Optimizing clustering metrics for patient stratification using heuristic approaches

Submitted By : Anjana BHAT

Supervisor : John Zobolas

Co-Supervisor : Alberto López

Advisor : Michel RIVEILL

Master of Data Science and Artificial Intelligence (MSc2)

University of Cote d'Azur

19 August 2024

Abstract

With the advancement of personalized medicine, integrating survival analysis with clustering techniques and multi-modal data has emerged as a promising avenue for identifying patient subgroups with distinct prognostic characteristics. However, the computational complexity of finding an optimal patient stratification that maximizes the difference in survival outcomes, even in cases with two groups, poses a significant challenge. This master's internship aims to address this challenge by developing heuristic approaches tailored to optimize patient clustering based solely on survival information. The primary objective is to leverage heuristic algorithms to identify a partition that maximizes relevant metrics such as the log-rank statistic, thereby offering valuable insights into patient stratification and prognosis. Furthermore, this methodology facilitates the establishment of an adjusted statistical score for assessing the performance of clustering models, enabling more interpretable benchmarking against a heuristic-driven optimal maximum value.

Various algorithms, including KMeans and Multiview KMeans, were utilized alongside a Genetic Algorithm (GA) developed as a metric-optimization method. The GA, which uses the log-rank test for measuring the difference in survival between patient groups, provided a heuristic benchmark for normalizing the scores of other methods. Additionally, Random Search (RS), a simple algorithm that generates random clustering assignments and obtains log-rank statistics for those assignments, was implemented as a baseline for comparison. Moreover, a multi-objective optimization approach has been implemented within the GA framework, considering metrics such as silhouette score and chi-square to incorporate multi-omics and other clinical factors association with patient stratification. A comprehensive comparison between single-objective and multi-objective optimization strategies has been performed. Overall, the results suggest that the GA leaves room for improvement in several clustering algorithms and serves as a good enough "oracle" algorithm to identify the best possible maximum for scores like the log-rank. It has also been determined that the single-objective strategy significantly outperforms the multi-objective strategies, whereas multi-objective strategies highlight the trade-offs between different optimization strategies.

Keywords : Clustering Techniques, Survival Analysis, Patient Stratification, Log-rank Statistics, Genetic Algorithm, Clustering Metrics, Explainable AI (XAI)

Contents

1 Introduction

1.1 Survival Analysis : Concepts and Challenges

Survival analysis encompasses a set of statistical methods designed to analyze time-to-event data, where the primary outcome of interest is the duration until a specified event occurs. Although historically focused on mortality, survival analysis is applicable to a variety of events, including disease onset, relapse, recovery or competing risks. Survival studies usually measure the time from a defined starting point, such as study enrollment, to the occurrence of the event, with durations expressed in years, months, weeks, or days.[1]

One of the major challenges in survival analysis is dealing with censored data. Censoring occurs when the event of interest has not yet happened by the end of the study period or when participants leave the study prematurely, resulting in incomplete observations. To manage this issue, survival analysis employs data analysis techniques such as life tables and the Kaplan–Meier (KM) estimator or even more advanced models such as the Cox's Proportional Hazards model [2]. These methods provide descriptive insights into survival times while accounting for censored observations, enabling researchers to estimate survival functions and understand the distribution of survival times in a population.[1]

Survival analysis is pivotal in cancer research, especially when confronted with censored data, where complete survival times are frequently unavailable. Estimating the survival function, a primary goal of such analysis, entails predicting the likelihood of a patient surviving past specific time points [3].

1.2 Limitations of Traditional Clustering Methods

Due to the absence of labels for patient data, recent years have seen significant advancements in clustering techniques to identify subgroups. Multi-modal data, encompassing diverse data types like clinical and molecular data, are frequently employed in this context [4], [5]. Clustering is an unsupervised learning method that groups similar data points based on their intrinsic features, without requiring predefined labels. This technique is particularly valuable when labels are not available, as it enables the identification of natural patterns and groupings within the data, facilitating deeper analysis and insight discovery. Traditional clustering methods, such as KMeans, often fall short in addressing the multifaceted nature of patient data, which include various clinical factors and molecular data. These methods may not adequately capture the intricate relationships between the different data types, leading to suboptimal patient stratification [6]. Multi-view clustering algorithms integrate data from multiple sources to overcome limitations of traditional methods but have drawbacks such as high computational overhead, algorithmic complexity, and issues with the 'curse of dimensionality'. This issue arises because many datasets are sparse, containing few observations (e.g., patients) but thousands of molecular features, which can lead to model overfitting and biased estimates of generalization performance. In such high-dimensional settings, models may struggle to balance the contribution of each view, leading to biased results and suboptimal patient stratification [7]. Lastly, existing approaches may struggle to find patient clusters associated with survival outcomes, as there might not be enough signal in the molecular data to extract meaningful associations between biological and clinical phenotypes.

1.3 Refining Patient Stratification Through Survival Analysis and Clustering

Building a clustering technique based on survival analysis and incorporating multi-modal data offers a promising approach for identifying patient subgroups with distinct prognostic characteristics. This approach capitalizes on the rich information provided by various data modalities to refine patient stratification and prognosis assessment. By including survival analysis with clustering methods, researchers can gain deeper insights into the underlying molecular mechanisms driving patient outcomes and assess if the patient stratification models are good enough, meaning that they effectively represent or distinguish molecular or clinical features, such as differences in survival. Additionally, leveraging multi-modal data enables a more comprehensive characterization of patient subgroups by integrating various types of molecular and clinical information that single-omics approaches cannot provide. This holistic view enhances our ability to tailor treatment strategies more precisely, potentially leading to improved patient outcomes.

Building on previous work, we aimed to enhance the interpretability of clustering metrics by incorporating survival analysis with multi-modal data, providing clearer insights into patient stratification. To achieve this, we aimed to maximize relevant metrics such as the log-rank statistic or equivalently, to minimize the p-value of the metrics considered, which can be framed as an optimization problem. One approach we employed was the use of a Genetic Algorithm (GA) due to its effectiveness in addressing complex optimization problems. The GA allows us to explore large search spaces and find optimal or near-optimal solutions for clustering, thereby ensuring the most meaningful patient subgroup divisions. We used this as a metric-optimized method with the default metric set to the log-rank test to find the patient clusters that correspond to larger survival differences through single-objective optimization (SOO). Additionally, a multi-objective optimization (MOO) approach was implemented within the GA framework, considering additional metrics such as silhouette score and chi-square test to incorporate multi-omics and other clinical factors' association with patient stratification. This ensured that the clustering not only optimized survival differences but also reflected the underlying biological and clinical diversity.

2 Related Works

To obtain clusters using omics data, many approaches have been developed such as KMeans, Spectral clustering, JIVE, Similarity Network Fusion (SNF) and so on [8]. Most of these tasks can be achieved using clustering approaches in scikit-learn libraries for single omics data. For multi-omics data, the snfpy package [9] can be used to obtain affinity networks, while the MOVICS package [10] provides a comprehensive pipeline for multi-omics integration and clustering. Once the clusters are obtained, the Kaplan-Meier estimator and the log-rank statistic is frequently employed to evaluate the survival differences between the clusters. In Lopez et al. (2024) [11], a deep clustering model was developed for PDAC patient stratification using integrated methylation (METH) and gene expression (GEX) data, emphasizing model explainability. The model identified two patient subgroups with different prognoses and biological factors. Follow-up analyses underscored the importance of DNA methylation in the clustering outcomes, demonstrating the model's ability to learn complex patterns through specialized neurons for individual or combined data modalities. This study advances explainable AI in clinical applications. GAs have been employed in optimization problems for many years. Two notable Python libraries that facilitate the use of GA are PyGAD and PyMoo.

