Feature Selection based on CCA

Szilvia Ujvary

2021 Summer UROP Project supervised by
Dr. Marina Evangelou

Abstract

The selection of sets of features promises to be more effective than univariate analysis in discovering links between groups of variables. In this report, an already published method based on Canonical Correlation Analysis is examined [1]. Seoane JA et al combine CCA with the genetic algorithm, an evolutionary optimisation algorithm in order to minimize the resulting p-values. The method is compared and contrasted to two other evolutionary algorithms commonly used in integer programming problems.

Contents

1 Motivation

The goal of the project was to use Canonical Correlation Analysis (CCA) in order to identify groups of variables in two datasets that are highly associated, and quantify this association with a p-value. Roughly, this can be thought of as a generalization of multiple regression, where the response is also a set of variables. However, by testing the association of two sets of variables, we expect to discover associations that might not have been found otherwise. The advantage of the methods presented in this report are their generality, in the sense that they operate on any two datasets (of appropriate dimensions). However, it is beneficial to keep in mind a few specific examples, for which the methods can be used successfully.

A genome-wide association studies (GWAS) are used in genetics research to associate specific genetic variations with particular diseases. They aims to find genetic markers by analizing genetic information from many different people, which can be used to predict the presence of a disease. The identification of such genetic markers can help researchers understand how genes contribute to the disease and thereby develop better prevention and treatment strategies.

OMICS data integration concerns data obtained by genomics, transcriptomics, proteomics, or metabolomics experiments. By integrating data from the different fields and understanding the between and within relationships of the OMICS datasets, the power to predict phenotypes, such as the presence of a disease, can be improved.

2 Statistical Methods

Let $X^{n\times p}$ and $Y^{n\times q}$ two datasets of observations of two different characteristics of the same individuals. In this section, we follow [1] in presenting a statistical methods for feature selection in X and Y: Canonical Correlation Analysis (CCA) and the Wilk's lambda statistic.

2.1 Canonical Correlation Analysis (CCA)

Canonical Correlation Analysis (CCA) was first proposed by Harold Hotelling in 1936. It is a statistical method that aims to find linear combinations of two sets of variables, such that they have highest correlation. More formally, given $X^{n\times p}$ and $Y^{n\times q}$ with $p, q < n$, we seek canonical factors $u_1 \in \mathbb{R}^p, v_1 \in \mathbb{R}^q$, such that

$$\rho_1 = corr(Xu_1, Yv_1)$$

is maximized. The random variables (U_1, V_1), where $U_1 = Xu_1$ and $V_1 = Yv_1$ are the first pair of canonical variates. The next step is finding vectors maximizing the same correlation subject to the constraint that they need to be uncorrelated with the first pair of canonical variates. Hereby we obtain the second pair of canonical variates (U_2, V_2). Continuing this method, we obtain $\min\{p, q\}$ pairs of canonical variates. We define

$$\rho_i = corr(U_i, V_i) \quad i = 1, 2, ..., \min\{p, q\}$$

as the i$^{\text{th}}$ canonical correlation. Let $\sum_X$ and $\sum_Y$ be the covariance matrices of X and Y, respectively, and let $\sum_{XY}$ be the cross covariance matrix. The solution can be found by a singular value decomposition of $M := \sum_X^{-1/2} \sum_{XY} \sum_Y^{-1/2}$. The condition $p, q < n$ is needed to ensure invertibility of the matrices.

2.2 Wilk's Lambda and Rao's F approximation

The significance of the canonical correlation pairs can be tested using the Wilk's lambda test statistic:

$$\lambda = \prod_{i=1}^{j} (1 - \rho_i)^2$$

Rao's F approximation can then be used.

$$F(df_1, df_2) = \Big(\frac{1 - \lambda^{1/s}}{\lambda^{1/s}}\Big) \cdot \Big(\frac{df_2}{df_1}\Big),$$

where $s = \sqrt{\frac{p^2 \cdot q^2 - 4}{p^2 + q^2 - 5}}$, $\quad df_1 = p \cdot q, \quad df_2 = \Big(n - 1.5 - \frac{p+q}{2}\Big)\Big(s - \frac{p \cdot q}{2} + 1\Big)$. Here df_1 and df_2 are the two degrees of freedom of the F distribution. Comparing to the corresponding CDF, we obtain the required p-value.

The next section continues to present the method of [1] showing its links with integer programming.

3 Optimisational Methods

Suppose we have two datasets $X^{n \times p}$ and $Y^{n \times q}$ containing measurements on the same individuals. Based on the previous section, we may quantify the association of any selection of columns $\{X_{i_k}\}$ and $\{Y_{j_l}\}$ by using CCA and the Wilk's statistic on the matrices X', Y', spanned by our selection of X and Y columns, respectively.

This can be thought of as a function, with $p + q$ binary variables, that stores the specific selection of columns of X and Y, and returns the corresponding p-value. The task is then to minimize this function. Since all of the variables are restricted to be integers, the problem falls into the category of integer programming.

Due to NP-completeness, the problem can be addressed using metaheuristic algorithms that search for reasonably good solutions in computationally feasible time. In particular, we will look at Evolutionary Algorithms. These procedures are inspired by biological processes, such as reproduction, mutation, recombination, and selection. The problem is initialized together with a set of candidate solutions that pay the role of individuals in a population. The quality of the solutions is determined by the so-called fitness function, which is to be minimized. During the project, three algorithms were analyzed, which are presented below. In each of the algorithms, the fitness function is the function with $p + q$ binary variables that returns the p-value for the selection determined by the binary variables, as explained above.

3.1 The Genetic Algorithm

As suggested by its name, the genetic algorithm aims, to some extent, to model the process of gene evolution. The input is the array of the binary parameters and is thought of as the "gene information" of a particular member of the population of candidate solutions.

3.2 Differential Evolution

Differential Evolution is very similar to the Genetic Algorithm, in the sense that it is also inspired by gene evolution. However, the child is obtained using a different formula.

