries of the CHFClBr molecule are enantiomers (as in Figure 4.1 (b)) using the `IOData` library to obtain their three-dimensional coordinates from XYZ files. This is done by testing whether their coordinates can be matched through translation and rotation (i.e., rotational Procrustes); the obtained Procrustes error of 26.09 Å reveals that these two structures are not identical. However, it is confirmed that the two coordinates are enantiomers because they can be matched through translation, rotation, and reflection (i.e., orthogonal Procrustes) gives a Procrustes error of 4.43×10^{-8}Å; thus, reflection is essential to match the structures.

```python
import numpy as np

from iodata import load_one
from procrustes import orthogonal, rotational

# load CHClFBr enantiomers' coordinates from XYZ files
a = load_one("enantiomer1.xyz").atcoords
b = load_one("enantiomer2.xyz").atcoords

# rotational Procrustes on a & b coordinates
result_rot = rotational(a, b, translate=True, scale=False)
print("Error = ", result_rot.error)    # output: 26.085545

# orthogonal Procrustes on a & b coordinates
result_ortho = orthogonal(a, b, translate=True, scale=False)
print("Error = ", result_ortho.error)    # output: 4.432878e-08
```

4.4.4 Atom-Atom Mapping

Given two molecular structures, it is important to be able to identify atoms that are chemically similar. This a commonly used in 3D QSAR pharmacore analysis [54, 55], substructure searching [56], metabolic pathway identification [57–59], and chemical machine learning [60, 61].

The code block below shows how easily the `Procrustes` library can be used to map atoms of but-1-en-3-yne and 3,3-dimethylpent-1-en-4-yne as depicted in Figure 4.1 (c). Based on our chemical intuition, we can tell that the triple and double bonds of both molecules "match" one another; however, this is not revealed by simple (geometric) molecular alignment based on three-dimensional coordinates does not identify that. The pivotal step is defining a representation that contains bonding information, and then using permutation Procrustes to match atoms between the two chemical structures. Inspired by graph theory, we represented each molecule with an "adjacency" matrix where the diagonal elements are the atomic numbers and the off-diagonal elements are the bond orders. The two-sided permutation Procrustes with one-transformation can be used to find the optimal matching of the two matrices.

It is important to note that the permutation Procrustes requires the two matrices to be of the same size, so the smaller matrix $\mathbf{A}$ is padded with zero rows and columns to have same shape as matrix $\mathbf{B}$. After obtaining the optimal permutation matrix $\mathbf{P}$, the transformed matrix $\mathbf{P}^{\dagger}\mathbf{AP}$ should be compared to matrix $\mathbf{B}$ for identifying the matching atoms; the zero rows/columns correspond to atoms in $\mathbf{B}$ for which there are no corresponding atoms in $\mathbf{A}$. The mapping between atoms can be also directly deduced from matrix $\mathbf{P}$.

```
import numpy as np

from procrustes import permutation_2sided

# Define molecule A representing "but-1-en-3-yne"
A = np.array([[6, 3, 0, 0],
              [3, 6, 1, 0],
              [0, 1, 6, 2],
              [0, 0, 2, 6]])

# Define molecule B representing "3,3-dimethylpent-1-en-4-yne"
B = np.array([[6, 3, 0, 0, 0, 0, 0],
              [3, 6, 1, 0, 0, 0, 0],
              [0, 1, 6, 1, 0, 1, 1],
              [0, 0, 1, 6, 2, 0, 0],
              [0, 0, 0, 2, 6, 0, 0],
              [0, 0, 1, 0, 0, 6, 0],
              [0, 0, 1, 0, 0, 0, 6]])

# two-sided permutation Procrustes
result = permutation_2sided(A, B,
                            method="approx-normal1",
```

```
                                single=True, pad=True)

# Compute the transformed molecule A
P = result.t
new_A = np.dot(P.T, np.dot(result.new_a, P)).astype(int)
print("Transformed A: \n", new_A)     # compare to B
```

4.4.5 Ranking by Reordering

The problem of ranking a set of objects is ubiquitous not only in everyday life, but also for many scientific problems such as information retrieval [62, 63], recommender systems [64], natural language processing [61], and drug discovery [65, 66].

Table 4.2 Team by team game score differential from [67]

Team	Duke	Miami	UNC	UVA	VT
Duke	0	0	0	0	0
Miami	45	0	18	8	20
UNC	3	0	0	2	0
UVA	31	0	0	0	0
VT	45	0	27	38	0

The code block below shows how easily the `Procrustes` library can be used to rank five American collegiate football teams, where each team plays one game against every other team, using their score-differentials as summarized in Table 4.2. Here, each team is given a zero score for a game they lost (e.g., Duke lost to every other team) and the score difference is calculated for games won (e.g., Miami beat Duke by 45 points and UNC by 18 points). These results are also summarized in the square score-differential matrix $\mathbf{A}$ in cf. Figure 4.1 (d). Two-sided permutation Procrustes can be used to rank these teams, but one needs to define a proper target matrix. Traditionally, the rank-differential matrix has been used for this purpose and is defined for n teams as,

$$\mathbf{R}_{n\times n} = \begin{bmatrix} 0 & 1 & 2 & \cdots & n-1 \\ & 0 & 1 & \cdots & n-2 \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & 1 \\ & & & & 0 \end{bmatrix} \tag{4.3}$$

The rank-differential matrix is an upper-triangular matrix and its ij-th element specifies the difference in ranking between team i and team j. This a sensible target for the score-differential matrix. Now, the two-sided permutation Procrustes method can be used to find the permutation matrix that maximizes the similarity between the score-differential matrix, $\mathbf{A}$, and the rank-differential matrix based on Equation 4.3, $\mathbf{B}$, resulting in $[5, 2, 4, 3, 1]$ as the final rankings of the teams.

```
import numpy as np

from procrustes import permutation_2sided

# input score-differential matrix
A = np.array([[ 0,  0,  0 ,  0,  0 ],     # Duke
              [45,  0, 18,  8,  20],     # Miami
              [ 3,  0,  0 ,  2,  0 ],     # UNC
              [31,  0,  0 ,  0,  0 ],     # UVA
              [45,  0, 27, 38,  0 ]])    # VT

# make rank-differential matrix
n = A.shape[0]
B = np.zeros((n, n))
for index in range(n):
    B[index, index:] = range(0, n - index)

# rank teams using two-sided Procrustes
result = permutation_2sided(A, B,
                            single=True,
                            method="approx-normal1")

# compute teams' ranks
_, ranks = np.where(result.t == 1)
ranks += 1
print("Ranks = ", ranks)      # displays [5, 2, 4, 3, 1]
```

4.5 Conclusions

This chapter introduces `Procrustes`, a free, open-source, and cross-platform Python 3 library implementing algorithmic solutions to a broad range of Procrustes problems, including the one- and two-sided variants of the permutation, rotation, orthogonal, and symmetric tasks. It also features an implementation of the softassign algorithm to solve two-sided permutation Procrustes problems with one transforma-

tion as well as the k-opt local search algorithm to improve the results of the two-sided permutation Procrustes problem. The scope and utility of this library for scientific computation is emphasized, featuring examples from chemical physics and molecular biology, which are released as Jupyter notebooks alongside the source code.

The algorithms implemented are detailed in the appendix emphasizing the more innovative aspects of the `Procrustes` library, most notably several valuable heuristics for two-sided permutation Procrustes problems. For future work, more flavors of Procrustes methods will be implemented, including generalized Procrustes analysis (GPA) [3, 68, 69], projected Procrustes analysis [70, 71], and continuous Procrustes methods [72, 73]. Furthermore, improvements to the scalability of the implemented algorithms will help make the analysis of large matrices more feasible.

This package functions as an alternative tool for shape analysis and solving quadratic assignment problems with many other potential applications in science and engineering.

References

[1] John R Hurley and Raymond B Cattell. The procrustes program: Producing direct rotation to test a hypothesized factor structure. *Behavioral Science*, 7(2): 258–262, 1962.

[2] Peter H. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, Mar 1966. ISSN 1860-0980. doi: 10.1007/BF02289451.

[3] John C. Gower. Procrustes methods. *WIREs Computational Statistics*, 2(4): 503–508, 2010. doi: 10.1002/wics.107.

[4] Keld Helsgaun. General k-opt submoves for the lin–kernighan tsp heuristic. *Mathematical Programming Computation*, 1(2-3):119–163, 2009.

[5] Georges A Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958.

[6] Shen Lin and Brian W Kernighan. An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21(2):498–516, 1973.

[7] Pythagoras Papadimitriou. *Parallel solution of SVD-related problems, with applications*. PhD thesis, University of Manchester, 1993.

[8] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov 2000. ISSN 0162-8828. doi: 10.1109/34.888718.

[9] John C Gower and Garmt B Dijksterhuis. *Procrustes problems*, volume 30. Oxford University Press, 2004.

[10] Frank B Brokken. Orthogonal procrustes rotation maximizing congruence. *Psychometrika*, 48(3):343–352, 1983.

[11] JL Farrell, JC Stuelpnagel, RH Wessner, JR Velman, and JE Brook. A least squares estimate of satellite attitude (grace wahba). *SIAM Review*, 8(3):384–386, 1966.

[12] Nicholas J Higham. The symmetric procrustes problem. *BIT Numerical Mathematics*, 28(1):133–143, 1988.

[13] René Escalante and Marcos Raydan. Dykstra's algorithm for constrained least-squares rectangular matrix problems. *Computers & Mathematics with Applications*, 35(6):73–79, 1998.

[14] Juan Peng, Xi-Yan Hu, and Lei Zhang. The (m, n)-symmetric procrustes problem. *Applied Mathematics and Computation*, 198(1):24–34, 2008.

[15] Farnaz Heidar Zadeh and Paul W Ayers. Molecular alignment as a penalized permutation procrustes problem. *Journal of Mathematical Chemistry*, 51(3): 927–936, 2013.

[16] Harold W Kuhn. Variants of the hungarian method for assignment problems. *Naval Research Logistics Quarterly*, 3(4):253–258, 1956.

[17] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics*, 2(1-2):83–97, 1955.

[18] Nick Higham and Pythagoras Papadimitriou. Matrix procrustes problems. *Rapport technique, University of Manchester*, 1995.

[19] Peter H Schönemann. On two-sided orthogonal procrustes problems. *Psychometrika*, 33(1):19–33, 1968.

[20] Shinji Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

[21] Chris Ding, Tao Li, and Michael I Jordan. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 183–192. IEEE, 2008.

[22] JJ Kosowsky and Alan L Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7(3):477–490, 1994.

[23] Steven Gold and Anand Rangarajan. Softassign versus softmax: Benchmarks in combinatorial optimization. In *Advances in Neural Information Processing Systems*, pages 626–632, 1996.

[24] Anand Rangarajan, Alan L Yuille, Steven Gold, and Eric Mjolsness. A convergence proof for the softassign quadratic assignment algorithm. In *Advances in Neural Information Processing Systems*, pages 620–626, 1997.

[25] Diwei Zhou, Ian L Dryden, Alexey A Koloydenko, and Li Bai. Procrustes analysis for diffusion tensor image processing. *International Journal of Computer Theory and Engineering*, 5(1):108, 2013.

[26] Graham McNeill and Sethu Vijayakumar. Hierarchical procrustes matching for shape retrieval. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 885–894. IEEE, 2006.

[27] Wanhyun Cho, Soonja Kang, Sangkyoon Kim, and Soonyoung Park. Human action recognition using product manifold theory and procrustes shape analysis. In *Computer Science and its Applications*, pages 831–836. Springer, 2015.

[28] Norm C Kenkel and Laszlo Orlóci. Applying metric and nonmetric multidimensional scaling to ecological studies: some new results. *Ecology*, 67(4): 919–928, 1986.

[29] F James Rohlf and Dennis Slice. Extensions of the procrustes method for the optimal superimposition of landmarks. *Systematic Biology*, 39(1):40–59, 1990.

[30] Sri Priya Ponnapalli, Michael A Saunders, Charles F Van Loan, and Orly Alter. A higher-order generalized singular value decomposition for comparison of global mrna expression from multiple organisms. *PloS ONE*, 6(12):e28072, 2011.

[31] Daniel G Oblinsky, Bryan MB VanSchouwen, Heather L Gordon, and Stuart M Rothstein. Procrustean rotation in concert with principal component analysis of molecular dynamics trajectories: Quantifying global and local differences between conformational samples. *The Journal of Chemical Physics*, 131(22): 12B612, 2009.

[32] Jose Manuel Andrade, María P. Gómez-Carracedo, Wojtek Krzanowski, and Mikael Kubista. Procrustes rotation in analytical chemistry, a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 72(2):123–132, 2004. ISSN 01697439. doi: 10.1016/j.chemolab.2004.01.007.

[33] Gareth W Richings and Scott Habershon. A new diabatization scheme for direct quantum dynamics: Procrustes diabatization. *The Journal of Chemical Physics*, 152(15):154108, 2020.

[34] M. D. Zoltowski and D. Stavrinides. Sensor array signal processing via a procrustes rotations based eigenanalysis of the esprit data pencil. *IEEE*

Transactions on Acoustics, Speech, and Signal Processing, 37(6):832–861, Jun 1989. ISSN 0096-3518. doi: 10.1109/ASSP.1989.28056.

[35] H. Zou, B. Huang, X. Lu, H. Jiang, and L. Xie. A robust indoor positioning system based on the procrustes analysis and weighted extreme learning machine. *IEEE Transactions on Wireless Communications*, 15(2):1252–1266, Feb 2016. ISSN 1536-1276. doi: 10.1109/TWC.2015.2487963.

[36] Takanori Maehara and Kazuo Murota. Simultaneous singular value decomposition. *Linear Algebra and its Applications*, 435(1):106–116, 2011.

[37] Peng Zhang, Zhou Sun, Chunbo Fan, and Yi Ding. Pre-scaling anisotropic orthogonal procrustes analysis based on gradient descent over matrix manifold. In *International Conference in Swarm Intelligence*, pages 371–379. Springer, 2015.

[38] Cencheng Shen, Joshua T Vogelstein, and Carey E Priebe. Manifold matching using shortest-path distance and joint neighborhood selection. *Pattern Recognition Letters*, 92:41–48, 2017.

[39] Frank Schoeneman, Suchismit Mahapatra, Varun Chandola, Nils Napp, and Jaroslaw Zola. Error metrics for learning reliable manifolds from streaming data. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 750–758. SIAM, 2017.

[40] Casimiro S Munita, Lúcia P Barroso, and Paulo MS Oliveira. Variable selection study using procrustes analysis. *Open Journal of Archaeometry*, 1(1):7, 2013.

[41] Eleonora Maset, Fabio Crosilla, and Andrea Fusiello. Errors-in-variables anisotropic extended orthogonal procrustes analysis. *IEEE Geoscience and Remote Sensing Letters*, 14(1):57–61, 2017.

[42] Steven Gold, Anand Rangarajan, et al. Softmax to softassign: Neural network algorithms for combinatorial optimization. *Journal of Artificial Neural Networks*, 2(4):381–399, 1996.

[43] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. URL http://www.scipy.org/.

[44] William J. Cook, William H. Cunningham, William R. Pulleyblank, and Alexander Schrijver. *Combinatorial Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1998. ISBN 0-471-55894-X.

[45] Rainer E Burkard, Eranda Cela, Panos M Pardalos, and Leonidas S Pitsoulis. The quadratic assignment problem. In *Handbook of Combinatorial Optimization*, pages 1713–1809. Springer, 1998.

[46] Stefan Roth. Analysis of a deterministic annealing method for graph matching and quadratic assignment problems in computer vision. *Master's thesis, CVGPR-group, University of Mannheim*, 2001.

[47] Alain Ibáñez de Opakua, Frederik Klama, Ikenna E Ndukwe, Gary E Martin, R Thomas Williamson, and Markus Zweckstetter. Determination of complex small-molecule structures using molecular alignment simulation. *Angewandte Chemie International Edition*, 59(15):6172–6176, 2020.

[48] Paul W Finn and Garrett M Morris. Shape-based similarity searching in chemical databases. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 3(3):226–241, 2013.

[49] Takeshi Kawabata and Haruki Nakamura. 3d flexible alignment using 2d maximum common substructure: dependence of prediction accuracy on target-reference chemical similarity. *Journal of Chemical Information and Modeling*, 54(7):1850–1863, 2014.

[50] Nicola J Richmond, Peter Willett, and Robert D Clark. Alignment of three-dimensional molecules using an image recognition algorithm. *Journal of Molecular Graphics and Modelling*, 23(2):199–209, 2004.

[51] Toon Verstraelen, William Adams, Leila Pujal, Alireza Tehrani, Braden D Kelly, Luis Macaya, Fanwang Meng, Michael Richer, Raymundo Hernández-Esparza, Xiaotian Derrick Yang, et al. Iodata: A python library for reading, writing, and converting computational chemistry file formats and generating input files. *Journal of Computational Chemistry*, 42(6):458–464, 2021.

[52] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

[53] Wolfgang Kabsch. A discussion of the solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 34(5):827–828, 1978.

[54] Nicola J Richmond, Peter Willett, and Robert D Clark. Alignment of three-dimensional molecules using an image recognition algorithm. *Journal of Molecular Graphics and Modelling*, 23(2):199–209, 2004.

[55] Martin Mann, Feras Nahar, Norah Schnorr, Rolf Backofen, Peter F Stadler, and Christoph Flamm. Atom mapping with constraint programming. *Algorithms for Molecular Biology*, 9(1):23, 2014.

[56] Lingran Chen, James G Nourse, Bradley D Christie, Burton A Leland, and David L Grier. Over 20 years of reaction access systems from mdl: a novel reaction substructure search algorithm. *Journal of Chemical Information and Computer Sciences*, 42(6):1296–1310, 2002.

[57] Syed Asad Rahman, Sergio Martinez Cuesta, Nicholas Furnham, Gemma L Holliday, and Janet M Thornton. Ec-blast: a tool to automatically search and compare enzyme reactions. *Nature Methods*, 11(2):171–174, 2014.

[58] Tatsuya Akutsu. Efficient extraction of mapping rules of atoms from enzymatic reaction data. *Journal of Computational Biology*, 11(2-3):449–462, 2004.

[59] Wojciech Jaworski, Sara Szymkuć, Barbara Mikulak-Klucznik, Krzysztof Piecuch, Tomasz Klucznik, Michał Kaźmierowski, Jan Rydzewski, Anna Gambin, and Bartosz A Grzybowski. Automatic mapping of atoms across both simple and complex chemical reactions. *Nature Communications*, 10(1):1–11, 2019.

[60] Connor W Coley, Regina Barzilay, Tommi S Jaakkola, William H Green, and Klavs F Jensen. Prediction of organic reaction outcomes using machine learning. *ACS Central Science*, 3(5):434–443, 2017.

[61] Filipp Nikitin, Olexandr Isayev, and Vadim Strijov. Dracon: disconnected graph neural network for atom mapping in chemical reactions. *Physical Chemistry Chemical Physics*, 22(45):26478–26486, 2020.

[62] W Bruce Croft, Donald Metzler, and Trevor Strohman. *Search engines: Information retrieval in practice*, volume 520. Addison-Wesley Reading, 2010.

[63] Tie-Yan Liu. *Learning to Rank for Information Retrieval*, chapter 1, pages 1–30. Springer Science & Business Media, 2011.

[64] Gediminas Adomavicius and YoungOk Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):896–911, 2011.

[65] Shivani Agarwal, Deepak Dugar, and Shiladitya Sengupta. Ranking chemical structures for drug discovery: a new machine learning approach. *Journal of Chemical Information and Modeling*, 50(5):716–731, 2010.

[66] Matteo Re and Giorgio Valentini. Network-based drug ranking and repositioning with respect to drugbank therapeutic categories. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(6):1359–1371, 2013.

[67] Amy N Langville and Carl D Meyer. *Ranking by Reordering Methods*, chapter 8, pages 97–112. Princeton University Press, 2012.

[68] Mikkel B Stegmann and David Delgado Gomez. A brief introduction to statistical shape analysis. *Informatics and Mathematical Modelling, Technical University of Denmark, DTU*, 15(11), 2002.

[69] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

[70] Xavier Perez-Sala, Fernando De la Torre, Laura Igual, Sergio Escalera, and Cecilio Angulo. Subspace procrustes analysis. *International Journal of Computer Vision*, 121(3):327–343, 2017.

[71] Konstantin Chernoff and Mads Nielsen. Projected generalized procrustes alignment. In *International Conference on Information Processing in Medical Imaging*, pages 503–514. Springer, 2009.

[72] Reema Al-Aifari, Ingrid Daubechies, and Yaron Lipman. Continuous procrustes distance between two surfaces. *Communications on Pure and Applied Mathematics*, 66(6):934–964, 2013.

[73] Laura Igual, Xavier Perez-Sala, Sergio Escalera, Cecilio Angulo, and Fernando De la Torre. Continuous generalized procrustes analysis. *Pattern Recognition*, 47(2):659–671, 2014.

Chapter 5

A Curated Diverse Molecular Database of Blood-Brain Barrier Permeability

5.1 Introduction

The blood-brain barrier (BBB) denotes a regulatory and protective mechanism of microvasculature in the central nervous system (CNS) that is central to regulating the homeostatis of the CNS [1, 2] and protecting the CNS from toxins, pathogens, and inflammations [3]. However, it is estimated that 98% of small molecules are not BBB permeable [4]. Therefore, predicting BBB permeability for small molecules is a vital but challenging task in drug discovery and development [4–7].

However, existing computational models for a molecule's BBB permeability are inadequate. In particular, they are restricted by the limited size and chemical diversity of existing sets of training data[8]. Moreover, although many different machine-learning (ML) models for predicting BBB permeability have been proposed, these models are not directly comparable because they use widely varying training data, ranging from as few as 45 molecules [9, 10] to as many as 7236 molecules [11]. The purpose of this paper is to curate an accessible, clean, well-documented, and reasonably comprehensive dataset of BBB permeability data and present it in a way that is convenient for those building new BBB predictive models. While our database, B3DB, is not the first attempt to curate data from the literature to construct a molecular BBB database, B3DB contains more molecules, and categorizes the molecules based on experimental uncertainty. Both features are very helpful when developing and validating ML models for BBB.

There are two types of data for BBB, numerical and categorical data. Numerical data is usually reported as $\log BB$, the logarithm of brain-plasma concentration ratio,

$$\log BB = \log \frac{C_{brain}}{C_{blood}} \tag{5.1}$$

Categorical data simply labels whether a compound is BBB permeable (BBB+) or not (BBB-).

Among existing studies of BBB permeability, we mention Zhuang *et. al.*, who built a machine-learning (ML) model with resampling using a binary dataset of 2358 molecules [12]. Similarly, Zhao *et al* [13] compiled a dataset of 1336 BBB crossing drugs (BBB+) and 360 BBB non-crossing drugs (BBB-). A recent study reported an ensemble ML model using of 1757 molecules as the training data [14]. To our knowledge, the largest dataset previously reported in the literature was used in developing the LightBBB model, which uses the Light Gradient Boosting Machine (LightGBM) algorithm to build a predictive model. The LightBBB model's database included 7162 entries. (These entries include duplicates (multiple entries with the same International Chemical Identifier (InChI)) and molecules that could not be recognized by `RDKit` package, so in the end there are only 4491 unique valid molecules). We curate data from these three efforts, and 47 other smaller efforts, in B3DB. Unlike many previous efforts, B3DB includes many (1058) molecules with numeric $\log BB$ values. The largest previous dataset we know was the data source for LightBBB, which has $\log BB$ values for 696 unique valid molecules.

Here ∗, we present a new Blood-Brain Barrier Database, B3DB, which is intended to provide a benchmark dataset for modelling BBB permeability of small molecules. The original data was collected from 50 peer-reviewed publications or open access datasets. As described in the next section, we processed and cleaned the data, then categorized it based on its reliability. By categorizing the data in this way, users can choose whether they want to focus on the smaller subsets with the highest reliability, or prefer to consider larger datasets with slightly lower reliability. We hope that our meticulous methods of preparing and sorting the data may be of interest those who wish to curate databases for other, similar, properties.

B3DB includes both numerical data (1058 $\log BB$ values) and categorical data (4956 BBB+ and 2851 BBB-). Here is summary of key features of B3DB dataset.

- This is the largest BBB data set we know, both for categorical labels and $\log BB$ numerical values.

∗I would like to acknowledge the help of Yang Xi, Jinfeng Huang with the validation of molecular representations in this study.

- Because the chirality of molecules plays an important role in BBB permeability [15, 16], isomeric SMILES is to used to incorporate chiral specifications of molecules.
- Because some molecules have been measured multiple times, using different experimental methods and under different conditions, we divide the value into groups based on the quantity of experimental data and the similarity between reported values, so that users of B3DB can easily select subsets of the data with varying degrees of reliability.
- B3DB is extended with molecular descriptors computed with `mordred` [17], so that it can be used out-of-the-box for building BBB predictive models.

5.2 Methods and Materials

The next three sections describe how raw data was collected from various sources, cleaned, and curated. We then describe how the dataset was extended with chemical descriptors (beyond the reference BBB value). This workflow is summarized in Figure 5.1.

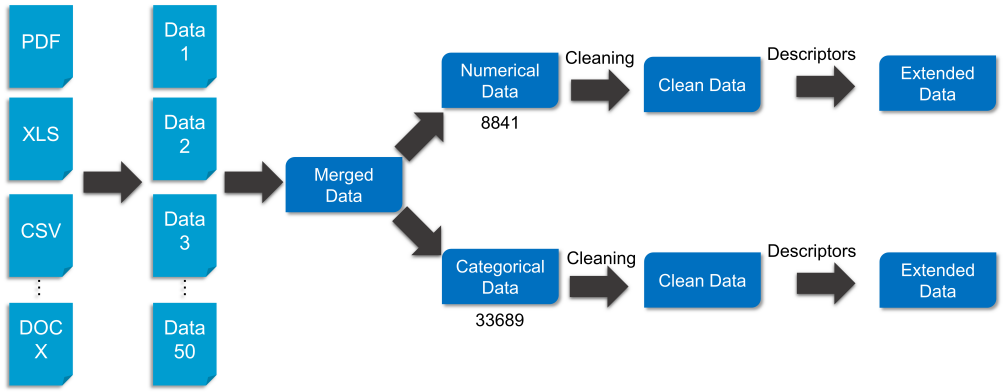


Fig. 5.1 Workflow for building B3DB. From left to right, the collection of raw BBB data, cleaning the raw data, categorization of cleaned data, and finally, extension of B3DB by computing other molecular descriptors.

5.2.1 Data Collecting

All the data was collected from the literature and open source databases. The dataset size, main available information, and data types are listed in Table 5.1 ∗. For each data source, a standard Excel workbook is formatted for further processing. If the original data is in portable document format (PDF), it is converted to a pandas [19] DataFrame and then stored in XLSX format with `tabula-py` [20]. For files in DOCX or DOC extension, as well as CSV, TXT and other Excel compatible formats, they are converted to Excel XLSX format directly, using Microsoft Office. We performed several automated consistency checks (e.g., numerical data should be reported as floating-point numbers) and manually verified a subset of the data to ensure that the data was faithfully transferred to *.xlsx format. In total, 33825 raw data records were collected.

The 50 datasets have various formats and include a wide range of information, so we constructed a template that contained only the most essential data, compound name, simplified molecular-input line-entry system (SMILES) string, PubChem compound identifier (CID), $\log BB$, BBB+/BBB- (whether a compound is BBB permeable or not), the IUPAC International Chemical Identifier (InChI), the threshold value used to determine categorical type of a compound, and the literature source for that data value.

5.2.2 Data Cleaning

In the data cleaning stage, an initial molecule specification (a SMILES string, PubChem CID, and/or compound name) is input; the output is also a SMILES string, but with transcription and typographical errors fixed, and with salts/solvents removed. In addition, molecules containing heavy metal atoms are removed from the database. A followed up standardization of molecular reorientation is performed which include updating valences, kekulizing and normalizing molecules, and neutralizing molecular charges. The basic procedure is shown in Figure 5.2 (A).

The first step is to fix invalid SMILES strings. For example, white spaces and line breaks in SMILES were removed. Some other issues (e.g., where a dash was used in lieu of a negative sign for the molecular charge) were manually remedied. Our data is drawn from 50 distinct sources, and a full molecule specification is not always provided. For example, some sources list only the compound names (and not the SMILES strings or PubChem CIDs); other sources list only PubChem CIDs. In these cases, `PubChemPy` [21] was used to access the PubChem [22] database to

∗Data of Ref. 18 accessed with `PyTDC` 0.1.5 as of Jan 25, 2021.

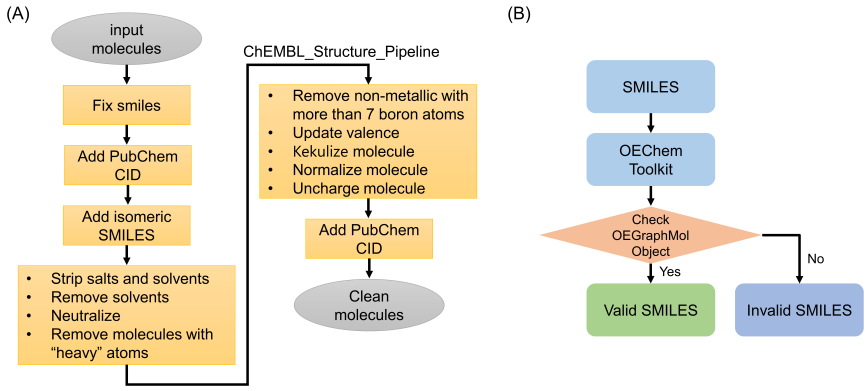


Fig. 5.2 Molecule representation cleaning and technical validation. (A). Flowchart of cleaning SMILES string representation of molecules. (B). Technical validation of molecular representation.

retrieve information about missing compound names, SMILES strings and PubChem CIDs. When only the compound name was available, there can be multiple PubChem instances. If this were to happen, the first Pubchem instance is selected and a note is added to the database flagging the potential ambiguity. Fortunately this does not seem to occur in this specific database. There are also a few molecules for which only molecular structures, and not SMILES or compound names, are provided. In these cases we built the molecules manually and searched for the Pubchem CID and SMILES string with the PubChem web interface. All the SMILES strings were loaded into `RdKit` [23] (version 2019.03.4) to build molecule objects. If the object is None, the SMILES is considered to be invalid. This leads to 33771 measured BBB instances.

Stereochemistry can play a significant role in a molecule's BBB permeability because of transporters' specific stereoselectivity [15, 16]. However, there is no stereochemical information in SMILES strings. To add stereochemical information to SMILES, and to deal with generic SMILES strings that were technically valid but not in canonical form, the original SMILES were upgraded to isomeric SMILES by using `PUG-REST API` [24] wherever possible. Otherwise, the canonical SMILES were retrieved from PubChem database with `PUG-REST API` [24]. The inclusion of stereochemical data about the molecules is an important, and (we believe) unique feature of B3DB.

Once the SMILES representations are fixed, `ChEMBL_Structure_Pipeline` [25] was used to strip the salts and neutralize the charge. Molecules containing metal atoms or heavy atom with atomic number greater than 20 (except for Zinc, Bromine, Krypton, Iodine, Xenon) were removed. Molecules with more than 7 boron atoms are also excluded due to problems of depicting borane compounds. Implicit valence and ring information were recomputed followed by kekulizing, normalization of molecules and molecular charges were neutralized. These revisions change the molecular structure, so the Pubchem CIDs were updated from the revised SMILES strings.

5.2.3 Data Curation

The curation procedures for numerical and categorical data are summarized in Figure 5.3. To curate the data, a unique chemical identifier is required. Although InChI is unique in principle, it cannot resolve tautomeric forms, which is a common source of ambiguity and error in chemical structure representation. Therefore, we examined the unique InChI generated with `RdKit` and the isomeric SMILES (and canonical SMILES where isomeric SMILES is unavailable). The number of unique SMILES is greater than the number of unique InChI values, but the redundancy is merely because each SMILES represents a specific resonance structure.

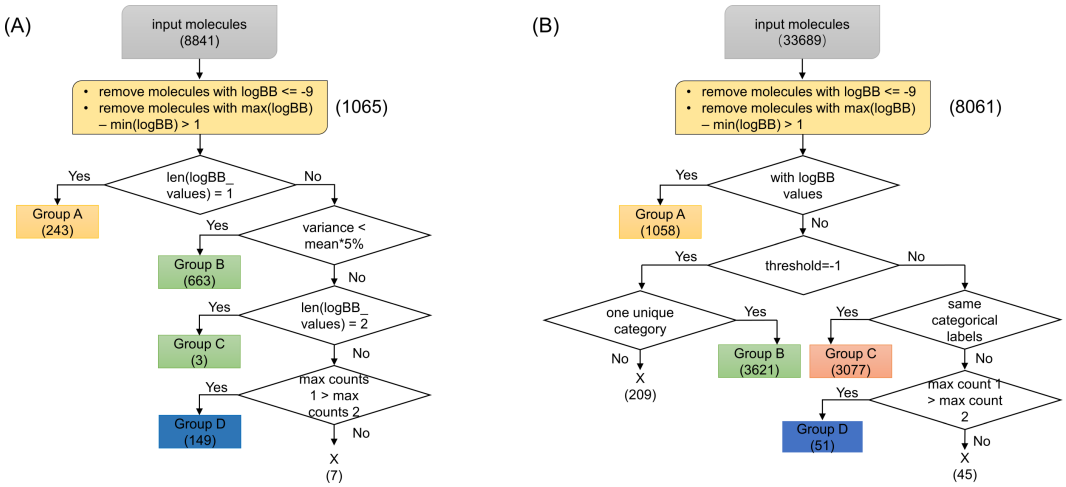


Fig. 5.3 Curation algorithm for numeric and categorical BBB data. (A). Curation pipeline for BBB data with $\log BB$ values. (B). Curation pipeline for BBB data with categorical information, either BBB+ or BBB-.

5.2.3.1 Curation of Numerical Data

To curate the 8841 numerical BBB data values, $\log BB$ values for each molecule were merged into a list. The 20 instances with $\log BB <= -9$ were regarded as outliers because, based on the distribution of $\log BB$ values, they seemed suspicious. Next, we identified molecules where there are multiple reported $\log BB$ values and eliminated those molecules from the database if the reported values differed significantly. Specifically, we eliminated 16 molecules where $\max(\log BB) - \min(\log BB) > 1$. The values that remain after curation are merged into 1065 molecular records. The molecular records are augmented, as necessary, to ensure that they are complete, including compound name, IUPAC name, isomeric (canonical) SMILES, etc..

Here is the detailed curation procedure for numeric data.

1. **Group A** (243 molecules). Molecules with only one unique $\log BB$ value.
2. **Group B** (663 molecules). Molecules with more than one $\log BB$ value, but all the the reported values differ by less than 5% from the mean value. In these cases, the mean value is used as the $\log BB$ value for the molecule.
3. **Group C** (3 molecules). Other molecules with two distinct $\log BB$ values. The (weighted) mean value is used as the curated value for group C (just as for group B).
4. **Group D** (149 molecules). Other molecules with more than two distinct values; whichever value occurs with greatest frequency is used. In three case, two distinct values were reported with maximum frequency; we discarded those molecules from the dataset.

The 7 molecules which failed to be categorized as group A, B, C or D, they are discarded. The final dataset therefore contains 1058 molecules; for most of these molecules (815 molecules) multiple, mutually consistent, values of $\log BB$ are reported in the literature.

5.2.3.2 Curation of Categorical Data

The 33689 data values were divided into two categories, numerical data and (binary) categorical data.

1. **Group A** (1058 molecules). Molecules with numerical data. Several threshold values for $\log BB$ have been used to determine if a molecule is BBB permeable or not, including 0 [26, 27], 0.1 [28], -1 [12, 13, 29–32], (-2, 1) [33]. The value of -1

is chosen as the threshold value to define if a compound is BBB+ or BBB- since this is the mostly widely used threshold and maximizes the ease of comparison with other studies.

2. **Group B** (3621 molecules). Molecules from sources that use $\log BB = -1$ as the threshold value, and where all sources agree on the categorical label. The unambiguous label is used.
3. **Group C** (3077 molecules). Molecules where all sources agree on the categorical label, but the sources that do not report their threshold value.
4. **Group D** (51 molecules). Molecules with two different BBB permeability labels. The most prevalent label is used. In the 45 cases where the two labels occurred with equal frequency, the molecule was discarded.

The 7807 remaining molecular records are augmented to ensure that they are complete, including compound name, IUPAC name, isomeric (canonical) SMILES, etc..

5.2.4 Validation of Molecular Representations

All the molecules are in canonical SMILES format and, if available from PubChem, also isomeric SMILES. We then attempt to load each SMILES string into `OEChem Toolkit` [34] as an OEGraphMol object ; if this is successful then this SMILES is regarded as valid. (See 5.2 (B).)

5.2.5 Data Extension with Chemical Descriptors

To better facilitate building BBB predictive models, the curated datasets were extended with chemical descriptors. Then 1613 chemical descriptors were calculated with `mordred version 1.1.1` [17]. The purpose of providing this extended data is to facilitate easy use of the B3DB, without requiring precomputation of cheminformatics descriptors.

5.2.6 Software and Packages

For consistency and reproducing purposes, all the data processing were performed in a `Python` 3.7.9 virtual environment created with `Conda` in CentOS Linux release 7.9.2009 which include `pandas` 1.2.1, `tabula-py` 2.2.0, `RDKit` 2020.09.1, `pubchempy` 1.0.4, `ChEMBL_Structure_Pipeline` 1.0.0, `OEChem Toolkit` [34]

provided by `openeye-toolkit` 2020.2.0, `SciPy` 1.5.2, `Numpy` 1.19.2, `mordred` 1.1.1, `PyTDC` 0.1.5. `ALOGPS` version 2.1 is also used for calculating octanol/water partition coefficient $\log P$. All the calculation were done with `Python 3.7.9` under a virtual environment created with Anaconda on Fedora 33.

5.3 Results and Discussions

5.3.1 Data Records

There are two datasets provided in this study, one with numeric $\log BB$ values (1058 molecules) and the other with categorical labels (7807 molecules with 4956 BBB+ and 2851 BBB-). B3DB data is stored in the comma-separated values (CSV) format and contains SMILES representations, compound name, IUPAC name, $\log BB$ value, threshold, BBB+/BBB- and the corresponding references along with 1613 molecular descriptors. This is summarized in Table 5.2. The data are openly accessible at GitHub, https://github.com/theochem/B3DB, as well as figshare platform [35].

To further valid the molecular representation of our dataset, `OEChem Toolkit` [34] was used and no problem was identified.

5.3.2 Analysis of Curated Datasets

The BBB data comes from 50 sources, and was acquired in different laboratories, under different conditions, and using different protocols. To characterize the experimental uncertainty, we examine the agreement between reported values, Figure 5.4. For 92.82% of the numerical data, there at most two unique $\log BB$ values are reported as shown in Figure 5.4 (a) and (c). Similarly, for 99.34% of the molecules, only a single categorical label is reported (Figure 5.4(d)); this is true even though the same molecule may appear in as many as 23 distinct sources (Figure 5.4(b)). More detailed data can be found in Table 5.3, 5.4, 5.5, 5.6.

Figure 5.5 reveals some features of the B3DB dataset. Presuming that the molecules in the dataset are relatively representative of (bio)organic molecules in general, the $\log BB$ for most of organic compound lie within the interval [-2, 2] (see Figure 5.5 (a)). The distribution of $\log BB$ values indicates that the numerical dataset is relatively balanced, though skewed towards BBB+ compounds.