PyGAD is an open-source, user-friendly Python library designed for building GAs. It provides extensive control over various aspects of the GA process, including population size, gene value range, gene data type, parent selection, crossover, and mutation. PyGAD is versatile and can be used for general optimization tasks, allowing users to define custom fitness functions. The typical workflow with PyGAD involves three main steps: defining the fitness function, creating an instance of the pygad.GA class, and executing the pygad.GA.run() method. PyGAD also supports the training of deep learning models, whether built within the library itself or using frameworks like Keras and PyTorch. [12]

PyMoo is a Python framework focused on MOO. It supports a wide range of optimization algorithms such as Pattern Search, Differential Evolution, including GAs. PyMoo offers customizable implementations and allows users to extend algorithms by providing custom operators. The framework includes a variety of single-objective, multi-objective, and many-objective test problems. It also features tools for automatic differentiation, parallelization of function evaluations, visualization of both low and high-dimensional spaces, and multi-criteria decision making.[13]

Building on these advancements, we focused on GA as a promising evolutionary computation method for optimizing clustering. GA is relevant in clustering because it can efficiently search large solution spaces and find clusters that optimize a given objective function. In the context of clustering using survival analysis, GA can be employed to identify patient subgroups with distinct survival patterns by optimizing cluster assignments to enhance the separation of Kaplan-Meier survival curves.

3 Objective

This Master internship is being conducted at the Oslo University Lab, Norway, within the Computational Systems Medicine group of the Cancer Genetics Department. The group specializes in integrating multi-omics profiling and clinical information from cancer patients using mathematical and statistical approaches, such as machine learning and network modeling. The primary medical objective is to optimize treatment outcomes for individual patients by developing maximally predictive models and minimal biomarker signatures that enable real-time and cost-effective routine diagnostics and prognosis.

The objective of this internship is to explore heuristic methodologies for solving the optimization problem. This involves investigating various optimization techniques, such as genetic algorithms, gradient-based methods, or Bayesian optimization, to efficiently navigate the patient stratification space. Additionally, the methodology will be implemented as a Python package to make it accessible and usable for further research and applications. Finally, several multi-modal clustering models will be benchmarked using the proposed adjusted metric, aiming to assess their performance and suitability for real-world clinical applications.

To achieve these objectives, a GA has been developed as a metric-optimized method, incorporating both SOO and MOO. For comparison, several clustering approaches have been considered, including single-view clustering algorithms, multi-view clustering algorithms, and deep clustering algorithm [11]. Additionally, to provide a baseline measure, a Random Search (RS) approach has been implemented. This straightforward method involves generating random clustering assignments and evaluating their log-rank statistics. These methods have been applied to two datasets, GBM and PAAD, utilizing clinical data, GEX, and METH data. Detailed information on these methodologies and datasets can be found in the subsequent sections of this report.

4 Methods

4.1 Data Overview and Preparation

4.1.1 Dataset Description

The two datasets considered in this project are GBM and PAAD. For GBM, GEX and METH data were obtained from the Broad GDAC Firehose, with clinical information sourced from TCGA. For PAAD, preprocessed data ready for analysis was obtained from [14].

- **GBM**:

  One of the datasets used in this project is the TCGA GBM dataset, which includes data on patients with glioblastoma multiforme (GBM) and was selected for its relatively low overall censoring rate [15]. In survival analysis, censoring refers to instances where the outcome event (e.g., death or disease progression) has not occurred by the end of the study period, making the exact survival time unknown [15]. Censored data are typically represented with a value of 0, indicating that the event has not yet occurred, while non-censored data are represented with a value of 1, signifying that the event has occurred. Less censoring generally means more complete survival data, which enhances the reliability of survival analysis.

  Among the 541 patients in the TCGA GBM dataset, 22.68% had censored data. After focusing on the 57 patients for whom both GEX and METH data are available, only 38.6% of this subset had censored data. The figure 1 illustrates the censoring rate within this subset, showing the sparse occurrence of censoring events marked by red crosses. Despite this, the majority of individuals in this dataset exhibit complete survival information. Consequently, the methods used or developed in this study are applied specifically to these 57 patients to ensure accurate integration of both clinical and omics data.

  Further, the GEX data initially contained 60,660 features, and the METH data had 172,788 features. To prepare the data for analysis, both datasets were reduced to 1,000 features each through preprocessing steps, which included feature selection and dimensionality reduction techniques. Detailed descriptions of these preprocessing steps are provided below.

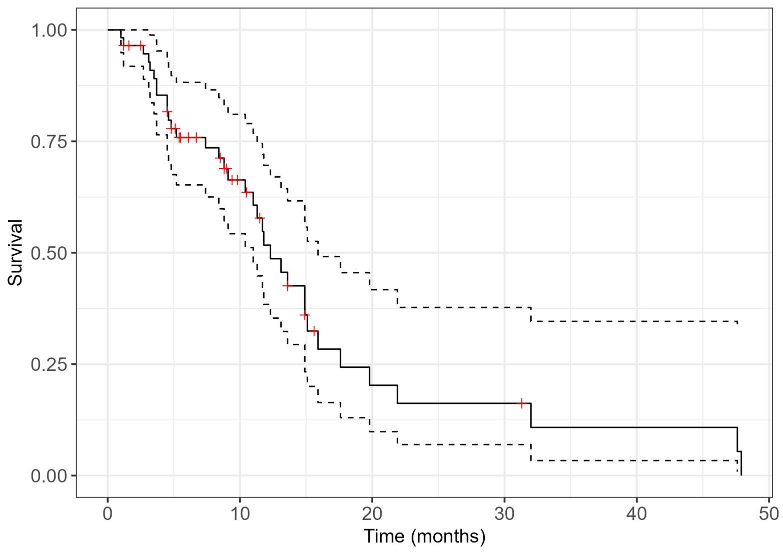

Figure 1: Kaplan-Meier survival curve for the TCGA GBM dataset, showing 38.6% censoring among 57 patients

- **PAAD** :
  Pancreatic Ductal Adenocarcinoma (PAAD) is the most prevalent type of pancreatic cancer, accounting for over 80% of all pancreatic cancer cases [16]. It is characterized by high mortality rates and a very poor prognosis [16]. The PAAD dataset includes clinical and omics data for 108 patients, with 44.44% of the data being censored, as illustrated in Figure 2.

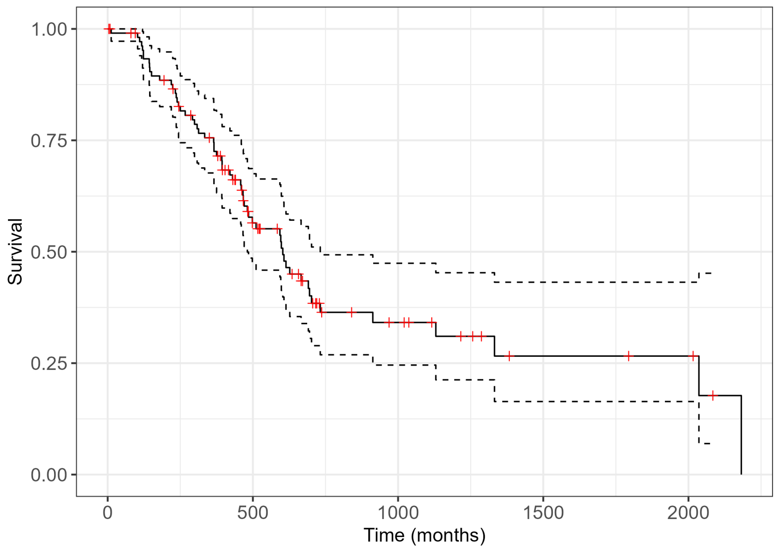

Figure 2: Kaplan-Meier survival curve for the TCGA PAAD dataset, showing 44.44% censoring among 108 patients

Initially, the GEX data contained 20,531 features, while the METH data had 22,537 features. The number of features in each dataset was reduced to 1,000 features after preprocessing steps.