Algorithm 1 Genetic Algorithm

generate initial population pop
while iteration <= maxiteration **do**
  iteration = iteration + 1
  Calculate fitness of each individual in pop
  Select elite group to automatically carry over to new generation
  for non elite group **do**
    Sample parents
    Sample crossover point
    Produce child
    Add child to new generation
    Perform mutation with probability mutationChance
  end for
end while

Algorithm 2 Differential Evolution

generate initial population pop
while Result is not within tolerance **do**
  if iteration >=maxiter **then**
    Max number of iterations reached without convergence
  end if
  iteration = iteration + 1
  for i in population index **do**
    Update scaling factor F, crossover prob CR and mutation prob pF with pre-given probabilities
    Generate three random integers base, $r_1, r_2, \in$ [1, popsize] with base $\neq r_1 \neq r_2 \neq i$
    function PERFORMREPRODUCTION(X_{base})

$$\text{childTrial} = \begin{cases} X_{\text{base}} + \text{Ftrial} * (X_{r1} - X_{r2}) & \text{if there is mutation} \\ X_{\text{base}} + 0.5 * (\text{Ftrial} + 1) * (X_{r1} + X_{r2} - 2 * X_{\text{base}}) & \text{else} \end{cases}$$

    end function
    Obtain and evaluate childTrial
    Find best indices
  end for
  Evaluate if there is convergence
end while

3.3 Particle Swarm Optimization

In Particle Swarm Optimization, the population of candidate solutions is thought of as particles. The optimization is achieved by moving these particles around in the search-space depending on the particle's position and velocity.

Algorithm 3 Particle Swarm Optimization

for each particle **do**
  Initialize the particle's position (uniform distr.)
  Initialize the particle's best known position to its initial position: BestPi $\leftarrow x_i$
  Initialize the particle's velocity
  if f(BestPi) $< f$(BestPinSwarm) **then**
    Update the swarm's best known position: BestPinSwarm $\leftarrow$ BestPi
  end if
end for
while iteration $<$ maxiteration && result is not within tolerance **do**
  iteration = iteration + 1
  for each particle **do**
    for each dimension $d = 1, ..., n$ **do**
      Pick random numbers: $r_p, r_g \sim U(0, 1)$
      Update the particle's velocity based on r_p, r_g and other params
    end for
    Update the particle's position
    if $f(x_i) < f$(BestPi) **then**
      Update the particle's best known position: BestPi $\leftarrow x_i$
      if f(BestPi) $< f$(BestPinSwarm) **then**
        Update the swarm's best known position: BestPinSwarm $\leftarrow$ BestPi
      end if
    end if
  end for
end while

4 An Evaluation Study

4.1 Setup

The performance of the algorithms were tested on simulated data of dimensions 100×100 and 500×500 for both matrices. The following simulation algorithm was used.

The parameters of the random noise and the *constant* coefficient of X_k were varied throughout the simulations, to produce data with different association structures. Within the simulation, I have also checked the significance of the created associations by univariate analysis. I then ran univariate regression and the three algorithms on the two datasets.

For the Genetic Algorithm, code from [1] was modified for the general setting of two matrices. For Differential Evolution, I've tailored the source code of the DEoptimR package, while my code for Particle Swarm Optimization is based on the R package pso.

Algorithm 4 Simulation Algorithm

Create X with rnorm
Sample column indices $inds_x$ that will be associated, with rate *xprop*
Create *beta* coefficients as indicators of the associated X columns, add noise
Create Y with rnorm
Sample column indices $inds_y$ that will be associated, with rate *yprop*
for i in $inds_y$ **do**
    sample a single Y column Y_k
    $Y_i = X \cdot beta + constant \cdot X_k +$ noise
    $inds_x = c(inds_x, k)$
end for
Check via univariate analysis if all elements of $inds_x$ and $inds_y$ are in significant pairs

When calibrating the parameters of each algorithm, I usually used the values recommended by the package, making sure that the results obtained were realistic. Parameters that were present in all three algorithms were given the same value, most notably the number of candidate solutions were set to 1000 in all three algorithms and the initial population or initial solution (binary array) was generated with the same zero-one ratio. The evolutionary algorithms were each run for 100 iterations and repeated 10 times. The repetitions were paralellized on 8 computer cores. The univariate regression analysis was not paralellized.

4.2 Results

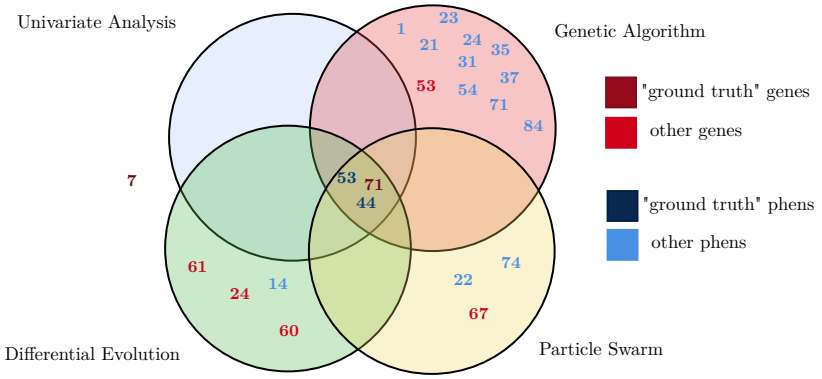
4.2.1 Simulated Data

In order to aid understanding and stress the possible applications of the method, throughout this section, the columns of X are called "genes", whereas the columns of Y are referred to as "phenotypes".

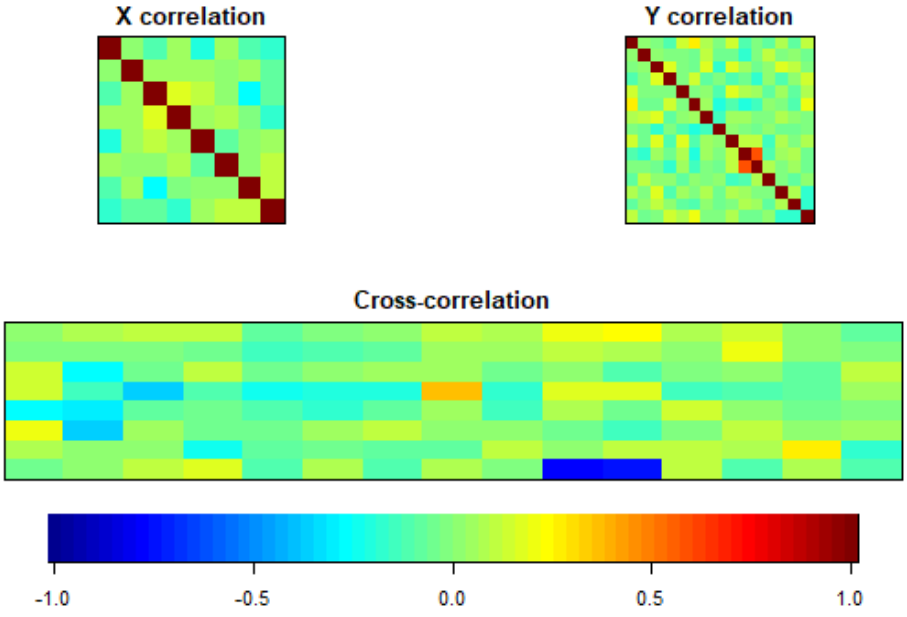
The below tables compare the performance of univariate regression and the three evolutionary algorithms. The *Construction* row collects the genes and phenotypes that were originally constructed to be associated. *Univariate check* confirms if they are indeed associated, by checking if all of them is present in an associated gene-phenotype pair. These walues are considered to be the "ground truth". The *Univariate* row shows the result of univariate analysis on the whole two matrices. The rows below show the results of the three algorithms, respectively. All three algorithms were run for 100 iterations and repeated 10 times to obtain a minimal p-value.