Lipinski's Rule of 5 is a simple rule-of-thumb for evaluating a molecule's drug-likeness. Specifically, Lipinski's Rule of 5 states that good absorption or permeation is more likely if a molecule has less than: 5 hydrogen-bond donors, 10 hydrogen-bond acceptors, 500 Dalton molecular weight, and a predicted $\log P$ value less than 5. It is

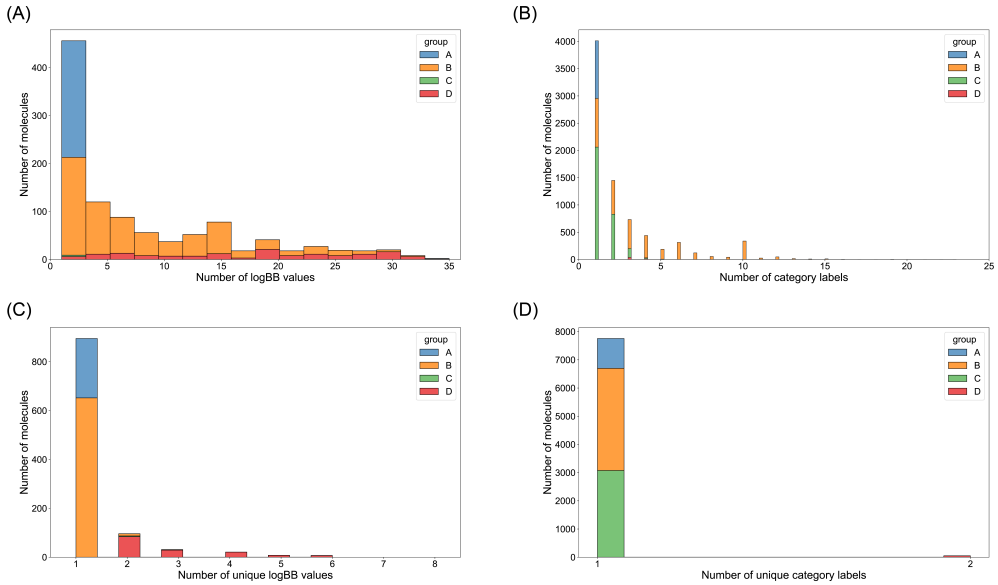


Fig. 5.4 Characterization of the nature and frequency of multiple/redundant data in B3DB. (A). Multiplicity of source $\log BB$ values in each group of the numerical dataset. (B). Prevalence of source BBB permeability labels in each group of the categorical dataset. (C). Multiplicity of unique $\log BB$ values in each group of the numerical dataset. (D). Prevalence of unique BBB permeability labels in each group of the categorical dataset. More data can be found at Table 5.3, 5.4, 5.5 and 5.6.

observed that the molecule weight of most BBB+ compounds (93.10%) is less than 500 Dalton. In contrast, there are many molecules with molecular weight greater than 500 Dalton (31.22%) that are BBB- compounds. Nonetheless, aside from the a long tail of heavy BBB- compounds, the distribution of molecular weights for BBB+ and BBB- molecules is not dissimilar (see Figures 5.5 (b) and (f)). 98.8% of BBB+ compounds and 23.4% of BBB- compounds have fewer than 5 hydrogen-bond donors; 97.6% of BBB+ compounds and 66.0% of BBB- compounds have fewer than 10 hydrogen-bond acceptors. This supports the idea that hydrophilic compounds find it difficult to cross the BBB, but this is not a hard-and-fast rule: there are BBB+ compounds that violate Lipinski's rule of 5. Finally, the octanol/water partition coefficient $\log P$ was estimated using `ALOGPS version 2.1` [36]. There is not much difference in the $\log P$ values for BBB+ and BBB- compounds: 93.8% of BBB+ and 95.1% of BBB- compounds have $\log P < 5$. Taken together, the analysis of the selected physiochemical descriptors suggest that no single parameter can determine the BBB-permeability of a compound. This confirms that predicting BBB permeability computationally is challenging, and emphasizes the value of the B3DB dataset.

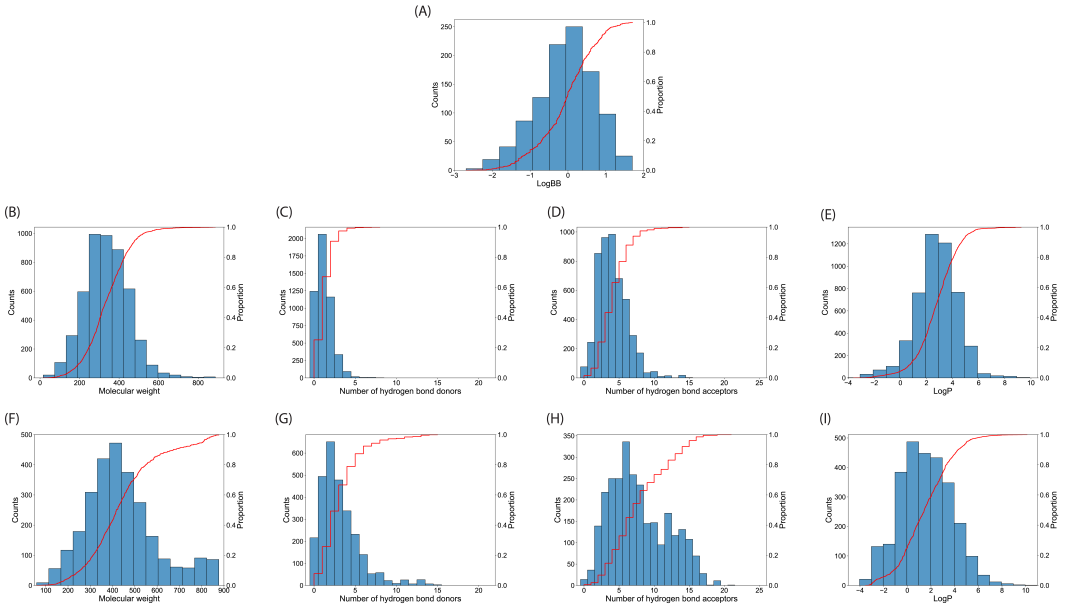


Fig. 5.5 Analysis of the curated datasets. (A). Distribution of $\log BB$ values for numeric dataset. (B)-(E) Distribution of molecular weight, number of hydrogen-bond donors, number of hydrogen acceptors and $\log P$ for BBB+ compounds. (F)-(I) Distribution of molecular weight, number of hydrogen-bond donors, number of hydrogen acceptors and $\log P$ for BBB- compounds.

5.4 Conclusions

The highly-selective blood-brain barrier (BBB) prevents neurotoxic substances in blood from crossing into the extracellular fluid of the central nervous system (CNS). As such, the BBB has a close relationship with CNS disease development and treatment, so predicting whether a substance crosses the BBB is a key task in lead discovery for CNS drugs.

There are two types of BBB data reported in literature and public accessible databases, numerical $\log BB$ defined in Eq. 5.1 and categorical labels to indicate if a compound can pass through BBB (BBB+ and BBB-). Therefore, many predictive models have been proposed for BBB predictions, such as resampling-based model [12], ensemble-based model [14], LightBBB [11] and *et al*. However, these models suffer from generalizability. That's to say the reported models cannot give accurate predictions of unseen molecules, which is restricted by the limited chemical diversity of initial dataset(s). The biggest dataset, as we know, is the one used by LightBBB, which contains only 4491 unique valid molecules with categorical labels and 696 unique valid molecule with $\log BB$ values and after checking with the unique molecule representation, InChI. It is believed that a molecule dataset with broader chemical diversity will help built a robust computational models.

To mitigate this issue, we present a large benchmark dataset, B3DB, complied from 50 published resources and categorized based on experimental uncertainty. A subset of the molecules in B3DB has numerical $\log BB$ values (1058 compounds), while the whole dataset has categorical (BBB+ or BBB-) BBB permeability labels (7807). None of the original data sources contain any quantification of uncertainty (e.g., the standard derivation), so it is recommended to incorporate the group categories when using the datasets. If one decides to use a different threshold to determine BBB+ and BBB- for a molecules, $\log BB$ can be used directly from the data reported in this study. The 1613 2D chemical descriptors, computed with `mordred` can facilitate building predictive models. Any further molecular pre-processing can be done with `RdKit`.

The dataset is freely available at https://github.com/theochem/B3DB and https://doi.org/10.6084/m9.figshare.15634230.v3 (version 3) [35]. We also provide some physicochemical properties of the molecules. By analyzing these properties, we can demonstrate some physiochemical similarities and differences between BBB+ and BBB- compounds. This dataset will facilitate BBB prediction for CNS drug discovery and we will build classification models of BBB+ and BBB- in the next chapter.

Table 5.1 Data source and the available corresponding information.

ID	Data Source Size	Information Available	Data Type	Reference
R1	2053	name, smiles	categorical data	29
R2	1210	name, smiles	categorical data, numerical data	33
R3	328	name, smiles	numerical data	37
R4	189	CAS, name, smiles	numerical data	38
R5	108	name, smiles	numerical data	39
R6	1692	name, smiles	categorical data	40
R7	224	name	categorical data	30
R8	439	smiles, CID	numerical data	26
R9	415	name, smiles	categorical data	41
R10	462	name, CID	categorical data	42
R11	151	name, logBB	numerical data	43
R12	182	name, smiles	numerical data	44
R13	2321	smiles	categorical data	12
R14	942	name, smiles	categorical data	45
R15	390	name	categorical data	46
R16	374	name, CID	categorical data	31
R17	55	name	numerical data	47
R18	332	name, smiles	numerical data	28
R19	1990	name, smiles	categorical data	13
R20	139	name	numerical data	48
R21	362	name, smiles, CID	numerical data	49
R22	27	name	numerical data	50
R23	1090	name, smiles	categorical data	51
R24	1866	smiles	categorical data	52
R25	581	name, smiles	numerical data	27
R26	448	CAS, name, smiles	categorical data, numerical data	53
R27	7236	smiles	categorical data, numerical data	11
R28	415	name, smiles	categorical data	32
R29	181	name	categorical data	54
R30	3620	name, smiles	categorical data	18 *
R31	12	name	numerical data	55
R32	26	name	numerical data	56
R33	26	name	numerical data	57
R34	153	name	numerical data	58
R35	145	smiles	numerical data	59
R36	525	name, smiles	categorical data	60
R37	111	name, smiles	categorical data	61
R38	291	name, smiles	numerical data	62
R39	122	name	numerical data	63
R40	405	name	numerical data	64
R41	296	smiles	numerical data	65
R42	45	smiles	numerical data	9
R43	328	name, smiles	numerical data	66
R44	89	name	numerical data	67
R45	8	smiles	numerical data	68
R46	483	smiles	numerical data	69
R47	529	name	numerical data	70
R48	115	smiles	numerical data	71
R49	181	name, smiles	numerical data	72
R50	113	name, smiles	categorical data, numerical data	73

Table 5.2 List of information in the curated datasets. The BBB+/BBB- and threshold columns are only available for categorical dataset. The 1613 2D chemical descriptors are not listed in this table.

Column Header	**Description**	**Data Type**
compound_name	Generic name of compound	string
IUPAC_name	Name of compound following the IUPAC nomenclature naming scheme	string
SMILES	SMILES representation of compound, isomeric SMILES if available	string
CID	PubChem compound identifier	string
$\log$ BB	$\log BB$ value of compound	float
BBB+/BBB-	Categorical labels to indicate if compound is BBB permeable (BBB+) or not (BBB-)	string
InChI	The IUPAC International Chemical Identifier of compound	string
threshold	Threshold value used to determine BBB permeability label	float
reference	Data sources	string
group	Group classification	string
comment	Complementary information	string

Table 5.3 Occurrences of source $\log BB$ values for different groups in numerical dataset.

Group / Frequency	A	B	C	D
1	243	0	0	0
2	0	32	3	0
3	0	172	0	6
4	0	38	0	7
5	0	71	0	4
6	0	46	0	8
7	0	29	0	5
8	0	19	0	3
9	0	29	0	5
10	0	13	0	4
11	0	17	0	3
12	0	19	0	3
13	0	26	0	4
14	0	58	0	6
15	0	8	0	6
16	0	5	0	0
17	0	10	0	3
18	0	6	0	11
19	0	4	0	7
20	0	10	0	3
21	0	4	0	5
22	0	6	0	3
23	0	7	0	4
24	0	9	0	7
25	0	4	0	5
26	0	7	0	3
27	0	2	0	6
28	0	5	0	5
29	0	3	0	10
30	0	1	0	6
31	0	1	0	3
32	0	1	0	3
33	0	1	0	0
34	0	0	0	0
35	0	0	0	1

Table 5.4 Occurrences of unique source $\log BB$ values for different groups in numerical dataset.

Group / Frequency	A	B	C	D
1	243	652	0	0
2	0	9	3	84
3	0	2	0	29
4	0	0	0	21
5	0	0	0	8
6	0	0	0	7

Table 5.5 Occurrences of source BBB permeability labels for different groups in categorical dataset.

Group / Frequency	A	B	C	D
1	1058	892	2062	0
2	0	618	831	0
3	0	533	162	37
4	0	409	17	13
5	0	181	5	1
6	0	313	0	0
7	0	121	0	0
8	0	55	0	0
9	0	41	0	0
10	0	338	0	0
11	0	26	0	0
12	0	47	0	0
13	0	15	0	0
14	0	8	0	0
15	0	11	0	0
16	0	3	0	0
17	0	2	0	0
18	0	1	0	0
19	0	3	0	0
20	0	1	0	0
21	0	1	0	0
22	0	1	0	0
23	0	1	0	0

Table 5.6 Occurrences of unique source BBB permeability labels for different groups in categorical dataset.

Group / **Frequency**	**A**	**B**	**C**	**D**
1	1058	3621	3077	0
2	0	0	0	51

References

[1] Werner Risau and Hartwig Wolburg. Development of the blood-brain barrier. *Trends in Neurosciences*, 13(5):174–178, 1990.

[2] Caterina P Profaci, Roeben N Munji, Robert S Pulido, and Richard Daneman. The blood–brain barrier in health and disease: Important unanswered questions. *Journal of Experimental Medicine*, 217(4), 2020.

[3] Richard Daneman and Alexandre Prat. The blood–brain barrier. *Cold Spring Harbor Perspectives in Biology*, 7(1):a020412, 2015.

[4] William M Pardridge. Blood–brain barrier delivery. *Drug Discovery Today*, 12 (1-2):54–61, 2007.

[5] William M Pardridge. The blood-brain barrier: bottleneck in brain drug development. *NeuroRx*, 2(1):3–14, 2005.

[6] Seetha Harilal, Jobin Jose, Rajesh Kumar, Mazhuvancherry Kesavan Unnikrishnan, Md Sahab Uddin, Githa Elizabeth Mathew, Rahul Pratap, Akash Marathakam, Bijo Mathew, et al. Revisiting the blood-brain barrier: A hard nut to crack in the transportation of drug molecules. *Brain Research Bulletin*, 2020.

[7] Szilvia Veszelka, Ágnes Kittel, and Mária A Deli. Tools of modelling blood–brain barrier penetrability. *Solubility, Delivery and ADME Problems of Drugs and Drug-Candidates, Bentham Science Publishers, Washington*, pages 166–188, 2011.

[8] Marta Glavatskikh, Jules Leguy, Gilles Hunault, Thomas Cauchy, and Benoit Da Mota. Dataset's chemical diversity limits the generalizability of machine learning predictions. *Journal of Cheminformatics*, 11(1):69, 2019.

[9] Krzesimir Ciura, Szymon Ulenberg, Hanna Kapica, Piotr Kawczak, Mariusz Belka, and Tomasz Bączek. Assessment of blood–brain barrier permeability using micellar electrokinetic chromatography and p_vsa-like descriptors. *Microchemical Journal*, 158:105236, 2020.

[10] Jan Kelder, Peter DJ Grootenhuis, Denis M Bayada, Leon PC Delbressine, and Jan-Peter Ploemen. Polar molecular surface as a dominating determinant for oral absorption and brain penetration of drugs. *Pharmaceutical Research*, 16 (10):1514–1519, 1999.

[11] Bilal Shaker, Myeong-Sang Yu, Jin Sook Song, Sunjoo Ahn, Jae Yong Ryu, Kwang-Seok Oh, and Dokyun Na. Lightbbb: computational prediction model of blood–brain-barrier penetration based on lightgbm. *Bioinformatics*, 2020.

[12] Zhuang Wang, Hongbin Yang, Zengrui Wu, Tianduanyi Wang, Weihua Li, Yun Tang, and Guixia Liu. In silico prediction of blood-brain barrier permeability of compounds by machine learning and resampling methods. *ChemMedChem*, 13(20):2189–2201, 2018.

[13] Yuan H Zhao, Michael H Abraham, Adam Ibrahim, Paul V Fish, Susan Cole, Mark L Lewis, Marcel J de Groot, and Derek P Reynolds. Predicting penetration across the blood-brain barrier from simple descriptors and fragmentation schemes. *Journal of Chemical Information and Modeling*, 47(1):170–175, 2007.

[14] Lili Liu, Li Zhang, Huawei Feng, Shimeng Li, Miao Liu, Jian Zhao, and Hongsheng Liu. Prediction of the blood–brain barrier (bbb) permeability of chemicals based on machine-learning and ensemble methods. *Chemical Research in Toxicology*, 2021.

[15] Kai Lun Chang, Hai Ning Pee, Shili Yang, and Paul C Ho. Influence of drug transporters and stereoselectivity on the brain penetration of pioglitazone as a potential medicine against alzheimer's disease. *Scientific Reports*, 5(1):1–7, 2015.

[16] Clifford W Fong. Permeability of the blood–brain barrier: molecular mechanism of transport of drugs and physiologically important compounds. *The Journal of Membrane Biology*, 248(4):651–669, 2015.

[17] Hirotomo Moriwaki, Yu-Shi Tian, Norihito Kawashita, and Tatsuya Takagi. Mordred: a molecular descriptor calculator. *Journal of Cheminformatics*, 10 (1):1–14, 2018.

[18] Kexin Huang, Tianfan Fu, Wenhao Gao, Yue Zhao, Yusuf Roohani, Jure Leskovec, Connor Coley, Cao Xiao, Jimeng Sun, and Marinka Zitnik. Therapeutics data commons: Machine learning datasets for therapeutics. https://tdcommons.ai, November 2020.

[19] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[20] Aki Ariga. tabula-py, September 2020. URL https://pypi.org/project/tabula-py/2.2.0/.

[21] Matt Swain. Pubchempy, Apr 2017. URL https://github.com/mcs07/PubChemPy.

[22] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, et al. Pubchem in 2021: new data content and improved web interfaces. *Nucleic Acids Research*, 49(D1):D1388–D1395, 2021.

[23] Greg Landrum. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling, 2013.

[24] Sunghwan Kim, Paul A Thiessen, Tiejun Cheng, Bo Yu, and Evan E Bolton. An update on pug-rest: Restful interface for programmatic access to pubchem. *Nucleic Acids Research*, 46(W1):W563–W570, 2018.

[25] A Patrícia Bento, Anne Hersey, Eloy Félix, Greg Landrum, Anna Gaulton, Francis Atkinson, Louisa J Bellis, Marleen De Veij, and Andrew R Leach. An open source chemical structure curation pipeline using rdkit. *Journal of Cheminformatics*, 12(1):1–16, 2020.

[26] Wenyi Wang, Marlene T Kim, Alexander Sedykh, and Hao Zhu. Developing enhanced blood–brain barrier permeability models: integrating external bioassay data in qsar modeling. *Pharmaceutical Research*, 32(9):3055–3065, 2015.

[27] Yoan Brito-Sánchez, Yovani Marrero-Ponce, Stephen J Barigye, Iván Yaber-Goenaga, Carlos Morell Perez, Huong Le-Thi-Thu, and Artem Cherkasov. Towards better bbb passage prediction using an extensive and curated data set. *Molecular Informatics*, 34(5):308–330, 2015.

[28] Fabien Plisson and Andrew M Piggott. Predicting blood–brain barrier permeability of marine-derived kinase inhibitors using ensemble classifiers reveals potential hits for neurodegenerative disorders. *Marine Drugs*, 17(2):81, 2019.

[29] Ines Filipa Martins, Ana L Teixeira, Luis Pinheiro, and Andre O Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of Chemical Information and Modeling*, 52(6):1686–1697, 2012.

[30] Claudia Andres and Michael C Hutter. Cns permeability of drugs predicted by a decision tree. *QSAR & Combinatorial Science*, 25(4):305–309, 2006.

[31] Zhen Gao, Yang Chen, Xiaoshu Cai, and Rong Xu. Predict drug permeability to blood–brain-barrier from clinical phenotypes: drug side effects and drug indications. *Bioinformatics*, 33(6):901–908, 2017.

[32] Hu Li, Chun Wei Yap, Choong Yong Ung, Ying Xue, Zhi Wei Cao, and Yu Zong Chen. Effect of selection of molecular descriptors on the prediction of blood- brain barrier penetrating and nonpenetrating agents by statistical learning methods. *Journal of Chemical Information and Modeling*, 45(5):1376–1384, 2005.

[33] Manvi Singh, Reshmi Divakaran, Leela Sarath Kumar Konda, and Rajendra Kristam. A classification model for blood brain barrier penetration. *Journal of Molecular Graphics and Modelling*, 96:107516, 2020.

[34] ToolKit OEChem. Openeye scientific software. *Inc., Santa Fe, NM, USA*, 2020.

[35] Fanwang Meng, Xi Yang, Jinfeng Huang, and Paul W. Ayers. B3db: A curated diverse molecular database of blood-brain barrier permeability with chemical descriptors. *figshare*, 8 2021. doi: https://doi.org/10.6084/m9.figshare.15634230.v3. URL https://figshare.com/articles/dataset/A_large_benchmark_dataset_Blood-Brain_Barrier_Database_B3DB_complied_from_50_published_resources_/15634230.

[36] Igor V Tetko and Vsevolod Yu Tanchuk. Application of associative neural networks for prediction of lipophilicity in alogps 2.1 program. *Journal of Chemical Information and Computer Sciences*, 42(5):1136–1145, 2002.

[37] Michael H Abraham, Adam Ibrahim, Yuan Zhao, and William E Acree Jr. A data base for partition of volatile organic compounds and drugs from blood/plasma/serum to brain, and an lfer analysis of the data. *Journal of Pharmaceutical Sciences*, 95(10):2091–2100, 2006.

[38] SR Mente and Franco Lombardo. A recursive-partitioning model for blood–brain barrier permeation. *Journal of Computer-Aided Molecular Design*, 19(7): 465–481, 2005.

[39] Angela Guerra, Juan A Páez, and Nuria E Campillo. Artificial neural networks in admet modeling: prediction of blood–brain barrier permeation. *QSAR & Combinatorial Science*, 27(5):586–594, 2008.

[40] Marc Adenot and Roger Lahana. Blood-brain barrier permeation models: discriminating between potential cns and non-cns drugs including p-glycoprotein substrates. *Journal of Chemical Information and Computer Sciences*, 44(1): 239–248, 2004.

[41] Subhabrata Majumdar, Subhash C Basak, Claudiu N Lungu, Mircea V Diudea, and Gregory D Grunwald. Finding needles in a haystack: determining key molecular descriptors associated with the blood-brain barrier entry of chemical compounds using machine learning. *Molecular Informatics*, 38(8-9):1800164, 2019.

[42] Rui Miao, Liang-Yong Xia, Hao-Heng Chen, Hai-Hui Huang, and Yong Liang. Improved classification of blood-brain-barrier drugs using deep learning. *Scientific Reports*, 9(1):1–11, 2019.

[43] Jie Shen, Yiping Du, Yaxue Zhao, Guixia Liu, and Yun Tang. In silico prediction of blood–brain partitioning using a chemometric method called genetic algorithm based variable selection. *QSAR & Combinatorial Science*, 27(6):704–717, 2008.

[44] Prabha Garg and Jitender Verma. In silico prediction of blood brain barrier permeability: an artificial neural network model. *Journal of Chemical Information and Modeling*, 46(1):289–297, 2006.

[45] Arup K Ghose, Torsten Herbertz, Robert L Hudkins, Bruce D Dorsey, and John P Mallamo. Knowledge-based, central nervous system (cns) lead selection and lead optimization for cns drug discovery. *ACS Chemical Neuroscience*, 3 (1):50–68, 2012.

[46] Sandhya Kortagere, Dmitriy Chekmarev, William J Welsh, and Sean Ekins. New predictive models for blood–brain barrier permeability of drug-like molecules. *Pharmaceutical Research*, 25(8):1836–1845, 2008.

[47] Xu-Chun Fu, Guo-Ping Wang, Hai-Li Shan, Wen-Quan Liang, and Jian-Qing Gao. Predicting blood–brain barrier penetration from molecular weight and number of polar atoms. *European Journal of Pharmaceutics and Biopharmaceutics*, 70(2):462–466, 2008.

[48] Kiril Lanevskij, Justas Dapkunas, Liutauras Juska, Pranas Japertas, and Remigijus Didziapetris. Qsar analysis of blood–brain distribution: The influence of plasma and brain tissue binding. *Journal of Pharmaceutical Sciences*, 100(6): 2147–2160, 2011.

[49] Markus Muehlbacher, Gudrun M Spitzer, Klaus R Liedl, and Johannes Kornhuber. Qualitative prediction of blood–brain barrier permeability on a large and refined dataset. *Journal of Computer-Aided Molecular Design*, 25(12): 1095–1106, 2011.

[50] David E Clark. Rapid calculation of polar molecular surface area and its application to the prediction of transport phenomena. 1. prediction of intestinal absorption. *Journal of Pharmaceutical Sciences*, 88(8):807–814, 1999.

[51] Mayuri Gupta, Hyeok Jun Lee, Christopher J Barden, and Donald F Weaver. The blood–brain barrier (bbb) score. *J. Med. Chem.*, 62(21):9824–9836, 2019.

[52] Dipankar Roy, Vijaya Kumar Hinge, and Andriy Kovalenko. To pass or not to pass: predicting the blood–brain barrier permeability with the 3d-rism-kh molecular solvation theory. *ACS Omega*, 4(16):16774–16780, 2019.

[53] Laura K Chico, Linda J Van Eldik, and D Martin Watterson. Targeting protein kinases in central nervous system disorders. *Nature Reviews Drug Discovery*, 8 (11):892–909, 2009.

[54] Govindan Subramanian and Douglas B Kitchen. Computational models to predict blood–brain barrier permeation and cns activity. *Journal of Computer-Aided Molecular Design*, 17(10):643–664, 2003.

[55] Timothy S Carpenter, Daniel A Kirshner, Edmond Y Lau, Sergio E Wong, Jerome P Nilmeier, and Felice C Lightstone. A method to predict blood-brain barrier permeability of drug-like compounds using molecular dynamics simulations. *Biophysical Journal*, 107(3):630–641, 2014.

[56] Franco Lombardo, James F Blake, and William J Curatolo. Computation of brain- blood partitioning of organic solutes via free energy calculations. *Journal of Medicinal Chemistry*, 39(24):4750–4755, 1996.

[57] Ulf Norinder, Per Sjöberg, and Thomas Österberg. Theoretical calculation and prediction of brain–blood partitioning of organic solutes using molsurf parametrization and pls statistics. *Journal of Pharmaceutical Sciences*, 87(8): 952–959, 1998.

[58] Fabio Broccatelli, Caroline A Larregieu, Gabriele Cruciani, Tudor I Oprea, and Leslie Z Benet. Improving the prediction of the brain disposition for orally administered drugs using bddcs. *Advanced Drug Delivery Reviews*, 64(1): 95–109, 2012.

[59] Yan Chen, Qing-Jun Zhu, Jian Pan, Yi Yang, and Xiao-Pei Wu. A prediction model for blood–brain barrier permeation and analysis on its parameter biologically. *Computer Methods and Programs in Biomedicine*, 95(3):280–287, 2009.

[60] Liying Zhang, Hao Zhu, Tudor I Oprea, Alexander Golbraikh, and Alexander Tropsha. Qsar modeling of the blood–brain barrier permeability for diverse organic compounds. *Pharmaceutical Research*, 25(8):1902–1914, 2008.

[61] Hongming Chen, Susanne Winiwarter, Markus Fridén, Madeleine Antonsson, and Ola Engkvist. In silico prediction of unbound brain-to-plasma concentration ratio using machine learning algorithms. *Journal of Molecular Graphics and Modelling*, 29(8):985–995, 2011.

[62] Dmitry A Konovalov, Danny Coomans, Eric Deconinck, and Yvan Vander Heyden. Benchmarking of qsar models for blood-brain barrier permeation. *Journal of Chemical Information and Modeling*, 47(4):1648–1656, 2007.

[63] Ali Shayanfar, Somaieh Soltani, and Abolghasem Jouyban. Prediction of blood–brain distribution: effect of ionization. *Biological and Pharmaceutical Bulletin*, 34(2):266–271, 2011.

[64] Santiago Vilar, Mayukh Chakrabarti, and Stefano Costanzi. Prediction of passive blood–brain partitioning: straightforward and effective classification models based on in silico derived physicochemical descriptors. *Journal of Molecular Graphics and Modelling*, 28(8):899–903, 2010.

[65] Andrey A Toropov, Alla P Toropova, Marten Beeg, Marco Gobbi, and Mario Salmona. Qsar model for blood-brain barrier permeation. *Journal of Pharmacological and Toxicological Methods*, 88:7–18, 2017.

[66] Maria Dichiara, Benedetto Amata, Rita Turnaturi, Agostino Marrazzo, and Emanuele Amata. Tuning properties for blood–brain barrier permeation: A statistics-based analysis. *ACS Chemical Neuroscience*, 11(1):34–44, 2019.

[67] Renata Bujak, Wiktoria Struck-Lewicka, Michał Kaliszan, Roman Kaliszan, and Michał J Markuszewski. Blood–brain barrier permeability mechanisms in view of quantitative structure–activity relationships (qsar). *Journal of Pharmaceutical and Biomedical Analysis*, 108:29–37, 2015.

[68] Bahram Hemmateenejad, Ramin Miri, Mohammad A Safarpour, and Ahmad R Mehdipour. Accurate prediction of the blood–brain partitioning of a large set

of solutes using ab initio calculations and genetic neural network modeling. *Journal of Computational Chemistry*, 27(11):1125–1135, 2006.

[69] Catrin Yojana Mendoza Valencia. Chemical composition of doc, 25b-nbome, 25c-nbome and in silico modeling of permeability to the blood-brain barrier (bbb). Master's thesis, Universidad Nacional de Colombia, Colombia, 2017.

[70] Eugene V Radchenko, Alina S Dyabina, and Vladimir A Palyulin. Towards deep neural network models for the prediction of the blood–brain barrier permeability for diverse organic compounds. *Molecules*, 25(24):5901, 2020.

[71] TJ Hou and XJ Xu. Adme evaluation in drug discovery. 3. modeling blood-brain barrier partitioning using simple molecular descriptors. *Journal of Chemical Information and Computer Sciences*, 43(6):2137–2152, 2003.

[72] Ulf Norinder and Markus Haeberlein. Computational approaches to the prediction of the blood–brain distribution. *Advanced Drug Delivery Reviews*, 54(3):291–313, 2002.

[73] Anna W Sobańska, Adam Hekner, and Elżbieta Brzezińska. Rp-18 hplc analysis of drugs' ability to cross the blood-brain barrier. *Journal of Chemistry*, 2019, 2019.

Chapter 6

Blood-Brain Barrier Permeability Predictions of Organic Molecules with XGBoost and Resampling Strategies

6.1 Introduction

The blood-brain barrier (BBB) protects and regulates the microvasculature of the central nervous system (CNS). Specifically, the BBB maintains homeostasis in the CNS [1, 2] and protects the CNS by inhibiting the passage of toxins and pathogens from the blood [3]. However, because of its resistance to exogenous compounds, the BBB also poses a challenge for the delivery of neuroactive molecules (i.e. drugs) into the CNS. It is estimated that 98% of drugs [4] and approximately 100% of bio-molecular pharmaceuticals (such as peptides and monoclonal antibodies) fail to penetrate the BBB. BBB permeability must also be considered when developing chimeric antigen receptor–modified T (CAR-T) cell-based therapy for brain tumors [5, 6]. Understanding small molecules' BBB permeability is therefore not only vital for CNS drug discovery, but must be considered at an early stage in the drug-development pipeline to avoid costly failures in (late stage) drug formulation studies.

To address this obstacle, various experimental approaches have been proposed for measuring molecules' BBB penetration *in vivo*, including biopartitioning micellar chromatography [7], the zebrafish animal model [8], Caco-2 cell monolayers [9–11], the iPSC-derived cell model [12, 13], the microfluidic multi-channel BBB chip [14],

an artificial membrane permeation assay (PAMPA) [15], nanoparticle enhanced delivery [16, 17], and many more [18, 19]. These methods mainly measure two types of BBB permeability data (1) logarithmic ratio of the molecular concentration in brain phase and blood phase in the steady-state (denoted as $\log BB$) [20] and (2) the permeability surface-area product (denoted as $\log PS$) [21]. As these experimental methods are expensive, time consuming, labour intensive and low-throughput, computational prediction of BBB permeability for small molecules is an attractive (but challenging) problem in CNS drug discovery and development [4, 22–24].

Various computational approaches for BBB permeability predictions have been proposed. The most fundamentally sound approaches use molecular dynamics (MD) simulations to directly simulate the solubility and/or transport of molecules across the BBB. An early MD study correlated the computed solvation free energy in water with $\log BB$ [25]. More recent studies have employed steered MD [26], enhanced sampling techniques [21], and unbiased MD [27], to provide a detailed characterization of physical interactions (e.g., hydrogen bonding, electrostatic, and Van der Waals interactions) between molecules and membranes and obtain atomic-level insights into molecules' (in)ability to cross the BBB.

However, MD simulations require substantial computational resources, along with an experienced scientist to set up and analyze the calculations. This motivates the increased popularity of machine learning (ML) approaches. ML methods seem feasible because it is believed that BBB permeability is largely determined by molecules' physicochemical properties (topological polar surface area (tPSA), number of hydrogen bond donors and acceptors, *etc.* pK_a [28, 29]). This insight directly motivates quantitative structure activity relationship (QSAR) models for BBB permeability [30–34]. ML methods are the natural extension of QSAR to include "synthetic" structural properties (e.g. molecular fingerprints) and more sophisticated mathematical models. There are two types of ML models for the permeability predictions of BBB (1) regression for $\log BB$ or $\log PS$ and (2) classification of molecules as BBB permeable (BBB+) or BBB impermeable (BBB-).

Probably because of the small size publically accessible datasets, there are relatively few ML models for regression of $\log BB$ values; studies include multiple linear regression (MLR) [20], neural networks [35–37], and support vector machine (SVM) [38]. By contrast, the number and diversity of ML models for the classification of molecular BBB permeability is substantial, including random forests (RF) [39, 40], classification and regression trees (CART) [41], binomial partial least squares (binomial-PLS) [42], decision tree induction (DTI) [43], SVM [44], generalized linear models (GLM) [45], recurrent neural network (RNN) [46] *etc.* [47]. Of special

relevance to this work is the LightBBB model, which uses the light gradient boosting machine (LightGBM) trained on 7162 molecular instances (only 4491 unique valid molecules after checking International Chemical Identifier (InChI)) with `RDKit`), achieving sensitivity of 0.93, specificity of 0.77, accuracy of 0.89, and area under the curve (AUC) of 0.93 [48]. To our knowledge, this is the best model using the largest dataset, but one should note the training set is somewhat redundant. (E.g., there are only 4491 unique valid molecules that can be recognized by `RDKit` package when checking for repeated International Chemical Identifiers (InChI).)

While there is more data for BBB classification than regression, model generalizability to unseen data is still limited because the chemical space covered by the available data is insufficient to build a good decision boundary [49], so models are prone to overfitting. This supports the finding that smaller datasets (1593 molecules) tend to give better evaluation metrics than bigger datasets (1990) using SVM for BBB classification [44].

Model performance is also compromised by the imbalance in the training molecule dataset. The BBB will prevent most of chemicals from entering the brain phase, suggesting that most molecules are not BBB permeable. However BBB+ molecules are the majority class in most available datasets. Given the difficulty of experimental measurements, it is infeasible to fix the data imbalance problem by high-throughput generation of additional minority class (BBB-) instances. This motivates resampling methods for BBB classification [50]. Zhuang *et al.* built a model by fusing SVM and synthetic minority over-sampling technique (SMOTE) with specificity 0.833 using 2358 molecules [51]. Using the same dataset, a recurrent neural network (RNN) was proposed in combination with SMOTE, which achieved high performance score for training data, but no training/testing protocol was utilized [46]. More recently, a computational model integrating SMOTE and extreme gradient boosting (XGBoost) [52] was reported for the same dataset with train/test split ratio 0.75:0.25 [53]. They achieved good precision and recall (sensitivity), with an F_1 score 0.91. But they did not report information about specificity, a measurement of true negative ratio. Another similar study used the same dataset by applying an oversampling strategy (SMOTE) with classification model built with a feed-forward artificial neural network (FFDNN) to the output of a simpler model (using SMOTE and kernel PCA (KPCA)), achieving overall accuracy of 97.11%, specificity of 98.42%, and sensitivity of 97.35% on the testing set [36].

In this work, we mitigate the problem of restricted generalizability caused by small datasets and their imbalanced labels by building predictive models for BBB permeability with ML and resampling strategies. The new dataset was curated

from 50 literature or public accessible resources, as discussed in [54]. We will first review how the molecular dataset was constructed, followed by feature generation and selection. Predictive models are constructed using multiple resampling strategies and several different ML classification algorithms Figure 6.1, then characterized and discussed.

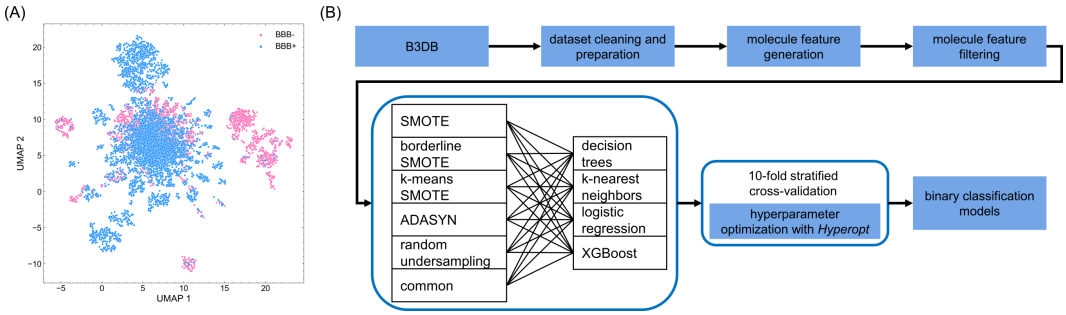


Fig. 6.1 Chemical diversity and computational framework for BBB prediction. (A). Chemical diversity of B3DB dataset with non-linear dimension reduction method, UMAP [55]. (B). General pipeline of constructing classification models for BBB penetration (B). Decision tree, k-nearest neighbour, logistic regression and XGBoost are combined with SMOTE and its variants (k-means SMOTE, borderline SMOT), ADAYSN, random undersampling as the classifiers. Computational models also include those without resampling strategies. The linking lines denotes combinations of classification algorithms and resampling strategies.

6.2 Methods and Materials

6.2.1 Dataset Preparation

The dataset used in this study is the B3DB database from [54]. B3DB contains 4956 BBB+ and 2851 BBB- molecules (7807 total). This formulates our problem as an imbalanced classification problem with imbalance ratio (IR) 1.74 as defined as $IR = \frac{N_{major}}{N_{minor}}$ [56]. Filtering out charged molecules and molecules for which `RDKit` could not generate a suitable 3D structure reduced this dataset to 4855 BBB+ and 2552 BBB- molecules; see Figure 6.2 (A).

To assess the molecular diversity of B3DB, nonlinear manifold projection (UMAP) was used. The difficulty of BBB classification is clear from the extensive overlap between the clouds of BBB+ and BBB- data, cf. Figure 6.1 (A).