4.1.2 Preprocessing

For the GEX features, those with zero values in more than 20% of patients were removed to eliminate features with minimal variability. Next, features were filtered to retain the 10% most variable ones based on mean absolute deviation. Highly correlated features with a Pearson correlation coefficient higher than 0.85 were also removed to mitigate redundancy. A log transformation was applied to stabilize variance and improve normality of the data distribution, which is beneficial for many statistical methods. Z-score normalization was performed to standardize the features, ensuring they have a mean of zero and a standard deviation of one, facilitating comparison across features. Finally, factor analysis was used to transform the data into a smaller set of latent factors that capture the most significant patterns, retaining 1,000 features that represent the underlying structure of the data.

For the METH features, those with NaN values in more than 20% of patients were removed to ensure data completeness. Subsequently, features were filtered to retain the 10% most variable ones based on mean absolute deviation. Non-Negative Matrix Factorization (NMF) was then applied to transform the data into a smaller set of latent factors, capturing the most significant patterns and retaining 1,000 features that reflect the underlying structure of the data. Z-score normalization was performed to standardize the features, ensuring a mean of zero and a standard deviation of one, which facilitates better comparison across features.

For single-view algorithms, the preprocessed GEX and METH data were concatenated, and FastICA was applied to retain 50 components for improved performance. In contrast, for multi-view algorithms, the preprocessed GEX and METH data were used separately, as obtained after the preprocessing steps. Also, it is important to note that, while using GA with MOO on omics data, we have considered only 50 features in each of the GEX and METH datasets. This is because GA performs a preprocessing step inside the algorithm to select and optimize a manageable subset of features, thereby reducing computational complexity and improving the efficiency of the optimization process. By limiting the feature set to

50, we balance the need for comprehensive data representation with the practical constraints of running the algorithm effectively on high-dimensional omics data.

4.2 Methodology

In this section, we discuss a list of implemented algorithms for identifying optimal patient clusters using only survival labels, without incorporating any omics data. The implementation of RS as a baseline method is explored, alongside the GA developed in this project. RS is employed due to its simplicity and ease of implementation, whereas GA is designed as a metric-optimized method, utilizing both SOO and MOO to enhance patient stratification based on survival analysis. An overview of the standard algorithms employed in this project is also included.

4.2.1 Implemented Algorithms

- **Random Search**:

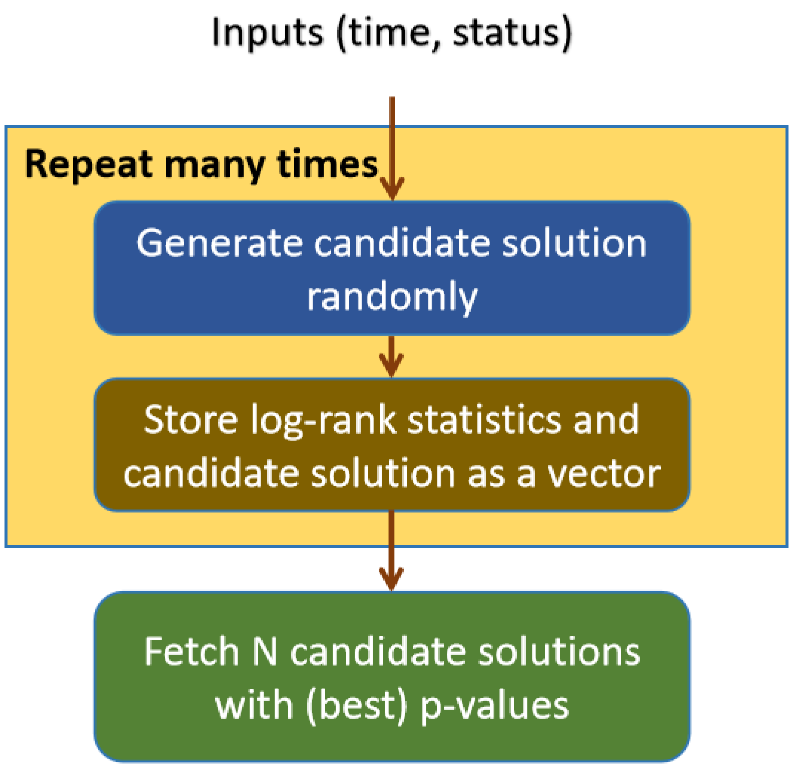
Inputs (time, status)
Repeat many times
Generate candidate solution randomly
Store log-rank statistics and candidate solution as a vector
Fetch N candidate solutions with (best) p-values

Figure 3: Workflow of Random Search method

To obtain the best patient stratification, ideally, one would need to evaluate all possible candidate solutions, which is infeasible in practice due to the exponential growth in combinations (e.g., 2^{20} for 20 samples). Therefore, a simpler approach is to randomly generate a certain number of candidate solutions and then calculate the log-rank statistics to identify the best clusters. A candidate solution refers to a potential grouping or clustering of patients generated during the patient stratification process. In our case, a candidate solution is represented as an integer vector, with each element indicating whether a patient belongs to cluster 1 or cluster 2. It is worth noting that our analysis focuses on two clusters, although the methodology can be applied to any number of clusters. We refer to this approach as the Random Search (RS) method, as illustrated in Figure 3.

RS utilizes patient data comprising ”Time” and ”Status” as inputs. Here, ”Status” is a binary indicator where 1 represents an event such as death, and 0 denotes censoring, meaning the event did not occur by the end of the study period. ”Time” refers to the number of months until the specified event. The RS method generates a set of random clustering assignments, calculates the log-rank statistics for each assignment, and evaluates the p-values to assess the statistical significance of the clustering solutions. However, this might not be feasible all the time because RS, while not as computationally intensive (e.g., running it for 1000 iterations), has the inherent disadvantage of being unguided by any optimization criterion. It essentially has a computational complexity of $O(\infty)$, meaning it may never find the best solution regardless of how long it runs.

- **Genetic Algorithm** :
  GAs are a powerful optimization technique inspired by the principles of genetics and natural selection. They are designed to solve complex problems by mimicking biological processes. In addition to optimization, GAs are also used in machine learning, research, and development. GAs operate analogously to biological evolution, using operations such as selection, crossover, and mutation to evolve solutions over successive generations. The process begins with a randomly generated population of potential solutions, which are then refined through genetic operations[17]. Compared to RS, GA, as shown in Figure 4, offer some theoretical guarantees for convergence to local optima, making it a more effective approach for patient stratification.

  The detailed steps of the GA implemented in this project are as follows:

1. Initialize Population : The first step in the process is setting up the initial population, where individuals are provided with a search space. Thus, a certain number of candidate solutions are generated, referred to as the initial population. As mentioned earlier, each candidate solution represents a clustering assignment of the patients, thus will be integer representation.
2. Fitness Function : A fitness function is used to assess the fitness level of each candidate solution. The function returns a fitness score for each candidate solution, indicating the chance of being selected for reproduction. The default fitness function provided here is the log-rank statistics where p-value is calculated and is used in SOO. Smaller the p-value, the chance of selection of candidate solution will be higher. There is also opportunity to select either log10 of p-value or the corresponding log-rank test-statistic, where the optimization aims at maximizing the fitness score. In MOO, user can select one of the following :
   - log-rank + Tarone-Ware + Wilcoxon (also known as Breslow) tests : when only "Time" and "Status" data are available
   - log-rank + Chi-square : when "Time", "Status" and any other clinical factors such as sex are available
   - log-rank + Silhouette or log-rank + Calinski-Harabasz score : when "Time", "Status" and omics data are available

   If omics data is available, the GA first preprocesses each omics dataset separately by standardizing the data and then applying Principal Component Analysis (PCA), where the number of components is determined using the 'mle' (Maximum Likelihood Estimation) method. After preprocessing the individual omics datasets, the different omics are concatenated, followed by another round of standardization and PCA with 'mle'. Since PCA with 'mle' requires that the number of samples be greater than or equal to the number of features, it is crucial to ensure this condition is met for each omics type. This preprocessing step allows for efficient data transformation and dimensionality reduction before passing the data to the algorithm. Users can perform these preprocessing steps in advance, as detailed in the 4.1.2 section, ensuring that the data is optimally prepared for subsequent analysis.
3. Stopping Criterion : Once the fitness scores are obtained for all the candidate solutions, stopping criterion is checked. In SOO, if the difference

between the current fitness score (curr) and the best fitness score (best) is less than epsilon (eps), a very small positive number, for a specified number of consecutive iterations (n), the algorithm terminates, returning the best score and the best candidate solution. If the user selects p-value for the fitness function, the negative logarithm (base 10) of the p-value (-log10(p)) is considered in stopping criterion for arithmetic stability. This is because, very small p-values (e.g., 10^{-30}) can make it difficult to set an appropriate epsilon, as calculations might exceed what is feasible for machine precision.