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	7, 22, 71	44, 53	NA
2	Univariate check	NA	7, 71	44, 53	NA
3	Univariate	NA	71	44, 53	4.88
4	Genetic Algorithm	-65.74601	53, 71	1, 21, 23, 24, 31, 35, 37, 44, 53, 54, 71, 84	6.05
5	Differential Evolution	-63.30580	24, 60, 61, 71	14, 44, 53	5.65
6	Particle Swarm Optimization	-61.09581	67, 71	22, 44, 53, 74	5.54

(a) Results for 100×100 matrices with few small associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



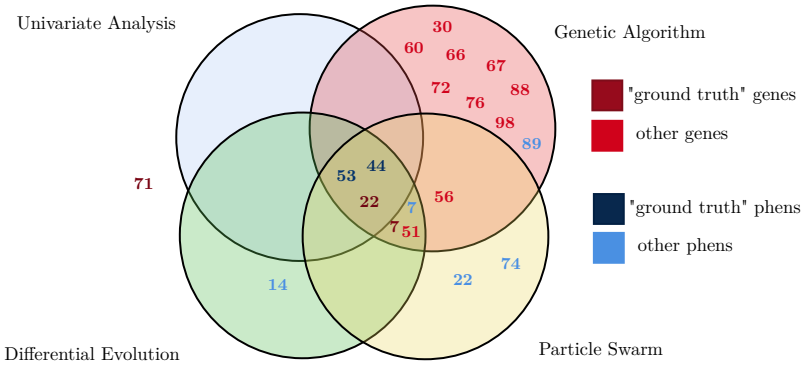
(b) Venn diagram of all genes and phenotypes appearing in table (a)



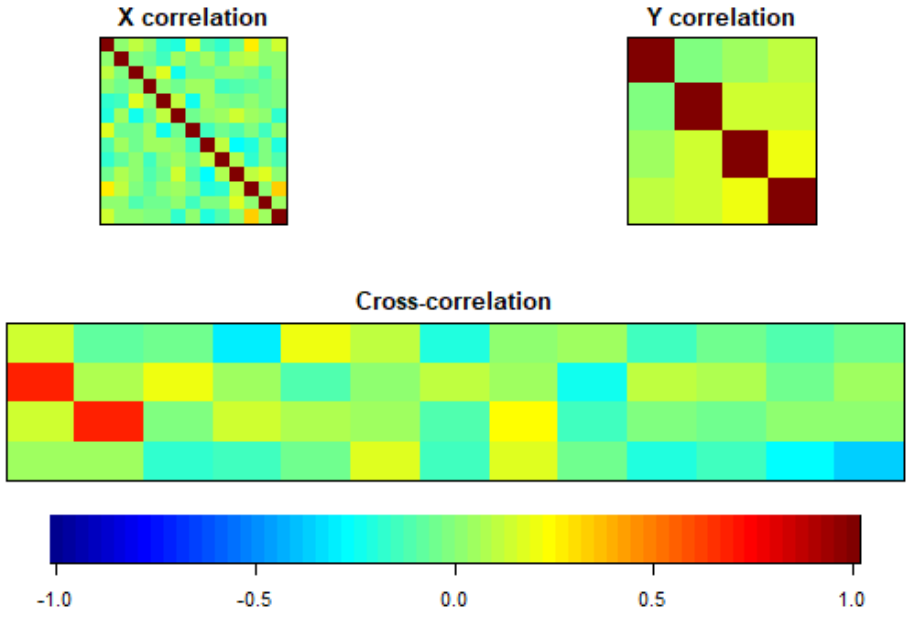
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	7, 22, 71	44, 53	NA
2	Univariate check	NA	7, 22, 71	44, 53	NA
3	Univariate	NA	7, 22	44, 53	4.86
4	Genetic Algorithm	-63.75849	7, 22, 30, 51, 56, 60, 66, 67, 72, 76, 88, 98	7, 44, 53, 89	5.61
5	Differential Evolution	-59.31640	7, 22, 51	7, 44, 53	6.14
6	Particle Swarm Optimization	-60.34500	7, 22, 51, 56	7, 44, 53	6.06

(a) Results for 100×100 matrices with few large associations. Best p-value is on log scale.Genes and phenotypes are referred to using their column index.



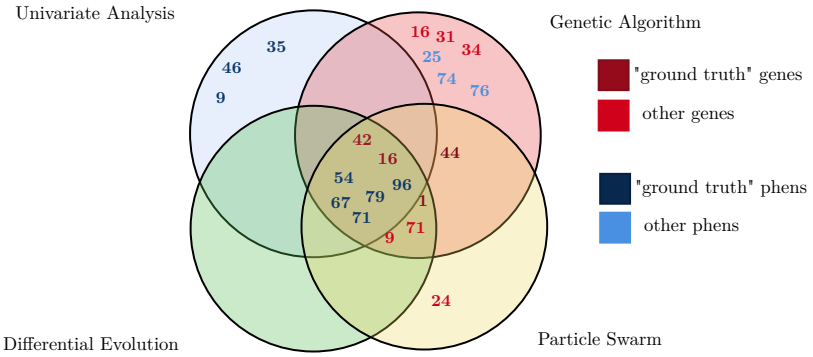
(b) Venn diagram of all genes and phenotypes appearing in table (a)