6.2.2 External Dataset Curation

The external dataset was curated from various sources: (1) 18 drug molecules (BBB+) from [57]; (2) 531 drug CNS drugs (BBB+) from *DrugBank* using WHO Anatomical Therapeutic Chemical (ATC) Classification ∗; (3) Topotecan [58]; (4) 24 first generation H1 anti-histaminergic drugs that can easily pass BBB; and 3 second generation H1 anti-histaminergic drugs that are more selective and do not readily pass the BBB (Bepotastine, Quifenadine and Rupatadine) [59, 60]; (5) The removed 30 molecules were used here where 5 monoatomic molecules were not considered.

Only those molecules which were not already present in B3DB were kept, resulting in 216 BBB+ molecules and 8 BBB- molecules. The SMILES strings of the external dataset are listed in Table D.3. Geometry optimization and feature generation procedures for the external dataset was consistent with B3DB.

6.2.3 3D Coordinate Generation and Geometry Optimization

When the PubChem CID is a available for a molecule, the 3D coordinates are downloaded from `PubChem` web server using `PubChemPy` [61] where available; otherwise the 2D coordinates are fetched. For database entries without a valid CID, the isomeric Simplified Molecular Input Line Entry System (SMILES) was used to generate 3D coordinates with `OpenBabel` [62]. Hydrogen atoms are added if necessary, and then the geometries of molecules are optimized with the MMFF94s force field as implemented in `OpenBabel` [62] using default parameters, except that 10,000 iterations were allowed. Molecules for which no satisfactory 3D coordinates could be generated or for which geometry optimization failed were eliminated, leaving 7447 molecules.

6.2.4 Molecule Feature Generation

Chemical descriptors were chosen to encode the molecules in the dataset not only because of its computational tractability but also motivated by recent studies indicating that such descriptors are preferable to graph neural networks [63]. We selected `PaDEL` to compute the 1875 descriptors, including 1D, 2D and 3D descriptors. Five molecules were removed because they were incompatible with `PaDEL`: H_2O, CH_4, HN_2O, N_2H_4 and H_4NO_2. There are 1875 descriptors (including 431 3D descriptors) for each molecule.

∗Downloaded from https://pubchem.ncbi.nlm.nih.gov/classification/#hid=79

6.2.5 Molecule Feature Selection

To build an robust model, one needs to perform feature selection (also known as variable elimination) [64–66]. A simple pipeline was used for feature selection, as shown in Figure 6.2 (B). Similar to previous classifiers for BBB permeability, we filter features based on their numerical and statistical properties [46, 48, 51, 53]. Specifically, features with infinite values ($-\infty$ and ∞), NaN values, or extremely large magnitude ($>10^5$) are removed from the feature matrix. We dropped constant features and removed correlated (and duplicate) features using 0.8 Pearson correlation value as the threshold; this left 475 features. Eliminating linearly correlated features helps remove redundant information from the feature matrix, Figure 6.2 (C) and (D); the performance advantages are not critical in this study, but will be advantageous when our model is used for large-scale database screening.

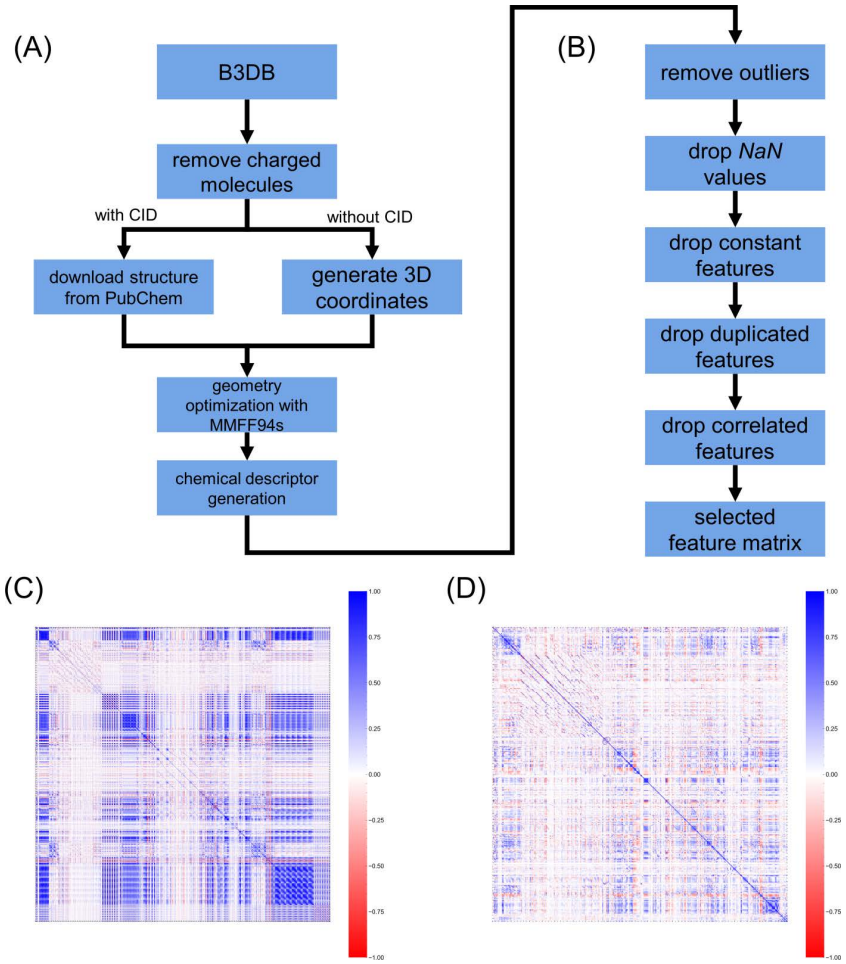


Fig. 6.2 Summary of dataset preparation, feature generation and feature filtering.

6.2.6 Model Construction and Selection

The general workflow of building classification models is shown in Figure 6.1 (B), including dataset pre-processing, feature engineering (feature generation and selection), stratified 10-fold cross-validation, and hyperparameter optimization. Because our dataset is imbalanced, we combined 4 basic classification algorithms (decision trees, k-nearest neighbours, logistic regression, and XGBoost) with 4 oversampling methods (SMOTE, borderline SMOTE and k-means SMOTE, and adaptive synthetic (ADASYN)), 1 undersampling approach (random undersampling). As a control, we also considered the performance of the 4 algorithms without any resampling strategies, denoted as *common* in Figure 6.1. Our figures and tables are labelled accordingly. For example, the label "xgb-borderline_SMOTE" refers to a model that uses XBGoost as the basic classifier using data oversampled with borderline_SMOTE.

The selected classification algorithms (decision trees, k-nearest neighbours, logistic regression and XGBoost) were integrated with different sampling strategies: SMOTE, k-means SMOTE, borderline SMOTE, adaptive synthetic (ADASYN), random undersampling, and the *common* (no sampling strategy) approach. There are thus $4 \times 6 = 24$ classification models considered; cf. Figure 6.1.

For each model, 10-fold cross-validation, stratified to keep the proportions of BBB+ and BBB- molecules fixed, was employed for model selection. 90% of the data was used for training. The remaining 10% was further split equally, in a stratified manner, into testing and validation subsets. Parameters were optimized using the training data, then validation data was was used to optimize hyperparameters, and testing data was used to ensure that the model was extensible to external data; see subsection 6.2.7. Each instance of 10-fold cross-validation thereby generates a set of hyperparameters. To identify the optimal hyperparameters (re)considered the original 10-fold stratified cross-validation, this time evaluating the error for each of the 10 sets of hyperparameters, using 90% of the data for training and the remaining 10% to assess the error. We then chose the hyperparameter set with the lowest error for further analysis Figure 6.3.

6.2.7 Hyperparameter Optimization

The hyperparameters were optimized with `Hyperopt` [67] using the tree of Parzen estimators (TPE) [68, 69] algorithm, which is a Sequential Model-Based Global Optimization (SMBO) algorithm.

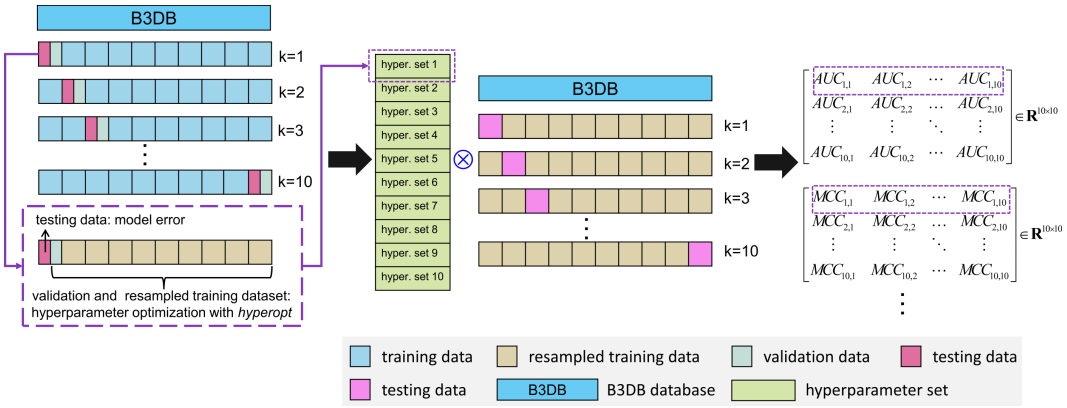


Fig. 6.3 Cross-validation for BBB predictions. The *B3DB* dataset was split into training, testing and validation dataset with ratio of 90%:5%:5% respectively following 10-fold-like data splitting scheme. For each fold, the training dataset was processed with resampling strategies (under-resampling or orver-resampling) and hyperparameters were optimized using testing dataset and resampled training dataset, providing a set of hyperparameter and model error evaluated on validation dataset. We are returned with 10 sets of different hyperparameters which was further evaluated with 10-fold data splitting where training data was used to fit the model and testing data was used to compute the errors (ROC_AUC, precision, recall, F_1 score *et al*) with predefined parameters.

6.2.8 Performance Evaluations of Classifiers

We assess the models' performance using popular assessment metrics. These metrics can be divided into threshold metrics (e.g., accuracy), ranking metrics (e.g., AUC), and probabilistic metrics. We will mainly focus on threshold metrics because they are appropriate for imbalanced classification problems, where sensitivity (also known as recall, hit rate, or true positive rate), specificity (also known as selectivity or true negative rate), precision (also known as positive predictive value), accuracy, F_1 score, Matthews correlation coefficient (MCC), geometric mean score (GEOM), and balanced accuracy score (BACC) were selected. These evaluation metrics are defined in subsection 6.2.8 based on confusion matrix elements, true positive (TP), true negative (TN), false positive (FP) and false negative (FN) [70, 71].

$$sensitivity = \frac{TP}{TP + FN} \tag{6.1}$$

$$specificity = \frac{TN}{FP + TN} \tag{6.2}$$

$$precision = \frac{TP}{TP + FP} \tag{6.3}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.4}$$

$$F_1 = 2 \cdot \frac{precision \cdot sensitivity}{precision + sensitivity} \tag{6.5}$$

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP) \cdot (TP + FN) \cdot (TN + FP) \cdot (TN + FN)}} \tag{6.6}$$

$$GEOM = \sqrt{sensitivity \cdot specificity} \tag{6.7}$$

$$BACC = \frac{sensitivity + specificity}{2} \tag{6.8}$$

The ROC_AUC is chosen to explore the relationship of true positives and false positives with different probability thresholds as a ranking method. The shape and the area under the curve (AUC) of ROC provides valuable information to understand classifiers' performance. Random classifiers give an averaged AUC of 0.5 and AUC can be used to rank the models. The precision-recall curve demonstrates the relationship between $precision$ and $recall$ and is more informative than ROC especially for binary imbalanced problems [72]. The average precision (AP) score is a commonly used ranking metric for information retrieval [73],

$$\text{AP} = \sum_n (R_n - R_{n-1}) P_n \tag{6.9}$$

where R_n and P_n are the precision and recall at the n-th threshold. Higher values of AP indicate better models.

6.2.9 Benchmarking *LightBBB* with External Data

The predictions of *LightBBB* on the curated dataset was performed using its Python API. All the evaluation metrics were computed except for ROC_AUC and AP, which we could not compute because the *LightBBB* software does not report prediction probabilities.

6.2.10 Software and Package

All the calculations were done using `Python` 3.7.9 on Compute Canada. Packages used in this study include `Scikit-Learn` 0.24.0, `pandas` 1.3.0, `numpy` 1.19.5, `scipy` 1.7.1, `hyperopt` 0.2.5, `xgboost` 1.3.0, `imbalanced-learn` 0.8.1

and `feature-engine` 1.1.2. Plots were generated with `seaborn` 0.10.0 and `mathplotlib` 3.4.3. Molecular descriptors wre obtained from `openbabel` 3.1.1, `rdkit` 2021.03.3, `PubChemPy` 1.0.4, and `PaDEL` 2.21 were used.

6.3 Results and Discussions

6.3.1 General Model Performance

The performance of all 24 classifiers is reported in (Figure D.1-Figure D.4). Reassuringly, all ten sets of hyperparameters give similar results, even though they were computed using different validation data. Results from different data samples are consistent (with small standard deviations; cf. Table D.1), supporting the robustness and generalizability of our models. The statistical variance of the models, computed by cross-validation, indicates an order-of-preference: XGBoost > kNN > logistical regression > decision trees (Figure 6.5 and Figure D.5), which suggests that XGBoost based models are the top models for BBB predictions. Moreover, the ROC and precision-recall curves of the top models in each sub-category shows the predictive power of our models, Figure 6.6. Generally speaking, the order of the predictive models are XGBoost > kNN > logistical regression > decision trees, cf. Figure D.5.

The xgb-classic_ADASYN, xgb-borderline_SMOTE and xgb-classic_SMOTE are the top 3 models after examining all the performance scores, especially the AUC_ROC and AP (see Figure 6.4 and Figure D.5). The xgb-borderline_SMOTE gives sensitivity of 0.9259, specificity of 0.8237, precision of 0.9091, accuracy of 0.8906, F_1 score of 0.9173, MCC of 0.7564, GEOM of 0.8732, BACC of 0.8748, AP of 0.9770, Table D.1. The averaged performance metrics of xgb-borderline_SMOTE and xgb-classic_ADASYN are almost identical and the only difference is that xgb-classic_ADASYN is more robust with slightly smaller variation with respect of different 10-fold data splitting (the last column in Figure D.6). This makes the ADASYN sampling strategy a superior choice for XGBoost. The sensitivity of xgb-keamns_SMOTE and raw XGBoost are slightly better than that of xgb-classic_ADASYN, xgb-borderline_SMOTE and xgb-classic_SMOTE, which is caused by the higher rate of false negative over true positive of xgb-keamns_SMOTE and raw XGBoost models, Figure 6.4 (B).

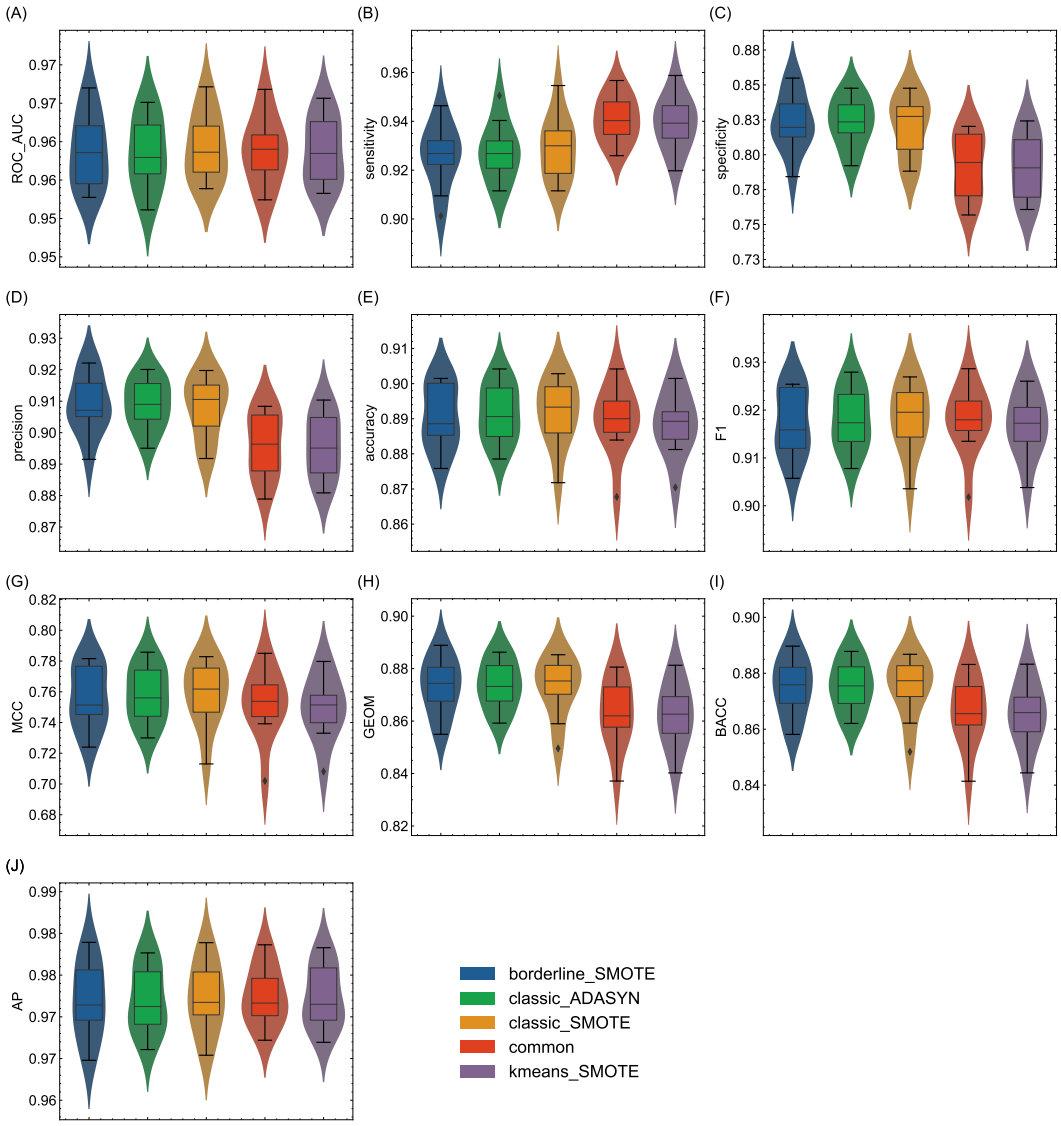


Fig. 6.4 Model performances for top XGBoost models with oversampling strategies including the raw form of XGBoost (denoted as *common*).

Table 6.1 Performance summary of selected top 5 optimal models.

model_name	ROC_AUC	sensitivity	specificity	precision	accuracy	F_1	MCC	GEOM	BACC	AP
dtree-kmeans_SMOTE	0.9000±0.0094	0.8814±0.0158	0.7472±0.0289	0.8692±0.0120	0.8352±0.0101	0.8751±0.0079	0.6332±0.0227	0.8113±0.0136	0.8143±0.0124	0.9300±0.0100
knn-classic_SMOTE	0.9436±0.0062	0.9067±0.0098	0.8248±0.0237	0.9079±0.0114	0.8785±0.0112	0.9073±0.0084	0.7312±0.0254	0.8647±0.0141	0.8658±0.0136	0.9649±0.0063
logreg-classic_SMOTE	0.9190±0.0120	0.8622±0.0192	0.8225±0.0315	0.9026±0.0163	0.8485±0.0168	0.8818±0.0135	0.6730±0.0363	0.8419±0.0187	0.8424±0.0186	0.9491±0.0105
xgb-classic_ADASYN	0.9585±0.0048	0.9275±0.0121	0.8233±0.0180	0.9090±0.0080	0.8916±0.0083	0.9181±0.0065	0.7583±0.0184	0.8737±0.0094	0.8754±0.0091	0.9770±0.0040

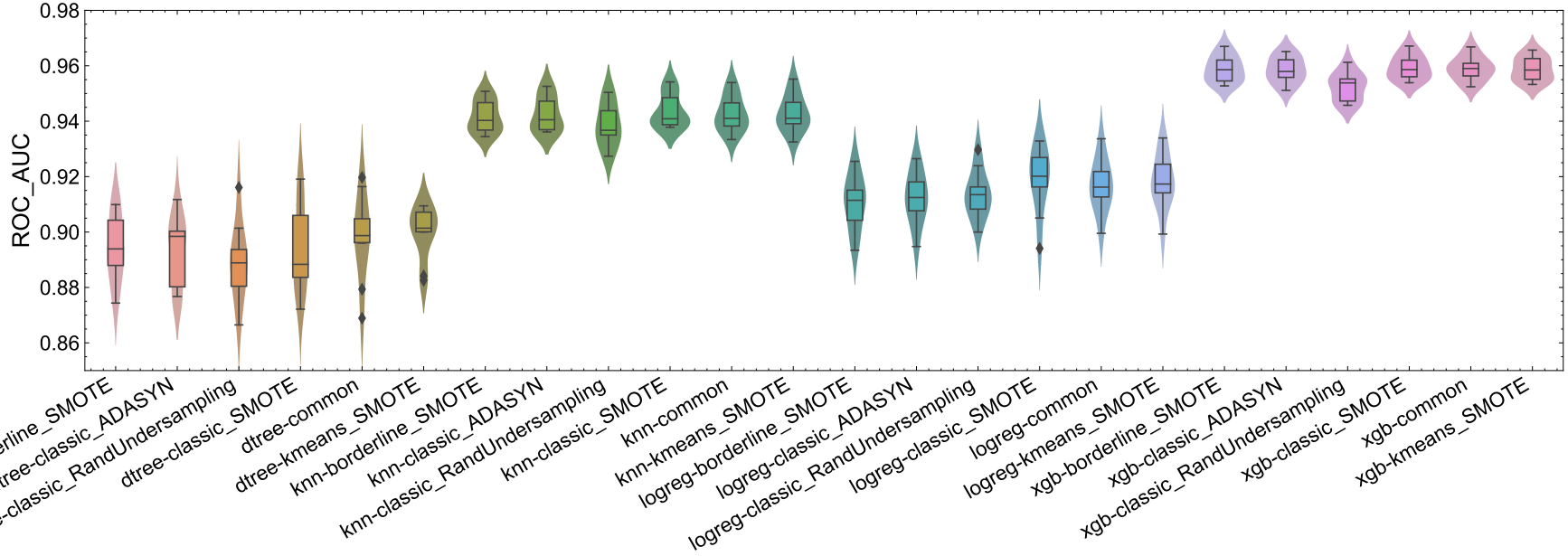


Fig. 6.5 Area under the curve (AUC) for ROC curves of 24 predictive models by combining XGBoost, kNN, logistical regression and decision trees with various sampling strategies respectively.

For the averaged model performances of 24 models, we can conclude that XGBoost > kNN > logistical regression > decision trees, as shown in Figure D.5. The kNN based models give AUC_ROC between 0.9387 and 0.9436. It is noticed that both knn-kmeans_SMOTE (0.9502) and knn-common (0.9547) give the best AP, implying that these two models are better at picking BBB+ molecules (true positive samples). The dramatic decrease of sensitivity for knn-borderline_SMOTE model and knn-classical_ADASYN compared with that of xgb-classical_ADASYN, 0.9275 suggests that knn-borderline_SMOTE model and knn-classical_ADASYN are suboptimal for selecting BBB+ molecules. For more detailed model performance, please see Figure D.5 and Table D.1.

6.3.2 Choice of Sampling Strategies

Imbalanced data is ubiquitous in practical applications, and different methods have been proposed to correct for the biases that imbalanced data can induce in classification mdoels, including methods at the algorithmic level (e.g. biased minimax probability machine (BMPM)) and resampling strategies (e.g., synthetic minority oversampling technique (SMOTE) [74]). There are mainly 3 different resampling strategies: undersampling on the majority class, oversampling over the minority class to generate synthetic data points, and hybrid models [75, 76]. Undersampling can help increase the sensitivity of a classifier to the minority class [74, 77], but tends to ignore information from the majority class. We only selected a random sampling strategy without replacement in this study. For oversampling, SMOTE [74], borderline SMOTE [78], k-means SMOTE [79], and adaptive synthetic (ADASYN) [80] were used.

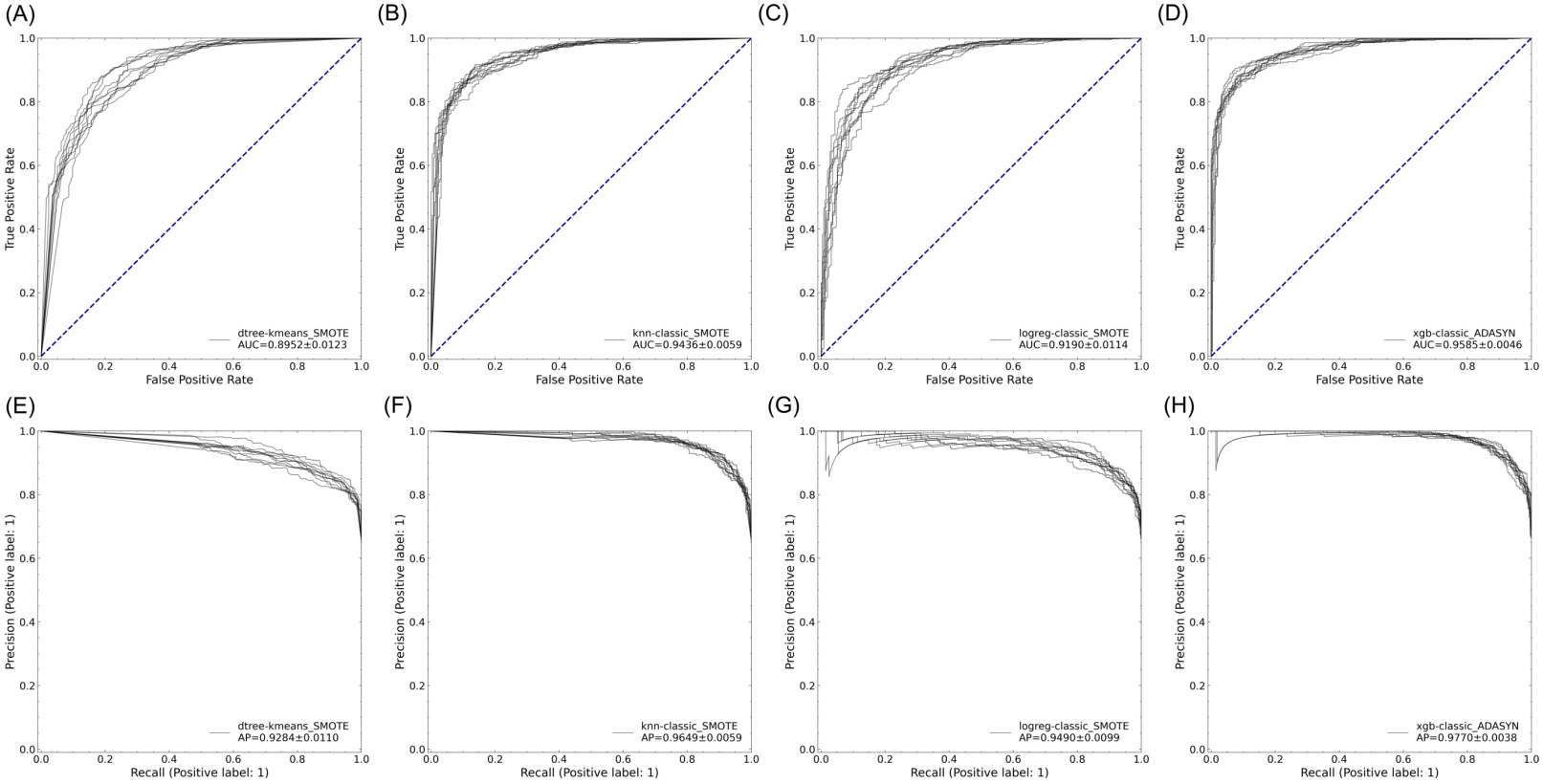


Fig. 6.6 ROC curves and prevision-recall curves for each classification algorithm with 10-fold cross-validation.

SMOTE, one of the most widely used used oversampling strategy, generates new synthetic data points for each minority class instance by interpolating existing data points [74]. Inspired by the success of SMOTE, many variants have been proposed,[81] including borderline SMOTE [78] and k-means SMOTE [79], adaptive synthetic (ADASYN) [80], and density-based SMOTEE (DBSMOTE) [82]. Borderline SMOTE, an adaptive SMOTE variant, generates artificial data points near the decision boundary to improve the classification performance because it is argued that this region is where minority examples are most susceptible to misclassification[78]. k-means SMOTE is designed to smooth over noise in the data and improve the description of the decision boundary [79]. ADASYN generates more synthetic minority class entries in an adaptive pattern by focussing on entries which are hard to learn, thus reducing the bias of the classifier [80]. In this study, SMOTE [78] and k-means SMOTE [79], and (ADASYN) [80] were selected as oversampling strategies for the imbalanced data.

The effect of sampling strategies was further analyzed by comparing to the common (non-resampled) classification algorithm. Undersampling method is inferior because it often leads to lower performance scores than the standard form, Figure D.6; this is unsurprising given that our dataset is relatively small (there is not an overabundance of data) and the imbalance between the majority (BBB+) and minority (BBB-) classes is not extreme. The XGBoost model with undersampling gives highest specificity score among all the XGBoost based resampling methods because the false positive rate is low, Figure D.6. But its sensitivity is smallest because of its tendency to give false negatives. Undersampling with kNN tends to give similar performance to traditional kNN, perhaps because kNN captures the pairwise similarity of input molecules.

As expected, oversampling can improve the model performance. Both xgb-borderline_SMOTE and xgb-ADASYN tend to outperform classic SMOTE and k-mean SMOTE, Figure D.6. For example, borderline SMOTE and ADASYN do a better job than classic SMOTE and k-mean SMOTE when using decision trees, KNN, and XGBoost. XGBoost does not greatly benefit from oversampling strategies, perhaps because our database is not very imbalanced, and XGBoost is already a powerful ensemble learning method. Oversampling tends to benefit other classification models more; the best-performing SMOTE variants are borderline SMOTE (for knn, decision tree), ADASYN (for knn), and k-means SMOTE (for logistic regression).

6.3.3 Evaluation with External Dataset

To further validate our models, the proposed models were tested against the external dataset as described in subsection 6.2.2. For consistency, we used the same procedures for molecular geometry optimization and feature generation as B3DB. Besides the 24 models reported in this study, we also compared their performances to *LightBBB*, a state-of-the-art model for BBB permeability prediction.

According to the results of model performances on the curated external dataset (see Table D.2), knn-common gets higher scores of accuracy, sensitivity, precision, F_1 score, MCC, GEOM, BACC, highest positive rate and lowest false negative rate out of all models. More specifically, the accuracy, sensitivity, precision of knn-common is 5%, 5%, 3% greater than that of `LightBBB` respectively. The performances of knn-kmeans_SMOTE are close to that of knn-common, all better than `LightBBB` except specificity. It is noticed that there is no information of ROC_AUC and AP for `LightBBB` because it cannot provide probabilities of the predictions.

One limitation of the external dataset is the limited number of BBB- molecules; this is a poor representation of our target application in molecular screening, as most drug-like molecules are BBB-. Due to the lack of BBB- molecules in our external data, the TN and FP are most equal with the specificity score being not statistical significant (ranging from 0.3750 to 0.5000).

6.3.4 `B3clf`: An Open-Source Python Package for BBB Predictions

To facilitate evaluation of molecule permeability at an early stage of the CNS drug design and development pipeline, we built a free and open-source command-line tool in Python, `B3clf`, which includes the predictive models reported in this study. This input is a text file containing SDF filenames or SMILES strings; it then generates 3D coordinates, optimizes the geometry, and computes molecular descriptors as descrbed in (cf. Figure 6.7 (A)). `B3clf` then selects features and uses the pre-trained predictive models to assess whether a molecule is BBB-permeable or not. This whole procedure is encapsulated in a single line of `bash` code,

```
b3clf -mol input_molecules.sdf -clf xgb -sampling classic_SMOTE
 ↪ -out BBB_pred_results.xlsx -verbose 1
```

The running time of `B3clf` scales almost linearly with number of molecules, approximately 3.96 seconds per molecule (Figure 6.7 (B)).

Predictions from `B3clf` are output as a `CSV` file which contains molecule name, predicted probability and BBB permeability labels (BBB+/BBB-). The probability information not only provides the confidence/uncertainty of the predictions, but also makes it easy to plot the ROC and precision-recall curves, which we hope will make `B3clf` a useful comparison in future studies. `B3clf` is freely available at https://github.com/theochem/B3clf with detailed documentation.

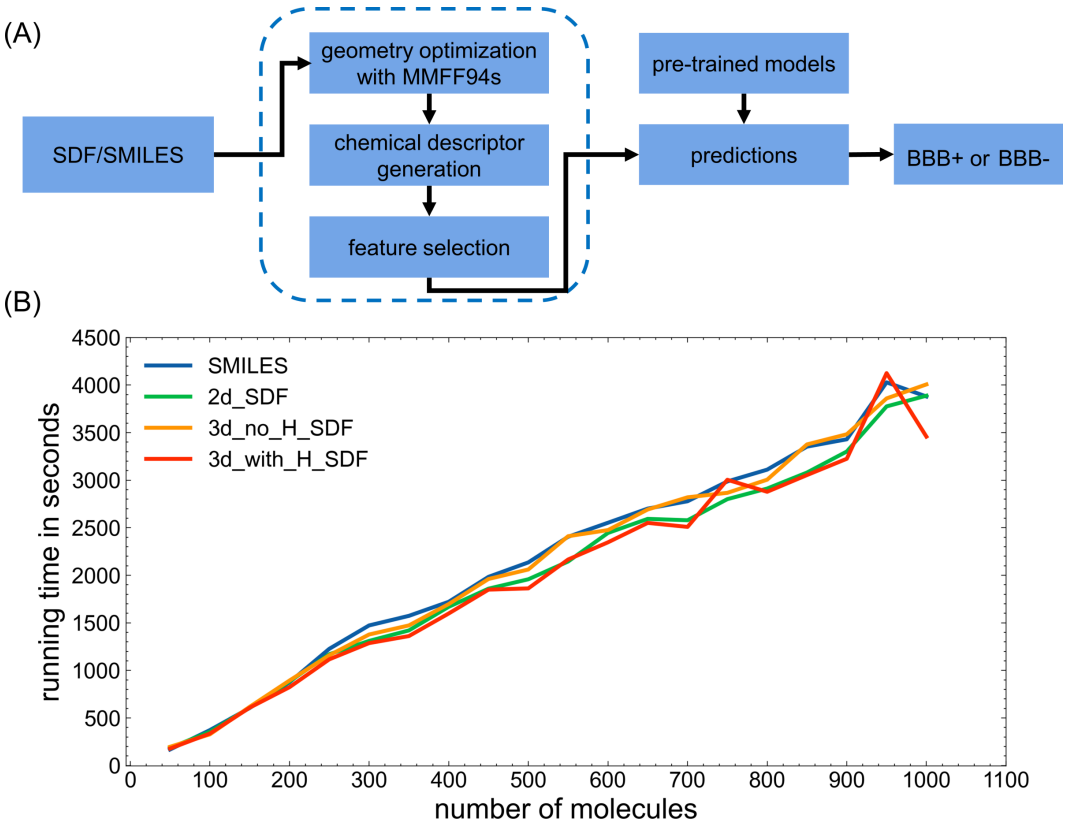


Fig. 6.7 Design structure (A) and performance profiling of `B3clf` (B).

6.4 Conclusions

BBB provides protective barrier mechanism for homeostatis of the CNS [1, 2, 83]. However, it is estimated that 98% of drugs [4] and approximately 100% of biomolecular pharmaceuticals cannot pass BBB, making the CNS drug discovery a challenging task. Various experimental methods have been proposed to enhance/facilitate the drug delivery into BBB phase attempting to tackle the obstacle for drug delivery of CNS drugs. But experimental measures of BBB permeability is labour extensive, time consuming and costly.

Therefore, computational predictions/modellings of BBB permeability of organic molecules an significant question in CNS drug discovery and development. Different computational models have been proposed for this purpose, especially the ML based predictors. But these models can not be generalizabled well to unseen molecules, which may account to: (1) limited chemical diversity covered by the training dataset (2) the imbalanced-class labels in molecule datasets for BBB.

We constructed a classifier to predict whether a molecule can cross the blood-brain-barrier (BBB+) or not (BBB-). Our model was trained and validated using data from B3DB (4956 BBB+ and 2851 BBB- molecules) [54] (see chapter 5) and finally validated against a external dataset of 224 molecules, which were curated especially for this work. We used decision trees, k-nearest neighbors (kNN), logistic regression, and XGBoost as classification models. Because the training data is somewhat imbalanced (imbalance ratio of 1.74), we combined these classifiers with various oversampling (SMOTE, k-means SMOTE, borderline SMOTE, and ADASYN) and undersampling (random) strategies to mitigate this bias. This results in 24 models (4 classifiers, 5 resampling strategies plus the raw (non-resampled) data), which were assessed by a panoply of metric (cf. subsection 6.2.8) and compared to a state-of-the-art model, *LightBBB*. We distribute our results in the free, open-source, and flexible computational tool, `B3clf` (https://github.com/theochem/B3clf).

The best model for most purposes seems to be XGBoost with ADASYN over-sampling, but many other models have comparable performance. The second-best oversampling method for XGBoost is borderline SMOTE. kNN with ADASYN and borderline SMOTE also perform well. Other classifiers and resampling methods are usually inferior, though they can be useful for applications where specific types of error (e.g., false positives) are especially problematic.

This study provides good predictive models for BBB permeability predictions for small molecules. But some questions remain to be answered. Ongoing work will explore the possibility of BBB permeability of all the drug molecules in DrugBank and this has great implication for drug repositioning. For example, all the antiviral and antibiotic drugs that are BBB permeability can be potentially used for brain infection related diseases [84].

Current feature selection is only based on simple statically method and it would be useful to explore more feature selection methods, such as information theory based feature filtering methods [85]. Moreover, the interpretability of the predictive models is not uncovered yet and some methods are worth trying, to name a few, shapley additive explanations (SHAP) [86]. Last but not least, understanding the relationship of chem-physical descriptors and predictions can help explain why

some models prefer specific resampling strategy, and how the decision boundary of different models were affect by the sampling strategies. For example, KNN tends to provide good performance when working with k-mean SMOTE.

References

[1] Werner Risau and Hartwig Wolburg. Development of the blood-brain barrier. *Trends in Neurosciences*, 13(5):174–178, 1990.

[2] Caterina P Profaci, Roeben N Munji, Robert S Pulido, and Richard Daneman. The blood–brain barrier in health and disease: Important unanswered questions. *Journal of Experimental Medicine*, 217(4), 2020.

[3] Richard Daneman. The blood–brain barrier in health and disease. *Annals of Neurology*, 72(5):648–672, 2012.

[4] William M Pardridge. Blood–brain barrier delivery. *Drug Discovery Today*, 12 (1-2):54–61, 2007.

[5] Juliane Gust, Kevin A Hay, Laïla-Aïcha Hanafi, Daniel Li, David Myerson, Luis F Gonzalez-Cuyar, Cecilia Yeung, W Conrad Liles, Mark Wurfel, Jose A Lopez, et al. Endothelial activation and blood–brain barrier disruption in neurotoxicity after adoptive immunotherapy with cd19 car-t cells. *Cancer Discovery*, 7(12):1404–1419, 2017.

[6] Hany E Marei, Asmaa Althani, Nahla Afifi, Anwarul Hasan, Thomas Caceci, Giacomo Pozzoli, and Carlo Cenciarelli. Current progress in chimeric antigen receptor t cell therapy for glioblastoma multiforme. *Cancer Medicine*, 2021.

[7] L Escuder-Gilabert, M Molero-Monfort, RM Villanueva-Camañas, S Sagrado, and MJ Medina-Hernández. Potential of biopartitioning micellar chromatography as an in vitro technique for predicting drug penetration across the blood–brain barrier. *Journal of Chromatography B*, 807(2):193–201, 2004.