Further, we have utilized the Pareto front for handling MOO problems due to its ability to provide a diverse set of optimal solutions. The Pareto front helps to identify solutions where no single option is superior across all objectives. By focusing on this set of non-dominated solutions, one can effectively balance trade-offs between competing objectives and select solutions that best meet the project's goals. This approach ensures that we explore a range of high-quality solutions, rather than converging on a single outcome, thereby improving the robustness and flexibility of our optimization process [18].Thus, to ensure the effectiveness of the optimization process, we employ a stopping criterion that halts the algorithm when the Pareto front stabilizes, indicating convergence. This process involves comparing the current Pareto front to the previously best-known Pareto front. If the two fronts are identical, or sufficiently close within a defined tolerance, it suggests that the solutions have not significantly changed over successive generations. Stability is tracked by maintaining a count of unchanged generations; if the front remains stable for a predetermined number of generations, the algorithm is considered to have converged and can be terminated.

If the criterion is not met in either case, a new population is generated using reproduction steps, and the fitness scores are recalculated.

4. Reproduction : Reproduction comprises three steps namely selection, cross-over and mutation.
   - Selection : In SOO, a defined percentage of the population with the minimum p-values or maximum test-statistic are selected based on the fitness scores. Whereas, in MOO, candidate solutions are selected based on Pareto dominance. To determine the Pareto front, each solution is compared to others to check if it is superior in at

least one objective and not worse in any. Once the Pareto front is identified, Pareto ranks are assigned to all solutions based on their dominance. Solutions on the Pareto front receive the highest rank, while others are ranked accordingly in subsequent iterations [19], [20]. Similar to SOO, a defined percentage of candidate solutions with top ranks are chosen. Using this selected population, crossover and mutation steps are performed to generate a new offspring population matching the initial population size. This process is done concurrently, utilizing half the number of processes needed, as each crossover and mutation step produces two offspring.

– Cross-over : Crossover, also known as recombination, is a GA operation used to combine the information of two parent solutions to generate new offspring. In this process, two candidate solutions are randomly selected from the chosen population and then combined to produce two new offspring solutions. This method allows the offspring to inherit characteristics from both parents, promoting diversity and potentially leading to improved solutions in future generations. Our GA employs multiple operators for performing crossover, including one-point, two-point, and uniform crossover, with one-point crossover as the default. In one-point crossover, a random index is selected, and the segments of the parents are swapped at this index to generate the offspring. In two-point crossover, two indices are selected, and the segments between these points are exchanged between the parents. For uniform crossover, two masks are generated; for child1, where mask1 equals mask2, it inherits the corresponding value from parent1, and where mask1 does not equal mask2, it inherits the value from parent2. Child2 is created using the inverse of this process.

– Mutation : Mutation involves making random changes to individual solutions. Our GA incorporates two different mutation operators namely flip-bit and swap, to maintain the diversity within the population. In flip-bit mutation, for each index, a random value is compared with the specified mutation rate. If the random value is less than the mutation rate, the value is flipped (i.e., 1 becomes 2 and vice versa). This flip-bit mutation also works for more than two clusters: a random cluster value, different from the current one, is selected and assigned and is the default method. In swap mutation, for each index, a random index is selected for swapping. If a random value is less than the mutation rate, the values at these indices are swapped.

After obtaining the offspring population, it is sent to the fitness function for evaluation and the process continues until the stopping criterion is met.

5. Convergence: When the stopping criterion is met, the GA returns the score distribution along with the candidate solutions from the final generation, which are considered the best set of solutions. Additionally, users can obtain a single best solution, configurable through a hyperparameter. While there is a risk of converging to local optima, this is mitigated by our focus on obtaining multiple high-quality candidate solutions, which reduces the impact of this potential drawback.

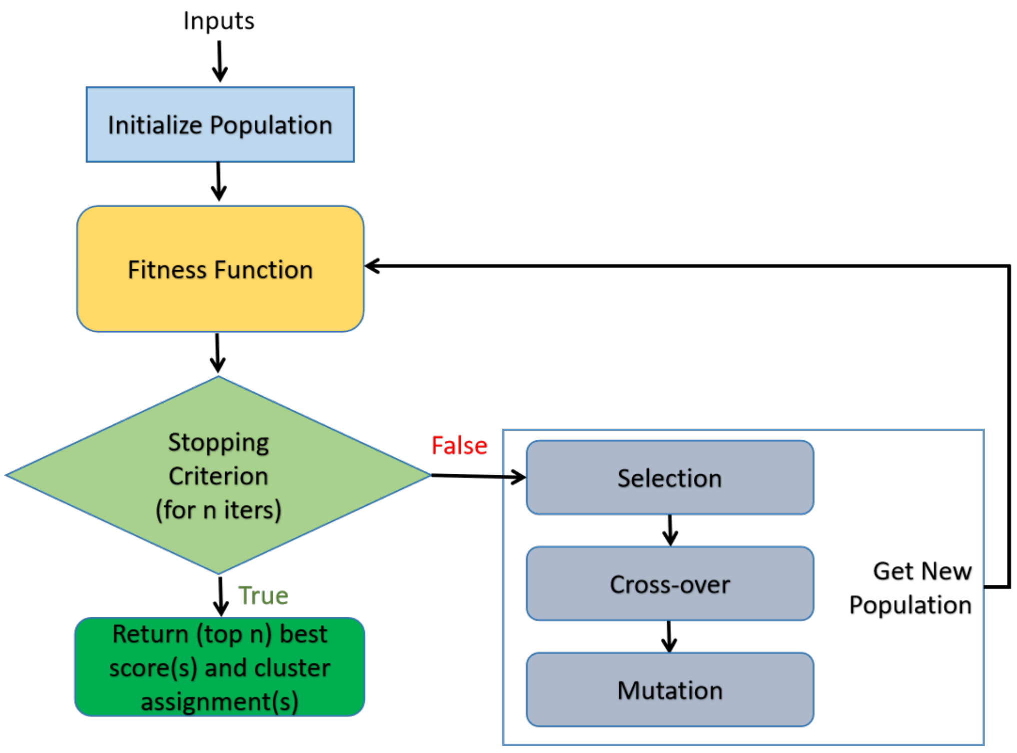

Figure 4: Genetic Algorithm used for optimizing clustering in patient stratification. The stopping criterion for n iters refers to the condition being met true for n successive iterations

- **Median-based Stratification**:
  The median is another effective method, particularly suited for our case since

the dataset has minimal censoring. We first calculate the median of the time data, then split the patient data into two groups based on this median. The p-value is subsequently calculated to assess the difference between the two groups. We name this method as median-based stratification.