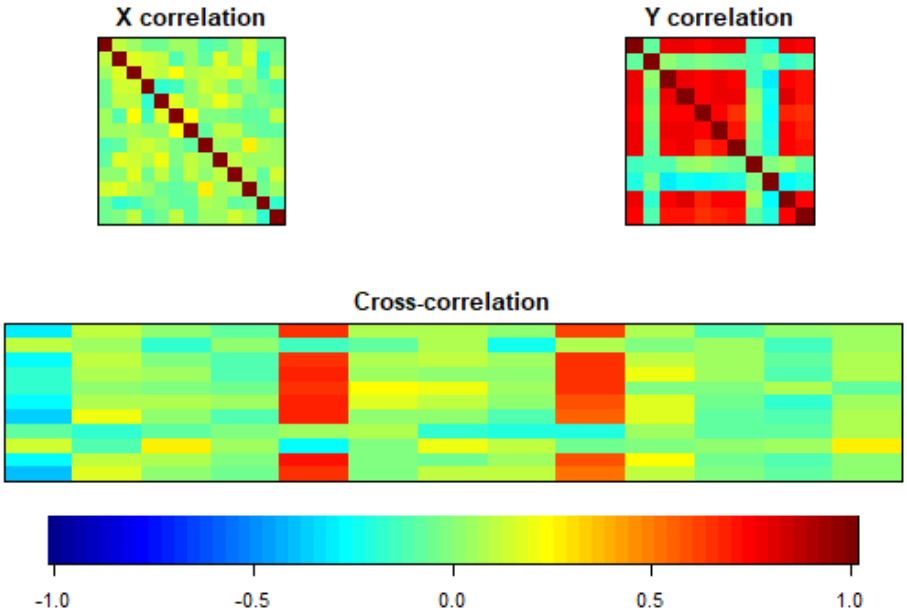
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	1, 7, 9, 9, 13, 16, 31, 42, 44, 66, 66, 71, 95	9, 35, 46, 54, 67, 71, 79, 96	NA
2	Univariate check	NA	1, 16, 42, 44	9, 35, 46, 54, 67, 71, 79, 96	NA
3	Univariate	NA	16, 42	9, 35, 46, 54, 67, 71, 79, 96	4.89
4	Genetic Algorithm	-121.7723	1, 9, 16, 31, 34, 42, 44, 71	25, 54, 67, 71, 74, 76, 79, 96	5.53
5	Differential Evolution	-112.9658	1, 9, 16, 42, 71	54, 67, 71, 79, 96	5.70
6	Particle Swarm Optimization	-114.3500	1, 9, 16, 24, 42, 44, 71	54, 67, 71, 79, 96	5.84

(a) Results for 100×100 matrices with a lot of small associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



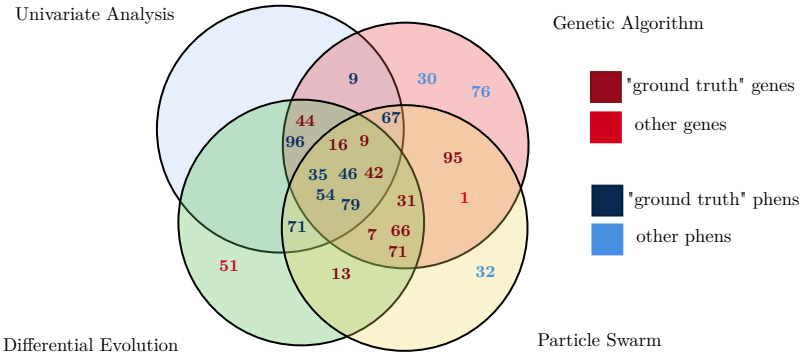
(b) Venn diagram of all genes and phenotypes appearing in table (a)



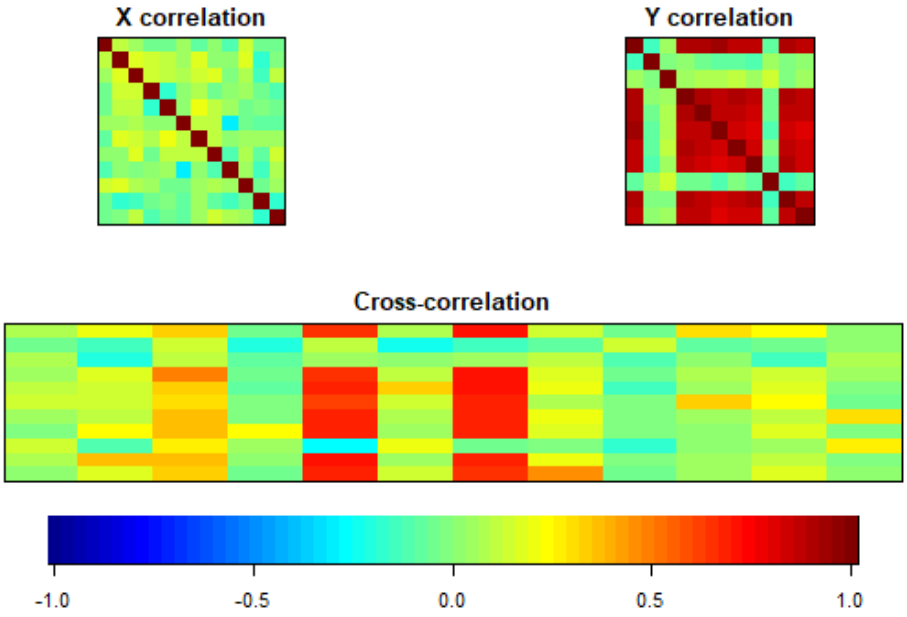
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	1, 7, 9, 9, 13, 16, 31, 42, 44, 66, 66, 71, 95	9, 35, 46, 54, 67, 71, 79, 96	NA
2	Univariate check	NA	7, 9, 13, 16, 31, 42, 44, 66, 71, 95	9, 35, 46, 54, 67, 71, 79, 96	NA
3	Univariate	NA	9, 16, 42, 44	9, 35, 46, 54, 67, 71, 79, 96	5.36
4	Genetic Algorithm	-242.3074	1, 7, 9, 16, 31, 42, 44, 66, 71, 95	9, 30, 35, 46, 54, 67, 76, 79, 96	6.83
5	Differential Evolution	-229.4114	7, 9, 13, 16, 31, 42, 44, 51, 66, 71	35, 46, 54, 71, 79, 96	5.81
6	Particle Swarm Optimization	-231.1792	1, 7, 9, 13, 16, 31, 42, 66, 71, 95	32, 35, 46, 54, 67, 71, 79	5.78