[8] Ye Li, Tongkai Chen, Xiaoqing Miao, Xiang Yi, Xueqing Wang, Haitao Zhao, Simon Ming-Yuen Lee, and Ying Zheng. Zebrafish: A promising in vivo model for assessing the delivery of natural products, fluorescence dyes and drugs across the blood-brain barrier. *Pharmacological Research*, 125:246–257, 2017.

[9] C Lohmann, S Hüwel, and H-J Galla. Predicting blood-brain barrier permeability of drugs: evaluation of different in vitro assays. *Journal of Drug Targeting*, 10 (4):263–276, 2002.

[10] Stefan Lundquist, Mila Renftel, Julien Brillault, Laurence Fenart, Roméo Cecchelli, and Marie-Pierre Dehouck. Prediction of drug transport through the

blood-brain barrier in vivo: a comparison between two in vitro cell models. *Pharmaceutical Research*, 19(7):976–981, 2002.

[11] Per Artursson, Katrin Palm, and Kristina Luthman. Caco-2 monolayers in experimental and theoretical predictions of drug transport. *Advanced Drug Delivery Reviews*, 46(1-3):27–43, 2001.

[12] Louise Delsing, Anna Herland, Anna Falk, Ryan Hicks, Jane Synnergren, and Henrik Zetterberg. Models of the blood-brain barrier using ipsc-derived cells. *Molecular and Cellular Neuroscience*, page 103533, 2020.

[13] Michael J Workman and Clive N Svendsen. Recent advances in human ipsc-derived models of the blood–brain barrier. *Fluids and Barriers of the CNS*, 17 (1):1–10, 2020.

[14] Sehoon Jeong, Sunja Kim, John Buonocore, Jaewon Park, C Jane Welsh, Jianrong Li, and Arum Han. A three-dimensional arrayed microfluidic blood–brain barrier model with integrated electrical sensor array. *IEEE Transactions on Biomedical Engineering*, 65(2):431–439, 2017.

[15] Li Di, Edward H Kerns, Kristi Fan, Oliver J McConnell, and Guy T Carter. High throughput artificial membrane permeability assay for blood–brain barrier. *European Journal of Medicinal Chemistry*, 38(3):223–232, 2003.

[16] Shichao Ding, Aminul Islam Khan, Xiaoli Cai, Yang Song, Zhaoyuan Lyu, Dan Du, Prashanta Dutta, and Yuehe Lin. Overcoming blood–brain barrier transport: Advances in nanoparticle-based drug delivery strategies. *Materials Today*, 2020.

[17] John JE Mulvihill, Eoghan M Cunnane, Aisling M Ross, Jason T Duskey, Giovanni Tosi, and Andreas M Grabrucker. Drug delivery across the blood–brain barrier: recent advances in the use of nanocarriers. *Nanomedicine*, 15(2): 205–214, 2020.

[18] Yan Liang and Jeong-Yeol Yoon. In situ sensors for blood-brain barrier (bbb) on a chip. *Sensors and Actuators Reports*, page 100031, 2021.

[19] Deeksha Saxena, Anju Sharma, Mohammed H Siddiqui, and Rajnish Kumar. Blood brain barrier permeability prediction using machine learning techniques: An update. *Current Pharmaceutical Biotechnology*, 20(14):1163–1171, 2019.

[20] Markus Muehlbacher, Gudrun M Spitzer, Klaus R Liedl, and Johannes Kornhuber. Qualitative prediction of blood–brain barrier permeability on a large and refined dataset. *Journal of Computer-Aided Molecular Design*, 25(12): 1095–1106, 2011.

[21] Timothy S Carpenter, Daniel A Kirshner, Edmond Y Lau, Sergio E Wong, Jerome P Nilmeier, and Felice C Lightstone. A method to predict blood-brain barrier permeability of drug-like compounds using molecular dynamics simulations. *Biophysical Journal*, 107(3):630–641, 2014.

[22] William M Pardridge. The blood-brain barrier: bottleneck in brain drug development. *NeuroRx*, 2(1):3–14, 2005.

[23] Seetha Harilal, Jobin Jose, Rajesh Kumar, Mazhuvancherry Kesavan Unnikrishnan, Md Sahab Uddin, Githa Elizabeth Mathew, Rahul Pratap, Akash Marathakam, Bijo Mathew, et al. Revisiting the blood-brain barrier: A hard nut to crack in the transportation of drug molecules. *Brain Research Bulletin*, 2020.

[24] Szilvia Veszelka, Ágnes Kittel, and Mária A Deli. Tools of modelling blood–brain barrier penetrability. *Solubility, Delivery and ADME Problems of Drugs and Drug-Candidates, Bentham Science Publishers, Washington*, pages 166–188, 2011.

[25] Franco Lombardo, James F Blake, and William J Curatolo. Computation of brain- blood partitioning of organic solutes via free energy calculations. *Journal of Medicinal Chemistry*, 39(24):4750–4755, 1996.

[26] Nguyen Quoc Thai, Panagiotis E Theodorakis, and Mai Suan Li. Fast estimation of the blood–brain barrier permeability by pulling a ligand through a lipid membrane. *Journal of Chemical Information and Modeling*, 60(6):3057–3067, 2020.

[27] Yukun Wang, Erin Gallagher, Christian Jorgensen, Evan P Troendle, Dan Hu, Peter C Searson, and Martin B Ulmschneider. An experimentally validated approach to calculate the blood-brain barrier permeability of small molecules. *Scientific Reports*, 9(1):1–11, 2019.

[28] Mayuri Gupta, Hyeok Jun Lee, Christopher J Barden, and Donald F Weaver. The blood–brain barrier (bbb) score. *Journal of Medicinal Chemistry*, 62(21): 9824–9836, 2019.

[29] Baichen Xiong, Yuanyuan Wang, Ying Chen, Shuaishuai Xing, Qinghong Liao, Yao Chen, Qi Li, Wei Li, and Haopeng Sun. Strategies for structural modification of small molecules to improve blood–brain barrier penetration: A recent perspective. *Journal of Medicinal Chemistry*, 2021.

[30] Liying Zhang, Hao Zhu, Tudor I Oprea, Alexander Golbraikh, and Alexander Tropsha. Qsar modeling of the blood–brain barrier permeability for diverse organic compounds. *Pharmaceutical Research*, 25(8):1902–1914, 2008.

[31] Juan M Luco and Eduardo Marchevsky. Qsar studies on blood-brain barrier permeation. *Current Computer-Aided Drug Design*, 2(1):31–55, 2006.

[32] Wenyi Wang, Marlene T Kim, Alexander Sedykh, and Hao Zhu. Developing enhanced blood–brain barrier permeability models: integrating external bioassay data in qsar modeling. *Pharmaceutical Research*, 32(9):3055–3065, 2015.

[33] Manisha Iyer, Rama Mishra, Yi Han, and AJ Hopfinger. Predicting blood–brain barrier partitioning of organic molecules using membrane–interaction qsar analysis. *Pharmaceutical Research*, 19(11):1611–1621, 2002.

[34] Ulf Norinder and Markus Haeberlein. Computational approaches to the prediction of the blood–brain distribution. *Advanced Drug Delivery Reviews*, 54(3):291–313, 2002.

[35] Prabha Garg and Jitender Verma. In silico prediction of blood brain barrier permeability: an artificial neural network model. *Journal of Chemical Information and Modeling*, 46(1):289–297, 2006.

[36] Shrooq Alsenan, Isra Al-Turaiki, and Alaaeldin Hafez. A deep learning approach to predict blood-brain barrier permeability. *PeerJ Computer Science*, 7:e515, 2021.

[37] Abena Achiaa Atwereboannah, Wei-Ping Wu, and Ebenezer Nanor. Prediction of drug permeability to the blood-brain barrier using deep learning. In *4th International Conference on Biometric Engineering and Applications*, pages 104–109, 2021.

[38] Sandhya Kortagere, Dmitriy Chekmarev, William J Welsh, and Sean Ekins. New predictive models for blood–brain barrier permeability of drug-like molecules. *Pharmaceutical Research*, 25(8):1836–1845, 2008.

[39] Manvi Singh, Reshmi Divakaran, Leela Sarath Kumar Konda, and Rajendra Kristam. A classification model for blood brain barrier penetration. *Journal of Molecular Graphics and Modelling*, 96:107516, 2020.

[40] Lili Liu, Li Zhang, Huawei Feng, Shimeng Li, Miao Liu, Jian Zhao, and Hongsheng Liu. Prediction of the blood–brain barrier (bbb) permeability of chemicals based on machine-learning and ensemble methods. *Chemical Research in Toxicology*, 2021.

[41] Eric Deconinck, Menghui H Zhang, Danny Coomans, and Yvan Vander Heyden. Classification tree models for the prediction of blood- brain barrier passage of drugs. *Journal of Chemical Information and Modeling*, 46(3):1410–1419, 2006.

[42] Yuan H Zhao, Michael H Abraham, Adam Ibrahim, Paul V Fish, Susan Cole, Mark L Lewis, Marcel J de Groot, and Derek P Reynolds. Predicting penetration across the blood-brain barrier from simple descriptors and fragmentation schemes. *Journal of Chemical Information and Modeling*, 47(1):170–175, 2007.

[43] Claudia Suenderhauf, Felix Hammann, and Jörg Huwyler. Computational prediction of blood-brain barrier permeability using decision tree induction. *Molecules*, 17(9):10429–10445, 2012.

[44] Yaxia Yuan, Fang Zheng, and Chang-Guo Zhan. Improved prediction of blood–brain barrier permeability through machine learning with combined use of molecular property-based descriptors and fingerprints. *The AAPS Journal*, 20(3):1–10, 2018.

[45] Dipankar Roy, Vijaya Kumar Hinge, and Andriy Kovalenko. To pass or not to pass: predicting the blood–brain barrier permeability with the 3d-rism-kh molecular solvation theory. *ACS Omega*, 4(16):16774–16780, 2019.

[46] Shrooq Alsenan, Isra Al-Turaiki, and Alaaeldin Hafez. A recurrent neural network model to predict blood–brain barrier permeability. *Computational Biology and Chemistry*, 89:107377, 2020.

[47] Ines Filipa Martins, Ana L Teixeira, Luis Pinheiro, and Andre O Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of Chemical Information and Modeling*, 52(6):1686–1697, 2012.

[48] Bilal Shaker, Myeong-Sang Yu, Jin Sook Song, Sunjoo Ahn, Jae Yong Ryu, Kwang-Seok Oh, and Dokyun Na. Lightbbb: computational prediction model of blood–brain-barrier penetration based on lightgbm. *Bioinformatics*, 37(8): 1135–1139, 2021.

[49] Gopinath Rebala, Ajay Ravi, and Sanjay Churiwala. *Classification*, pages 57–66. Springer International Publishing, 2019. ISBN 978-3-030-15729-6. doi: 10.1007/978-3-030-15729-6_5.

[50] Tiago Pereira, Maryam Abbasi, José Luis Oliveira, Bernardete Ribeiro, and Joel Arrais. Optimizing blood–brain barrier permeation through deep reinforcement learning for *de novo* drug design. *Bioinformatics*, 37(Supplement_1):i84–i92, 07 2021. ISSN 1367-4803. doi: 10.1093/bioinformatics/btab301.

[51] Zhuang Wang, Hongbin Yang, Zengrui Wu, Tianduanyi Wang, Weihua Li, Yun Tang, and Guixia Liu. In silico prediction of blood–brain barrier permeability of compounds by machine learning and resampling methods. *ChemMedChem*, 13(20):2189–2201, 2018.

[52] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[53] Zhiwen Shi, Yanyi Chu, Yonghong Zhang, Yanjing Wang, and DongQing Wei. Prediction of blood-brain barrier permeability of compounds by fusing resampling strategies and extreme gradient boosting. *IEEE Access*, 9:9557–9566, 2020.

[54] Fanwang Meng, Yang Xi, Jinfeng Huang, and Paul W. Ayers. A curated diverse molecular database of blood-brain barrier permeability with chemical descriptors. *Scientific Data*, 8(289), 2021. doi: 10.1038/s41597-021-01069-5.

[55] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[56] Nurulfitrah Noorhalim, Aida Ali, and Siti Mariyam Shamsuddin. Handling imbalanced ratio for class imbalance problem using smote. In *Proceedings of the Third International Conference on Computing, Mathematics and Statistics (iCMS2017)*, pages 19–30. Springer, 2019.

[57] Zhen Gao, Yang Chen, Xiaoshu Cai, and Rong Xu. Predict drug permeability to blood–brain-barrier from clinical phenotypes: drug side effects and drug indications. *Bioinformatics*, 33(6):901–908, 2017.

[58] Vito Lorusso, Domenico Galetta, Francesco Giotta, Antonio Rinaldi, Sante Romito, Cosimo Brunetti, Nicola Silvestris, and Giuseppe Colucci. Topotecan in the treatment of brain metastases. a phase ii study of goim (gruppo oncologico dell'italia meridionale). *Anticancer Research*, 26(3B):2259–2263, 2006.

[59] Diana S Church and Martin K Church. Pharmacology of antihistamines. *World Allergy Organization Journal*, 4(3):S22–S27, 2011.

[60] Ikumi Tamai, Yasuto Kido, Junko Yamashita, Yoshimichi Sai, and Akira Tsuji. Blood-brain barrier transport of h1-antagonist ebastine and its metabolite carebastine. *Journal of Drug Targeting*, 8(6):383–393, 2000.

[61] Matt Swain. Pubchempy, Apr 2017. URL https://github.com/mcs07/PubChemPy.

[62] Noel M O'Boyle, Michael Banck, Craig A James, Chris Morley, Tim Vandermeersch, and Geoffrey R Hutchison. Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3(1):1–14, 2011.

[63] Dejun Jiang, Zhenxing Wu, Chang-Yu Hsieh, Guangyong Chen, Ben Liao, Zhe Wang, Chao Shen, Dongsheng Cao, Jian Wu, and Tingjun Hou. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of Cheminformatics*, 13(1):1–23, 2021.

[64] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):1–45, 2017.

[65] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[66] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(1-4):131–156, 1997.

[67] James Bergstra, Brent Komer, Chris Eliasmith, Dan Yamins, and David D Cox. Hyperopt: a python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, 8(1):014008, 2015.

[68] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[69] James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International Conference on Machine Learning*, pages 115–123. PMLR, 2013.

[70] Nathalie Japkowicz. *Assessment Metrics for Imbalanced Learning*, chapter 8, pages 187–206. John Wiley & Sons, Ltd, 2013. ISBN 9781118646106. doi: https://doi.org/10.1002/9781118646106.ch8. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118646106.ch8.

[71] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[72] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS ONE*, 10(3):e0118432, 2015.

[73] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

[74] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[75] Ajinkya More. Survey of resampling techniques for improving classification performance in unbalanced datasets. *arXiv preprint arXiv:1608.06048*, 2016.

[76] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20 (1):18–36, 2004.

[77] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Proceedings of the International Conference on Machine Learning (ICML 2003) Workshop on Learning from Imbalanced Data Sets II*, volume 11, pages 1–8. Citeseer, 2003.

[78] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer, 2005.

[79] Felix Last, Georgios Douzas, and Fernando Bacao. Oversampling for imbalanced learning based on k-means and smote. *arXiv preprint arXiv:1711.00837*, 2017.

[80] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328. IEEE, 2008.

[81] Alberto Fernández, Salvador Garcia, Francisco Herrera, and Nitesh V Chawla. Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research*, 61:863–905, 2018.

[82] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Dbsmote: density-based synthetic minority over-sampling technique. *Applied Intelligence*, 36(3):664–684, 2012.

[83] Richard Daneman and Alexandre Prat. The blood–brain barrier. *Cold Spring Harbor Perspectives in Biology*, 7(1):a020412, 2015.

[84] Roland Nau, Fritz Sögel, and Helmut Eiffert. Penetration of drugs through the blood-cerebrospinal fluid/blood-brain barrier for treatment of central nervous system infections. *Clinical Microbiology Reviews*, 23(4):858–883, 2010.

[85] Marc Sebban and Richard Nock. A hybrid filter/wrapper approach of feature selection using information theory. *Pattern Recognition*, 35(4):835–846, 2002.

[86] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4768–4777, 2017.

Appendix A

Supporting Information for Allostery Prediction

A.1 Supporting Figures for Allostery Prediction

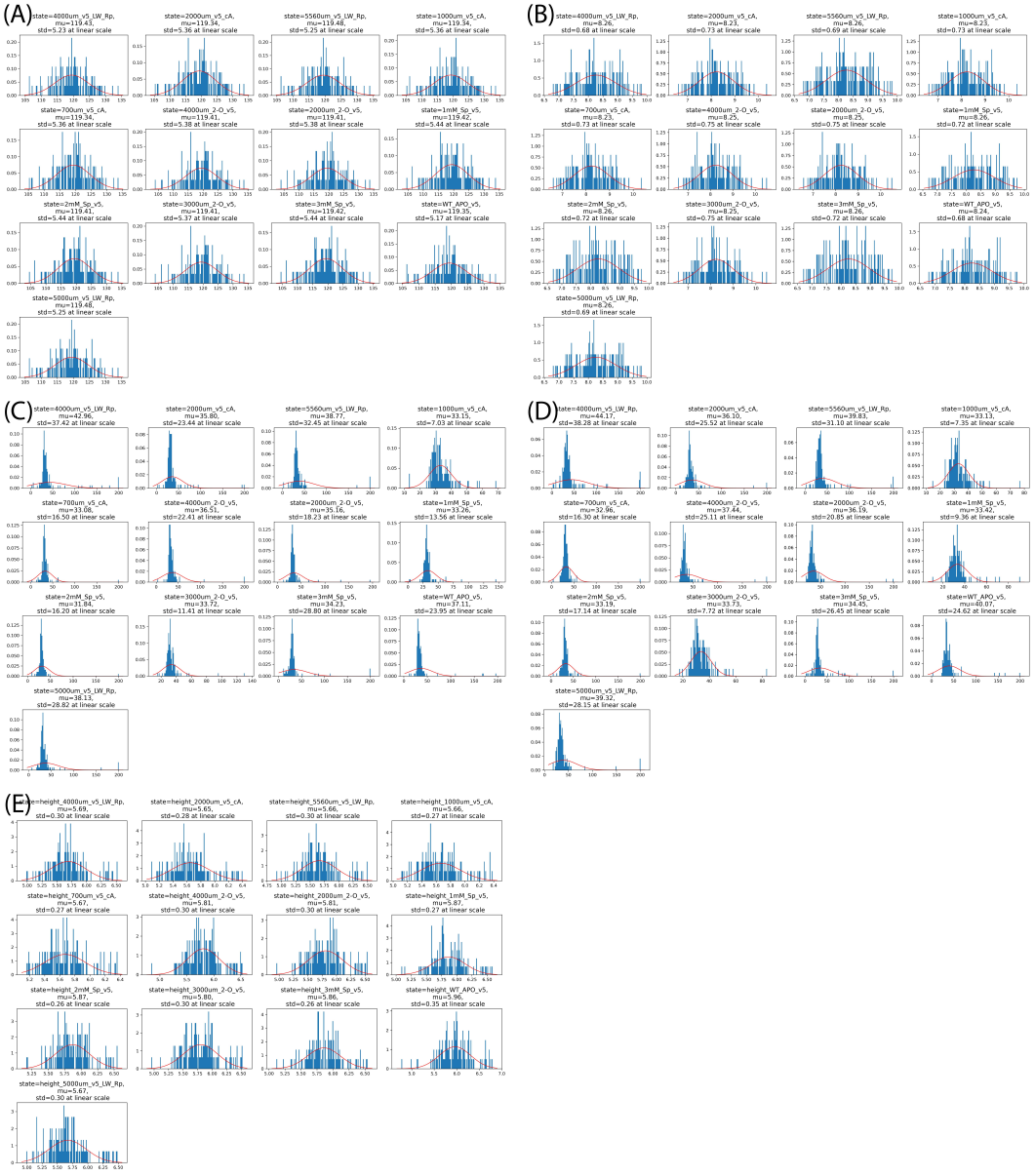


Fig. A.1 Distributions of observed NMR data at linear scale. (A). Chemical shifts of N atoms. (B). Chemical shifts of H atoms. (C). Line width of N atoms. (D). Line width of H atoms. (E). Data height of NMR spectroscopy.

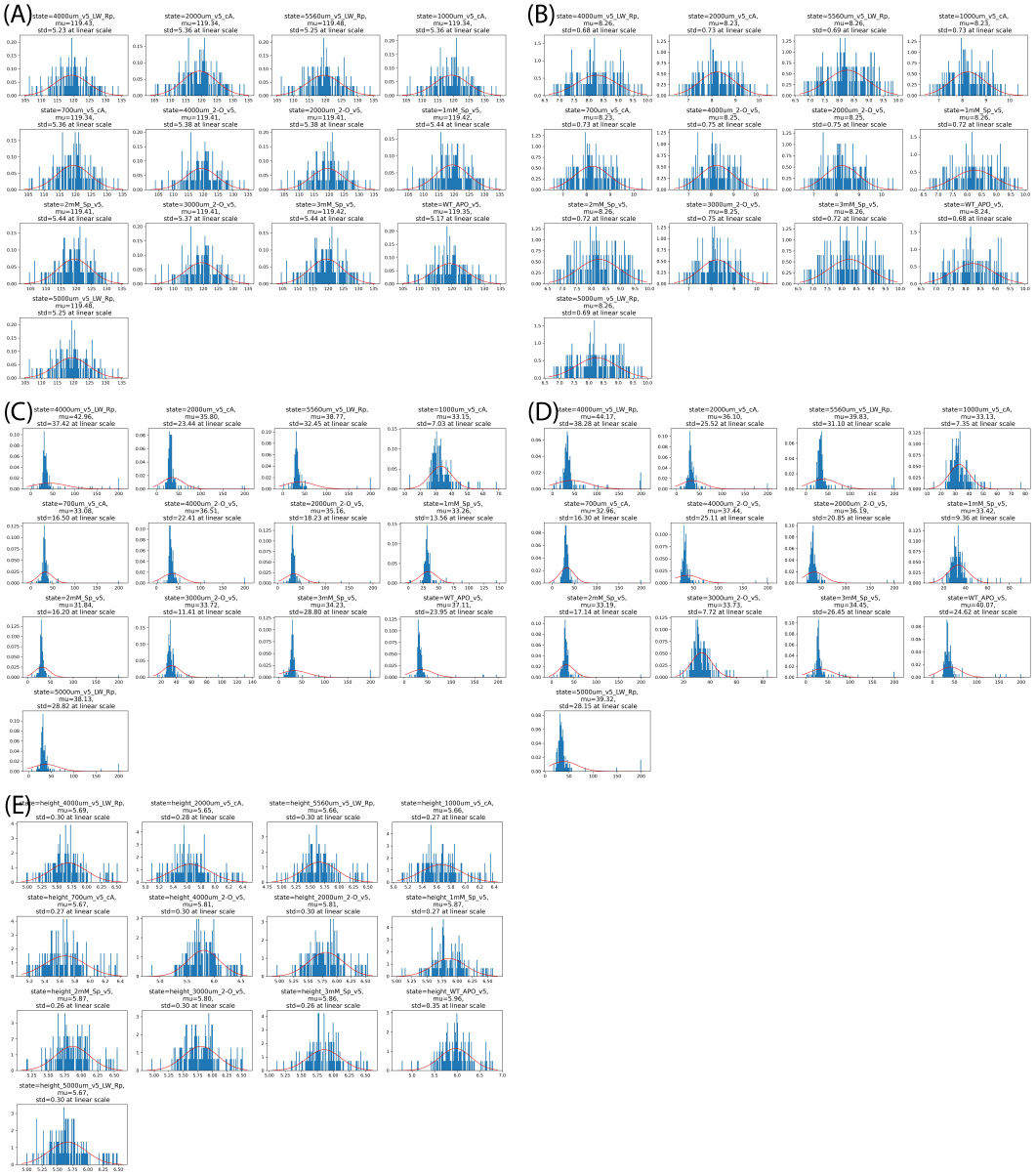


Fig. A.2 Distributions of observed NMR data at log10 scale. (A). Chemical shifts of N atoms. (B). Chemical shifts of H atoms. (C). Line width of N atoms. (D). Line width of H atoms. (E). Data height of NMR spectroscopy.

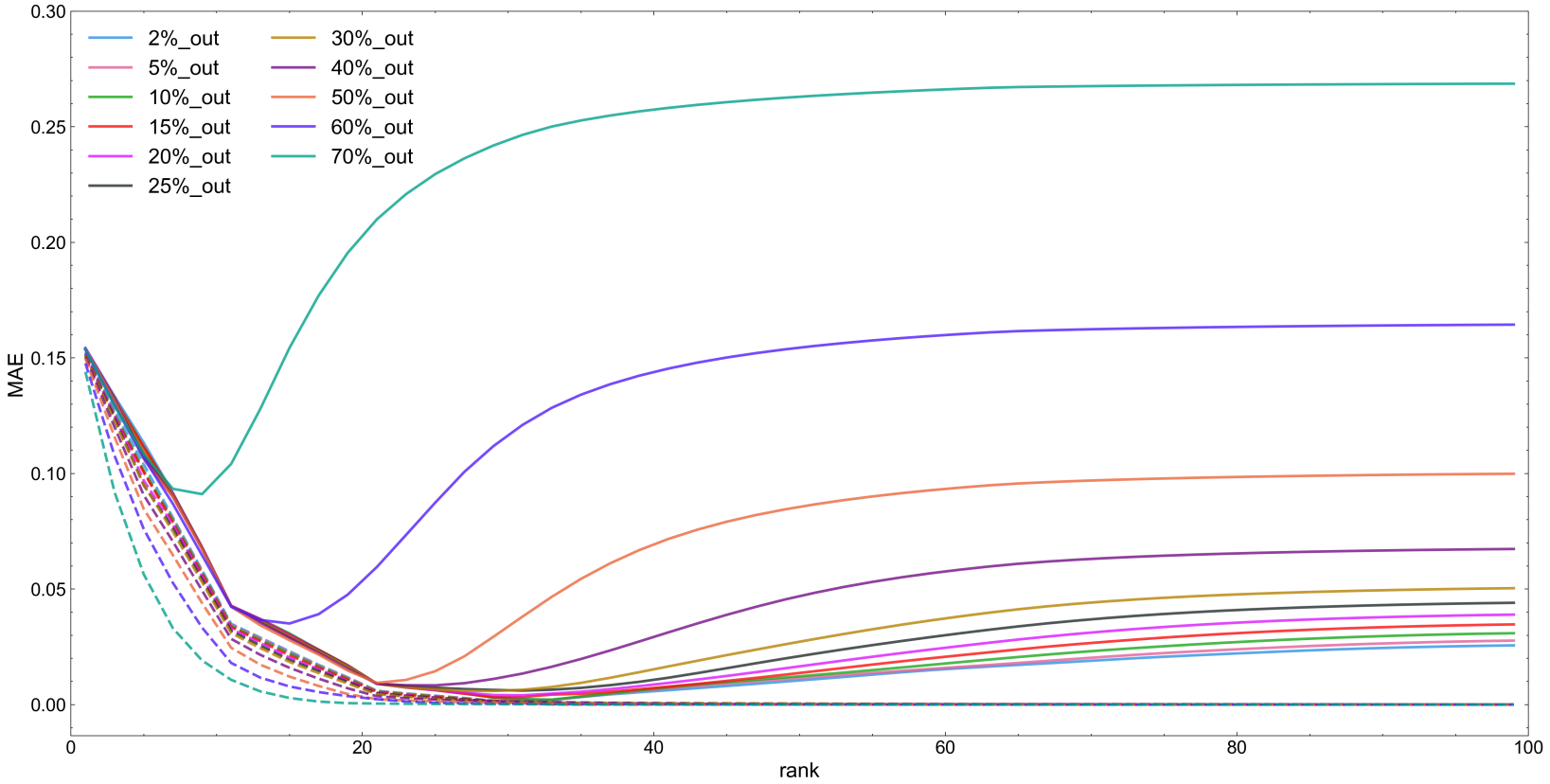


Fig. A.3 Learning curves for synthetic data $\mathbf{X} \in \mathbb{R}^{500\times100}$ without noise (15% missing data) with different percent of hold-out testing data. The training data is denoted as dashed line and the testing data curve is shown in solid line. MAE is selected as error measurement.

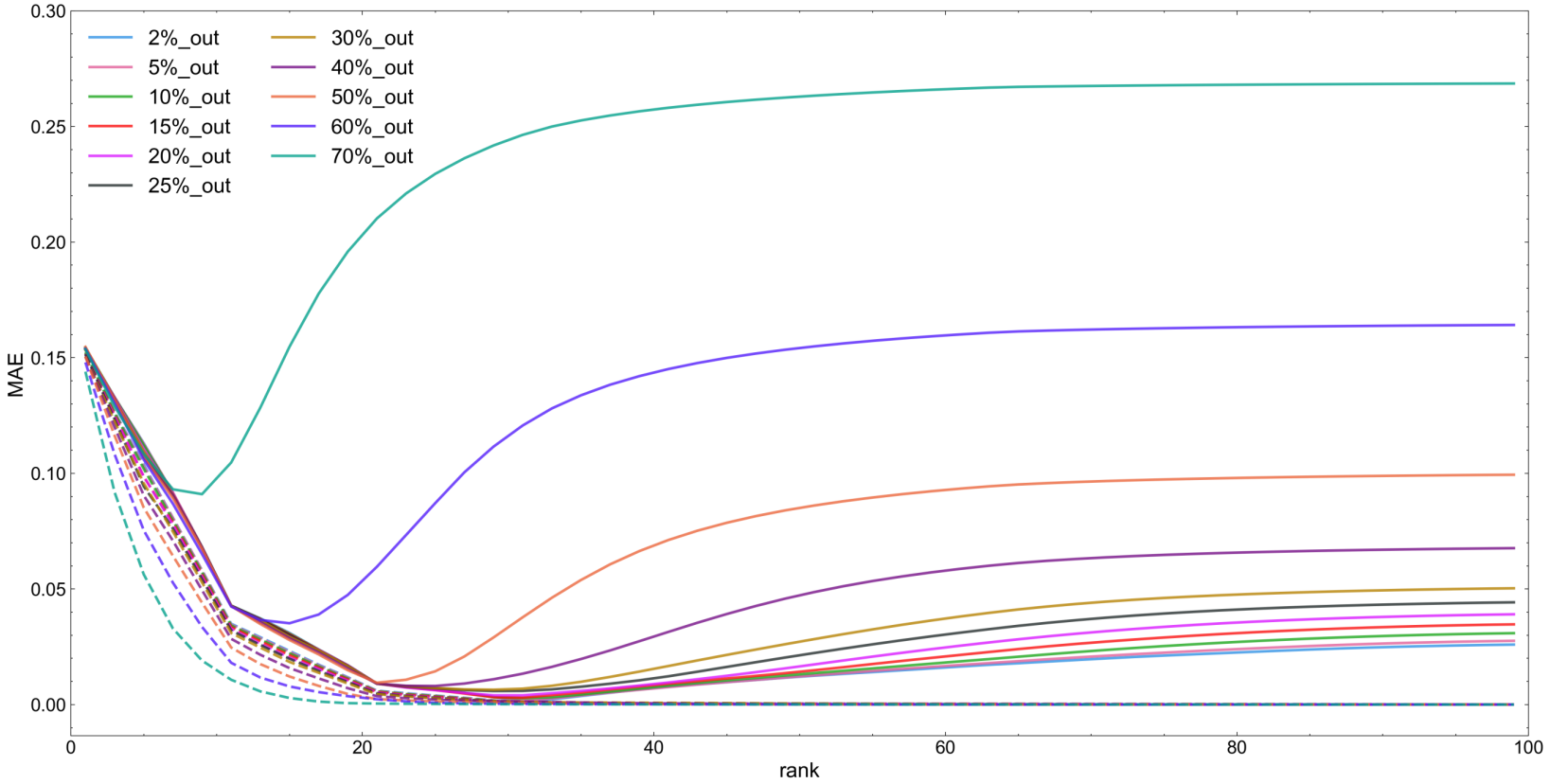


Fig. A.4 Learning curves for synthetic data $\mathbf{X} \in \mathbb{R}^{500\times100}$ at noise level of 0.01 (15% missing data) with different percent of hold-out testing data. The training data is denoted as dashed line and the testing data curve is shown in solid line. MAE is selected as error measurement.

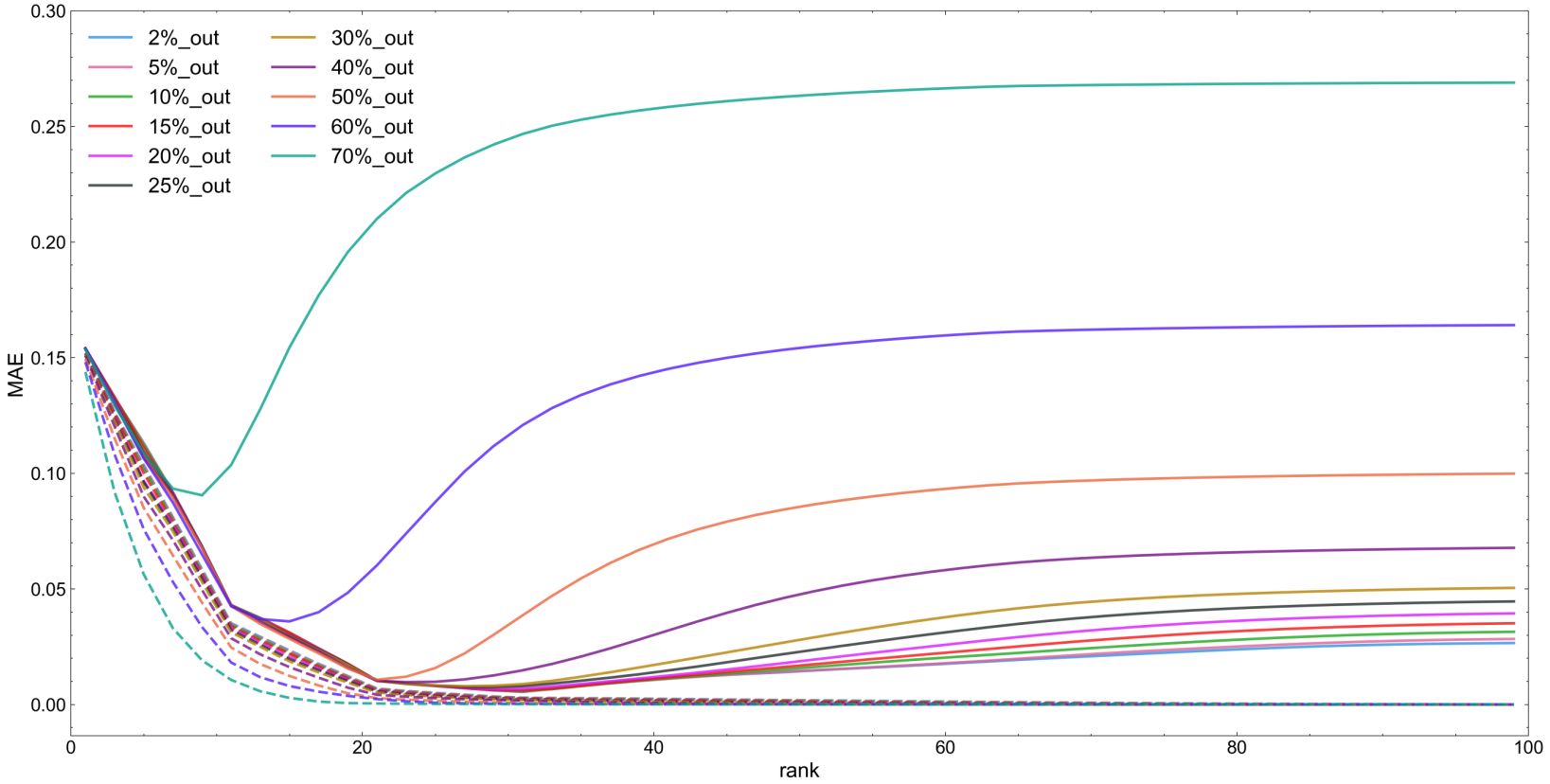


Fig. A.5 Learning curves for synthetic data $\mathbf{X} \in \mathbb{R}^{500\times100}$ at noise level of 0.05 (15% missing data) with different percent of hold-out testing data. The training data is denoted as dashed line and the testing data curve is shown in solid line. MAE is selected as error measurement.

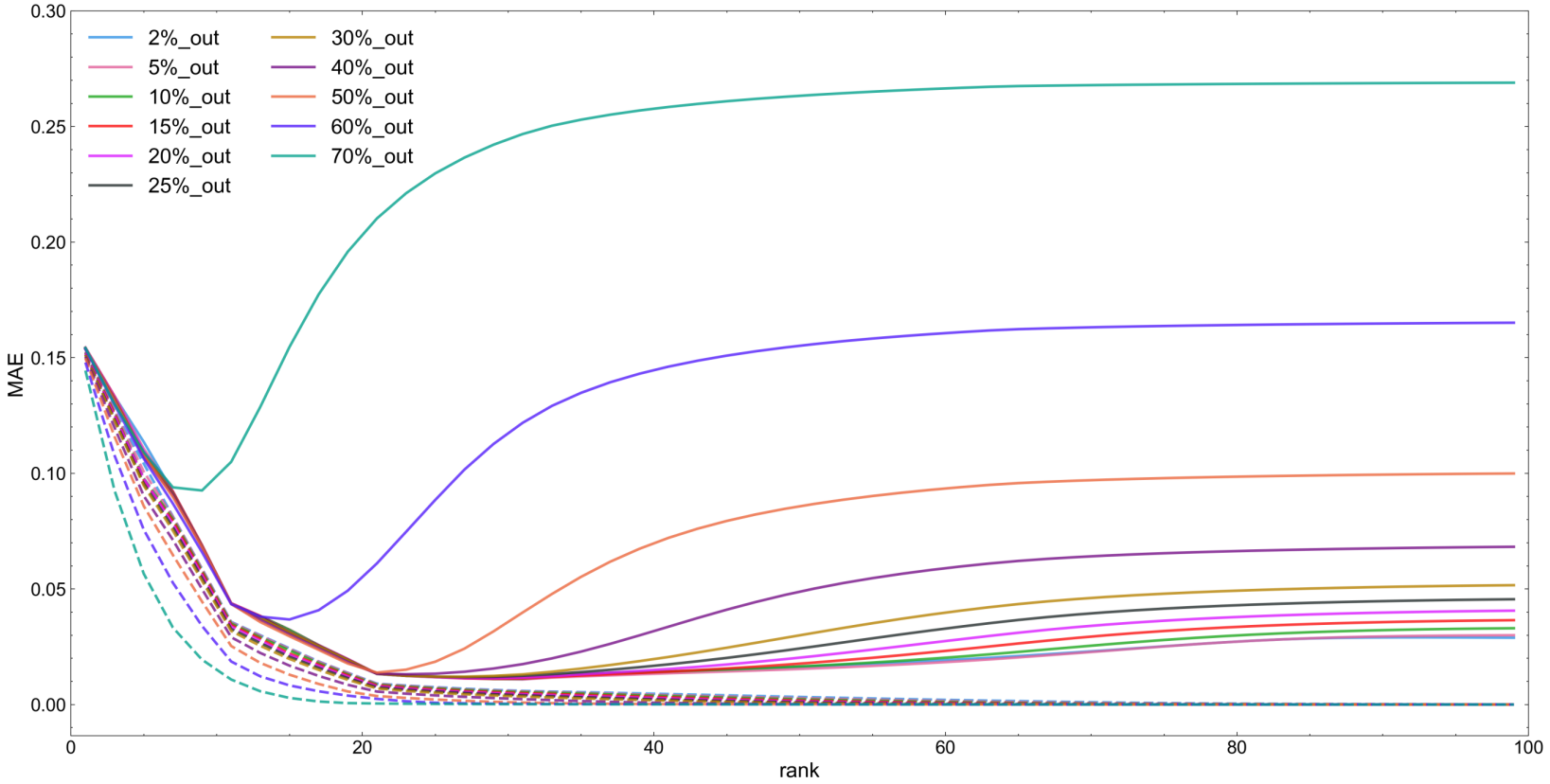


Fig. A.6 Learning curves for synthetic data $\mathbf{X} \in \mathbb{R}^{500\times100}$ at noise level of 0.10 (15% missing data) with different percent of hold-out testing data. The training data is denoted as dashed line and the testing data curve is shown in solid line. MAE is selected as error measurement.