4.2.2 Standard Algorithms

1. **Single-view algorithms** :
   Single-view algorithms consider only one type of data at a time. As mentioned earlier, we have concatenated both GEX and METH for these algorithms.

   - KMeans (KM): KM clustering partitions the data into a predefined number of clusters by minimizing the variance within each cluster. It iteratively assigns data points to clusters based on the nearest cluster centroid and updates centroids to the mean of assigned points.
   - Spectral Clustering (SC): SC uses the eigenvalues of a similarity matrix to perform dimensionality reduction before clustering in fewer dimensions. This method is effective for identifying clusters with complex shapes and non-convex boundaries.

2. **Multi-view algorithms** :
   These algorithms integrate multiple data types or views to improve clustering performance. MvLearn is a Python library designed for multi-view machine learning, providing tools to implement and evaluate multi-view algorithms [21]. Ajive, MultiviewKMeans, and MultiviewCoRegSpectralClustering fall under the category of multi-view algorithms supported by mvlearn. Additionally, deep clustering by Lopez et al.[11] was also considered for analysis.

   - Ajive : Ajive (Angle-based Joint and Individual Variation Explained) decomposes the data into joint and individual components to identify shared and unique structures in multi-view data. It enhances interpretability by separating common signals from view-specific variations.
   - MultiviewKMeans (MKm) : MKm extends traditional KM by integrating multiple data views to improve clustering performance. It optimizes a joint objective function that balances the contributions from each view, resulting in more robust and accurate clusters.

- MultiviewCoRegSpectralClustering (MReg) : MReg leverages co-regularization to enforce consistency between different data views in the spectral clustering process. By aligning the clustering results from each view, it ensures more cohesive and reliable multi-view clustering outcomes.
- Deep Clustering : The model integrates an autoencoder with a KM algorithm through a joint loss function. The multi-branch autoencoder encodes both methylation and gene expression profiles into a reduced-dimensional matrix. This encoded matrix serves as the input for the KM algorithm to cluster patient samples. By optimizing the joint loss function, the model effectively enhances both dimensionality reduction and clustering processes simultaneously.

4.2.3 Benchmark and Analysis Workflow

Below are the steps involved in the complete methodology for analysis:

- Algorithms: For single-view and multi-view algorithms, the first step involves passing the preprocessed multi-omics data as their input. Single-view algorithms take concatenated data of preprocessed GEX and METH, while multi-view algorithms take GEX and METH separately in a multi-view format. Once the cluster assignments are obtained, the log-rank test is applied on "Time" and "Status" data to obtain the p-value. For all algorithms except KMeans, the process is run only once since the obtained cluster assignments or p-values either remain the same or exhibit very slight changes that do not significantly impact the results. Further, for GA and RS, the input comprises "Time" and "Status". However, if MOO is desired in the case of GA, the user must also pass multi-omics data or other clinical factors, depending on the metrics used. RS returns a set of random solutions with their p-values, while GA provides the final population after termination along with its p-values, considered the best population distribution.
- Metric Normalization: Once the p-values from all algorithms are obtained, normalization is performed to add interpretability to the clustering metrics, as shown in Equation 1. Here, A represents any algorithm, Baseline represents RS due to its simplicity, and Best is considered GA due to its optimization to achieve the best possible metric, detailed in the results section. Baseline

and Best are randomly sampled from the RS and GA score sets, respectively, and a set of normalized distribution values (e.g., 1000) is generated to ensure unbiased normalization and robust comparison [22]. A negative normalized score indicates that the algorithm's performance is worse than the baseline. To normalize RS scores, the mean is subtracted to set their mean to 0, and to normalize GA scores, they are divided by their mean to set the mean to 1.

$$\frac{\text{score}_A - \text{score}_{\text{Baseline}}}{|\text{score}_{\text{Best}} - \text{score}_{\text{Baseline}}|} \tag{1}$$

- Comparison: After obtaining the normalized score distribution for all algorithms, density plots and 95% confidence interval (CI) tables are generated for comparison.

5 Results

For GA, initial parameter values were selected through a process of trial and error, recognizing that fine-tuning can be computationally expensive. Instead of aiming for perfect solutions, the focus was on identifying sufficiently good configurations. Various settings were experimented with, and those that performed well were retained for further analysis. Consequently, no customized tuning process was undertaken. The GA was run for 500 iterations with a population size of 100. The optimization criterion was set as the p-value, with a mutation rate of 0.01, epsilon value of 1e-4 , and a maximum of 5 consecutive generations without improvement before stopping. In the selection step, 25% of the population was chosen for reproduction. The crossover was performed using the one-point method, and mutations were applied using the flip-bit technique. Additionally, a hyperparameter was set for the minimum cluster size at 10% to avoid outliers. Depending on the analysis requirement, the objective was either single or multi-objective. Moreover, all results presented were based on a two-cluster configuration.

5.1 Comparison of GA with other clustering algorithms

The tables 1 and 2 show the initial p-values obtained for different algorithms used on the GBM and PAAD datasets, respectively. For most algorithms, except RS, GA, and KM, a single p-value is reported in the "S / 95% CI" column, as these algorithms were run only once. In contrast, RS, GA, and KM were executed multiple times, so a range of p-values is provided. The 95% CI is calculated for all algorithms based on the set of p-values generated. The results demonstrate that GA consistently returns solutions with very small p-values, indicating high statistical significance in both datasets. Given that a p-value of less than 0.05 is considered statistically significant, GA's performance highlights its effectiveness in the context of survival analysis. The 95% CI for the p-values obtained by GA, as presented in the tables 1 and 2, range between 10^{-08} to 10^{-06} in case of GBM dataset and e-22 to e-20 in case of PAAD dataset. However, it is important to note that the full set of scores includes p-values as low as 10^{-20} for GBM dataset and 10^{-36} for PAAD dataset. The CI provides a statistical measure of the most commonly observed range of p-values, but does not capture the full extent of the algorithm's capability in generating extreme values. This indicates that while the majority of the p-values fall within the specified CI, the algorithm is sensitive enough to produce highly significant results.

We note that the clustering algorithms are unsupervised (i.e. they don't utilize the clinical labels or features for the stratification), and therefore their performance is expected to be much less compared to the GA, which utilizes labels to optimize the log-rank metric. Same reasoning follows the median-based stratification approach, which shows significant performance as it directly leverages the distribution of survival times. One drawback of this method is that it doesn't incorporate censoring information, leading to suboptimal results compared to the GA approach.

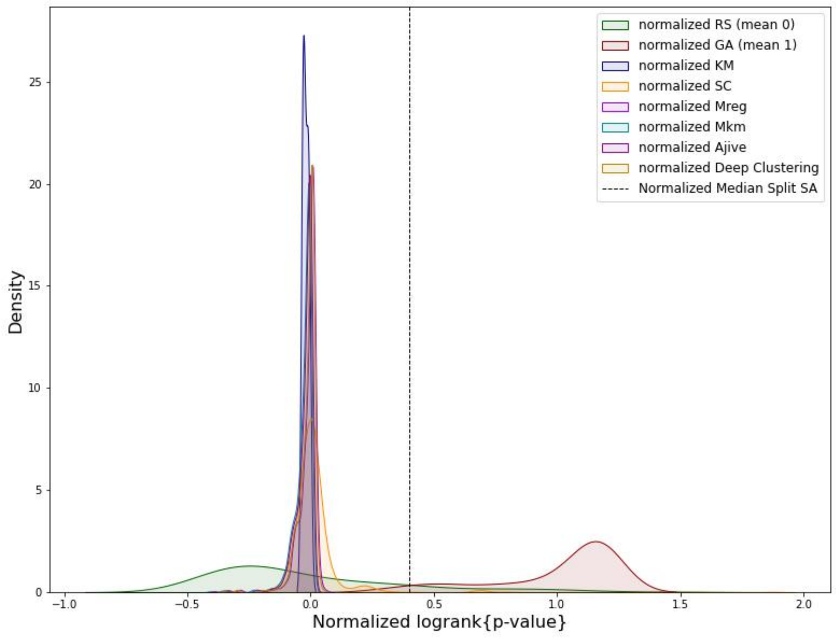

Figure 5: Density Plots of Normalized p-Values for All Algorithms on the GBM Dataset

To enhance the interpretability of the clustering metrics, normalization is performed as explained in Section 4.2. The 95% CI is also obtained for these normalized scores. From the figures 5 and 6, as expected, GA performs the best, while the RS demonstrates random performance as expected. The median-based stratification approach, although slightly less effective than GA, still falls within the top-performing category. For the remaining methods, instead of statistically significant p-values, we observe normalized scores close to zero, indicating that these algorithms are not performing better than random chance.