(a) Results for 100×100 matrices with a lot of strong associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



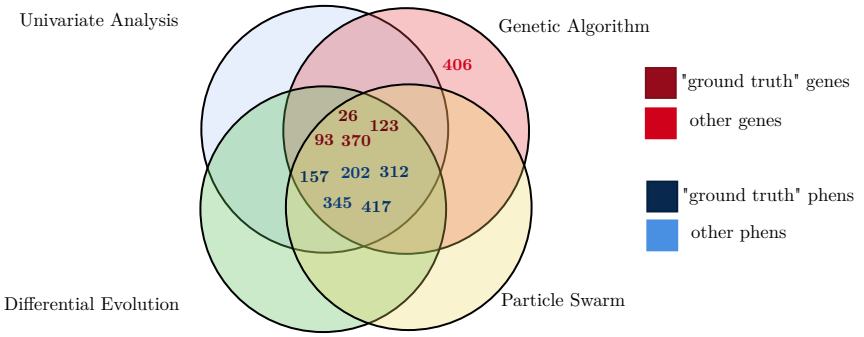
(b) Venn diagram of all genes and phenotypes appearing in table (a)



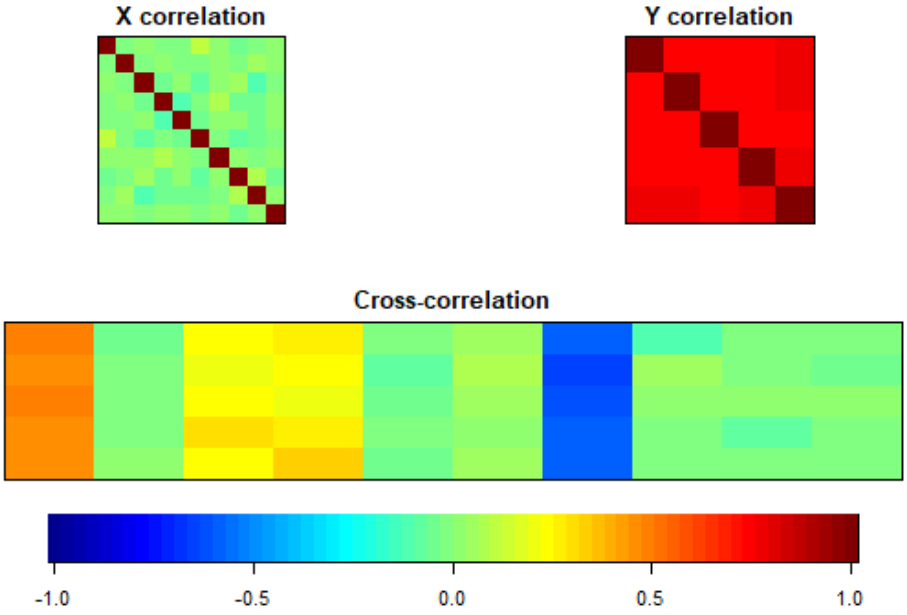
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	26, 59, 93, 123, 141, 257, 370, 427, 438	157, 202, 312, 345, 417	NA
2	Univariate check	NA	26, 93, 123, 370	157, 202, 312, 345, 417	NA
3	Univariate	NA	26, 93, 123, 370	157, 202, 312, 345, 417	163.31
4	Genetic Algorithm	-669.9492	26, 93, 123, 370, 406	157, 202, 312, 345, 417	4.92
5	Differential Evolution	-667.7773	26, 93, 123, 370	157, 202, 312, 345, 417	4.65
6	Particle Swarm Optimization	-667.7773	26, 93, 123, 370	157, 202, 312, 345, 417	4.28

(a) Results for 500×500 matrices with few small associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



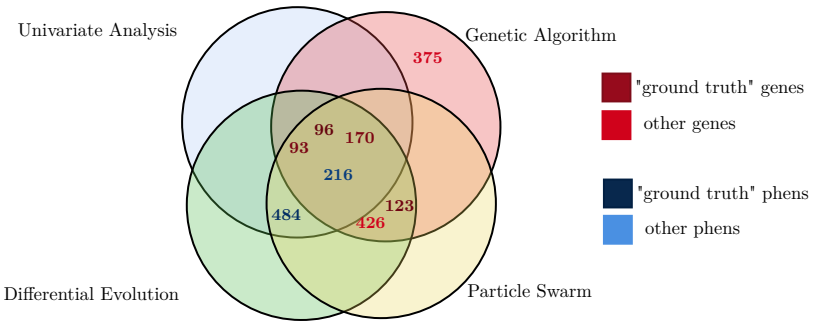
(b) Venn diagram of all genes and phenotypes appearing in table (a)



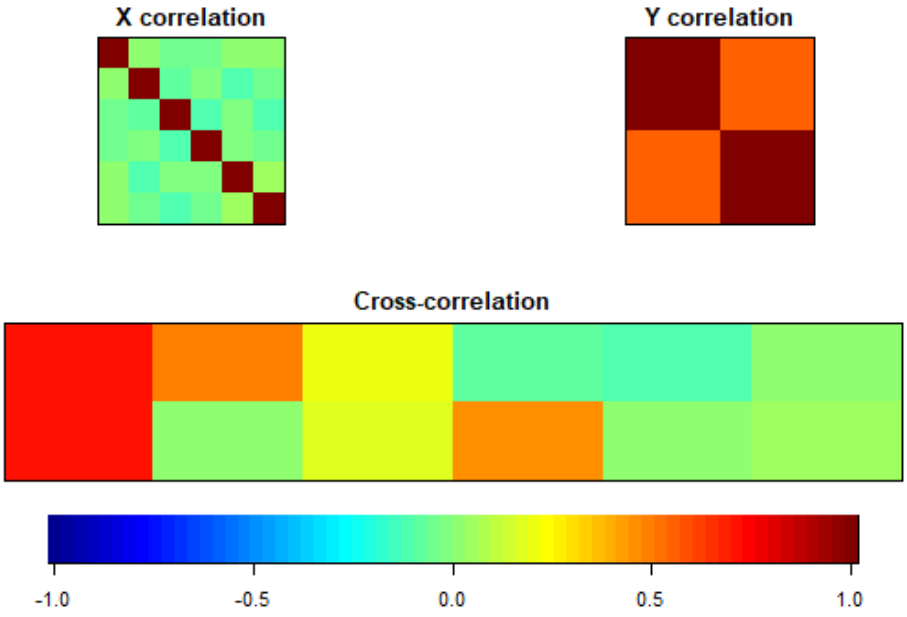
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	93, 96, 123, 170	216, 484	NA
2	Univariate check	NA	93, 96, 123, 170	216, 484	NA
3	Univariate	NA	93, 96, 170	216, 484	167.98
4	Genetic Algorithm	-388.9400	93, 96, 123, 375	216	5.06
5	Differential Evolution	-665.9556	93, 96, 123, 170, 426	216, 484	5.36
6	Particle Swarm Optimization	-665.9556	93, 96, 123, 170, 426	216, 484	5.16