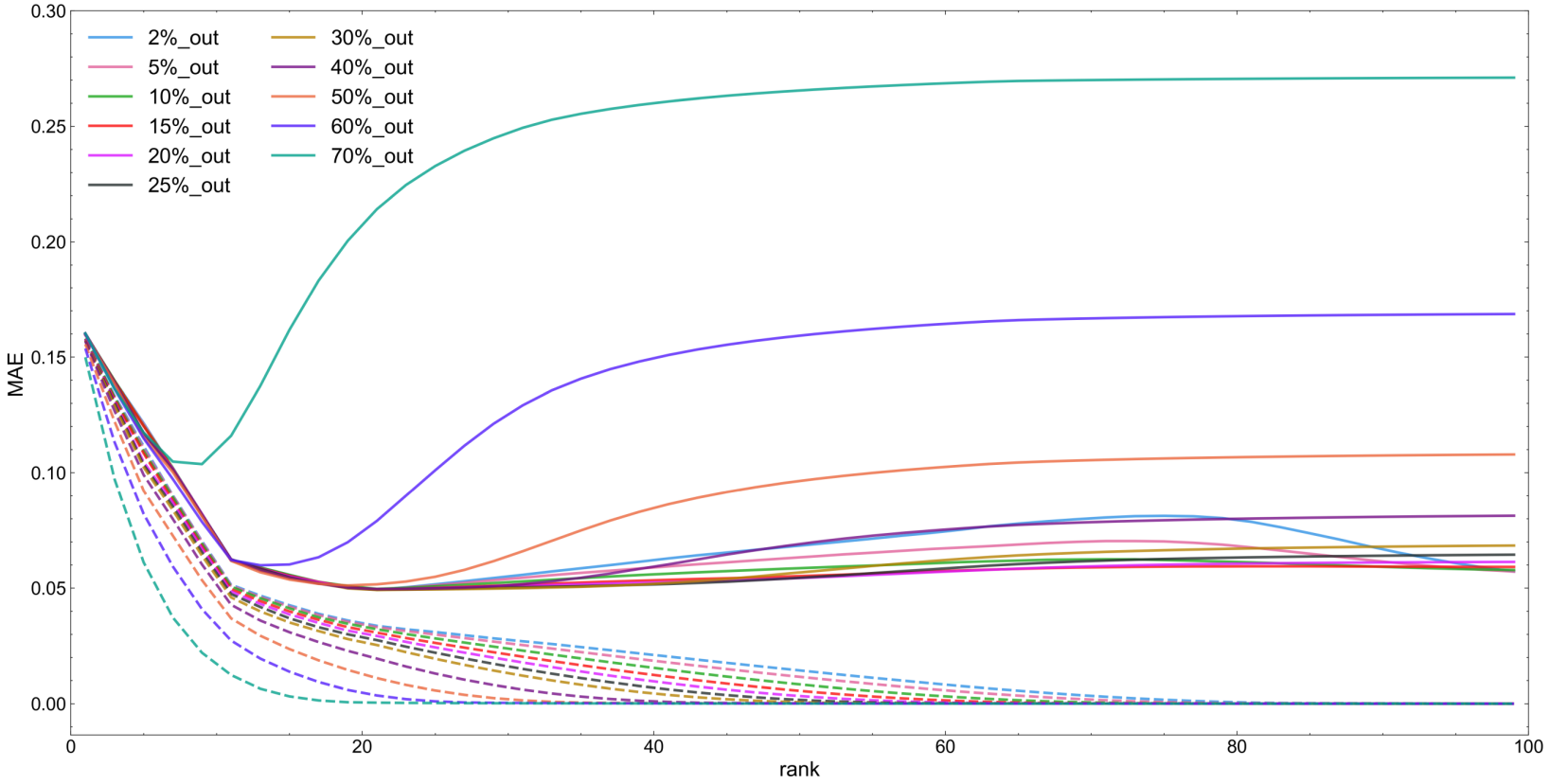


Fig. A.7 Learning curves for synthetic data $\mathbf{X} \in \mathbb{R}^{500\times100}$ at noise level of 0.50 (15% missing data) with different percent of hold-out testing data. The training data is denoted as dashed line and the testing data curve is shown in solid line. MAE is selected as error measurement.

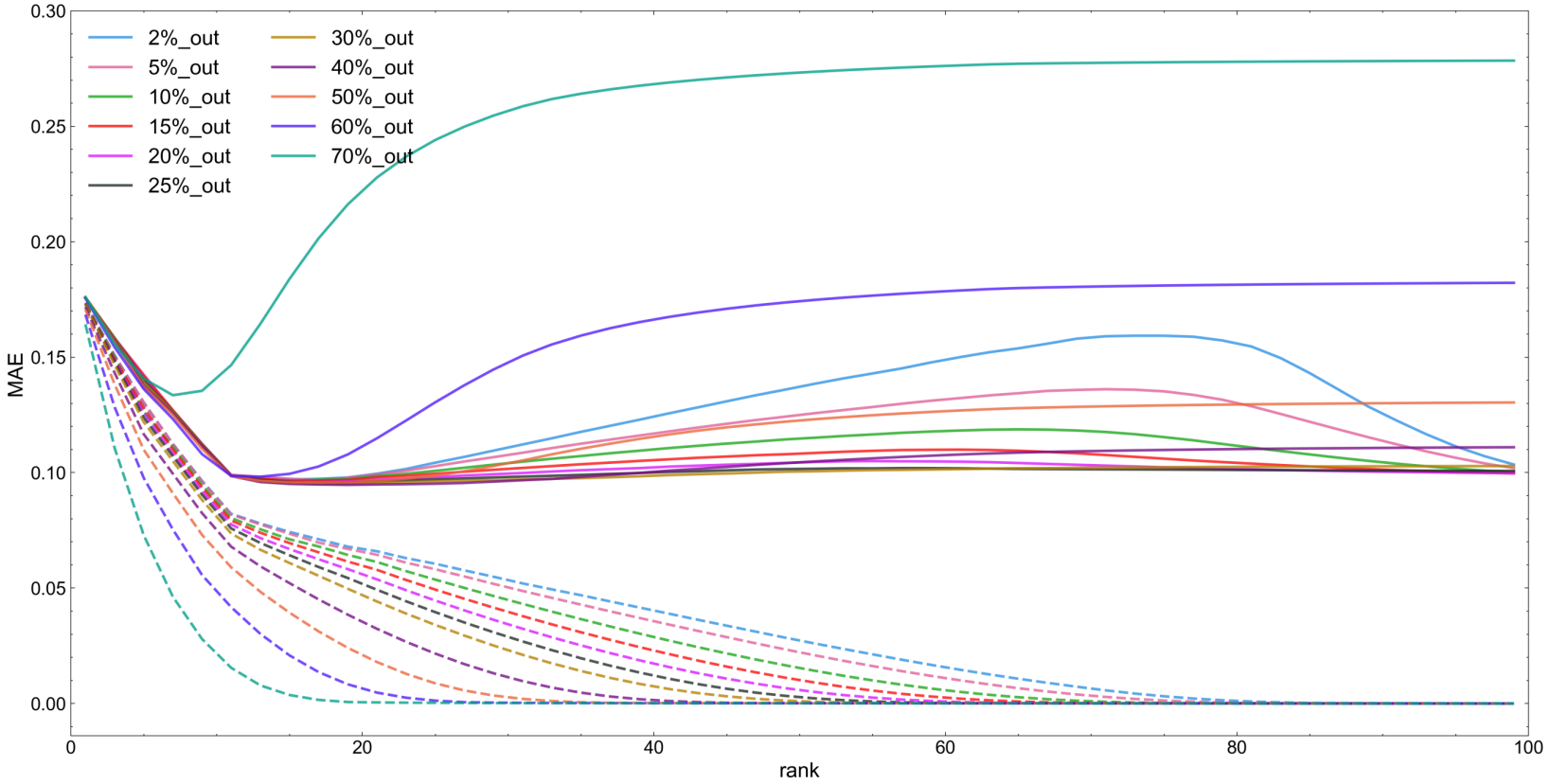


Fig. A.8 Learning curves for synthetic data $\mathbf{X} \in \mathbb{R}^{500\times100}$ at noise level of 1.00 (15% missing data) with different percent of hold-out testing data. The training data is denoted as dashed line and the testing data curve is shown in solid line. MAE is selected as error measurement.

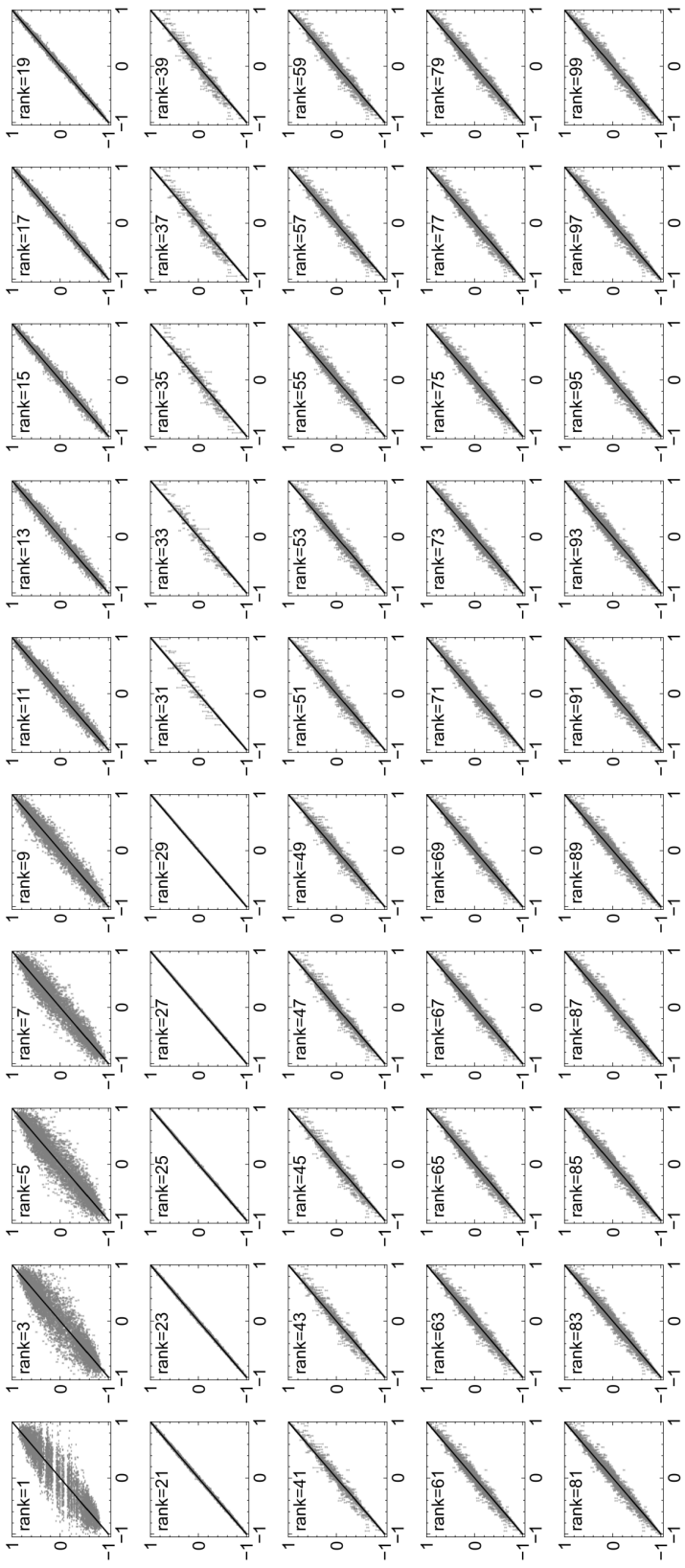


Fig. A.9 Predictions of missing values for synthetic data without noise by holding 2% data out as testing data. The y-axis denotes the averaged predictions for missing values and x-axis denotes the actual values. The error bars in grey are for std values.

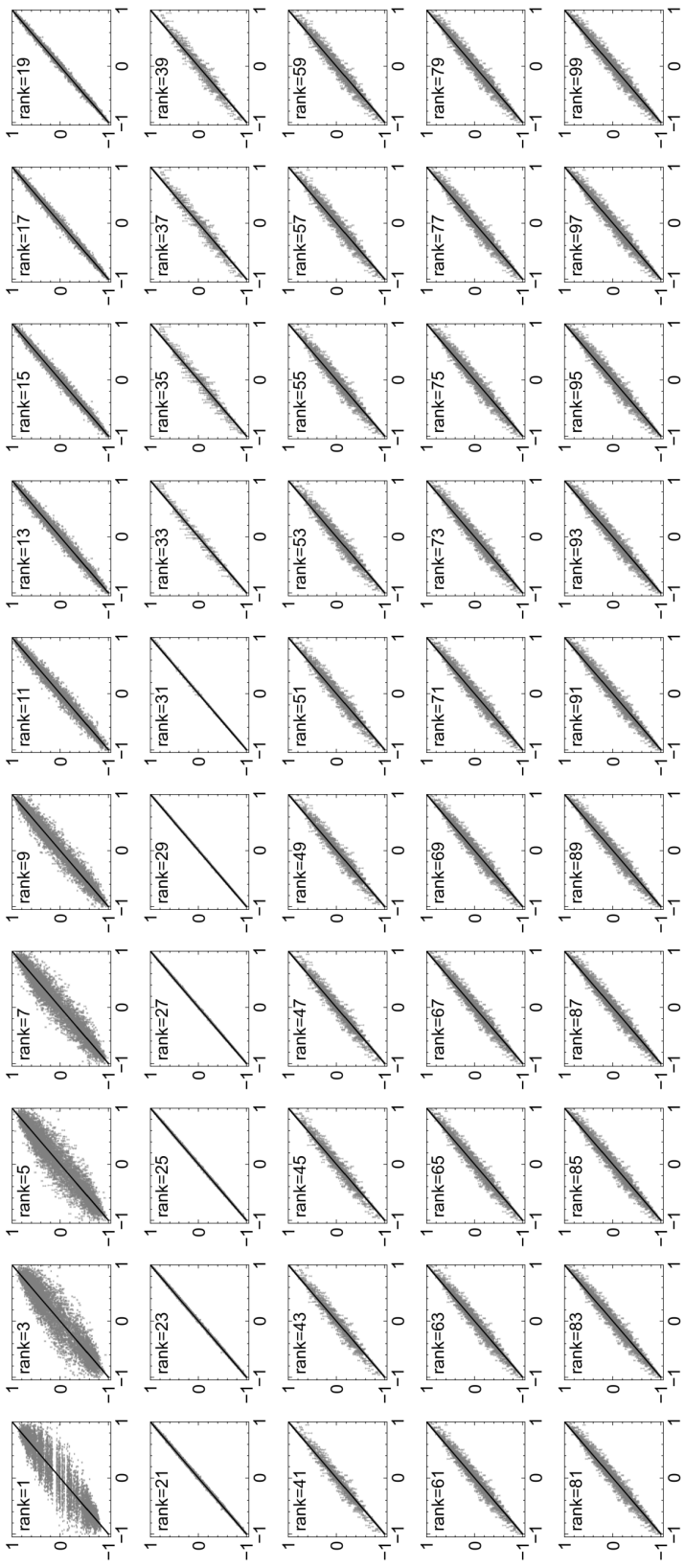


Fig. A.10 Predictions of missing values for synthetic data with noise level of 0.01 by holding 2% data out as testing data. The y-axis denotes the averaged predictions for missing values and x-axis denotes the actual values. The error bars in grey are for std values.

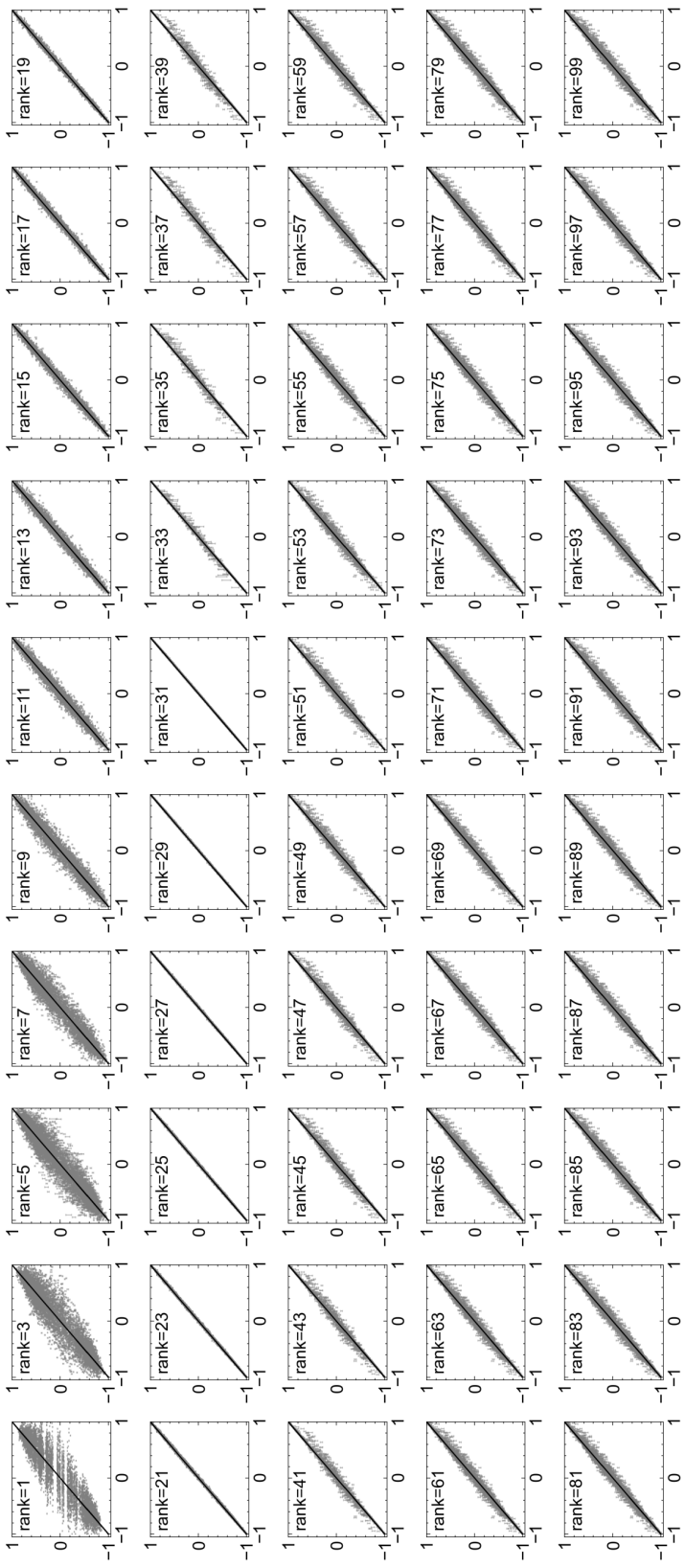


Fig. A.11 Predictions of missing values for synthetic data with noise level of 0.05 by holding 2% data out as testing data. The y-axis denotes the averaged predictions for missing values and x-axis denotes the actual values. The error bars in grey are for std values.

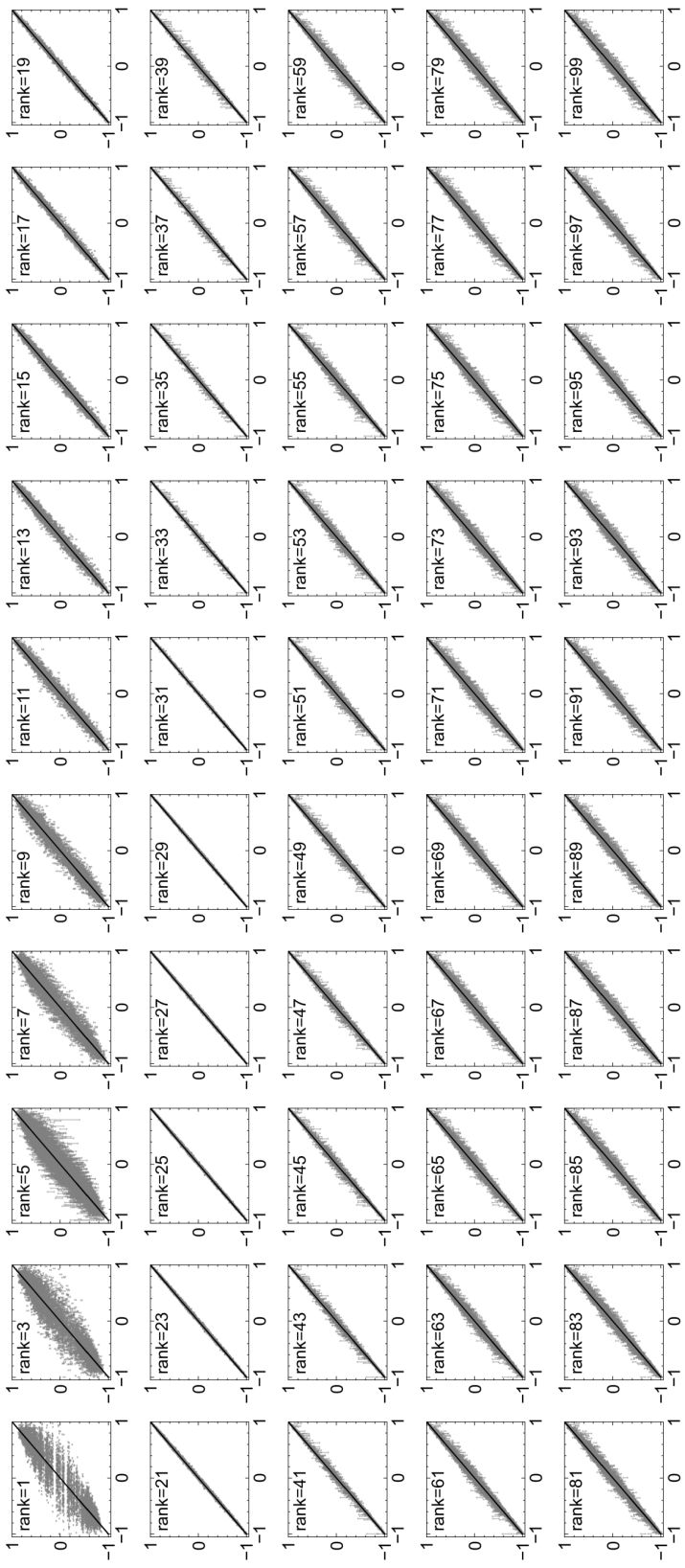


Fig. A.12 Predictions of missing values for synthetic data with noise level of 0.10 by holding 5% data out as testing data. The y-axis denotes the averaged predictions for missing values and x-axis denotes the actual values. The error bars in grey are for std values.

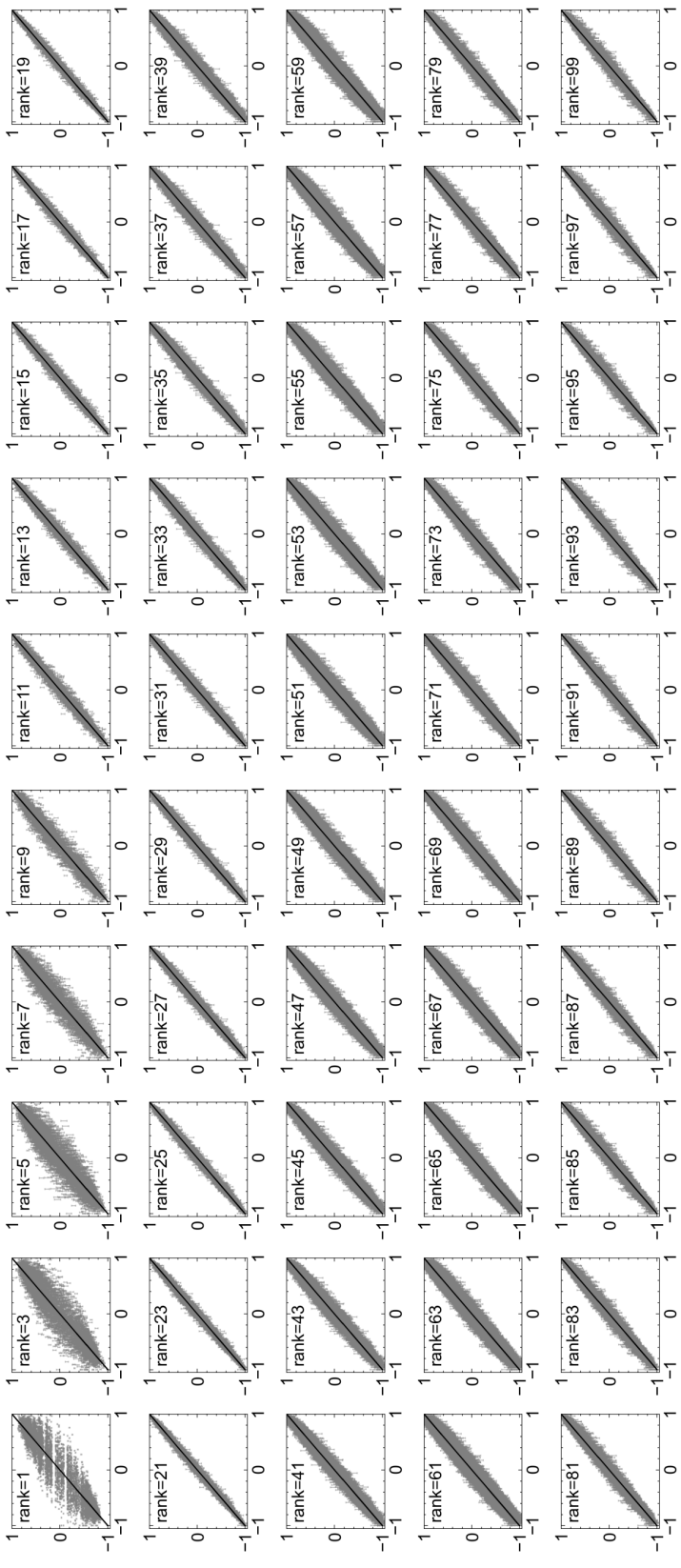


Fig. A.13 Predictions of missing values for synthetic data with noise level of 0.50 by holding 15% data out as testing data. The y-axis denotes the averaged predictions for missing values and x-axis denotes the actual values. The error bars in grey are for std values.

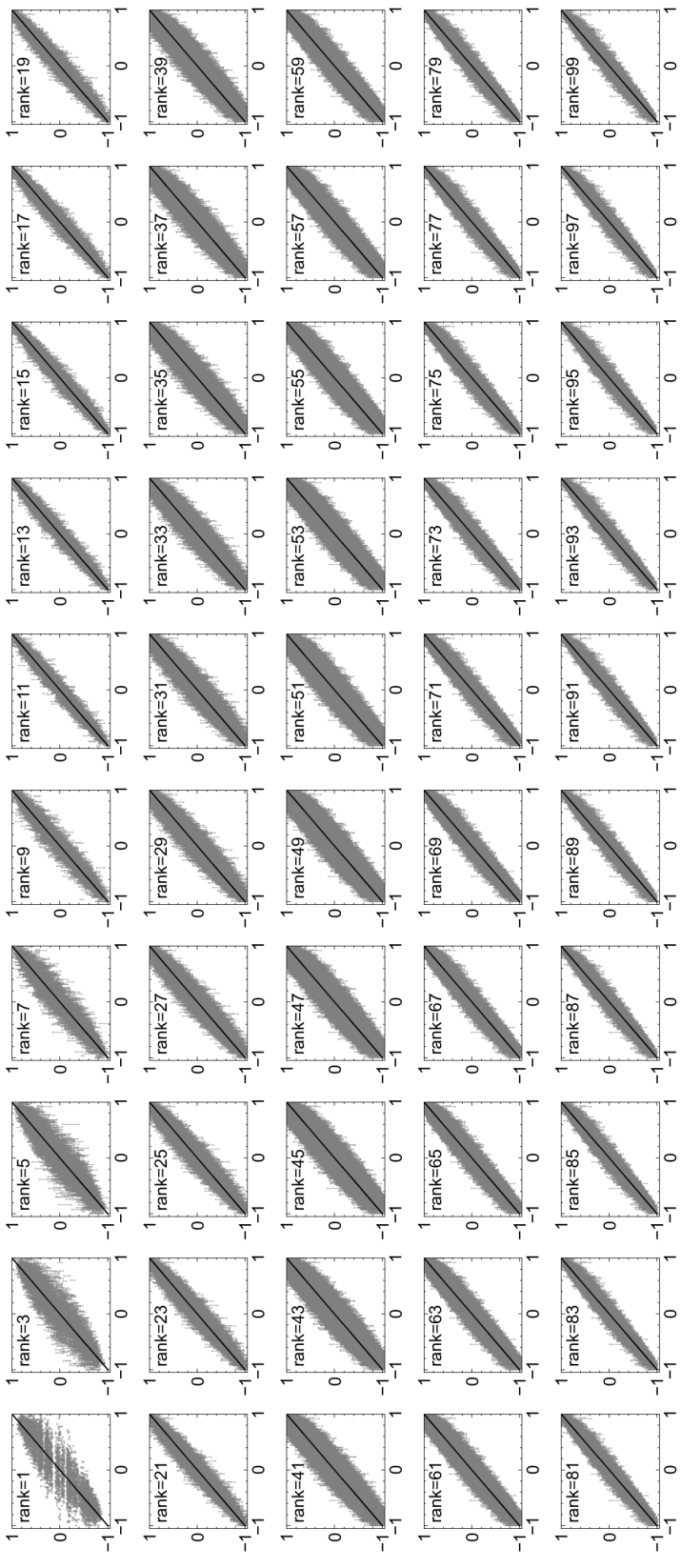


Fig. A.14 Predictions of missing values for synthetic data with noise level of 1.00 by holding 25% data out as testing data. The y-axis denotes the averaged predictions for missing values and x-axis denotes the actual values. The error bars in grey are for std values.

A.2 Welford's Online Algorithm

A standard way of computing the sample variance is by its definition,

$$\sigma^2 = \frac{1}{n(n-1)}\left(n\sum_{i=1}^{n}x_i^2 - \left(\sum_{i=1}^{n}x_i\right)^2\right) \tag{A.1}$$

and taking the square root of the variance returns us the standard deviation (std), σ. Calculating the variance by implementing the sum of squares method, Equation A.1, results to numerical instability and precision loss [1, 2]. Welford's online algorithm provides a solution to compute the variance/std and mean value [3, 4].

The Welford's online algorithm computes the mean and std in a recurrent manner. Suppose we are given a stream of values, $[x_0, x_2, x_3, \cdots, x_t, \cdots]$, where x_t denotes the value of x at time t. In order to get the mean and std, we initialize the mean values M_1 with x_1,

$$M_1 = x_1 \quad k = 1 \tag{A.2}$$

and sum of squares with 0,

$$S_1 = 0 \quad k = 1 \tag{A.3}$$

Then for incoming values x_k, we have

$$M_k = M_{k-1} + (x_k - M_{k-1})/k \quad k \geq 2 \tag{A.4}$$

and

$$S_k = S_{k-1} + (x_k - M_{k-1}) \times (x_k - M_k) \quad k \geq 2 \tag{A.5}$$

The mean value is given by M_k. It is also easy to calculate the sample std with $\sqrt{S_k/(k-1)}$ and population std with $\sqrt{S_k/k}$. Our implementation not only supports of matrices instead of single scalars, but also compatible with sparse matrices, which is achieved by introducing element-wise operations.

A.3 Minimum Description Length

For a generalized linear regression problem (Eq. 34 in [5]), model selection can be achieved by

$$gMDL = \frac{n}{2}\ln S + \frac{p}{2}\ln F + \ln n \tag{A.6}$$

where n is the number of data points, p is number of parameters defined as

$$p = r \times m + r \times n \tag{A.7}$$

Then we can compute S with

$$S = \frac{RSS}{n-p} \tag{A.8}$$

where RSS is the residual sum of squares,

$$RSS = \sum_i \left(\hat{\mathbf{X}}_i - \mathbf{X}_i\right)^2 \tag{A.9}$$

The often-used MSE is just the mean of RSS.

F is the the F-ratio and can be computed with

$$F = \frac{1}{p}\left(\frac{\mathbf{y}\cdot\mathbf{y} - \mathrm{RSS}}{S}\right) \tag{A.10}$$

The optimal rank is selected under the assumption that the coefficient of determination $R^2 >= p/n$.

A.4 Supporting Tables for Allostery Prediction

Table A.1 NMR measurements for EPAC. The perturbation library is composed by integrating different ligands (Figure 2.1) under different concentrations.

concentrations	chemical shift of C atom	chemical shift of H atom	line width of C atom	line width of H atom
4000um	w1_4000um_v5_LW_Rp	w2_4000um_v5_LW_Rp	lw1_4000um_v5_LW_Rp	lw2_4000um_v5_LW_Rp
2000um	w1_2000um_v5_cA	w2_2000um_v5_cA	lw1_2000um_v5_cA	lw2_2000um_v5_cA
5560um	w1_5560um_v5_LW_Rp	w2_5560um_v5_LW_Rp	lw1_5560um_v5_LW_Rp	lw2_5560um_v5_LW_Rp
1000um	w1_1000um_v5_cA	w2_1000um_v5_cA	lw1_1000um_v5_cA	lw2_1000um_v5_cA
700um_	w1_700um_v5_cA	w2_700um_v5_cA	lw1_700um_v5_cA	lw2_700um_v5_cA
4000um	w1_4000um_2-O_v5	w2_4000um_2-O_v5	lw1_4000um_2-O_v5	lw2_4000um_2-O_v5
2000um	w1_2000um_2-O_v5	w2_2000um_2-O_v5	lw1_2000um_2-O_v5	lw2_2000um_2-O_v5
1mM_Sp	w1_1mM_Sp_v5	w2_1mM_Sp_v5	lw1_1mM_Sp_v5	lw2_1mM_Sp_v5
2mM_Sp	w1_2mM_Sp_v5	w2_2mM_Sp_v5	lw1_2mM_Sp_v5	lw2_2mM_Sp_v5
3000um	w1_3000um_2-O_v5	w2_3000um_2-O_v5	lw1_3000um_2-O_v5	lw2_3000um_2-O_v5
3mM_Sp	w1_3mM_Sp_v5	w2_3mM_Sp_v5	lw1_3mM_Sp_v5	lw2_3mM_Sp_v5
WT_APO	w1_WT_APO_v5	w2_WT_APO_v5	lw1_WT_APO_v5	lw2_WT_APO_v5
5000um	w1_5000um_v5_LW_Rp	w2_5000um_v5_LW_Rp	lw1_5000um_v5_LW_Rp	lw2_5000um_v5_LW_Rp

Table A.2 Clustering of residues in EPAC for allostery effect.

residue	pred_labels
V150	1
G151	0
T152	1
H153	1
M155	0
E156	1
E158	1
L159	1
A160	1
E161	1
A162	1
V163	1
A164	1
L165	1
S167	0
R169	0
G170	1
D172	1
A173	0
L174	0
L175	0
T176	1
V177	1
A178	1
L179	1
R180	1
K181	1
G184	1
Q185	1
R186	1
T187	1
D188	1
E189	1
E190	1

Table A.2 (continued)

residue	pred_labels
D192	1
L193	1
I194	1
F195	1
E196	1
E197	1
L198	1
L199	1
H200	1
I201	1
A203	1
A205	1
H206	1
L207	1
S208	1
V211	0
K212	0
R213	1
E214	0
L215	0
A216	0
A217	1
V218	1
L219	0
L220	0
F221	1
E222	1
K226	0
A227	1
G228	1
T229	1
V230	1
L231	1
F232	1

Table A.2 (continued)

residue	pred_labels
S233	1
Q234	1
G235	1
D236	0
K237	1
G238	0
W241	0
I243	0
I244	0
W245	1
K246	0
G247	1
S248	1
V249	1
N250	1
V251	1
V252	1
T253	1
H254	1
G255	0
K256	1
G257	1
L258	1
V259	0
T260	0
T261	1
L262	0
H263	1
E264	0
G265	0
D266	0
D267	1
F268	0
G269	0

Table A.2 (continued)

residue	pred_labels
Q270	0
L271	0
A272	0
L273	0
V274	0
N275	0
D276	0
A277	0
R279	0
A280	0
A281	0
T282	1
I283	1
I284	1
L285	1
R286	1
E287	0
N289	1
C290	1
H291	1
L293	1
R294	1
V295	1
D296	1
K297	1
Q298	0
D299	0
F300	0
N301	0
R302	0
I303	0
I304	1
K305	0
D306	0

Table A.2 (continued)

residue	pred_labels
V307	0
E308	0
K310	0
T311	0
M312	1
R313	0
L314	0
E316	0
G318	0

References

[1] Robert F Ling. Comparison of several algorithms for computing sample means and variances. *Journal of the American Statistical Association*, 69(348):859–866, 1974.

[2] Tony F Chan, Gene H Golub, and Randall J LeVeque. Algorithms for computing the sample variance: Analysis and recommendations. *The American Statistician*, 37(3):242–247, 1983.

[3] Donald E Knuth. *Art of computer programming, volume 2: Seminumerical algorithms*. Addison-Wesley Professional, 2014.

[4] BP Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[5] Mark H Hansen and Bin Yu. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454): 746–774, 2001.

Appendix B

Supporting Information for Procrustes Methods

B.1 Notations and Definitions

Let $\mathbb{R}^{m\times n}$ denote the space of $m \times n$ matrices with real entries. The Frobenius inner product $\mathrm{Tr}(\mathbf{A}^T\mathbf{B})$ which induces the norm $||\mathbf{A}|| := \sqrt{\mathrm{Tr}(\mathbf{A}^T\mathbf{A})}$ on $\mathbb{R}^{n\times m}$.

A square matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ is said to be symmetric if $\mathbf{A}^T = \mathbf{A}$ and said to be orthogonal if $\mathbf{A}^T\mathbf{A} = \mathbf{A}\mathbf{A}^T = \mathbf{I}$ (i.e. $\mathbf{A}^T = \mathbf{A}^{-1}$). An orthogonal matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ is said to be a permutation matrix if its elements are all either zero or one, (i.,e., $p_{ij} \in 0, 1$)) and summation of rows and columns gives one (i.e. $\sum_{j=1}^{n} p_{ij} = \sum_{i=1}^{n} p_{ij} = 1$. An orthogonal matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ is said to be a rotation matrix if its determinant, denoted $|\mathbf{R}|$, is one. Every permutation matrix is a rotation matrix.

Every matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ has a singular value decomposition (SVD) of the form

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \tag{B.1}$$

where that $\mathbf{U} \in \mathbb{R}^{m\times m}$ and $\mathbf{V} \in \mathbb{R}^{n\times n}$ are unitary matrices and $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ is a diagonal matrix with $r = \min\{m, n\}$ non-negative diagonal entries $\sigma_1 \geq \cdots \geq \sigma_r \geq 0$ called the singular values of $\mathbf{A}$ and zeros elsewhere.