Moreover, as shown in Table 2, the PAAD dataset, which contains a larger number of samples compared to the GBM dataset (as shown in Table 1), demonstrates that both GA and the median-based stratification approach perform even better. This improvement is likely due to the increased statistical power afforded by the larger sample size, allowing these methods to more accurately capture the underlying patterns in the data.

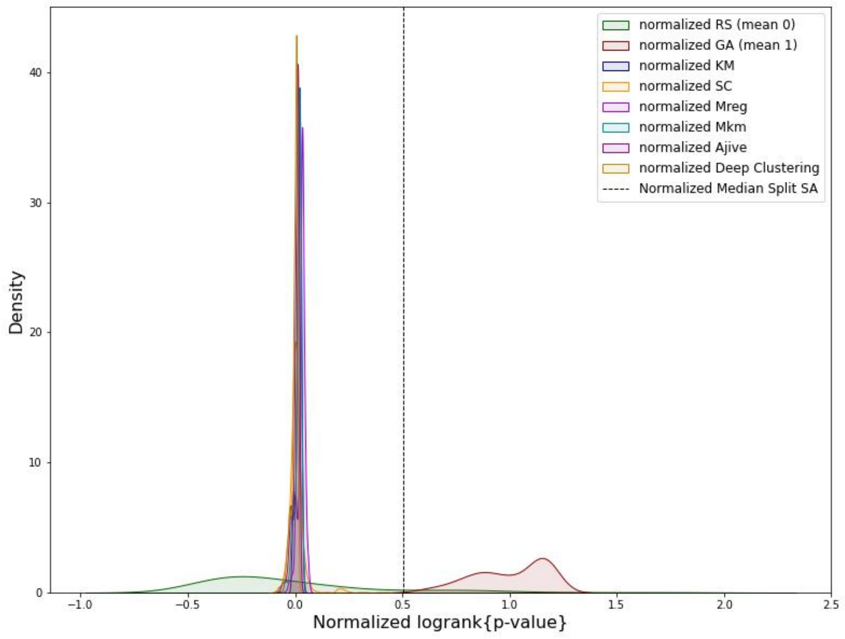

Figure 6: Density Plots of Normalized p-Values for All Algorithms on the PAAD Dataset

5.2 Single-Objective vs. Multi-Objective Optimization in GA

In the previous section, all algorithms were evaluated and compared against the GA using SOO with the log-rank p-value as the metric. In this section, we extend the comparison by evaluating both SOO and MOO within the GA framework to

A	**GBM**	
	S/ 95% CI	**Norm 95% CI**
GA	(1.185e-08, 1.810e-06)	(0.950, 1.049)
Median	2.212e-08	(0.382, 0.382)
SC	0.295	(0.011, 0.027)
KM	(0.425, 0.534)	(-0.147, -0.026)
RS	(0.436, 0.546)	(-0.079, 0.079)
MReg	0.488	(-0.012, -0.007)
Deep C	0.553	(-0.016, -0.011)
Ajive	0.769	(-0.027, -0.021)
MKm	0.897	(-0.032, -0.026)

Table 1: 95% CI of Original p-Values and Normalized Scores for the GBM dataset; A : Algorithms; S/ 95% CI : indicates the original score for algorithms run once and the 95% CI for algorithms with a set of scores for algorithms that has set of scores; Norm 95% CI : normalized 95% CI. Scores are presented in increasing order of p-values, with algorithms listed accordingly.

gain insight into the similarities and differences between the metrics. This analysis also helps us understand the trade-offs involved in achieving the best possible performance for the log-rank test when optimization is extended to include additional metrics, such as the silhouette score or other relevant measures. The figures 7 and 8 presented display three sets of scores:

- SOO Results: The scores from the SOO with the log-rank statistic, as discussed previously.
- MOO with Log-rank and Silhouette Score: Scores obtained using MOO, incorporating both the log-rank statistic and silhouette score. This approach evaluates the clustering quality by using metrics that assess the difference in survival distribution while considering clinical outcomes, alongside clustering performance metrics applied to the preprocessed GEX and METH data.
- MOO with Log-rank and chi-square: Scores from MOO, using the log-rank statistic alongside the chi-square test. The chi-square test in this context assesses the association between the cluster assignments and the clinical factor "Sex," measuring how well the clusters align with this clinical attribute

A	**PAAD**	
	S/ 95% CI	**Norm 95% CI**
GA	(1.495e-22, 5.461e-20)	(0.968, 1.034)
Median	9.982e-19	(0.693, 0.693)
MReg	0.051	(0.028, 0.030)
MKm	0.146	(0.012, 0.014)
Ajive	0.268	(0.003, 0.005)
SC	0.310	(0.005, 0.010)
KM	(0.384, 0.498)	(0.013, 0.026)
RS	(0.416,0.521)	(-0.089, 0.090)
Deep C	0.442	(-0.005, -0.003)

Table 2: 95% CI of Original p-Values and Normalized Scores for the PAAD dataset; A : Algorithms; S/ 95% CI : indicates the original score for algorithms run once and the 95% CI for algorithms with a set of scores for algorithms that has set of scores; Norm 95% CI : normalized 95% CI. Scores are presented in increasing order of p-values, with algorithms listed accordingly.

and its impact on survival outcomes. Here, "Sex" was chosen because it is a fundamental demographic characteristic included in most clinical datasets and which often has significant implications for survival outcomes and disease progression. By including "Sex" in the analysis, we can assess whether there are any notable differences in the distribution of this clinical factor across different patient clusters, helping to ensure that the stratification approach effectively captures clinically relevant variations in patient outcomes [23], [24].Specifically, -log10(p) of the chi-square p-value is considered as the metric, where a lower p-value (and hence a higher -log10(p) value) indicates a stronger association between the clusters and the clinical factor.

These comparisons will provide insights into the effectiveness of SOO versus MOO in enhancing patient stratification, with a focus on how the integration of multiple objectives influences the clustering performance and the overall interpretability of the results. The subsequent analysis aims to highlight whether the additional objectives in MOO contribute to better clustering outcomes and more robust survival predictions compared to the SOO approach.

In Figures 7 and 8, we present density plots of the -log10(log-rankp) for the three different cases mentioned above. Higher values indicate better clustering performance. From these figures, it is evident that SOO consistently outperforms MOO across both datasets, GBM and PAAD. The performance of MOO is notably hindered when the Silhouette score is used alongside the log-rank test. This is primarily because the Silhouette score focuses on the geometrical alignment of data points, which may not be directly correlated with survival outcomes. Additionally, biological phenotypes and clinical outcomes might not always align with the metrics used to represent them, making direct correlations between clustering performance and clinical relevance less straightforward. Specifically, if the data points for two groups are overlapping, the Silhouette score becomes very small. For the GBM dataset, the Silhouette scores ranged from 0.24 to 0.48, with a 95% CI of (0.350, 0.365), indicating relatively poor clustering performance. Similarly, for the PAAD dataset, the Silhouette scores ranged from 0.01 to 0.1, with a 95% CI of (0.051, 0.056), indicating almost random performance for this metric. This further emphasizes the limitations of using the Silhouette score in this context.