(a) Results for 500×500 matrices with few strong associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



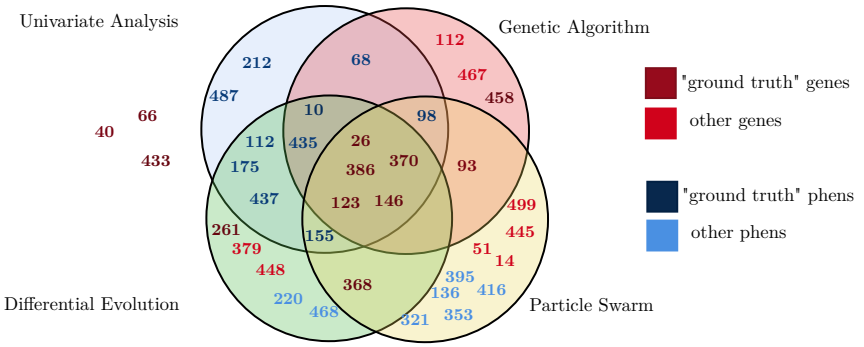
(b) Venn diagram of all genes and phenotypes appearing in table (a)



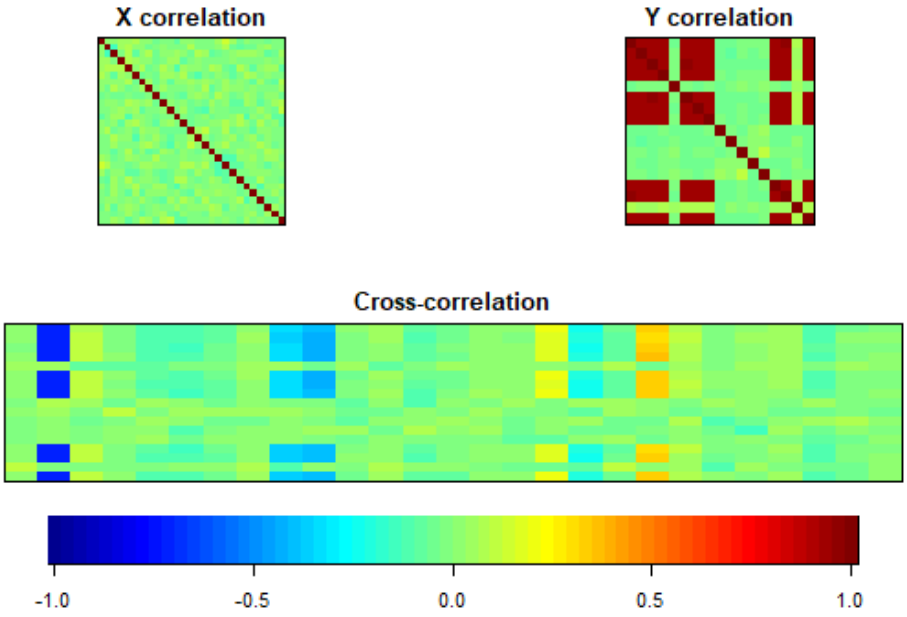
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	26, 40, 66, 93, 112, 122, 123, 146, 183, 218, 261, 279, 301, 318, 368, 370, 386, 433, 450, 458	10, 68, 98, 112, 155, 175, 212, 435, 437, 487	NA
2	Univariate check	NA	26, 40, 66, 93, 123, 146, 261, 368, 370, 386, 433, 458	10, 68, 98, 112, 155, 175, 212, 435, 437, 487	NA
3	Univariate	NA	26, 123, 146, 370, 386	10, 68, 98, 112, 155, 175, 212, 435, 437, 487	173.91
4	Genetic Algorithm	-675.8042	26, 93, 112, 123, 146, 370, 386, 458, 467	10, 68, 98, 435	4.94
5	Differential Evolution	-Inf	26, 123, 146, 261, 368, 370, 379, 386, 448	10, 112, 155, 175, 220, 435, 437, 468	4.30
6	Particle Swarm Optimization	-Inf	14, 26, 51, 93, 123, 146, 368, 370, 386, 445, 499	98, 136, 155, 321, 353, 395, 416	4.00

(a) Results for 500×500 matrices with a lot of small associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



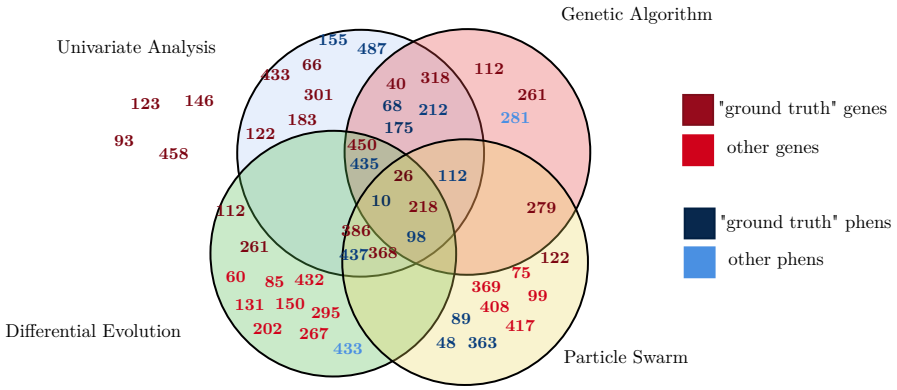
(b) Venn diagram of all genes and phenotypes appearing in table (a)