B.2 Procrustes Methods

B.2.1 Generic Procrustes

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and a reference matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$, find the unconstrained transformation matrix $\mathbf{T} \in \mathbb{R}^{n\times n}$ that transforms $\mathbf{A}$ to best match $\mathbf{B}$. The optimal transformation matrix $\mathbf{T}_{\text{opt}}$ is given by [1]

$$\mathbf{T}_{\text{opt}} = \arg\min \|\mathbf{AT} - \mathbf{B}\|_F^2 = \left(\mathbf{A}^\dagger\mathbf{A}\right)^{-1}\mathbf{A}^\dagger\mathbf{B} \tag{B.2}$$

If $m < n$, the transformation matrix $\mathbf{T}_{\text{opt}}$ is not unique, because the system of equations is underdetermined (i.e., there are fewer equations than unknowns). When $\mathbf{T}$ is restricted to a special form, such as orthogonal matrix, symmetric matrix, permutation matrix, the one-sided Procrustes problem will evolve into some of the following Procrustes methods.

B.2.2 Orthogonal Procrustes Problem

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and the reference matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$, find the orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{n\times n}$ that rotates and reflects $\mathbf{A}$ to best fit $\mathbf{B}$ [2]. This problem was originally solved by Peter Schönemann in 1964 [3] by taking the singular value decomposition of the product of two matrices, $\mathbf{A}^\dagger\mathbf{B}$. The first step in Schönemann's approach, which is shared in all other forms of Procrustes analysis, is to rewrite the minimization of the distance between the matrices as a maximization of their overlap

$$\begin{aligned}
&\arg \underbrace{\min}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^\dagger\}} \|\mathbf{AQ} - \mathbf{B}\|_F^2 \\
&= \arg \underbrace{\min}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^\dagger\}} \operatorname{Tr}\left[(\mathbf{AQ} - \mathbf{B})^\dagger(\mathbf{AQ} - \mathbf{B})\right] \\
&= \arg \underbrace{\min}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^\dagger\}} \operatorname{Tr}\left[(\mathbf{AQ})^\dagger\mathbf{AQ} - \mathbf{Q}^\dagger\mathbf{A}^\dagger\mathbf{B} - \mathbf{B}^\dagger\mathbf{AQ} + \mathbf{B}^\dagger\mathbf{B}\right] \\
&= \arg \underbrace{\min}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^\dagger\}} \operatorname{Tr}\left[-\mathbf{Q}^\dagger\mathbf{A}^\dagger\mathbf{B} - \mathbf{B}^\dagger\mathbf{AQ}\right] \\
&= \arg \underbrace{\max}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^\dagger\}} \operatorname{Tr}\left[\mathbf{Q}^\dagger\mathbf{A}^\dagger\mathbf{B}\right]
\end{aligned} \tag{B.3}$$

This procedure of this derivation is used here to convert the minimization problem to a maximization problem and will be utilized several times in the following sections. The solution [3, 4] of orthogonal Procrustes problem can be found by taking the singular value decomposition (SVD) of the product,

$$\mathbf{A}^{\dagger}\mathbf{B} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\dagger} \tag{B.4}$$

Now the problem becomes

$$\begin{aligned}
&\max_{\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}} \mathrm{Tr}\left[\mathbf{Q}^{\dagger}\mathbf{A}^{\dagger}\mathbf{B}\right] \\
&= \max_{\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}} \mathrm{Tr}\left[\mathbf{Q}^{\dagger}\tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\dagger}\right] \\
&= \max_{\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}} \mathrm{Tr}\left[\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{\dagger}\mathbf{Q}^{\dagger}\tilde{\mathbf{U}}\right] \\
&= \max_{\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}} \mathrm{Tr}\left[\tilde{\mathbf{\Sigma}}\mathbf{Z}\right] \\
&= \max_{\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}} \sum_{i} \tilde{\mathbf{\Sigma}}_{\mathbf{i},\mathbf{i}}\mathbf{Z}_{i,i}
\end{aligned} \tag{B.5}$$

where $\mathbf{Z} = \tilde{\mathbf{V}}^{\dagger}\mathbf{Q}^{\dagger}\tilde{\mathbf{U}}$. Because matrix $\mathbf{Q}$ is orthogonal, the objective function is maximized when $\mathbf{Z} = \mathbf{I}$. Therefore,

$$\mathbf{Q}_{\mathrm{opt}} = \tilde{\mathbf{U}}\tilde{\mathbf{V}}^{\dagger} \tag{B.6}$$

The same derivation works for complex matrices, but then $\mathbf{Q}$ is a unitary matrix.

This derivation works even for matrices $\mathbf{A}$ and $\mathbf{B}$ that have different matrix dimensions. This is done by zero padding both matrices which adds zeros to the matrix with the smaller number of rows/columns until both of the matrices have the same dimension. Similarly, the last few rows/columns of a matrix that do not contain any information (i.e., they are filled with zeros) are considered as redundant information and are removed.

Translation and scaling are also important operations that will affect the alignment of two objects; users can choose to center and scale matrices for all Procrustes methods. User defined data-point weighting is also implemented as an optional argument for Procrustes methods. This enables one to consider cases where different points (e.g., due to their mass, or the experimental precision with which they were measured) should have different priorities when computing the optimal transformation.

B.2.3 Rotational-Orthogonal Procrustes Problem

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and the reference matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$, find the rotation matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$ that rotates $\mathbf{A}$ to best fit $\mathbf{B}$ [2, 5, 6]. Analogous to the orthogonal Procrustes problem, the optimal rotational matrix $\mathbf{R}_{\text{opt}}$ is given by

$$\mathbf{R}_{\text{opt}} = \arg \underbrace{\min}_{\left\{\mathbf{R}\middle|\substack{\mathbf{R}^{-1}=\mathbf{R}^\dagger \\ |\mathbf{R}|=1}\right\}} \|\mathbf{AR}-\mathbf{B}\|_F^2 = \arg \underbrace{\max}_{\left\{\mathbf{R}\middle|\substack{\mathbf{R}^{-1}=\mathbf{R}^\dagger \\ |\mathbf{R}|=1}\right\}} \operatorname{Tr}\left[\mathbf{R}^\dagger \mathbf{A}^\dagger \mathbf{B}\right] \tag{B.7}$$

The solution is obtained by taking SVD of the product of the matrix,

$$\mathbf{A}^\dagger \mathbf{B} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^\dagger \tag{B.8}$$

$$\mathbf{R}_{\text{opt}} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^\dagger \tag{B.9}$$

The $\tilde{\mathbf{S}}_{n\times m}$ is almost an identity matrix, e.g.,

$$\tilde{\mathbf{S}}_{n\times m} \equiv \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \ddots & \vdots & 0 \\ 0 & \ddots & \ddots & 0 & \vdots \\ 0 & 0 & \ddots & 1 & 0 \\ 0 & 0 & \ddots & 0 & \left(\left|\mathbf{UV}^\dagger\right|\right) \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \end{bmatrix} \tag{B.10}$$

but the last diagonal element is replaced by

$$\operatorname{sgn}\left(\left|\tilde{\mathbf{U}}\tilde{\mathbf{V}}^\dagger\right|\right) = \begin{cases} +1 & \left|\tilde{\mathbf{U}}\tilde{\mathbf{V}}^\dagger\right| \geq 0 \\ -1 & \left|\tilde{\mathbf{U}}\tilde{\mathbf{V}}^\dagger\right| < 0 \end{cases} \tag{B.11}$$

This ensures the determinant of $\mathbf{R}_{\text{opt}}$ is one. Equation B.11 can be simplified into

$$\operatorname{sgn}\left(\left|\tilde{\mathbf{U}}\tilde{\mathbf{V}}^\dagger\right|\right) = \left|\tilde{\mathbf{U}}\tilde{\mathbf{V}}^\dagger\right| \tag{B.12}$$

B.2.4 Symmetric Procrustes Problem

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and the reference $\mathbf{B} \in \mathbb{R}^{m\times n}$, with $m \geqslant n$, find the symmetric matrix $\mathbf{X} \in \mathbb{R}^{n\times n}$ that makes $\mathbf{AX}$ as close as possible to $\mathbf{B}$ [7–9]. The

symmetric Procrustes problem arises when determining the strain matrix of an elastic structure; in this context $\mathbf{X}$ is a symmetric stiffness (or compliance) matrix that is associated with the deformation of $\mathbf{A}$ towards $\mathbf{B}$. I.e., the symmetric Procrustes problem is:

$$\mathbf{X}_{\text{opt}} = \underbrace{\min}_{\{\mathbf{X}|\mathbf{X}=\mathbf{X}^\dagger\}} \|\mathbf{AX}-\mathbf{B}\|_F^2 = \underbrace{\min}_{\{\mathbf{X}|\mathbf{X}=\mathbf{X}^\dagger\}} \text{Tr}\left[(\mathbf{AX}-\mathbf{B})^\dagger(\mathbf{AX}-\mathbf{B})\right] \tag{B.13}$$

The matrix factorization of $\mathbf{A}$ is done with SVD,

$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\begin{bmatrix}\mathbf{\Sigma}_{n\times n}\\ \mathbf{0}_{(m-n)\times n}\end{bmatrix}\mathbf{V}^\dagger_{n\times n} \tag{B.14}$$

where $\mathbf{\Sigma}_{n\times n}$ is a square diagonal matrix with nonnegative elements denoted by σ_i listed in decreasing order. Define

$$\mathbf{C}_{m\times n} = \mathbf{U}^\dagger_{m\times m}\mathbf{B}_{m\times n}\mathbf{V}_{n\times n}. \tag{B.15}$$

with elements denoted c_{ij}. Then we compute the symmetric matrix $\mathbf{Y}\in\mathbb{R}^{n\times n}$ with

$$\mathbf{Y}_{ij} = \begin{cases}0 & i \text{ and } j > \text{rank}(\mathbf{A})\\ \frac{\sigma_i c_{ij}+\sigma_j c_{ji}}{\sigma_i^2+\sigma_j^2} & \text{otherwise}\end{cases} \tag{B.16}$$

It is worth noting that the first part of this definition only applies in the unusual case where $\mathbf{A}$ has rank less than n. The optimal solution $\mathbf{X}_{opt}$ can be obtained with

$$\mathbf{X}_{\text{opt}} = \mathbf{VYV}^\dagger \tag{B.17}$$

B.2.5 Permutation Procrustes Problem

Given a matrix $\mathbf{A}\in\mathbb{R}^{m\times n}$ and the reference matrix $\mathbf{B}\in\mathbb{R}^{m\times n}$, find the permutation matrix $\mathbf{P}\in\mathbb{R}^{n\times n}$ that permutes the columns of $\mathbf{A}$ to best fit $\mathbf{B}$. The optimal permutation matrix $\mathbf{P}_{\text{opt}}$ is given by

$$\mathbf{P}_{\text{opt}} = \underbrace{\arg\min}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\}\\ \sum_{i=1}^n p_{ij}=\sum_{j=1}^n p_{ij}=1}\right\}} \|\mathbf{AP}-\mathbf{B}\|_F^2 = \underbrace{\arg\max}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\}\\ \sum_{i=1}^n p_{ij}=\sum_{j=1}^n p_{ij}=1}\right\}} \text{Tr}\left[\mathbf{P}^\dagger\mathbf{A}^\dagger\mathbf{B}\right], \tag{B.18}$$

where $\mathbf{P}$ is the permutation matrix acting on matrix $\mathbf{A}$.

The solution is to relax the problem into a linear programming problem and note that the solution to a linear programming problem is always at the boundary of the feasible region, which means that the solution can always be written as a permutation matrix,

$$\underbrace{\max}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\}\\ \Sigma_{i=1}^{n} p_{ij}=\Sigma_{j=1}^{n} p_{ij}=1}\right\}} \mathrm{Tr}\left[\mathbf{P}^{\dagger}\mathbf{A}^{\dagger}\mathbf{B}\right] = \underbrace{\max}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\geq 0\\ \Sigma_{i=1}^{n} p_{ij}=\Sigma_{j=1}^{n} p_{ij}=1}\right\}} \mathrm{Tr}\left[\mathbf{P}^{\dagger}\mathbf{A}^{\dagger}\mathbf{B}\right] \tag{B.19}$$

This is a matching problem and can be solved by the Hungarian algorithm [10–12], which solves the assignment problem in polynomial time and which anticipated later primal-dual methods. Note that if $\mathbf{A}$ and $\mathbf{B}$ have different numbers of items, one uses zero-padding.

B.2.6 Two-Sided Orthogonal Procrustes Problem

Given a square matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and a reference matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ of the same dimension, find distinct unitary/orthogonal transformation of the rows and columns of $\mathbf{A}$ that makes it as as close as possible to $\mathbf{B}$ [13]. In other words,

$$\underbrace{\arg\min}_{\left\{\mathbf{Q}_1,\mathbf{Q}_2\middle|\substack{\mathbf{Q}_1^{-1}=\mathbf{Q}_1^{\dagger}\\ \mathbf{Q}_2^{-1}=\mathbf{Q}_2^{\dagger}}\right\}} \|\mathbf{Q}_1^{\dagger}\mathbf{A}\mathbf{Q}_2 - \mathbf{B}\|_F^2 = \underbrace{\arg\max}_{\left\{\mathbf{Q}_1,\mathbf{Q}_2\middle|\substack{\mathbf{Q}_1^{-1}=\mathbf{Q}_1^{\dagger}\\ \mathbf{Q}_2^{-1}=\mathbf{Q}_2^{\dagger}}\right\}} \mathrm{Tr}\left[\mathbf{Q}_2^{\dagger}\mathbf{A}^{\dagger}\mathbf{Q}_1\mathbf{B}\right] \tag{B.20}$$

The two-sided orthogonal Procrustes problem is solved by taking the SVD of the matrices

$$\begin{aligned}\mathbf{A} &= \mathbf{U}_A\mathbf{\Sigma}_A\mathbf{V}_A^{\dagger},\\ \mathbf{B} &= \mathbf{U}_B\mathbf{\Sigma}_B\mathbf{V}_B^{\dagger}.\end{aligned} \tag{B.21}$$

Then,

$$\begin{aligned}\mathbf{Q}_1 &= \mathbf{U}_A\mathbf{U}_B^{\dagger},\\ \mathbf{Q}_2 &= \mathbf{V}_A\mathbf{V}_B^{\dagger}.\end{aligned} \tag{B.22}$$

B.2.7 Two-Sided Orthogonal Procrustes Problem with One Transformation

This problem arises when the transformation matrices of the two-sided orthogonal Procrustes are forced to be the same, $\mathbf{Q}_1 = \mathbf{Q}_2$ [13]. This variation of the two-sided orthogonal Procrustes problem arises primarily for symmetric/Hermitian matrices, where the rows and columns of the matrix are equivalent. Formally, for symmetric matrices $\mathbf{A}$ and $\mathbf{B}$ in $\mathbb{R}^{n\times n}$,

$$\underbrace{\arg\min}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}\}} \|\mathbf{Q}^{\dagger}\mathbf{A}\mathbf{Q}-\mathbf{B}\|_F^2 = \underbrace{\arg\max}_{\{\mathbf{Q}|\mathbf{Q}^{-1}=\mathbf{Q}^{\dagger}\}} \operatorname{Tr}\left[\mathbf{Q}^{\dagger}\mathbf{A}^{\dagger}\mathbf{Q}\mathbf{B}\right]. \tag{B.23}$$

Because the matrices are symmetric, their eigenvalue decomposition has the form,

$$\begin{aligned}\mathbf{A} &= \mathbf{U}_A\mathbf{\Lambda}_A\mathbf{U}_A^{\dagger},\\ \mathbf{B} &= \mathbf{U}_B\mathbf{\Lambda}_B\mathbf{U}_B^{\dagger}.\end{aligned} \tag{B.24}$$

The solution to the two-sided orthogonal Procrustes problem with one transformation is then

$$\mathbf{Q} = \mathbf{U}_A\mathbf{S}\mathbf{U}_B^{\dagger}, \tag{B.25}$$

where $\mathbf{S}$ is any diagonal matrix with entries from $\{-1, 1\}$,

$$\mathbf{S} = \begin{bmatrix} \pm 1 & 0 & \cdots & 0 \\ 0 & \pm 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \pm 1 \end{bmatrix}. \tag{B.26}$$

Since any choice of $\mathbf{S}$ provides an optimal solution, our software package chooses $\mathbf{S}$ to be the identity matrix.

B.2.8 Two-Sided Permutation Procrustes Problem

Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ and the reference matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$, find permutations of the rows and columns of $\mathbf{A}$ so that it best-matches $\mathbf{B}$. In other words, given permutation matrices $\mathbf{P}_1 \in \mathbb{R}^{m\times m}$ and $\mathbf{P}_2 \in \mathbb{R}^{n\times n}$

$$\begin{aligned} &\arg\min \|\mathbf{P}_1\mathbf{A}\mathbf{P}_2 - \mathbf{B}\|_F^2 \\ =\ &\arg\max \mathrm{Tr}\left[\mathbf{B}^\dagger\mathbf{P}_1\mathbf{A}\mathbf{P}_2\right] \\ =\ &\arg\max \mathrm{Tr}\left[\mathbf{P}_1\mathbf{A}\mathbf{P}_2\mathbf{B}^\dagger\right] \end{aligned} \tag{B.27}$$

This problem can be solved by the flip-flop algorithm [14] or k-opt algorithm.

Algorithm 1 Flip-flop Algorithm for Two-sided Permutation Procrustes

function TSPP_FLIP_FLOP($\mathbf{A}, \mathbf{B}, iteration, \epsilon$)

Require: $\mathbf{A}$, $\mathbf{B}$ are matrices, and ϵ is the threshold value

Ensure: Two permutation matrices $\mathbf{P}_1$ and $\mathbf{P}_2$

$k \leftarrow 0$

$\mathbf{P}_1^0 \leftarrow \mathbf{I}_m$ (or $\mathbf{P}_2^0 \leftarrow \mathbf{I}_n$)

$\mathbf{P}_2^0 \leftarrow \arg\max \mathrm{Tr}\left[\mathbf{B}^\dagger\mathbf{A}\mathbf{P}_2^0\right]$ (or $\mathbf{P}_1^0 \leftarrow \arg\max \mathrm{Tr}\left[(\mathbf{P}_1^0)^\dagger\mathbf{A}\mathbf{B}^\dagger\right]$)

$\rho_0 \leftarrow \|\mathbf{P}_1\mathbf{A}\mathbf{P}_2 - \mathbf{B}\|_F^2$

while $\rho_k > \epsilon$ AND $k < iteration$ **do**

$k = k + 1$

$\mathbf{P}_1^k \leftarrow \arg\max \mathrm{Tr}\left[\mathbf{P}_1^k\mathbf{A}\mathbf{P}_2^{k-1}\mathbf{B}^\dagger\right]$ (or $\mathbf{P}_2^k \leftarrow \arg\max \mathrm{Tr}\left[\mathbf{B}^\dagger\mathbf{P}_1^{k-1}\mathbf{A}\mathbf{P}_2^k\right]$)

$\rho_k \leftarrow \|\mathbf{P}_1^k\mathbf{A}\mathbf{P}_2^{k-1} - \mathbf{B}\|_F^2$ (or $\rho_k \leftarrow \|\mathbf{P}_1^{k-1}\mathbf{A}\mathbf{P}_2^k - \mathbf{B}\|_F^2$)

if $\rho_{k-1} - \rho_k \leq \epsilon$ **then**

break

$\mathbf{P}_2^k \leftarrow \arg\max \mathrm{Tr}\left[\mathbf{B}^\dagger\mathbf{P}_k\mathbf{A}\mathbf{P}_2^k\right]$ (or $\mathbf{P}_1^k \leftarrow \arg\max \mathrm{Tr}\left[\mathbf{P}_1^k\mathbf{A}\mathbf{P}_2^k\mathbf{B}^\dagger\right]$)

$\rho_k \leftarrow \|\mathbf{P}_1^k\mathbf{A}\mathbf{P}_2^k - \mathbf{B}\|_F^2$

return $\mathbf{P}_1$, $\mathbf{P}_2$

B.2.9 Two-sided Permutation Procrustes Problem with One Transformation

This problem arises when the permutation matrices of the two-sided permutation Procrustes are forced to be the same, $\mathbf{P}_1 = \mathbf{P}_2$ [3]. The case when both $\mathbf{A}$ and $\mathbf{B}$ are symmetric is dealt first, followed by the non-symmetric case.

Assuming $\mathbf{A}$ and $\mathbf{B}$ are both symmetric, the goal is to solve the following

$$\underbrace{\arg\min}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\}\\ \sum_{i=1}^{n} p_{ij}=\sum_{j=1}^{n} p_{ij}=1}\right\}} \|\mathbf{P}^{\dagger}\mathbf{A}\mathbf{P}-\mathbf{B}\|_F^2 = \underbrace{\arg\max}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\}\\ \sum_{i=1}^{n} p_{ij}=\sum_{j=1}^{n} p_{ij}=1}\right\}} \operatorname{Tr}\left[\mathbf{P}^{\dagger}\mathbf{A}^{\dagger}\mathbf{P}\mathbf{B}\right]. \tag{B.28}$$

There are no polynomial-cost methods for solving the two-sided permutation Procrustes (TSPP) exactly. The practical approach [15] we considered here have three main steps:

1. Make an initial guess of a permutation matrix, usually based on some heuristic,
2. Iteratively update the permutation matrix,
3. Solve for the closest permutation matrix.

Initial Guesses Given an initial guess, various optimizers can be used to identify a nearby local minimum [15]. The following outlines four different ways of obtaining initial guesses.

The first two are based on a heuristic due to Umeyama's approach [16]. Specifically, note that permutation matrices are a subset of orthogonal matrices and thus one can obtain an upper-bound to this problem B.28 by treating it as the same optimization problem except it is optimized over orthogonal matrices (B.2.7). However, for the two-sided orthogonal Procrustes problem there are 2^n choices of $\mathbf{S}$ to choose from. Further, when the solution is exact there exists at-least one $\mathbf{S}$ that results in $\mathbf{U}_A\mathbf{S}\mathbf{U}_B^{\dagger}$ being a permutation matrix. Umeyama suggested to take the element-wise absolute value of the elements of the unitary transformations

$$\mathbf{U}_{\text{Umeyama}} = \operatorname{abs}(\mathbf{U}_A)\cdot\operatorname{abs}(\mathbf{U}_B^{\dagger}), \tag{B.29}$$

then solve for the closest permutation matrix to $\mathbf{U}_{\text{Umeyama}}$ (cf. B.2.5). This solution is optimal when both $\mathbf{A}$ and $\mathbf{B}$ satisfy $\mathbf{P}^{\dagger}\mathbf{A}\mathbf{P} = \mathbf{B}$ for some permutation matrix $\mathbf{P}$. Otherwise, this provides a initial guess for the iterative procedures.

This matrix $\mathbf{U}_{\text{Umeyama}}$ can be used to find a second initial guess by first obtaining the singular value decomposition of $\mathbf{U}_{\text{Umeyama}} = \tilde{\mathbf{U}}\tilde{\mathbf{E}}\tilde{\mathbf{V}}$. Then define a unitary matrix $\mathbf{U}^{\text{approx}}_{\text{Umeyama}} = \tilde{\mathbf{U}}\tilde{\mathbf{V}}^{\dagger}$ and similarly find the closest permutation matrix to it.

The next strategy for obtaining another initial guess is to convert the TSPP to an one-sided permutation Procrustes problem. This is done by considering the entries in a given row/column to be the properties of an entity [17]. One can then build the auxiliary matrices, $\mathbf{A}^0$ for $\mathbf{A}$ and $\mathbf{B}^0$ for $\mathbf{B}$, with columns

$$\begin{bmatrix} a_{ii} \\ p \cdot \text{sgn}(a_{ij_{max}}) \underbrace{\max}_{1\le j\le n}(|a_{ij}|) \\ p^2 \cdot \text{sgn}(a_{ij_{max-1}}) \underbrace{\text{max-1}}_{1\le j\le n}(|a_{ij}|) \\ p^3 \cdot \text{sgn}(a_{ij_{max-2}}) \underbrace{\text{max-2}}_{1\le j\le n}(|a_{ij}|) \\ \vdots \end{bmatrix}. \tag{B.30}$$

Here the subscript $(max-1)$ denotes the second-largest element in absolute value, $(max-2)$ is the third-largest element in absolute value, *etc.* The first row of $\mathbf{A}^0$ is filled with diagonal element of $\mathbf{A}$. The second row has the largest off-diagonal element in row $\mathbf{i}$ of $\mathbf{A}$, *etc.* The weighting factor $p \in (0,1)$ ensures that the smaller off-diagonal elements become progressively less important (the default value of p is $2^{-\frac{1}{2}}$). The map sgn is the sign function which extracts the sign of a real number, e.g,

$$\text{sgn}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ 1 & \text{if } x > 0 \end{cases}. \tag{B.31}$$

Once we get $\mathbf{A}^0$ for $\mathbf{A}$ and $\mathbf{B}^0$ for $\mathbf{B}$, we can compute the third initial guess for the optimal permutation matrix by solving the one-sided permutation Procrustes problem

$$\underbrace{\arg\min}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\} \\ \sum_{i=1}^n p_{ij}=\sum_{j=1}^n p_{ij}=1}\right\}} \|\mathbf{A}^0\mathbf{P} - \mathbf{B}^0\|_F^2 = \underbrace{\arg\max}_{\left\{\mathbf{P}\middle|\substack{p_{ij}\in\{0,1\} \\ \sum_{i=1}^n p_{ij}=\sum_{j=1}^n p_{ij}=1}\right\}} \text{Tr}\left[\mathbf{P}^{\dagger}\mathbf{A}^{0\dagger}\mathbf{B}^0\right] \tag{B.32}$$

The fourth method to construct the initial guess is motivated by the fact that, in many contexts, the diagonal element can be considered as a label for the entry

(e.g., an atom type). In these cases, it is helpful to encode not only the off-diagonal elements, but also the corresponding diagonal elements. We do so by replacing each row in Equation B.32 by two rows, one of which lists the diagonal elements and the other of which lists the associated off-diagonal element [17],

$$\begin{bmatrix} a_{ii} \\ p \cdot a_{j_{max} j_{max}} \\ p \cdot \mathrm{sgn}(a_{i j_{max}}) \underbrace{\max}_{1 \le j \le n} (|a_{ij}|) \\ p^2 \cdot a_{j_{max-1} j_{max-1}} \\ p^2 \cdot \mathrm{sng}(a_{i j_{max-1}}) \underbrace{\text{max-1}}_{1 \le j \le n} (|a_{ij}|) \\ \vdots \end{bmatrix} \tag{B.33}$$

As before, one can then determine an approximate permutation using the one-sided permutation Procrustes method.

Algorithm 2 Two-sided Permutation Procrustes for Undirected Graph Matching

function TSPP_UNDIRECTED($\mathbf{A}, \mathbf{B}, \mathbf{P}_{guess}, miter, tol$)
Require: $\mathbf{A}$ and $\mathbf{B}$ are matrices, $\mathbf{P}_{guess}$ is the initial guess of permutation matrix, $miter$ is the maximum number of iteration and tol is the tolerance
Ensure: Local minimum two-sided permutation matrix $\mathbf{P}$
    $\mathbf{P} \leftarrow \arg\min \left\| \mathbf{P}^T \tilde{\mathbf{A}} \mathbf{P} - \tilde{\mathbf{B}} \right\|_F^2$ ▷ Solve the symmetric case
    if $\mathbf{P}_{guess}$ is None **then**
        $\mathbf{P} \leftarrow \mathbf{I}$
    else
        $\mathbf{P} \leftarrow \mathbf{P}_{guess}$
    $step \leftarrow 0$
    $change \leftarrow \infty$
    while $change > tol$ and $step \le miter$ **do**
        $\alpha = \frac{\mathbf{P}^T \mathbf{APB} + (\mathbf{P}^T \mathbf{APB})^T}{2}$
        $\mathbf{T} = \mathbf{APB}$
        $\mathbf{P}_{ij} = \mathbf{P}_{ij} \sqrt{\frac{(\mathbf{APB})_{ij}}{(\mathbf{P}\alpha)_{ij}}}$
        $change \leftarrow \mathrm{Tr}\left[(\mathbf{P}^{(n+1)} - \mathbf{P}^{(n)})^T (\mathbf{P}^{(n+1)} - \mathbf{P}^{(n)}) \right]$
        $step \leftarrow step + 1$
    $\mathbf{P} \leftarrow \arg\min_{\tilde{\mathbf{P}}} \left\| \tilde{\mathbf{P}} - \mathbf{P} \right\|_F^2$
    return $\mathbf{P}$

Directed Graph Matching When $\mathbf{A}$ and $\mathbf{B}$ are both not symmetric, then one can use an algorithm based on directed graph matching instead [15],

Algorithm 3 Two-sided Permutation Procrustes for Directed Graph Matching

function TSPP_DIRECTED($\mathbf{A}, \mathbf{B}, \mathbf{P}_{guess}, miter, tol$)
Require: $\mathbf{A}$ and $\mathbf{B}$ are matrices, $\mathbf{P}_{guess}$ is the initial guess of permutation matrix, $miter$ is the maximum number of iteration and tol is the tolerance
Ensure: Local minimum two-sided permutation matrix $\mathbf{P}$
    $\tilde{\mathbf{A}} \leftarrow \frac{\mathbf{A}+\mathbf{A}^T}{2} + i\frac{\mathbf{A}-\mathbf{A}^T}{2}$ ▷ $i = \sqrt{-1}$
    $\tilde{\mathbf{B}} \leftarrow \frac{\mathbf{B}+\mathbf{B}^T}{2} + i\frac{\mathbf{B}-\mathbf{B}^T}{2}$
    $\mathbf{P} \leftarrow \arg\min \left\|\mathbf{P}^T\tilde{\mathbf{A}}\mathbf{P} - \tilde{\mathbf{B}}\right\|_F^2$
    if $\mathbf{P}_{guess}$ is None **then**
        $\mathbf{P} \leftarrow \mathbf{I}$
    else
        $\mathbf{P} \leftarrow \mathbf{P}_{guess}$
    $step \leftarrow 0$
    $change \leftarrow \infty$
    while $change > tol$ and $step \leq miter$ **do**
        $\alpha = \frac{\mathbf{P}^T(\mathbf{A}\mathbf{P}\mathbf{B}^T + \mathbf{A}^{\mathrm{TPB}}) + (\mathbf{A}\mathbf{P}\mathbf{B}^T + \mathbf{A}^{\mathrm{TPB}})\mathbf{P}}{4}$
        $\mathbf{P}_{ij} \leftarrow \mathbf{P}_{ij}\sqrt{\frac{(\mathbf{A}\mathbf{P}\mathbf{B}^T + \mathbf{A}^T\mathbf{P}\mathbf{B})_{ij}}{(2\mathbf{P}\alpha)_{ij}}}$
        $change \leftarrow \mathrm{Tr}\left[(\mathbf{P}^{(n+1)} - \mathbf{P}^{(n)})^T(\mathbf{P}^{(n+1)} - \mathbf{P}^{(n)})\right]$
        $step \leftarrow step + 1$
    $\mathbf{P} \leftarrow \arg\min_{\tilde{\mathbf{P}}} \left\|\tilde{\mathbf{P}} - \mathbf{P}\right\|_F^2$
    return $\mathbf{P}$

The non-symmetric case is harder than the symmetric case and similarly this algorithm does not guarantee an optimal solution. When $\mathbf{A}$ and $\mathbf{B}$ are both symmetric, then the Directed Graph Matching algorithm reduces to the previous algorithm with the initial guess chosen to be the closest Permutation matrix to $\mathbf{U}_{Umeyama}$.

B.2.10 Softassign

The softassign method was developed for the quadratic assignment problem in 1993, [18] and has subsequently been developed theoretically[19, 20] and extended to combinatorial optimization problems, graph matching, photometric calibration, multiple object tracking, *etc.* [19, 21–24]. Because the two-sided permutation

Procrustes problem is a special quadratic assignment problem it can be used here. First one assigns an energy functional [19, 25] and the objective function is defined in Eq. B.34.

$$\begin{aligned} E_{qap}(\mathbf{M}, \mu, \nu) = &- \frac{1}{2} \sum_{aibj} \mathbf{C}_{ai;bj} \mathbf{M}_{ai} \mathbf{M}_{bj} \\ &+ \sum_{a} \mu_a \left(\sum_{i} \mathbf{M}_{ai} - 1 \right) + \sum_{i} \nu_i \left(\sum_{a} \mathbf{M}_{ai} - 1 \right) \\ &- \frac{\gamma}{2} \sum_{ai} \mathbf{M}_{ai}^2 + \frac{1}{\beta} \sum_{ai} \mathbf{M}_{ai} \log \mathbf{M}_{ai} \end{aligned} \tag{B.34}$$

Here, the $\mathbf{M} \in \mathbb{R}^{N \times N}$ is the permutation matrix. The constraints are imposed with Lagrange multipliers μ and ν and the self-amplification term is scaled by γ; this term modulates the response to ensure that the global minimum is found. The barrier function (entropy function), $\mathbf{M}_{ai} \log \mathbf{M}_{ai}$, guarantees the positively of $\mathbf{M}_{ai}$ using the deterministic annealing control variables β, which is the inverse temperature. When the temperature $T = \frac{1}{\beta}$ is high enough, the object function is nearly convex [26]. $\mathbf{C}_{ai;bj}$ represents the benefit matrix and, for two-sided permutation Procrustes problems with one transformation, it is simply the Kronecker product of $\mathbf{A}$ and $\mathbf{B}$:

$$\mathbf{C}_{ai;bj} = \mathbf{A}_{ab} \otimes \mathbf{B}_{ij} \tag{B.35}$$

Algorithm 4 Softassign for Two-sided Permutation Procrustes

function SOFTASSIGN($A, B, \beta_0, \beta_f, iteration_r, iteration_s, tol$)

Require: A and B are matrices, β_0, β_f are the initial and final inverse temperature respectively, $miter_r$ is maximum number of steps in relaxation, $miter_s$ is maximum number of steps in Sinkhorn normalization and tol is the threshold value

Ensure: Permutation matrix $\mathbf{P}$

    $\beta \leftarrow \beta_0$

    $M_{ai} \leftarrow \frac{1}{N} + \epsilon_{ai}$

    while $\beta \leq \beta_f$ **do** ▷ Deterministic annealing

        while $\mathbf{M}_{ai}$ is not converged or iteration number $\leq miter_r$ **do** ▷ Relaxation

            $\mathbf{Q}_{ai} \leftarrow \sum_{bj} \mathbf{C}_{ai;bj}\mathbf{M}_{bj} + \gamma\mathbf{M}_{ai}$

            while $\mathbf{M}_{ai}$ not converged **do** ▷ Softassign

                $\mathbf{M}_{ai} \leftarrow \exp(\beta\mathbf{Q}_{ai})$

                while $\mathbf{M}_{ai}$ not converged or iteration number $\leq miter_s$ **do** ▷ Sinkhorn

                    $\mathbf{M}_{ai} \leftarrow \frac{\mathbf{M}_{ai}}{\sum_i \mathbf{M}_{ai}}$ ▷ Column normalization

                    $\mathbf{M}_{ai} \leftarrow \frac{\mathbf{M}_{ai}}{\sum_a \mathbf{M}_{ai}}$ ▷ Row normalization

        $\beta \leftarrow \beta_r\beta_f$

    return P ▷ **P** is a permutation matrix

References

[1] John C. Gower. Procrustes methods. *WIREs Computational Statistics*, 2(4): 503–508, 2010. doi: 10.1002/wics.107. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/wics.107.

[2] John C Gower and Garmt B Dijksterhuis. *Procrustes problems*, volume 30. Oxford University Press, 2004.

[3] Peter H. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1):1–10, Mar 1966. ISSN 1860-0980. doi: 10.1007/BF02289451.

[4] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, Nov 2000. ISSN 0162-8828. doi: 10.1109/34.888718.

[5] Frank B Brokken. Orthogonal procrustes rotation maximizing congruence. *Psychometrika*, 48(3):343–352, 1983.

[6] JL Farrell, JC Stuelpnagel, RH Wessner, JR Velman, and JE Brook. A least squares estimate of satellite attitude (grace wahba). *SIAM Review*, 8(3):384–386, 1966.

[7] Nicholas J Higham. The symmetric procrustes problem. *BIT Numerical Mathematics*, 28(1):133–143, 1988.

[8] René Escalante and Marcos Raydan. Dykstra's algorithm for constrained least-squares rectangular matrix problems. *Computers & Mathematics with Applications*, 35(6):73–79, 1998.

[9] Juan Peng, Xi-Yan Hu, and Lei Zhang. The (m, n)-symmetric procrustes problem. *Applied Mathematics and Computation*, 198(1):24–34, 2008.

[10] Harold W Kuhn. Variants of the hungarian method for assignment problems. *Naval Research Logistics Quarterly*, 3(4):253–258, 1956.

[11] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics*, 2(1-2):83–97, 1955.

[12] Nick Higham and Pythagoras Papadimitriou. Matrix procrustes problems. *Rapport technique, University of Manchester*, 1995.

[13] Peter H Schönemann. On two-sided orthogonal procrustes problems. *Psychometrika*, 33(1):19–33, 1968.

[14] Pythagoras Papadimitriou. *Parallel solution of SVD-related problems, with applications*. PhD thesis, University of Manchester, 1993.

[15] Chris Ding, Tao Li, and Michael I Jordan. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 183–192. IEEE, 2008.

[16] Shinji Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

[17] Farnaz Heidar Zadeh and Paul W Ayers. Molecular alignment as a penalized permutation procrustes problem. *Journal of Mathematical Chemistry*, 51(3): 927–936, 2013.

[18] JJ Kosowsky and Alan L Yuille. The invisible hand algorithm: Solving the assignment problem with statistical physics. *Neural Networks*, 7(3):477–490, 1994.

[19] Steven Gold and Anand Rangarajan. Softassign versus softmax: Benchmarks in combinatorial optimization. In *Advances in Neural Information Processing Systems*, pages 626–632, 1996.

[20] Anand Rangarajan, Alan L Yuille, Steven Gold, and Eric Mjolsness. A convergence proof for the softassign quadratic assignment algorithm. In *Advances in Neural Information Processing Systems*, pages 620–626, 1997.

[21] Jiahui Wang, Xiaoshuang Zeng, Wenjie Luo, and Wei An. The application of neural network in multiple object tracking. *DEStech Transactions on Computer Science and Engineering*, (csse):358–264, 2018.

[22] Steven Gold, Anand Rangarajan, et al. Softmax to softassign: Neural network algorithms for combinatorial optimization. *Journal of Artificial Neural Networks*, 2(4):381–399, 1996.

[23] Yu Tian, Junchi Yan, Hequan Zhang, Ya Zhang, Xiaokang Yang, and Hongyuan Zha. On the convergence of graph matching: Graduated assignment revisited. In *European Conference on Computer Vision*, pages 821–835. Springer, 2012.

[24] Z Sheikhbahaee, R Nakajima, T Erben, P Schneider, H Hildebrandt, and AC Becker. Photometric calibration of the combo-17 survey with the softassign procrustes matching method. *Monthly Notices of the Royal Astronomical Society*, 471(3):3443–3455, 2017.

[25] Alan L Yuille and JJ Kosowsky. Statistical physics algorithms that converge. *Neural Computation*, 6(3):341–356, 1994.

[26] Stefan Roth. Analysis of a deterministic annealing method for graph matching and quadratic assignment problems in computer vision. *Master's thesis, CVGPR-group, University of Mannheim*, 2001.