On the other hand, MOO with the chi-square test performs slightly better than MOO with the Silhouette score, as it directly considers the clinical factor "Sex," which is more relevant to survival analysis. For the GBM dataset, -log10(chi-squarep) ranged from 1.878 to 3.885, with a 95% CI of (2.738, 2.905), suggesting a moderate alignment of cluster assignments with the clinical factor and survival outcomes. In the PAAD dataset, the (chi-squarep) ranged more broadly from 3.979 to 13.291, with a 95% CI of (7.955, 8.734), indicating a stronger association between clusters and the clinical factor "Sex."

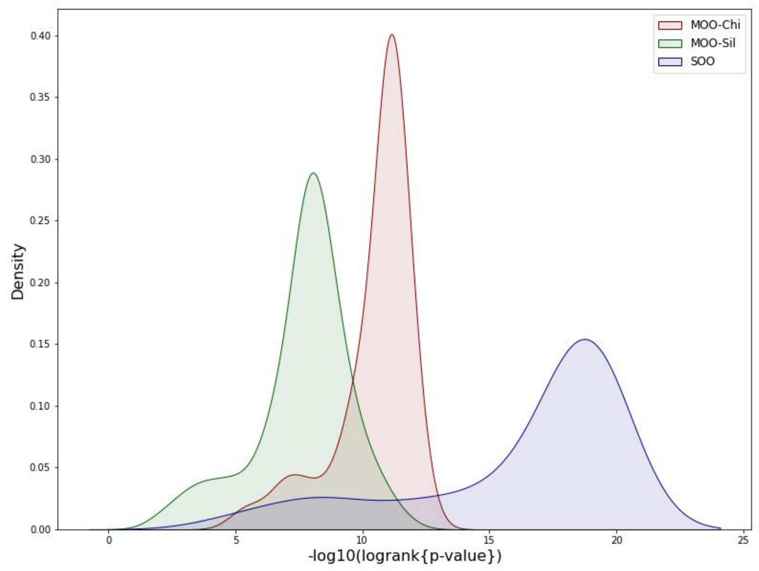

Figure 7: Density Plots of Log-Transformed p-Values Comparing SOO and MOO in GA for the GBM Dataset; MOO Sil : MOO with log-rank+Silhouette; MOO Chi : MOO with log-rank+chi-square; SOO : SOO with log-rank only

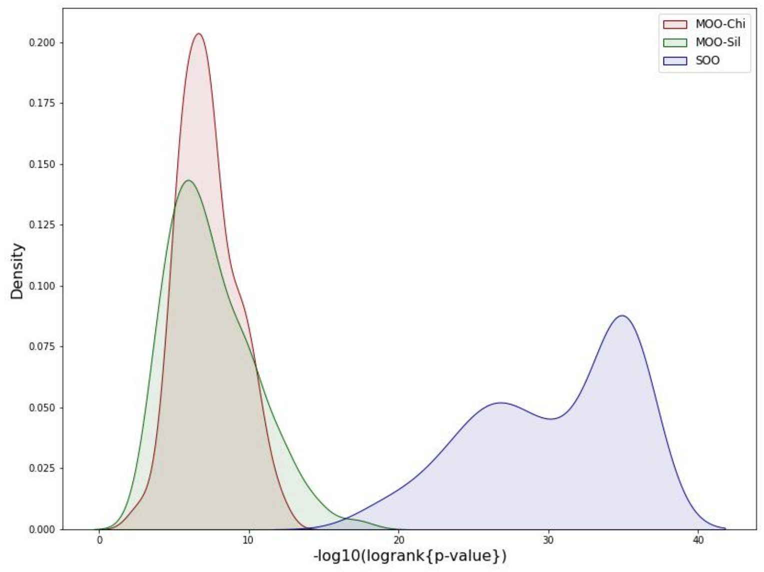

Figure 8: Density Plots of Log-Transformed p-Values Comparing SOO and MOO in GA for the PAAD Dataset; MOO Sil : MOO with log-rank+Silhouette; MOO Chi : MOO with log-rank+chi-square; SOO : SOO with log-rank only

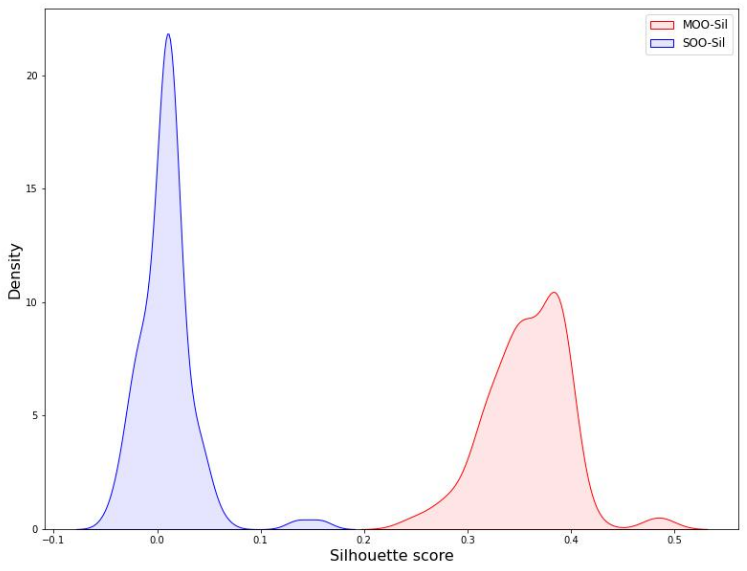

Figure 9: Comparison of Silhouette Scores: MOO Log-rank + Silhouette vs. SOO on GBM dataset; MOO-sil : Silhouette score obtained when used as objective in MOO; SOO-sil : Silhouette score calculated Post SOO Log-rank Optimization

In conclusion, while SOO consistently yields better performance in terms of survival analysis, the choice of secondary objectives in MOO provides a better reflection of the best possible performance for a clustering algorithm, as it incorporates additional clustering metrics alongside survival outcomes. The use of the Silhouette score as a secondary objective appears less effective, particularly when clusters overlap geometrically, whereas the chi-square test shows more promise by integrating relevant clinical factors into the optimization process. This highlights the importance of carefully selecting objectives that align well with the primary goal of patient stratification based on survival outcomes.

However, despite the superior performance of SOO in terms of log-rank p-values for survival analysis, MOO offers a more balanced consideration of multiple objectives. For instance, Figures 9 and 10 show the density plots for Silhouette scores obtained using MOO with log-rank and silhouette score as objectives, compared to the Silhouette scores calculated after deriving the final candidate solutions from GA for SOO, for both the GBM and PAAD datasets. These figures illustrate that the Silhouette scores obtained when using it as an objective in MOO, along with the log-rank statistic, are significantly higher than those obtained after running SOO, especially on GBM dataset (figure 9). However, this is dataset-dependent: on the PAAD dataset, we observe generally worse results, likely due to overlapping data points, indicating poor cluster solutions (figure 10). This indicates that, depending on the dataset, Multi-Objective Optimization (MOO) can enhance the geometric separation of clusters, as observed in the GBM dataset, but may not necessarily improve patient stratification. For example, in figure 8, the differences between the distributions through statistical significance are practically the same, suggesting that the enhancement of geometric separation in MOO does not translate into improved survival performance for PAAD.