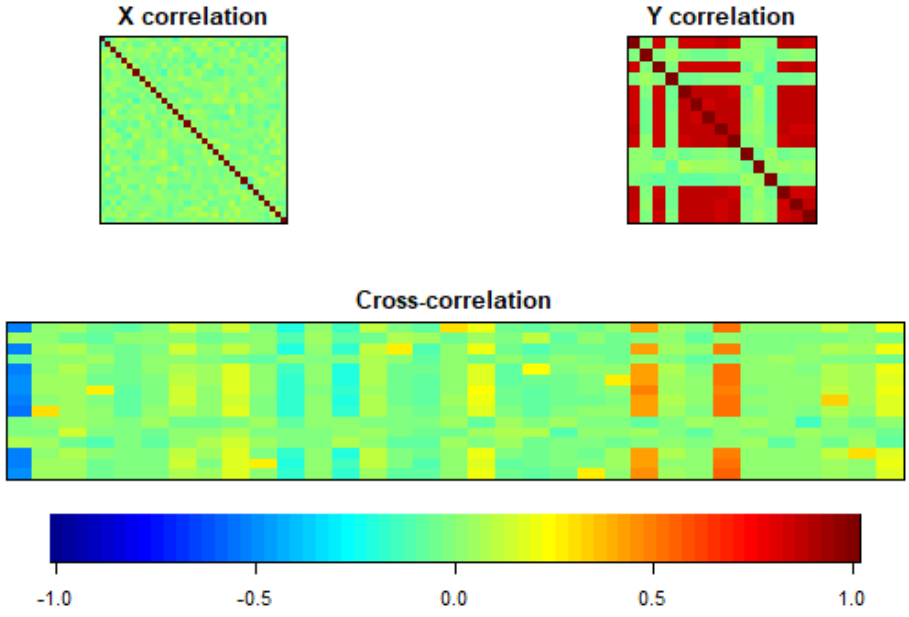
(c) Correlation structure of all genes and phenotypes appearing in table (a)

	Algorithm	Best p-value	Genes	Phenotypes	Running Time
1	Construction	NA	26, 40, 66, 93, 112, 122, 123, 146, 183, 218, 261, 279, 301, 318, 368, 370, 386, 433, 450, 458	10, 68, 98, 112, 155, 175, 212, 435, 437, 487	NA
2	Univariate check	NA	26, 40, 66, 93, 112, 122, 123, 146, 183, 218, 261, 279, 301, 318, 368, 386, 433, 450, 458	10, 68, 98, 112, 155, 175, 212, 435, 437, 487	NA
3	Univariate	NA	26, 40, 66, 122, 183, 218, 301, 318, 368, 386, 433, 450	10, 68, 98, 112, 155, 175, 212, 435, 437, 487	150.46
4	Genetic Algorithm	-Inf	26, 40, 112, 218, 261, 279, 318, 450	10, 68, 98, 112, 175, 212, 281, 435	5.09
5	Differential Evolution	-Inf	26, 60, 85, 112, 131, 150, 202, 218, 261, 267, 295, 368, 386, 432, 450	10, 98, 433, 435, 437	5.22
6	Particle Swarm Optimization	-Inf	26, 75, 99, 122, 218, 279, 368, 369, 386, 408, 417	10, 48, 89, 98, 112, 363, 437	4.66

(a) Results for 500×500 matrices with a lot of strong associations. Best p-value is on log scale. Genes and phenotypes are referred to using their column index.



(b) Venn diagram of all genes and phenotypes appearing in table (a)



(c) Correlation structure of all genes and phenotypes appearing in table (a)

The table below summarizes the performance of the four methods on the simulated datasets. It seems that the three evolutionary algorithms are very similar in terms of performance and running time. We could not find any algorithm-specific characteristics, as the algorithms'

characteristics varied with the datasets (e.g. FPR of the genetic algorithms is high on 100×100 datasets but lower on 500×500 datasets). These differences might even be due to parameter calibration issues.

The FPR of the EAs is encouraging, however, their TPR are to be improved. An option could be to run all four methods, and discard the found genes and phenotypes, that are found by only one method. This would significantly increase TPR in most cases, without raising the FPR by much.

Univariate analysis has performed significantly better than the EAs, most notably with a 0 FPR. This questions the advantages of CCA over univariate anaylsis. However, the success of univariate analysis might be explained by the simulated datasets being too simple. Adding more noise might have made the CCA-based methods more accurate. However, this would make it harder to determine the "ground truth" associations.

		Univariate Analysis	Genetic Algorithm	Differential Evolution	Particle Swarm Optimization
1	True Positive Rate	0.76	0.69	0.63	0.64
2	False Negative Rate	0.24	0.31	0.37	0.36
3	True Negative Rate	1.00	0.97	0.98	0.98
4	False Positive Rate	0.00	0.03	0.02	0.02

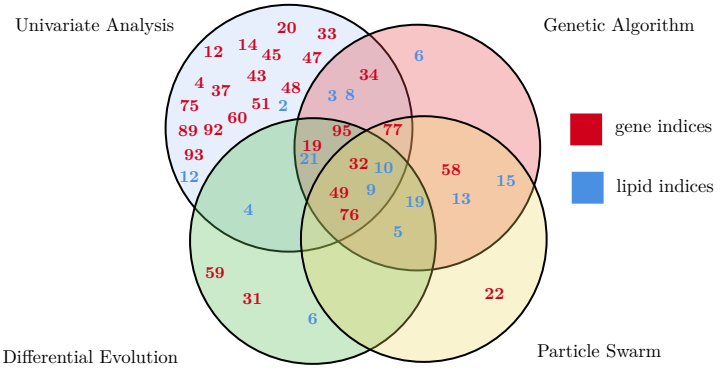
Figure 9: The performance of the four methods on all simulated datasets