Appendix C

Supporting Information of Δ-Learning for Hydration Free Energy Prediction

Table C.1 List of QM Descriptors

QM descriptor type number	**QM descriptors**	**comment**
1	dipole moment	dipole moment in X, Y, Z axis and total
2	quadrupole moment	quadrupole moments in XX, YY, ZZ, XY, XZ, YZ
3	electronic spatial extent (au)	
4	HUMO	
5	LUMO	
6	rotational constants	rotational constants in X, Y, Z
7	nuclear repulsion energy (au)	
8	R6Disp: Grimme-D2 Dispersion energy (au)	
9	nuclear repulsion after empirical dispersion term	
10	PCM non-electrostatic energy	
11	PCM non-electrostatic energy	
12	nuclear repulsion after PCM non-electrostatic terms	
13	KE	
14	PE	
15	EE	
16	SMD-CDS (non-electrostatic) energy (kcal/mol)	
17	number of generator spheres in GePol	
18	total number of spheres in GePol	
19	number of exposed spheres in GePol	
20	percent of exposed spheres in GePol	
21	number of points in GePol	
22	average weight of points in GePol	
23	minimum weight of points in GePol	
24	maximum weight of points in GePol	
25	number of points with low weight	
26	fraction of low-weight points GePol (<1% of avg)	
27	cavity surface area in GePol (ang**2)	
28	cavity volume in GePol (ang ** 3)	

Table C.2 Top 10 models from commonly used algorithms.

Regressor	**feature_type**	**MAE_test**	**RMSE_test**	R^2**_test**
KernelRidge	150	0.6581±0.0995	1.0568±0.1839	0.4717±0.1052
KernelRidge	100	0.6721±0.0829	1.0551±0.1998	0.4756±0.1089
KNN	500	0.6819±0.0949	1.0895±0.2143	0.4359±0.1408
KNN	150	0.6828±0.0878	1.1027±0.2145	0.4222±0.1485
KernelRidge	200	0.6860±0.1078	1.0938±0.1901	0.4336±0.1131
KNN	602	0.6895±0.0943	1.1084±0.2249	0.4178±0.1484
KernelRidge	SECFP6	0.6913±0.0989	1.0595±0.1779	0.4690±0.1097
KernelRidge	MaCCSKeys	0.6923±0.0929	1.0504±0.1865	0.4753±0.1202
KernelRidge	50	0.6944±0.0897	1.0527±0.1951	0.4776±0.1061
KernelRidge	ECFP6	0.6949±0.0835	1.0704±0.1922	0.4578±0.1144

Table C.3 Top 10 models from GPR.

kernel	**feature_type**	**MAE_test**	**RMSE_test**	R^2**_test**
Matern12*Constant + White	desc_150	0.6249±0.0868	1.0164±0.1867	0.5128±0.0967
Exponential	desc_150	0.6249±0.0868	1.0164±0.1867	0.5128±0.0967
Matern12	desc_150	0.6249±0.0868	1.0164±0.1867	0.5128±0.0967
Matern12 + Constant	desc_150	0.6249±0.0868	1.0164±0.1867	0.5128±0.0967
Exponential*Constant + White	desc_150	0.6249±0.0868	1.0165±0.1866	0.5128±0.0967
Matern32 + Constant	desc_150	0.6291±0.0859	1.0206±0.1784	0.5087±0.0907
Matern32	desc_150	0.6291±0.0859	1.0206±0.1784	0.5087±0.0907
Matern32*Constant + White	desc_150	0.6291±0.0859	1.0206±0.1784	0.5087±0.0907
Matern32 + White	desc_150	0.6291±0.0859	1.0206±0.1784	0.5087±0.0907
Matern52*Constant + White	desc_150	0.6333±0.0859	1.0242±0.1773	0.5052±0.0898

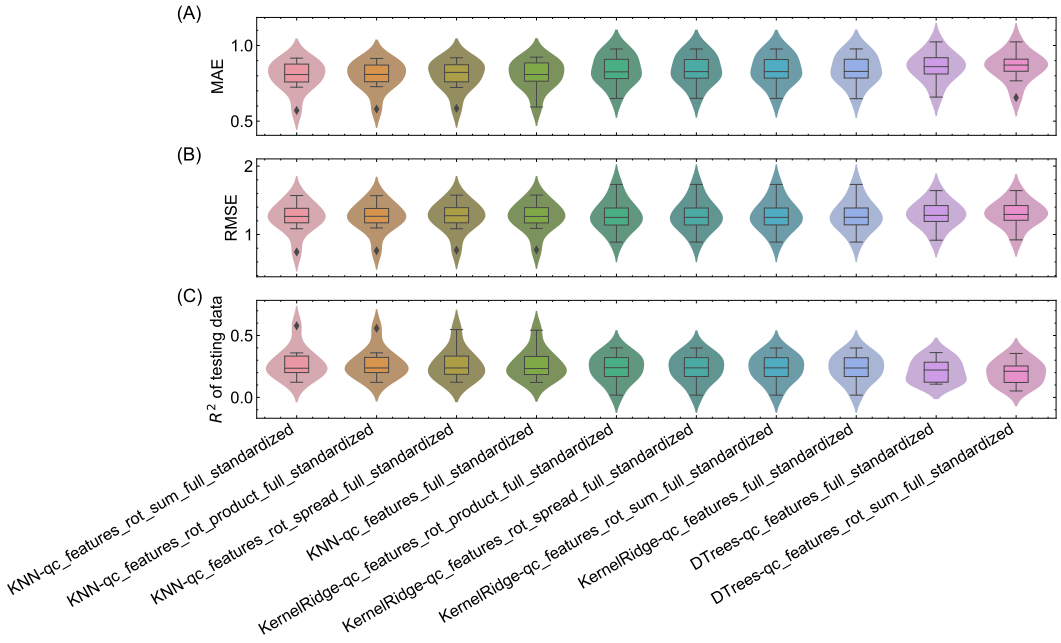


Fig. C.1 Box plots and violin plots for top 10 QM based regression models with commonly used algorithms.

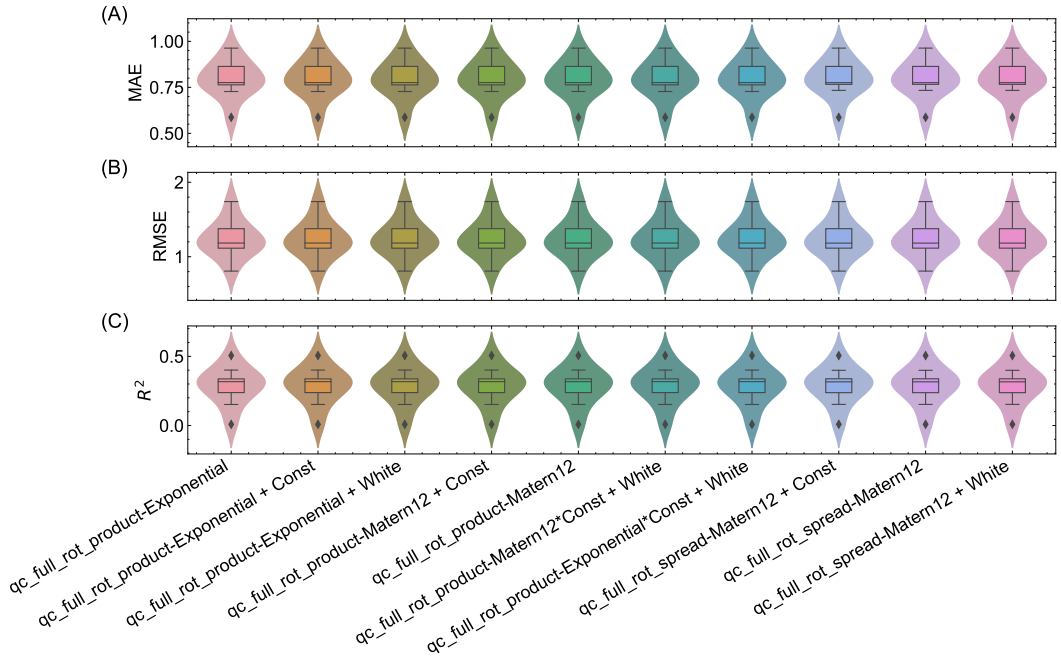


Fig. C.2 Box plots and violin plots for top 10 QM based GPR models.

Appendix D

Supporting Information for BBB Permeability Predictions

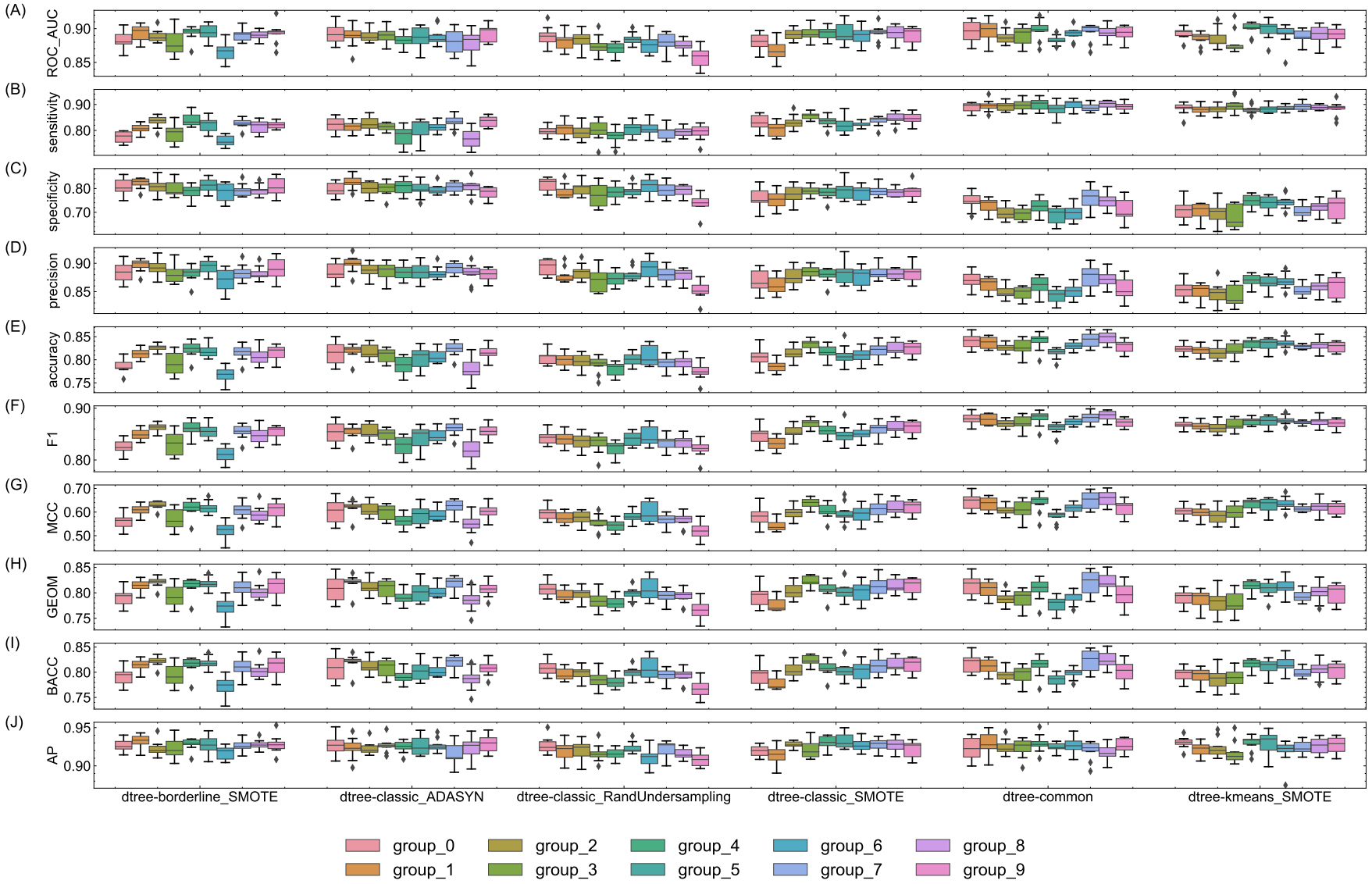


Fig. D.1 Model performances of decision trees based classifiers for 10 groups of hyperparameters.

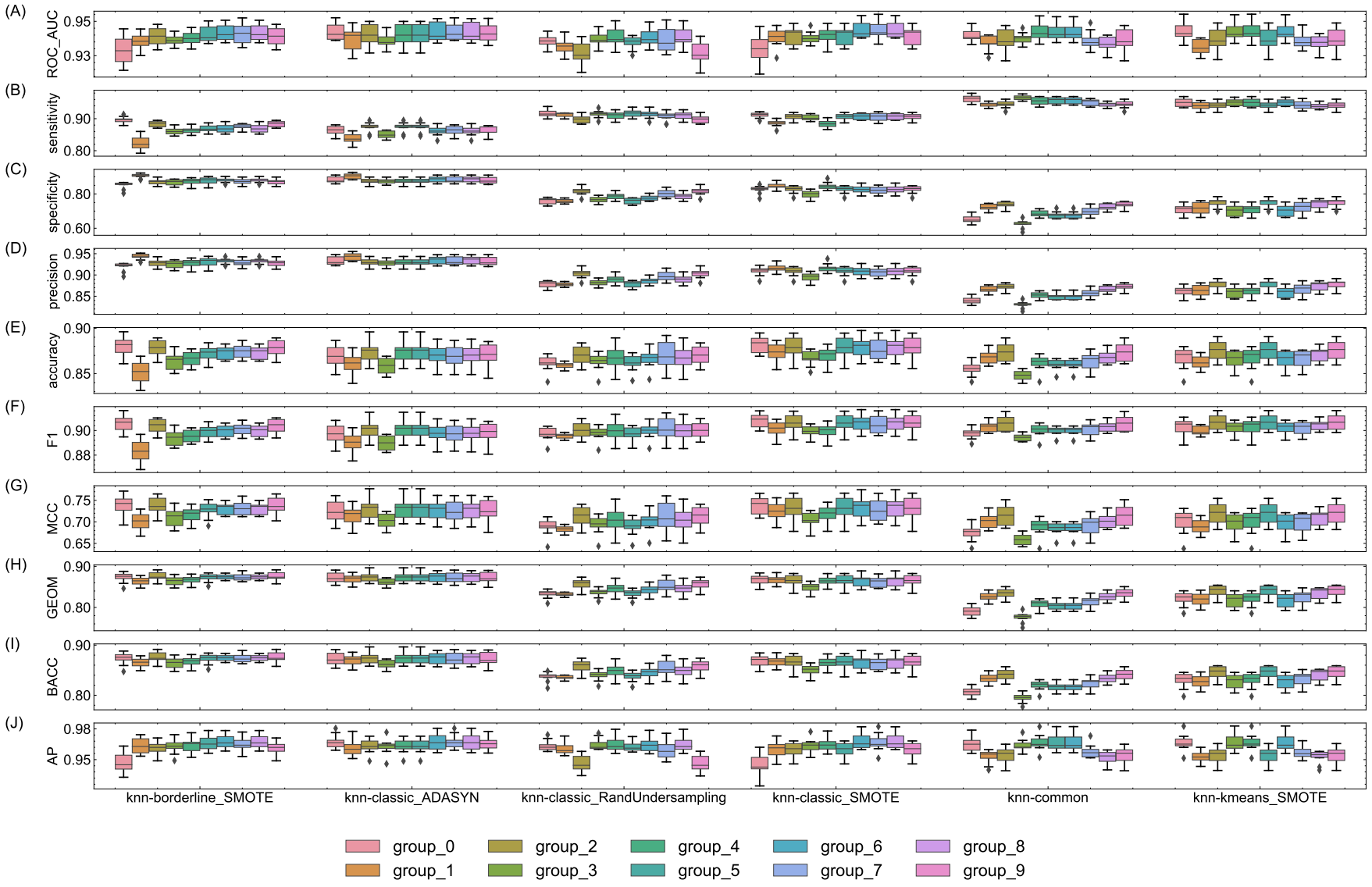


Fig. D.2 Model performances of kNN based classifiers for 10 groups of hyperparameters.

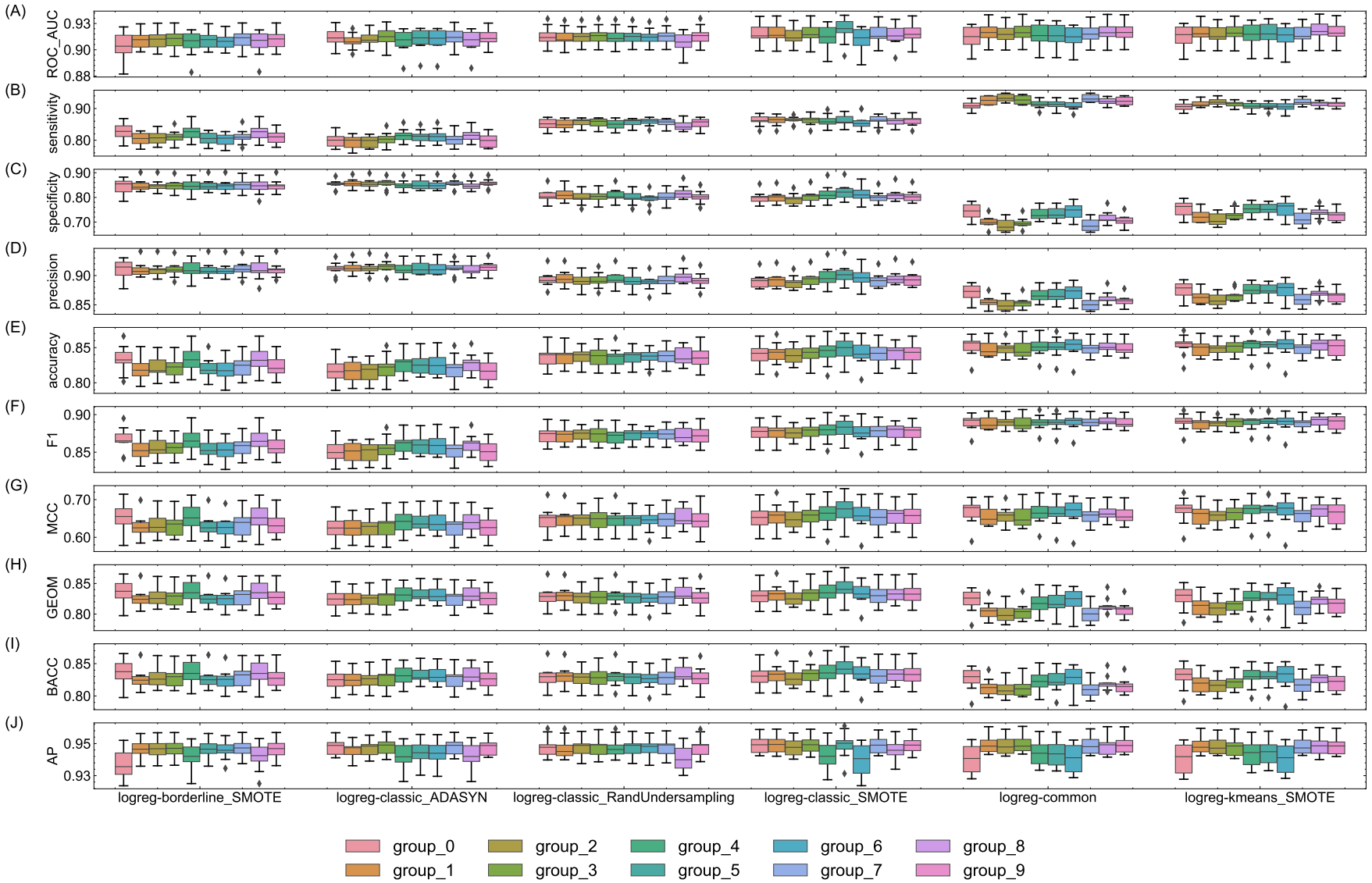


Fig. D.3 Model performances of logistical regression based classifiers for 10 groups of hyperparameters.

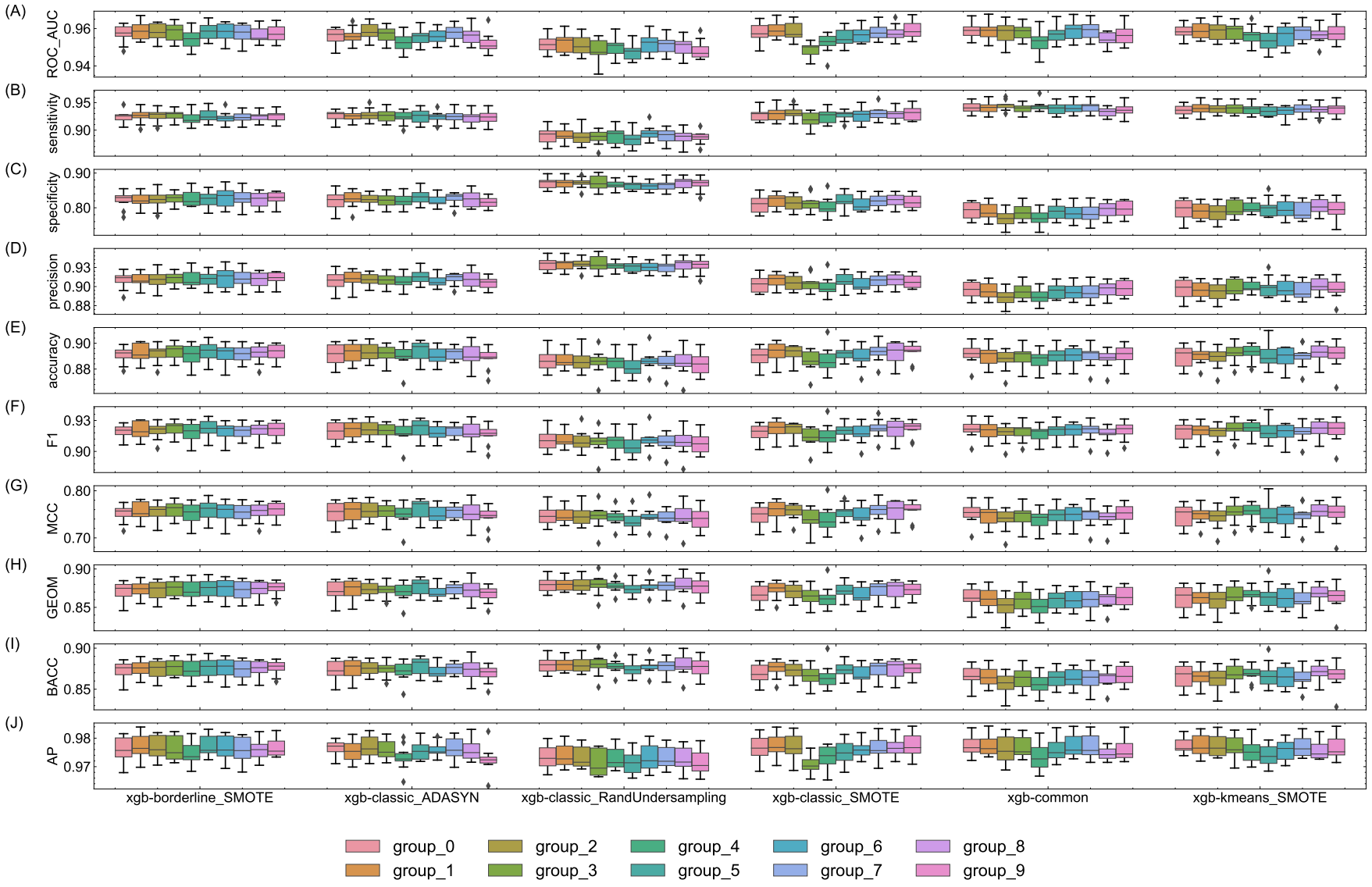


Fig. D.4 Model performances of XGBoost based classifiers for 10 groups of hyperparameters.

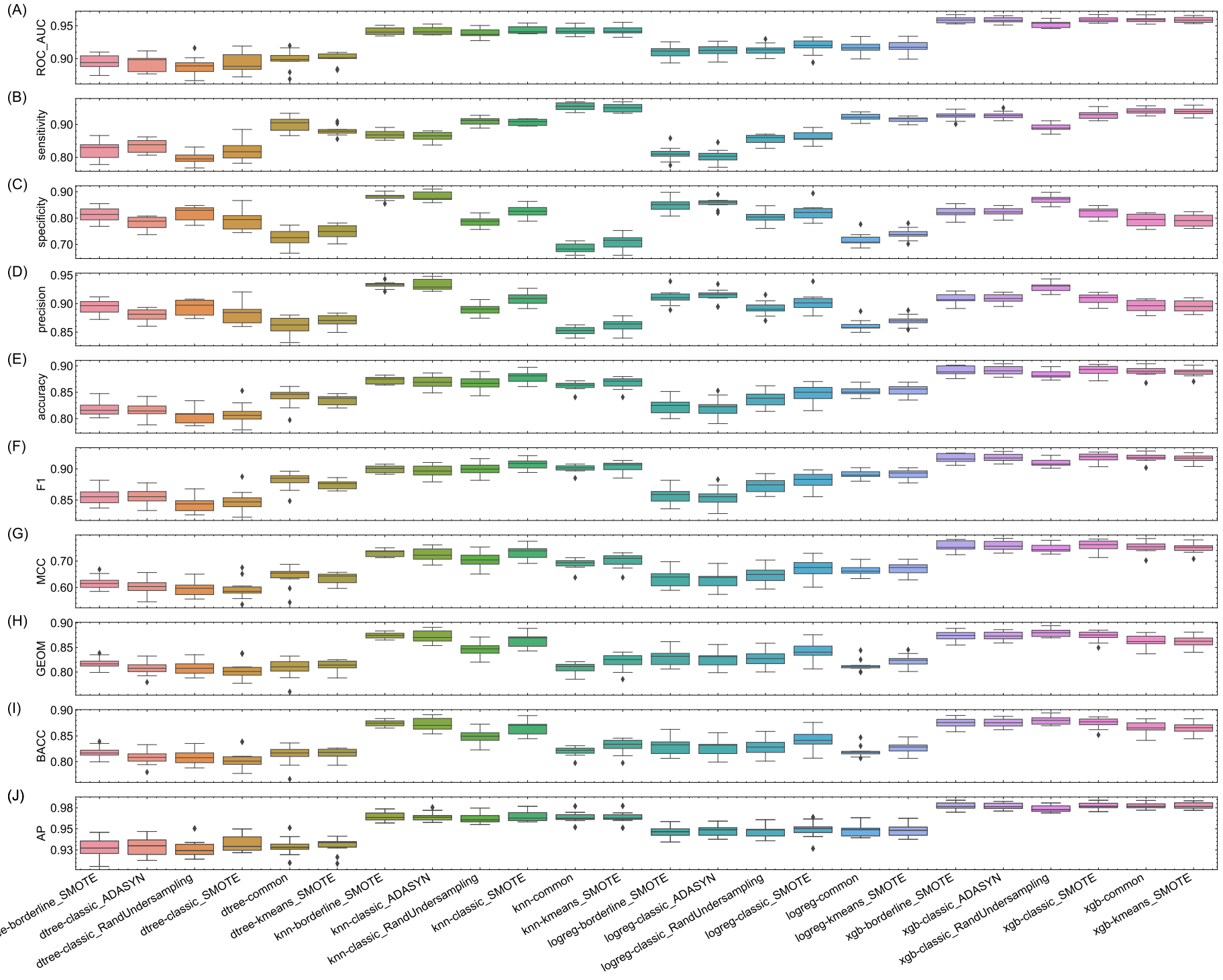


Fig. D.5 Model performance summary of all the classification models for BBB permeability.

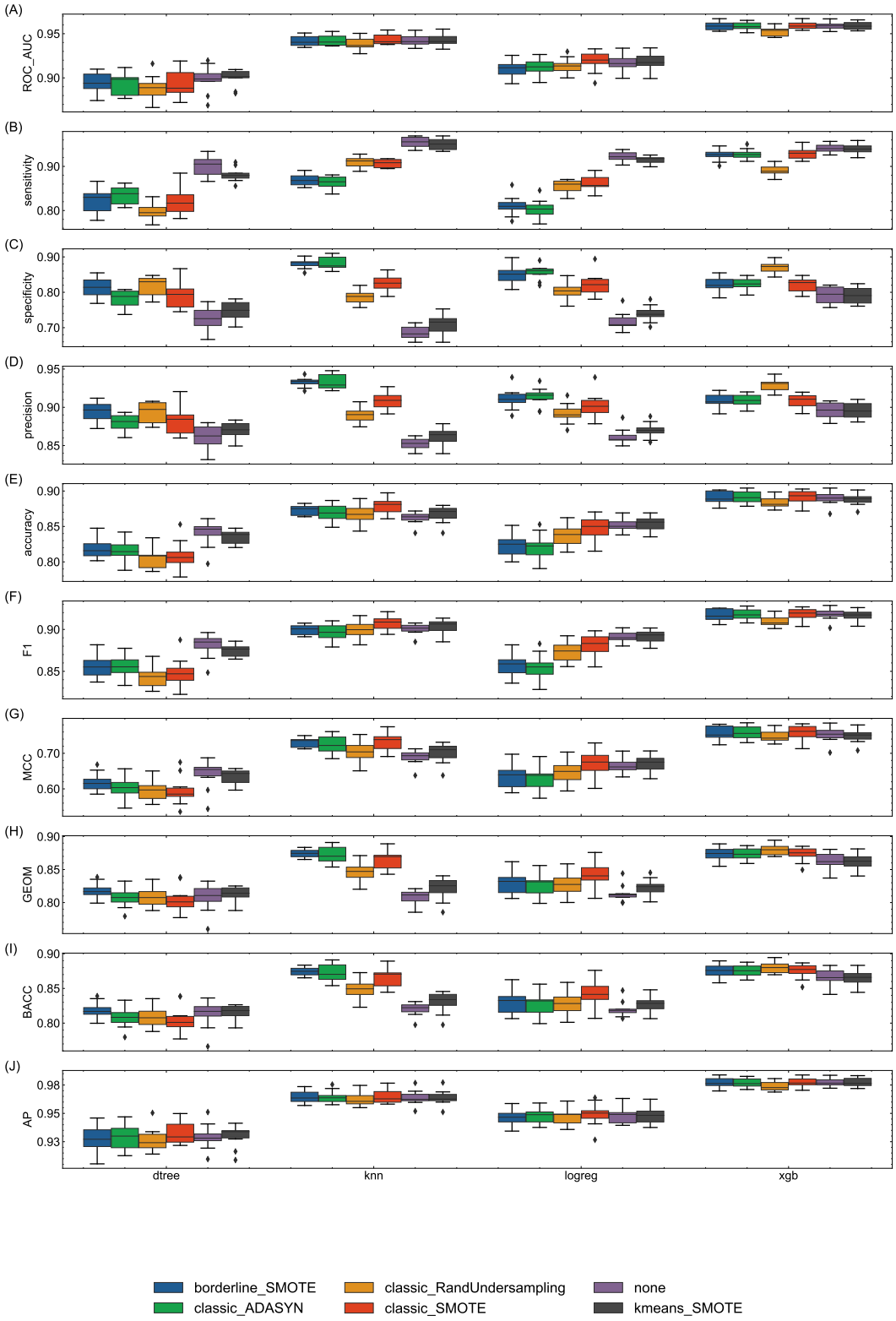


Fig. D.6 Model performance summary based on classification algorithms.

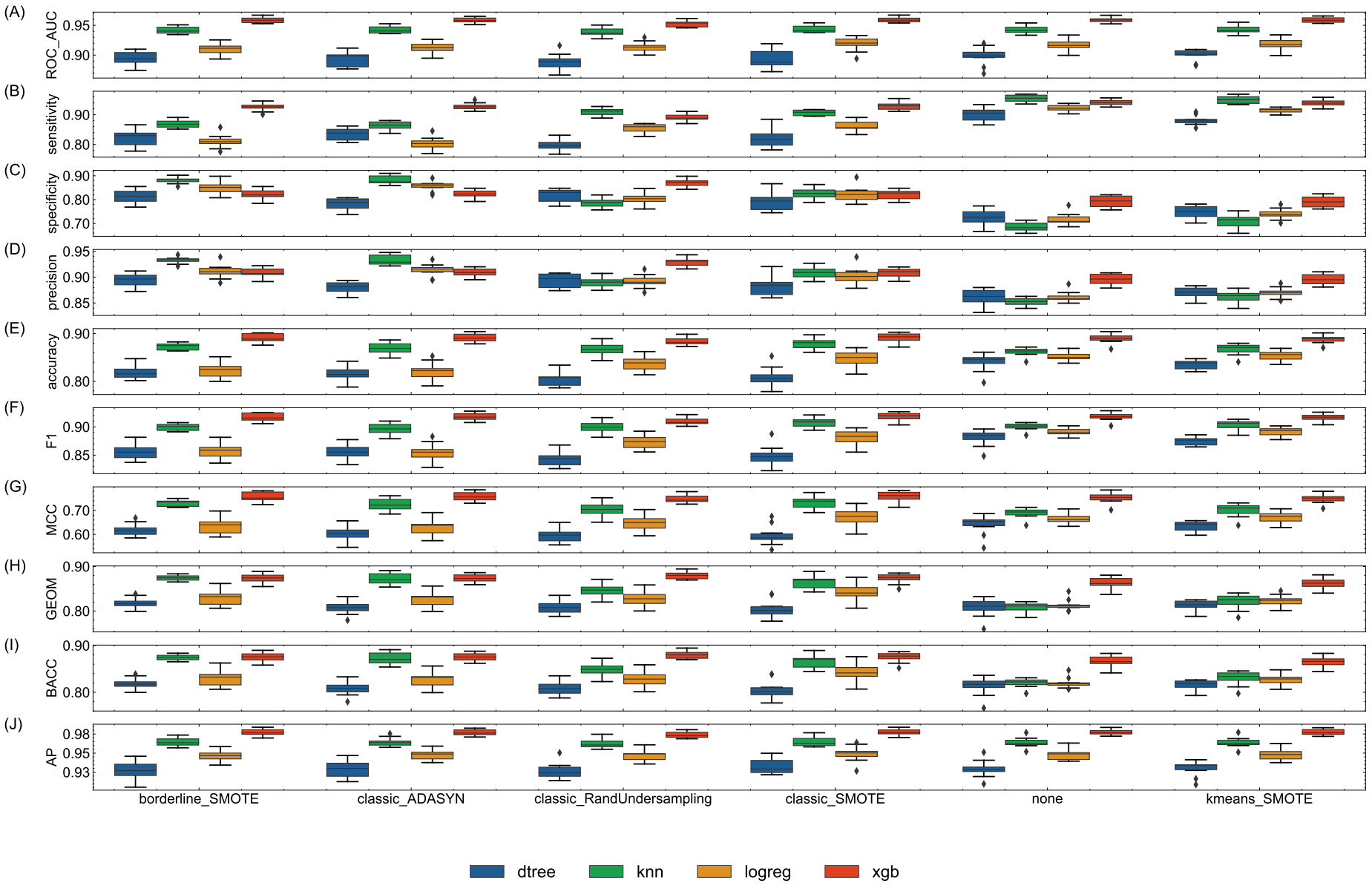


Fig. D.7 Model performance summary based on resampling strategies.

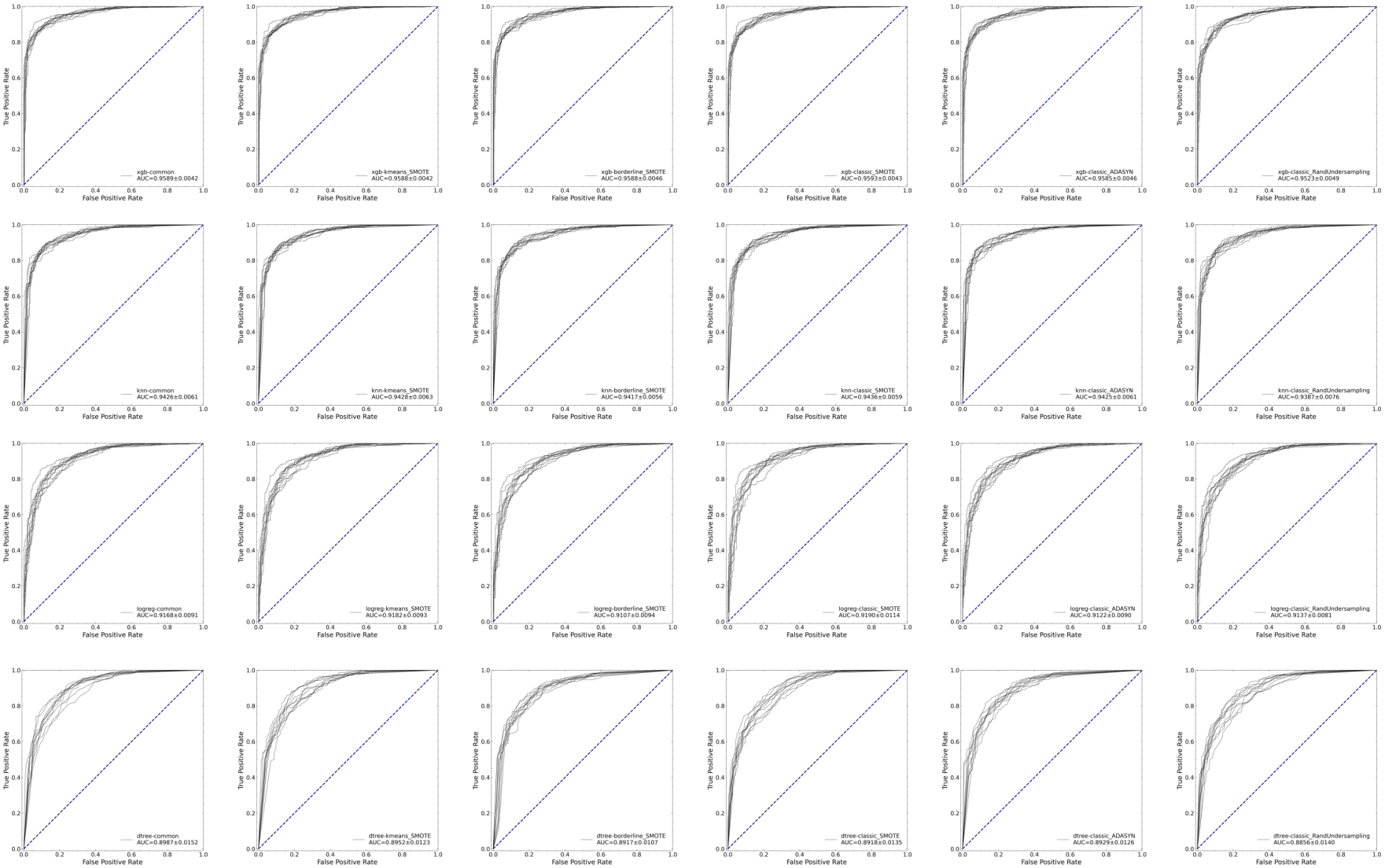


Fig. D.8 Receiver operating characteristic (ROC) curves of 24 predictive models.