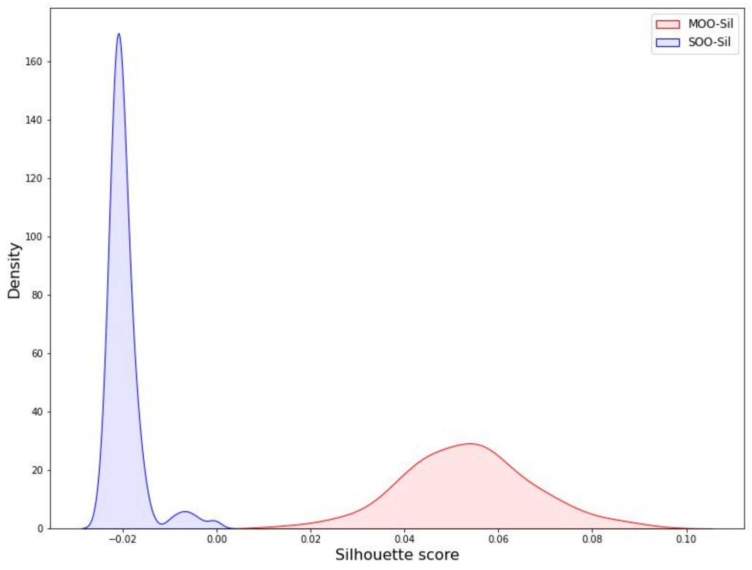

Figure 10: Comparison of Silhouette Scores: MOO Log-rank + Silhouette vs. SOO on PAAD dataset; MOO-Sil : Silhouette score obtained when used as objective in MOO; SOO-Sil : Silhouette score calculated Post SOO Log-rank Optimization

Time complexity is a critical consideration in evaluating the efficiency of any algorithm. In the context of GA, SOO proves to be significantly faster, typically requiring a maximum of 5 minutes to complete. In contrast, MOO is more computationally intensive, often taking between 10 to 20 minutes to converge. The increased time required for MOO is due to the complexity of optimizing multiple objectives simultaneously, which involves a more extensive search across the solution space and a greater number of calculations. While MOO offers the advantage of balancing multiple criteria, its longer runtime highlights the trade-off between computational efficiency and the depth of optimization achieved. This difference in time complexity must be considered when choosing between SOO and MOO based on the specific needs and constraints of the analysis.

6 Conclusion

The main goal of this project is to optimize clustering metrics for patient stratification using various algorithms. This study presents a comprehensive analysis of clustering techniques, specifically focusing on their application to multi-omics and clinical data for enhanced patient stratification in survival analysis. The analysis was conducted on two datasets, GBM and PAAD, providing robust evidence that GA, particularly in its SOO form, yields more statistically significant results as indicated by consistently lower p-values in survival analysis. Nevertheless, MOO, while potentially yielding lower survival performance, enhances the geometric separation of clusters, highlighting the trade-offs between different optimization strategies. Additionally, the study's methodology emphasized the importance of normalization and careful comparison of algorithmic performance, ensuring that the results are interpretable and statistically sound. The detailed implementation of this project can be found on this GitHub repository.

As of now, the results have been tested only for two clusters. In the future, we plan to expand our analysis to include more than two clusters to evaluate GA's performance across different scenarios. Additionally, a comparison between our GA implementation and existing packages like PyGAD and PyMOO will be documented. While this project focused primarily on the log-rank test for patient stratification, we aim to enhance the framework to allow users the flexibility to choose from a variety of metrics or a combination of them in future iterations.

References

[1] S. Rai, P. Mishra, and U. C. Ghoshal, "Survival analysis: A primer for the clinician scientists," *Indian Journal of Gastroenterology*, vol. 40, no. 5, pp. 541–549, 2021.

[2] D. R. Cox, "Regression models and life-tables," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 34, no. 2, pp. 187–202, 1972.

[3] J. P. Klein, H. C. Van Houwelingen, J. G. Ibrahim, and T. H. Scheike, *Handbook of survival analysis*. CRC Press Boca Raton, FL: 2014.

[4] Y. Chen *et al.*, "Mocss: Multi-omics data clustering and cancer subtyping via shared and specific representation learning," *Iscience*, vol. 26, no. 8, 2023.

[5] J. Pateras, M. Lodi, P. Rana, and P. Ghosh, "Data clustering with improved expectation maximization for multiomics data integration," *Authorea Preprints*, 2023.

[6] A. K. Jain, "Data clustering: 50 years beyond k-means," *Pattern recognition letters*, vol. 31, no. 8, pp. 651–666, 2010.

[7] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *arXiv preprint arXiv:1304.5634*, 2013.

[8] N. Rappoport and R. Shamir, "Multi-omic and multi-view clustering algorithms: Review and cancer benchmark," *Nucleic acids research*, vol. 46, no. 20, pp. 10 546–10 562, 2018.

[9] B. Wang *et al.*, "Similarity network fusion for aggregating data types on a genomic scale," *Nature methods*, vol. 11, no. 3, pp. 333–337, 2014.

[10] X. Lu, J. Meng, Y. Zhou, L. Jiang, and F. Yan, "Movics: An r package for multi-omics integration and visualization in cancer subtyping," *Bioinformatics*, vol. 36, no. 22-23, pp. 5539–5541, 2020.

[11] López *et al.*, "Explainable multi-omics deep clustering reveals an important role of dna methylation in pdac," *PANCAIM - Pancreatic Cancer AI for Genomics and Personalized Medicine*, 2024.

[12] A. F. Gad, *Pygad: An intuitive genetic algorithm python library*, 2021. arXiv: `2106.06158 [cs.NE]`. [Online]. Available: `https://arxiv.org/abs/2106.06158`.

[13] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *Ieee access*, vol. 8, pp. 89 497–89 509, 2020.

[14] D. Wissel, D. Rowson, and V. Boeva, "Systematic comparison of multi-omics survival models reveals a widespread lack of noise resistance," *Cell Reports Methods*, 2023. [Online]. Available: https://doi.org/10.1016/j.crmeth.2023.100461.

[15] J. Hao, Y. Kim, T. Mallavarapu, J. H. Oh, and M. Kang, "Interpretable deep neural network for cancer survival analysis by integrating genomic and clinical data," *BMC medical genomics*, vol. 12, pp. 1–13, 2019.

[16] D. Hariharan, A. Saied, and H. Kocher, "Analysis of mortality rates for pancreatic cancer across the world," *Hpb*, vol. 10, no. 1, pp. 58–62, 2008.

[17] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm- a literature review," *IEEE Xplore*, 2019.

[18] R. O. Ottosson *et al.*, "The feasibility of using pareto fronts for comparison of treatment planning systems and delivery techniques," *Acta Oncologica*, vol. 48, no. 2, pp. 233–237, 2009, PMID: 18752085. DOI: 10.1080/02841860802251559. eprint: https://doi.org/10.1080/02841860802251559. [Online]. Available: https://doi.org/10.1080/02841860802251559.

[19] D. Grygar and R. Fabricius, "An efficient adjustment of genetic algorithm for pareto front determination," *Transportation Research Procedia*, vol. 40, pp. 1335–1342, 2019, TRANSCOM 2019 13th International Scientific Conference on Sustainable, Modern and Safe Transport. DOI: https://doi.org/10.1016/j.trpro.2019.07.185. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146519303539.

[20] J. Tracey *et al.*, "Prioritizing conserved areas threatened by wildfire and fragmentation for monitoring and management," *PLOS ONE*, vol. 13, e0200203, Sep. 2018. DOI: 10.1371/journal.pone.0200203.

[21] R. Perry *et al.*, "Mvlearn: Multiview machine learning in python," *Journal of Machine Learning Research*, vol. 22, no. 109, pp. 1–7, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1370.html.

[22] M. Schemper, "Predictive accuracy and explained variation," *Statistics in medicine*, vol. 22, no. 14, pp. 2299–2308, 2003.

[23] D. G. Kleinbaum and M. Klein, *Survival analysis a self-learning text*. Springer, 1996.

[24] S. K. Pal and A. Hurria, “Impact of age, sex, and comorbidity on cancer therapy and disease progression,” *Journal of clinical oncology*, vol. 28, no. 26, pp. 4086–4093, 2010.