4.2.2 Real Data

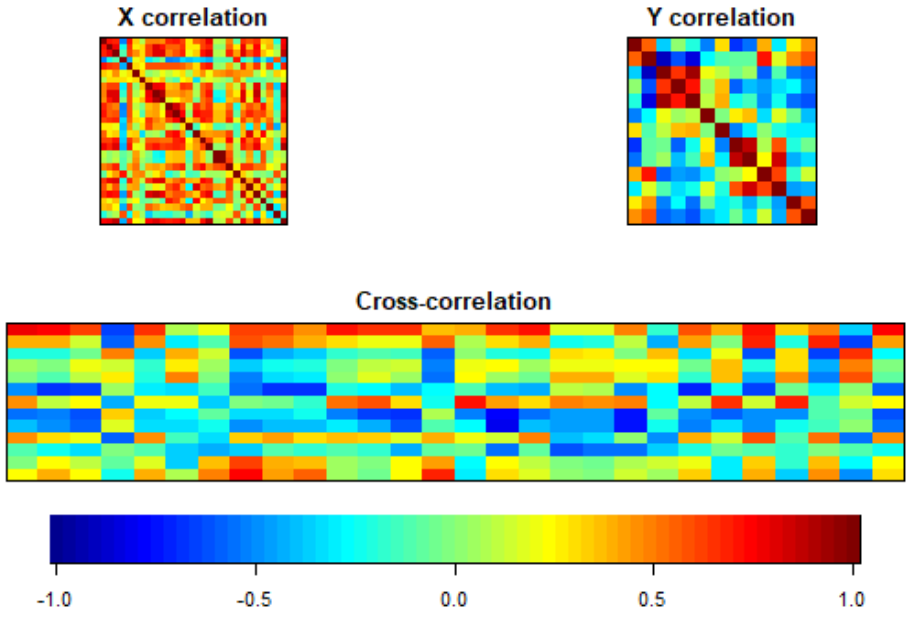
The algorithms were also tested on the heavily-correlated nutrimouse dataset from the whitening R package. The X matrix is the gene dataset in nutrimouse, collecting gene expression of 120 genes in liver tissue for 40 mice. The Y matrix is defined as the lipid component, which collects concentrations of 21 lipids for the same 40 mice.

Algorithm	Best p-value (log scale)	Genes	Lipids	Running Ti
Univariate	NA	4, 12, 14, 19, 20, 32, 33, 34, 37, 43, 45, 47, 48, 49, 51, 60, 75, 76, 77, 89, 92, 93, 95	2, 3, 4, 8, 9, 10, 12, 21	5.27
Genetic Algorithm	-78.04711	19, 32, 34, 49, 58, 76, 77, 95	3, 5, 6, 8, 9, 10, 13, 15, 19, 21	5.81
Differential Evolution	-64.16611	19, 31, 32, 49, 59, 76, 95	4, 5, 6, 9, 10, 15, 21	6.03
Particle Swarm Optimization	-69.96739	22, 32, 49, 58, 63, 76, 77	5, 9, 10, 13, 15, 19	4.69

(a) Performance of the algorithms on the nutrimouse dataset.



(b) Venn diagram of all genes and phenotypes appearing in table (a)



(c) Correlation structure of all genes and phenotypes appearing in table (a)

4.3 Discussion

It is worth to mention some of the challenges encountered in the project, that can also serve as directions of future research.

4.3.1 Challenges of the Simulation

The challenge of the evaluation of our methods is largely due to the unsupervised nature of the problem. When simulating data, there were two objectives to respect: the simulated data should have an association structure that is easy to control, in other words, there should be a "ground truth" that we can compare the results to. However in the same time, the produced data should be complex enough (many within and cross-correlations), so that the advantages of CCA against univariate analysis are apparent.

As discussed before, it seems that our simulations fall into the "too simple" category, where univariate analysis is performing comparably, and in some cases even better than CCA. Therefore, as an improvement, the simulation of more complex datasets could be explored in the future.

4.3.2 Correction Measures

Due to testing multiple hypotheses at once, correction measures must be applied on the p-value. Seoane et al. [1] used a Bonferroni correction to find the real threshold p-value. The number of hypotheses tested were estimated from the maximum number of genes and phenotypes found in an association rule (an association rule is a combination of a number of phenotypes associated with a number of genes), after many repetitions of the genetic algorithm. For example, if out of 100 genes and 100 phenotypes, the maximum number of genes observed in an association rule are 15, and the maximum number of phenotypes is 10, then the adjusted threshold p-value is given by

$$p = \frac{0.05}{\sum_{k=1}^{15} \binom{100}{k} \cdot \sum_{k=1}^{10} \binom{100}{k}}$$

To my understanding, this method only gives an upper bound (and we would need a lower bound) on the threshold p-value, and it is nontrivial to assess the strength of this upper bound. In fact, it is hard to imagine how a strong lower bound could be theoretically obtained, since the algorithms have no cap on the number of genes or phenotypes they can select in an association rule.

For these reasons, we did not use any correction on the p-values in our evaluation study, and instead compare the selected features to the results of univariate analysis.

References

[1] Seoane JA, Campbell C, Day INM, Casas JP, Gaunt TR (2014). Canonical Correlation Analysis for Gene-Based Pleiotropy Discovery. *PLoS Comput Biol* 10(10): e1003876. https://doi.org/10.1371/journal.pcbi.1003876

[2] Hastie T, Tibshirani R, Friedman J (2009). *The elements of statistical learning: data mining, inference, and prediction.* 2nd ed. Available at https://web.stanford.edu/~hastie/Papers/ESLII.pdf?fbclid=IwAR1SZLgs6S6H8l9rw2TrEBia24kxmkTUAPuqP0uenhlaKwRyhow3MEdKoWk

[3] Evangelou M. OMICS Integration. [Lecture notes] In: *MATH97135 - Statistical Genetics and Bioinformatics 2020-2021.* Imperial College London. 2021.

[4] Turgeon M. Canonical Correlation Analysis. [Lecture notes] In: *STAT 4690–Applied Multivariate Analysis.* University of Manitoba. 2020.

https://www.maxturgeon.ca/f19-stat4690/slides/canonical-correlation-analysis.pdf

[5] Momeni Z, Hassanzadeh E, Abadeh M S, Bellazzi R (2020). A survey on single and multi omics data mining methods in cancer data classification. *Journal of Biomedical Informatics* 107 (103466). Available from: https://www.sciencedirect.com/science/article/pii/S1532046420300939

[6] Elbeltagi E, Hegazy T, Grierson D (2005). Comparison among five evolutionary-based optimization algorithms. *Advanced Engineering Informatics* 19 (1), 43-53. Available from: https://www.sciencedirect.com/science/article/pii/S1474034605000091

[7] Corne D, Lones MA (2018) Evolutionary Algorithms. In: Martí R, Panos P, Resende M (eds) *Handbook of Heuristics.* Springer, Cham. https://doi.org/10.1007/978-3-319-07153-4_27-1

[8] Conceicao E L T, Maechler M (2021). Package 'DEoptimR'. Available at: https://cran.r-project.org/web/packages/DEoptimR/DEoptimR.pdf

[9] Bendtsen C (2012). Package 'pso'. Available at: https://cran.r-project.org/web/packages/pso/pso.pdf