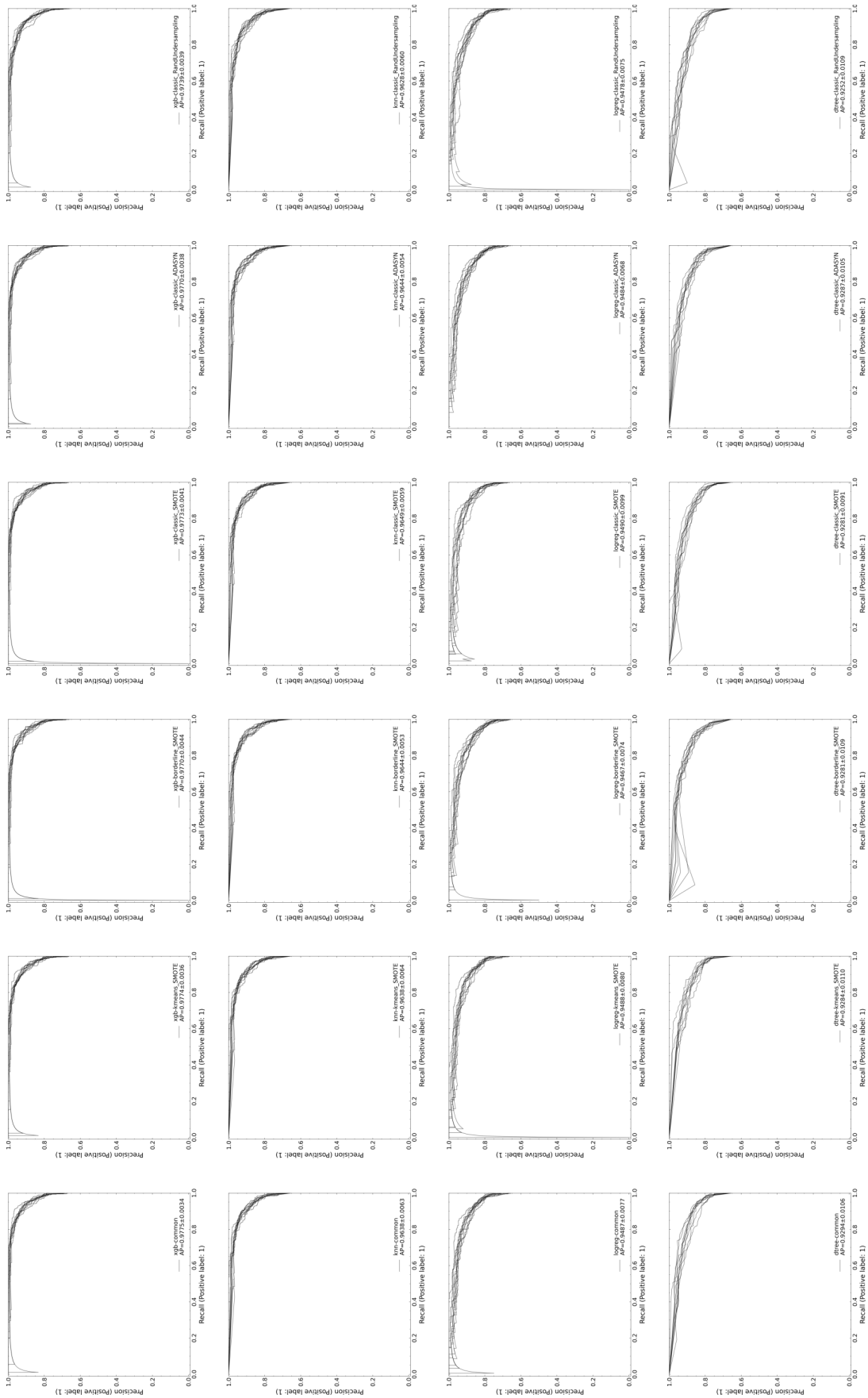


Fig. D.9 Precision-recall curves of 24 predictive models.

Table D.1 Model performance of 24 different classifiers by combing basic algorithms and resampling strategies.

model_name	ROC_AUC	sensitivity	specificity	precision	accuracy	F1	MCC	GEOM	BACC	AP
dtree-kmeans_SMOTE	0.9000±0.0094	0.8814±0.0158	0.7472±0.0289	0.8692±0.0120	0.8352±0.0101	0.8751±0.0079	0.6332±0.0227	0.8113±0.0136	0.8143±0.0124	0.9300±0.0100
dtree-common	0.8982±0.0152	0.9005±0.0225	0.7241±0.0347	0.8615±0.0153	0.8397±0.0183	0.8804±0.0141	0.6399±0.0411	0.8072±0.0211	0.8123±0.0200	0.9291±0.0112
dtree-classic_SMOTE	0.8942±0.0159	0.8194±0.0301	0.7904±0.0372	0.8818±0.0182	0.8094±0.0203	0.8491±0.0174	0.5948±0.0412	0.8043±0.0203	0.8049±0.0204	0.9332±0.0105
dtree-classic_RandUndersampling	0.8892±0.0137	0.7983±0.0198	0.8178±0.0275	0.8931±0.0140	0.8050±0.0147	0.8429±0.0127	0.5948±0.0299	0.8078±0.0155	0.8081±0.0155	0.9265±0.0106
dtree-classic_ADASYN	0.8931±0.0124	0.8346±0.0208	0.7817±0.0246	0.8793±0.0118	0.8164±0.0152	0.8562±0.0129	0.6045±0.0308	0.8076±0.0154	0.8082±0.0152	0.9291±0.0116
dtree-borderline_SMOTE	0.8941±0.0121	0.8218±0.0276	0.8139±0.0285	0.8940±0.0130	0.8191±0.0147	0.8560±0.0137	0.6182±0.0266	0.8175±0.0129	0.8178±0.0128	0.9278±0.0127
knn-kmeans_SMOTE	0.9428±0.0067	0.9502±0.0132	0.7096±0.0306	0.8618±0.0123	0.8673±0.0123	0.9037±0.0087	0.7009±0.0291	0.8209±0.0176	0.8299±0.0155	0.9638±0.0068
knn-common	0.9426±0.0065	0.9547±0.0119	0.6853±0.0186	0.8524±0.0072	0.8619±0.0089	0.9006±0.0066	0.6888±0.0214	0.8088±0.0110	0.8200±0.0098	0.9638±0.0066
knn-classic_SMOTE	0.9436±0.0062	0.9067±0.0098	0.8248±0.0237	0.9079±0.0114	0.8785±0.0112	0.9073±0.0084	0.7312±0.0254	0.8647±0.0141	0.8658±0.0136	0.9649±0.0063
knn-classic_RandUndersampling	0.9387±0.0080	0.9100±0.0130	0.7860±0.0187	0.8900±0.0093	0.8673±0.0131	0.8999±0.0101	0.7037±0.0290	0.8457±0.0142	0.8480±0.0139	0.9628±0.0063
knn-classic_ADASYN	0.9425±0.0065	0.8636±0.0144	0.8820±0.0186	0.9331±0.0101	0.8700±0.0126	0.8970±0.0104	0.7256±0.0261	0.8727±0.0130	0.8728±0.0131	0.9644±0.0057
knn-borderline_SMOTE	0.9417±0.0059	0.8686±0.0134	0.8805±0.0129	0.9326±0.0061	0.8727±0.0071	0.8994±0.0063	0.7302±0.0130	0.8744±0.0059	0.8745±0.0059	0.9644±0.0056
logreg-kmeans_SMOTE	0.9182±0.0098	0.9145±0.0090	0.7398±0.0227	0.8700±0.0101	0.8543±0.0105	0.8917±0.0076	0.6719±0.0241	0.8225±0.0138	0.8272±0.0128	0.9488±0.0084
logreg-common	0.9168±0.0095	0.9219±0.0115	0.7183±0.0251	0.8617±0.0104	0.8518±0.0089	0.8907±0.0065	0.6650±0.0205	0.8136±0.0129	0.8201±0.0115	0.9487±0.0082
logreg-classic_SMOTE	0.9190±0.0120	0.8622±0.0192	0.8225±0.0315	0.9026±0.0163	0.8485±0.0168	0.8818±0.0135	0.6730±0.0363	0.8419±0.0187	0.8424±0.0186	0.9491±0.0105
logreg-classic_RandUndersampling	0.9137±0.0086	0.8548±0.0156	0.8025±0.0252	0.8918±0.0129	0.8368±0.0153	0.8728±0.0122	0.6465±0.0327	0.8281±0.0169	0.8286±0.0167	0.9478±0.0079
logreg-classic_ADASYN	0.9122±0.0095	0.8021±0.0226	0.8562±0.0202	0.9138±0.0121	0.8207±0.0190	0.8542±0.0165	0.6331±0.0366	0.8286±0.0183	0.8291±0.0182	0.9484±0.0072
logreg-borderline_SMOTE	0.9107±0.0099	0.8109±0.0226	0.8491±0.0247	0.9110±0.0137	0.8241±0.0179	0.8579±0.0155	0.6367±0.0353	0.8297±0.0176	0.8300±0.0177	0.9467±0.0078
xgb-kmeans_SMOTE	0.9588±0.0044	0.9392±0.0112	0.7919±0.0235	0.8958±0.0103	0.8885±0.0092	0.9170±0.0068	0.7498±0.0209	0.8623±0.0123	0.8656±0.0114	0.9774±0.0038
xgb-common	0.9589±0.0044	0.9409±0.0097	0.7923±0.0235	0.8961±0.0105	0.8897±0.0099	0.9179±0.0072	0.7524±0.0226	0.8633±0.0132	0.8666±0.0123	0.9775±0.0036
xgb-classic_SMOTE	0.9593±0.0045	0.9292±0.0131	0.8205±0.0217	0.9079±0.0098	0.8917±0.0095	0.9184±0.0072	0.7585±0.0213	0.8730±0.0113	0.8748±0.0108	0.9773±0.0043
xgb-classic_RandUndersampling	0.9523±0.0052	0.8908±0.0117	0.8699±0.0172	0.9288±0.0085	0.8836±0.0081	0.9094±0.0066	0.7485±0.0173	0.8802±0.0089	0.8804±0.0088	0.9739±0.0041
xgb-classic_ADASYN	0.9585±0.0048	0.9275±0.0121	0.8233±0.0180	0.9090±0.0080	0.8916±0.0083	0.9181±0.0065	0.7583±0.0184	0.8737±0.0094	0.8754±0.0091	0.9770±0.0040
xgb-borderline_SMOTE	0.9588±0.0048	0.9259±0.0131	0.8237±0.0210	0.9091±0.0095	0.8906±0.0092	0.9173±0.0071	0.7564±0.0204	0.8732±0.0108	0.8748±0.0104	0.9770±0.0046

Table D.2 Model performance of 24 different classifiers by combing basic algorithms and resampling strategies.

model_name	accuracy	sensitivity	specificity	precision	F1	MCC	GEOM	BACC	ROC_AUC	AP	TN	FP	FN	TP
dtree-borderline_SMOTE	0.7500	0.7639	0.3750	0.9706	0.8549	0.0603	0.5352	0.5694	0.6152	0.9739	3	5	51	165
dtree-classic_ADASYN	0.6741	0.6806	0.5000	0.9735	0.8011	0.0715	0.5833	0.5903	0.6157	0.9703	4	4	69	147
dtree-classic_RandUndersampling	0.7321	0.7407	0.5000	0.9756	0.8421	0.1009	0.6086	0.6204	0.6201	0.9742	4	4	56	160
dtree-classic_SMOTE	0.7098	0.7222	0.3750	0.9689	0.8276	0.0401	0.5204	0.5486	0.5686	0.9636	3	5	60	156
dtree-kmeans_SMOTE	0.8214	0.8380	0.3750	0.9731	0.9005	0.1053	0.5606	0.6065	0.6898	0.9805	3	5	35	181
dtree-common	0.8170	0.8333	0.3750	0.9730	0.8978	0.1020	0.5590	0.6042	0.5686	0.9681	3	5	36	180
knn-borderline_SMOTE	0.7098	0.7176	0.5000	0.9748	0.8267	0.0890	0.5990	0.6088	0.6861	0.9779	4	4	61	155
knn-classic_ADASYN	0.6964	0.7037	0.5000	0.9744	0.8172	0.0822	0.5932	0.6019	0.7179	0.9807	4	4	64	152
knn-classic_RandUndersampling	0.8125	0.8241	0.5000	0.9780	0.8945	0.1541	0.6419	0.6620	0.6716	0.9767	4	4	38	178
knn-classic_SMOTE	0.7679	0.7778	0.5000	0.9767	0.8660	0.1221	0.6236	0.6389	0.6820	0.9777	4	4	48	168
knn-kmeans_SMOTE	0.8795	0.8935	0.5000	0.9797	0.9346	0.2243	0.6684	0.6968	0.6947	0.9795	4	4	23	193
knn-common	**0.9018**	**0.9167**	0.5000	**0.9802**	**0.9474**	**0.2598**	**0.6770**	**0.7083**	0.7028	0.9798	4	4	**18**	**198**
logreg-borderline_SMOTE	0.7455	0.7546	0.5000	0.9760	0.8512	0.1085	0.6143	0.6273	0.6568	0.9801	4	4	53	163
logreg-classic_ADASYN	0.7188	0.7269	0.5000	0.9752	0.8329	0.0936	0.6028	0.6134	0.6510	0.9796	4	4	59	157
logreg-classic_RandUndersampling	0.7902	0.8009	0.5000	0.9774	0.8804	0.1372	0.6328	0.6505	0.6256	0.9749	4	4	43	173
logreg-classic_SMOTE	0.7589	0.7685	0.5000	0.9765	0.8601	0.1165	0.6199	0.6343	0.6007	0.9705	4	4	50	166
logreg-kmeans_SMOTE	0.8482	0.8611	0.5000	0.9789	0.9163	0.1868	0.6562	0.6806	0.6238	0.9758	4	4	30	186
logreg-common	0.8482	0.8611	0.5000	0.9789	0.9163	0.1868	0.6562	0.6806	0.6227	0.9762	4	4	30	186
xgb-borderline_SMOTE	0.8170	0.8287	0.5000	0.9781	0.8972	0.1577	0.6437	0.6644	0.7072	0.9741	4	4	37	179
xgb-classic_ADASYN	0.8259	0.8380	0.5000	0.9784	0.9027	0.1654	0.6473	0.6690	**0.7303**	**0.9813**	4	4	35	181
xgb-classic_RandUndersampling	0.7634	0.7731	0.5000	0.9766	0.8630	0.1193	0.6218	0.6366	0.6736	0.9734	4	4	49	167
xgb-classic_SMOTE	0.8393	0.8519	0.5000	0.9787	0.9109	0.1778	0.6526	0.6759	0.7066	0.9757	4	4	32	184
xgb-kmeans_SMOTE	0.8304	0.8426	0.5000	0.9785	0.9055	0.1694	0.6491	0.6713	0.6927	0.9716	4	4	34	182
xgb-common	0.8304	0.8426	0.5000	0.9785	0.9055	0.1694	0.6491	0.6713	0.7066	0.9764	4	4	34	182
LightBBB	0.8527	0.8657	0.5000	0.9791	0.9189	0.1915	0.6579	0.6829			4	4	29	187

Table D.3 SMILES for external dataset with 216 BBB+ molecules and 8 BBB- molecules. The 30 molecules originally in *B3DB* are not shown here.

Name	**SMILES**	**PubChem_CID**	**class**
acetyl-dl-carnitine	CC(=O)O[C@@H](CC(=O)[O-])C[N+](C)(C)C	1	BBB+
melatonin	COc1ccc2[nH]cc(CCN=C(C)O)c2c1	896	BBB+
N-acetyl-dl-leucine	CC(O)=N[C@@H](CC(C)C)C(=O)O	1995	BBB+
amantadine	N[C@]12C[C@H]3C[C@H](C[C@H](C3)C1)C2	2130	BBB+
amisulpride	CCN1CCC[C@@H]1CN=C(O)c1cc(S(=O)(=O)CC)c(N)cc1OC	2159	BBB+
biperiden	O[C@](CCN1CCCCC1)(c1ccccc1)[C@@H]1C[C@@H]2C=C[C@H]1C2	2381	BBB+
bupivacaine	CCCCN1CCCC[C@H]1C(O)=Nc1c(C)cccc1C	2474	BBB+
cevimeline	C[C@@H]1O[C@]2(CS1)CN1CCC2CC1	2684	BBB+
chlordiazepoxide	C/N=C1/CN(O)C(c2ccccc2)=c2cc(Cl)ccc2=N1	2712	BBB+
cloxazolam	OC1=Nc2ccc(Cl)cc2[C@@]2(c3ccccc3Cl)OCCN2C1	2816	BBB+
diflunisal	O=C(O)c1cc(-c2ccc(F)cc2F)ccc1O	3059	BBB+
fonazine	C[C@H](CN1c2ccccc2Sc2ccc(S(=O)(=O)N(C)C)cc21)N(C)C	3089	BBB+
disulfiram	CCN(CC)C(=S)SSC(=S)N(CC)CC	3117	BBB+
fenfluramine	CCN[C@H](C)Cc1cccc(C(F)(F)F)c1	3337	BBB+
floctafenine	O=C(OC[C@H](O)CO)c1ccccc1Nc1ccnc2c(C(F)(F)F)cccc12	3360	BBB+
glafenine	O=C(OC[C@H](O)CO)c1ccccc1Nc1ccnc2cc(Cl)ccc12	3474	BBB+
maprotiline	CNCCC[C@]12CC[C@H](c3ccccc31)c1ccccc12	4011	BBB+
mazaticol	CN1[C@@H]2CCC(C)(C)[C@H]1C[C@H](OC(=O)C(O)(c1cccs1)c1cccs1)C2	4019	BBB+
memantine	C[C@]12C[C@@H]3C[C@](C)(C1)C[C@@](N)(C3)C2	4054	BBB+
mepivacaine	Cc1cccc(C)c1N=C(O)[C@H]1CCCCN1C	4062	BBB+
methadone	CCC(=O)C(C[C@@H](C)N(C)C)(c1ccccc1)c1ccccc1	4095	BBB+
metixene	CN1CCC[C@H](CC2c3ccccc3Sc3ccccc32)C1	4167	BBB+
modafinil	N=C(O)C[S@](=O)C(c1ccccc1)c1ccccc1	4236	BBB+
mosapramine	OC1=N[C@H]2CCCCN2C12CCN(CCCN1c3ccccc3CCc3ccc(Cl)cc31)CC2	4257	BBB+
neostigmine	CN(C)C(=O)Oc1cccc([N+](C)(C)C)c1	4456	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
nifenazone	Cc1c(N=C(O)c2cccnc2)c(=O)n(-c2ccccc2)n1C	4487	BBB+
phenacetin	CCOc1ccc(N=C(C)O)cc1	4754	BBB+
prilocaine	CCCN[C@H](C)C(O)=Nc1ccccc1C	4906	BBB+
propentofylline	CCCn1cnc2c1c(=O)n(CCCCC(C)=O)c(=O)n2C	4938	BBB+
pyridostigmine	CN(C)C(=O)Oc1ccc[n+](C)c1	4991	BBB+
salicylamide	N=C(O)c1ccccc1O	5147	BBB+
salsalate	O=C(Oc1ccccc1C(=O)O)c1ccccc1O	5161	BBB+
scopolamine	CN1[C@H]2C[C@@H](OC(=O)[C@@H](CO)c3ccccc3)C[C@@H]1[C@H]1O[C@@H]21	5184	BBB+
tetracaine	CCCCNc1ccc(C(=O)OCCN(C)C)cc1	5411	BBB+
triclofos	O=P(O)(O)OCC(Cl)(Cl)Cl	5563	BBB+
trihexyphenidyl	O[C@@](CCN1CCCCC1)(c1ccccc1)C1CCCCC1	5572	BBB+
pilocarpine	CC[C@@H]1C(=O)OC[C@@H]1Cc1cncn1C	5910	BBB+
3,4-diaminopyridine	N=c1cc[nH]cc1N	5918	BBB+
mebutamate	CC[C@@H](C)C(C)(COC(=N)O)COC(=N)O	6151	BBB+
dl-2-aminobutyric acid	CC[C@H](N)C(=O)O	6657	BBB+
paramethadione	CC[C@]1(C)OC(=O)N(C)C1=O	8280	BBB+
chloroprocaine	CCN(CC)CCOC(=O)c1ccc(N)cc1Cl	8612	BBB+
methohexital	C=CC[C@]1([C@@H](C)C#CCC)C(=O)N(C)C(=O)N=C1O	9034	BBB+
methysergide	CC[C@@H](CO)N=C(O)[C@@H]1C=C2c3cccc4c3c(cn4C)C[C@H]2N(C)C1	9681	BBB+
cytisine	O=c1cccc2n1C[C@@H]1CNC[C@H]2C1	10235	BBB+
captodiame	CCCCSc1ccc([C@H](SCCN(C)C)c2ccccc2)cc1	10240	BBB+
ethadione	CCN1C(=O)OC(C)(C)C1=O	10630	BBB+
apronal	C=CC[C@H](C(O)=NC(=N)O)C(C)C	10715	BBB+
metabutethamine	CC(C)CNCCOC(=O)c1cccc(N)c1	11115	BBB+
phenoperidine	CCOC(=O)C1(c2ccccc2)CCN(CC[C@H](O)c2ccccc2)CC1	11226	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
citicoline	C[N+](C)(C)CCOP(=O)(O)OP(=O)([O-])OC[C@H]1O[C@@H](n2ccc(=N)nc2=O)[C@H](O)[C@@H]1O	03804	BBB+
citicoline	C[N+](C)(C)CCOP(=O)(O)OP(=O)(O)OC[C@H]1O[C@@H](n2ccc(=N)nc2O)[C@H](O)[C@@H]1O	13805	BBB+
phenibut	NC[C@H](CC(=O)O)c1ccccc1	14113	BBB+
bucetin	CCOc1ccc(NC(=O)C[C@@H](C)O)cc1	14130	BBB+
fencamfamin	CCN[C@H]1[C@H]2CC[C@H](C2)[C@@H]1c1ccccc1	14584	BBB+
phenazocine	C[C@@H]1[C@@H]2Cc3ccc(O)cc3[C@@]1(C)CCN2CCc1ccccc1	14707	BBB+
dixyrazine	C[C@H](CN1CCN(CCOCCO)CC1)CN1c2ccccc2Sc2ccccc21	17182	BBB+
proxibarbal	C=CCC1(C[C@@H](C)O)C(O)=NC(=O)N=C1O	17336	BBB+
narcobarbital	C=C(Br)C[C@@]1(C(C)C)C(=O)N(C)C(=O)N=C1O	18735	BBB+
reposal	CCC1(C2=C[C@@H]3CC[C@H](C2)C3)C(O)=NC(=O)N=C1O	19254	BBB+
fenethylline	C[C@H](Cc1ccccc1)NCCn1cnc2c1c(=O)n(C)c(=O)n2C	19527	BBB+
valnoctamide	CC[C@@H](C)[C@@H](CC)C(=N)O	20140	BBB+
benorilate	CC(=O)Oc1ccccc1C(=O)Oc1ccc(N=C(C)O)cc1	21102	BBB+
butriptyline	C[C@H](CC1c2ccccc2CCc2ccccc21)CN(C)C	21772	BBB+
butanilicaine	CCCCNCC(O)=Nc1c(C)cccc1Cl	22379	BBB+
melevodopa	COC(=O)[C@@H](N)Cc1ccc(O)c(O)c1	23497	BBB+
(1s,2r)-2-phenylcyclopropanamine	N[C@H]1C[C@@H]1c1ccccc1	26070	BBB+
benzoctamine	CNC[C@]12CC[C@H](c3ccccc31)c1ccccc12	28425	BBB+
bornaprine	CCN(CC)CCCOC(=O)[C@@]1(c2ccccc2)C[C@@H]2CC[C@H]1C2	30160	BBB+
lofexidine	C[C@@H](Oc1c(Cl)cccc1Cl)C1=NCCN1	30668	BBB+
dexetimide	O=C1N=C(O)CC[C@@]1(c1ccccc1)C1CCN(Cc2ccccc2)CC1	30843	BBB+
paraldehyde	C[C@H]1O[C@@H](C)O[C@@H](C)O1	31264	BBB+
articaine	CCCN[C@@H](C)C(O)=Nc1c(C)csc1C(=O)OC	32170	BBB+
oxetorone	CN(C)CCC=C1c2ccccc2OCc2c1oc1ccccc21	36846	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
etidocaine	CCCN(CC)[C@@H](CC)C(O)=Nc1c(C)cccc1C	37497	BBB+
doxefazepam	O=C1[C@@H](O)N=C(c2ccccc2F)c2cc(Cl)ccc2N1CCO	38668	BBB+
progabide	N=C(O)CCC/N=C(\c1ccc(Cl)cc1)c1cc(F)ccc1O	44115	BBB+
ethallobarbital	C=CCC1(CC)C(O)=NC(=O)N=C1O	48542	BBB+
rotigotine	CCCN(CCc1cccs1)[C@H]1CCc2c(O)cccc2C1	59227	BBB+
bezitramide	CCC(=O)n1c(=O)n(C2CCN(CCC(C#N)(c3ccccc3)c3ccccc3)CC2)c2ccccc21	61791	BBB+
iprazochrome	CC(C)N1C[C@@H](O)c2cc(/N=N/C(=N)O)c(O)cc21	65594	BBB+
viminol	CC[C@@H](C)N(C[C@H](O)c1cccn1Cc1ccccc1Cl)[C@H](C)CC	65697	BBB+
chlorproethazine	CCN(CC)CCCN1c2ccccc2Sc2ccc(Cl)cc21	65750	BBB+
dipyrocetyl	CC(=O)Oc1cccc(C(=O)O)c1OC(C)=O	68093	BBB+
guacetisal	COc1ccccc1OC(=O)c1ccccc1OC(C)=O	68749	BBB+
acamprosate	CC(O)=NCCCS(=O)(=O)O	71158	BBB+
niaprazine	C[C@H](CCN1CCN(c2ccc(F)cc2)CC1)N=C(O)c1cccnc1	71919	BBB+
rimazolium	CCOC(=O)c1c[n+](C)c2n(c1=O)[C@@H](C)CCC2	71940	BBB+
frovatriptan	CN[C@@H]1CCc2[nH]c3ccc(C(=N)O)cc3c2C1	77992	BBB+
agomelatine	COc1ccc2cccc(CCN=C(C)O)c2c1	82148	BBB+
cevimeline	C[C@@H]1O[C@@]2(CS1)CN1CCC2CC1	83898	BBB+
quinupramine	c1ccc2c(c1)CCc1ccccc1N2[C@H]1CN2CCC1CC2	93154	BBB+
tirilazad	C[C@@H]1C[C@H]2[C@@H]3CCC4=CC(=O)C=C[C@]4(C)C3=CC[C@]2(C)[C@@]1(O)OC(=O)CN1CCN(c2cc(N3CCCC3)n	100490	BBB+
mebicar	CN1C(=O)N(C)[C@H]2[C@@H]1N(C)C(=O)N2C	122282	BBB+
desvenlafaxine	CN(C)C[C@@H](c1ccc(O)cc1)C1(O)CCCCC1	125017	BBB+
safinamide	C[C@H](NCc1ccc(OCc2cccc(F)c2)cc1)C(=N)O	131682	BBB+
scopolamine	CN1[C@H]2C[C@@H](OC(=O)[C@H](CO)c3ccccc3)C[C@@H]1[C@H]1O[C@@H]21	168331	BBB+
varenicline	c1cnc2cc3c(cc2n1)[C@H]1CNC[C@@H]3C1	170361	BBB+
ropivacaine	CCCN1CCCC[C@H]1C(O)=Nc1c(C)cccc1C	175805	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
tropatepine	CN1[C@H]2CC[C@@H]1CC(=C1c3ccccc3CSc3ccccc31)C2	198068	BBB+
lurasidone	O=C1[C@H]2[C@@H]3CC[C@@H](C3)[C@H]2C(=O)N1C[C@@H]1CCCC[C@H]1CN1CCN(c2nsc3ccccc23)CC1	216467	BBB+
lacosamide	COC[C@@H](N=C(C)O)C(O)=NCc1ccccc1	219078	BBB+
phenazocine, (-)-	C[C@H]1[C@H]2Cc3ccc(O)cc3[C@]1(C)CCN2CCc1ccccc1	443405	BBB+
cocaine	COC(=O)[C@H]1[C@@H](OC(=O)c2ccccc2)C[C@@H]2CC[C@H]1N2C	446220	BBB+
ipidacrine	N=c1c2c([nH]c3c1CCC3)CCCC2	604519	BBB+
scopolamine	CN1[C@H]2C[C@@H](OC(=O)[C@@H](CO)c3ccccc3)C[C@@H]1[C@@H]3O[C@H]23	638540	BBB+
cannabidiol	C=C(C)[C@@H]1CCC(C)=C[C@H]1c1c(O)cc(CCCCC)cc1O	644019	BBB+
choline alfoscerate	C[N+](C)(C)CCOP(=O)([O-])OC[C@H](O)CO	657272	BBB+
cytisine	O=c1cccc2n1C[C@H]1CNC[C@@H]2C1	683511	BBB+
benztropine	CN1[C@H]2CC[C@@H]1C[C@@H](OC(c1ccccc1)c1ccccc1)C2	1201549	BBB+
adrafinil	O=[S@](CC(O)=NO)C(c1ccccc1)c1ccccc1	3033226	BBB+
asenapine	CN1C[C@H]2c3ccccc3Oc3ccc(Cl)cc3[C@@H]2C1	3036780	BBB+
neocitrullamon	N[C@@H](CCCN1C(=O)C(c2ccccc2)(c2ccccc2)N=C1O)C(=O)O	3084726	BBB+
tolcapone	Cc1ccc(C(=O)c2cc(O)c(O)c([N+](=O)[O-])c2)cc1	4659569	BBB+
istradefylline	CCn1c(=O)c2c(nc(/C=C/c3ccc(OC)c(OC)c3)n2C)n(CC)c1=O	5311037	BBB+
dexmedetomidine	Cc1cccc([C@H](C)c2cnc[nH]2)c1C	5311068	BBB+
stiripentol	CC(C)(C)[C@H](O)/C=C/c1ccc2c(c1)OCO2	5311454	BBB+
oxetorone	CN(C)CC/C=C1\c2ccccc2OCc2c1oc1ccccc21	6434185	BBB+
oxetorone	CN(C)CC/C=C1/c2ccccc2OCc2c1oc1ccccc21	6436540	BBB+
carisbamate	N=C(O)OC[C@@H](O)c1ccccc1Cl	6918474	BBB+
acetyl-l-carnitine	CC(=O)O[C@H](CC(=O)[O-])C[N+](C)(C)C	7045767	BBB+
brivaracetam	CCC[C@@H]1CC(=O)N([C@@H](CC)C(=N)O)C1	9837243	BBB+
tapentadol	CC[C@@H](c1cccc(O)c1)[C@@H](C)CN(C)C	9838022	BBB+
fabomotizole	CCOc1ccc2nc(SCCN3CCOCC3)[nH]c2c1	9862937	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
remimazolam	COC(=O)CC[C@@H]1N=C(c2ccccn2)c2cc(Br)ccc2-n2c(C)cnc21	9867812	BBB+
eslicarbazepine	N=C(O)N1c2ccccc2C[C@H](O)c2ccccc21	9881504	BBB+
scopolamine	CN1[C@H]2C[C@@H](OC(=O)[C@H](CO)c3ccccc3)C[C@@H]1[C@@H]1O[C@@H]21	9916237	BBB+
perampanel	N#Cc1ccccc1-c1cc(-c2ccccn2)cn(-c2ccccc2)c1=O	9924495	BBB+
pitolisant	Clc1ccc(CCCOCCCN2CCCCC2)cc1	9948102	BBB+
vortioxetine	Cc1ccc(Sc2ccccc2N2CCNCC2)c(C)c1	9966051	BBB+
pimavanserin	CC(C)COc1ccc(CN=C(O)N(Cc2ccc(F)cc2)C2CCN(C)CC2)cc1	10071196	BBB+
solriamfetol	N=C(O)OC[C@H](N)Cc1ccccc1	10130337	BBB+
tasimelteon	CCC(O)=NC[C@@H]1C[C@H]1c1cccc2c1CCO2	10220503	BBB+
cariprazine	CN(C)C(O)=N[C@H]1CC[C@@H](CCN2CCN(c3cccc(Cl)c3Cl)CC2)CC1	11154555	BBB+
lisdexamfetamine	C[C@@H](Cc1ccccc1)N=C(O)[C@@H](N)CCCCN	11597698	BBB+
cenobamate	N=C(O)O[C@@H](Cn1ncnn1)c1ccccc1Cl	11962412	BBB+
brexpiprazole	Oc1ccc2ccc(OCCCCN3CCN(c4cccc5sccc45)CC3)cc2n1	11978813	BBB+
(-)-tilidine	CCOC(=O)[C@@]1(c2ccccc2)CCC=C[C@@H]1N(C)C	12546498	BBB+
cevimeline	C[C@H]1O[C@]2(CS1)CN1CCC2CC1	18642481	BBB+
ethybenztropine	CCN1[C@H]2CC[C@@H]1C[C@@H](OC(c1ccccc1)c1ccccc1)C2	20055089	BBB+
flumedroxone	CC(=O)[C@@]1(O)CC[C@H]2[C@@H]3C[C@H](C(F)(F)F)C4=CC(=O)CC[C@]4(C)[C@H]3CC[C@@]21C	20055652	BBB+
valbenazine	COc1cc2c(cc1OC)[C@H]1C[C@@H](OC(=O)[C@@H](N)C(C)C)[C@H](CC2)C2	24795069	BBB+
suvorexant	Cc1ccc(-n2nccn2)c(C(=O)N2CCN(c3nc4cc(Cl)ccc4o3)CC[C@H]2C)c1	24965990	BBB+
cdp-choline(1-)	C[N+](C)(C)CCOP(=O)(O)OP(=O)([O-])OC[C@H]1O[C@@H](n2ccc(=N)nc2=O)[C@H](O)[C@@H]1O	25202509	BBB+
laquinimod	CCN(C(=O)c1c(O)c2c(Cl)cccc2n(C)c1=O)c1ccccc1	54677946	BBB+
mirogabalin	CCC1=C[C@@H]2[C@H](C1)C[C@]2(CN)CC(=O)O	59509752	BBB+
mazaticol	CN1[C@H]2CCC(C)(C)[C@@H]1C[C@@H](OC(=O)C(O)(c1cccs1)c1cccs1)C2	20471539	BBB+
mazaticol	CN1[C@H]2C[C@H](OC(=O)C(O)(c3cccs3)c3cccs3)C[C@H]1CCC2(C)C	118984411	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
etifoxine	CCN=C1Nc2ccc(Cl)cc2[C@@](C)(c2ccccc2)O1	135413553	BBB+
opicapone	Cc1c(Cl)c(C)[n+]([O-])c(Cl)c1-c1noc(-c2cc(O)c(O)c([N+](=O)[O-])c2)n1	135565903	BBB+
chloramphenicol	CN(Cc1nc(-c2cccs2)no1)C(=O)c1ccc(CSc2ccccc2)cc1	100000298	BBB+
bupropion	Cc1ccc([C@H](C)NC(=O)c2ccc(CSc3ccc(Cl)cc3)cc2)cc1C	100000444	BBB+
estradiol	N=C(S)[C@@H]1C[C@H]1S(=O)(=O)c1ccc(Cl)cc1	100000450	BBB+
allopurinol	Cc1nn(CCC(O)=NC[C@@H]2CCCN(c3ccccn3)C2)c(C)c1C	100002094	BBB+
theophylline	Cc1nn(CC(O)=N[C@H](c2ccc(F)cc2F)C2CC(O)C2)cc1Cl	100002153	BBB+
aspirin	Cc1cccc(-c2noc(CN(C)C(=O)CSCc3ccc([N+](=O)[O-])cc3)n2)c1	100002244	BBB+
verapamil	C[C@H](Cc1cnccn1)N(C)C(=O)c1noc2c1CCCC2	100002520	BBB+
carbenicillin	N[C@]1(CO)[C@H](c2ccccc2)[C@@H]1S(=O)(=O)c1ccc(Cl)cc1	100002559	BBB+
domperidone	COc1ccc(-c2noc(CNC(=O)c3ccc(CN4CCOCC4)cc3)n2)cc1	100003151	BBB+
ganciclovir	CCOC[C@@]1(CN)[C@H](c2ccc(OC)cc2)[C@@H]1S(=O)(=O)CC	100003454	BBB+
atropine	CC(=O)c1c(C)nn(CC(O)=N[C@H](c2ccccc2)c2nccn2C)c1C	100003661	BBB+
ibuprofen	CN(C)C(=O)C[C@@H](O)c1cc2n(n1)CCN(C(=O)CN1CCCCCC1)C2	100003672	BBB+
isoniazid	CC(C)[C@@H](CO)N=C(O)c1cnc(O)c2ccccc12	100003767	BBB+
progesterone	COC[C@]1(C(=O)O)[C@@H](S(=O)(=O)c2ccc(Cl)cc2)[C@@H]1c1ccc(OC)cc1	100004920	BBB+
propranolol	CC[C@@H](C)[C@@H](CO)N=C(O)[C@H]1CC(=O)N(CCc2ccccc2)C1	100004946	BBB+
testosterone	Cc1ccc(S[C@H](C)C(O)=N[C@@H](C)COc2ccccc2C)cc1	100005408	BBB+
zidovudine	Cn1cc([C@H](N=C(O)CCc2nc3ccccc3[nH]2)C2CC(O)C2)cn1	100005726	BBB+
doxycycline	Cc1ccc(NC(=O)COc2ccc(/C=N\\N=C(O)CN(c3cccc(Br)c3)S(C)(=O)=O)cc2)cc1	125880656	BBB+
H1_mepyramine	COc1ccc(CN(CCN(C)C)c2ccccn2)cc1	4992	BBB+
H1_carbinoxamine	CN(C)CCO[C@@H](c1ccc(Cl)cc1)c1ccccn1	2564	BBB+
H1_alimemazine	C[C@@H](CN(C)C)CN1c2ccccc2Sc2ccccc21	5574	BBB+
H1_promethazine	C[C@H](CN1c2ccccc2Sc2ccccc21)N(C)C	4927	BBB+
H1_pheniramine	CN(C)CC[C@@H](c1ccccc1)c1ccccn1	4761	BBB+

Continued on next page

Table D.3 (continued)

Name	SMILES	PubChem_CID	class
H1_cyclizine	CN1CCN(C(c2ccccc2)c2ccccc2)CC1	6726	BBB+
H1_chlorcyclizine	CN1CCN([C@H](c2ccccc2)c2ccc(Cl)cc2)CC1	2710	BBB+
H1_meclizine	Cc1cccc(CN2CCN([C@H](c3ccccc3)c3ccc(Cl)cc3)CC2)c1	4034	BBB+
H1_diphenhydramine	CN(C)CCOC(c1ccccc1)c1ccccc1	3100	BBB+
H1_brompheniramine	CN(C)CC[C@@H](c1ccc(Br)cc1)c1ccccn1	6834	BBB+
H1_chloropyramine	CN(C)CCN(Cc1ccc(Cl)cc1)c1ccccn1	25295	BBB+
H1_doxylamine	CN(C)CCO[C@](C)(c1ccccc1)c1ccccn1	3162	BBB+
H1_tripelennamine	CN(C)CCN(Cc1ccccc1)c1ccccn1	5587	BBB+
H1_dimetindene	C[C@H](C1=C(CCN(C)C)Cc2ccccc21)c1ccccn1	21855	BBB+
H1_cyproheptadine	CN1CCC(=C2c3ccccc3C=Cc3ccccc32)CC1	2913	BBB+
H1_antazoline	c1ccc(CN(CC2=NCCN2)c2ccccc2)cc1	2200	BBB+
H1_chlorphenamine	CN(C)CC[C@@H](c1ccc(Cl)cc1)c1ccccn1	2725	BBB+
H1_clemastine	CN1CCC[C@@H]1CCO[C@](C)(c1ccccc1)c1ccc(Cl)cc1	26987	BBB+
H1_hydroxyzine	OCCOCCN1CCN([C@H](c2ccccc2)c2ccc(Cl)cc2)CC1	3658	BBB+
H1_orphenadrine	Cc1ccccc1[C@@H](OCCN(C)C)c1ccccc1	4601	BBB+
H1_bromazine	CN(C)CCO[C@H](c1ccccc1)c1ccc(Br)cc1	2444	BBB+
H1_triprolidine	Cc1ccc(/C(=C\CN2CCCC2)c2ccccn2)cc1	5282443	BBB+
topotecan	CC[C@@]1(O)C(=O)OCc2c1cc1n(c2=O)Cc2cc3c(CN(C)C)c(O)ccc3nc2-1	60700	BBB+
H1_bepotastine	O=C(O)CCCN1CCC(O[C@H](c2ccc(Cl)cc2)c2ccccn2)CC1	2350	BBB-
H1_quifenadine	OC(c1ccccc1)(c1ccccc1)[C@@H]1CN2CCC1CC2	65600	BBB-
H1_rupatadine	Cc1cncc(CN2CCC(=C3c4ccc(Cl)cc4CCc4cccnc43)CC2)c1	133017	BBB-