Санкт–Петербургский государственный университет

Милов Артём Игоревич

Выпускная квалификационная работа

Вычислительные методы при идентификации малых молекул по масс-спектрам

Уровень образования: бакалавриат
Направление 01.03.01 «Математика»
Основная образовательная программа СВ.5000.2019 «Математика и анализ данных»

Научный руководитель:
доцент, факультет математики и компьютерных наук,
д.ф. - м.н. Степанов Алексей Владимирович

Рецензент:
научный сотрудник, Саарский университет,
к.ф. - м.н. Тагирджанов Азат Мухаммедович

Санкт-Петербург
2023 г.

Содержание

1. Введение

1.1 Обзор

1.1.1 Как устроена масс-спектрометрия

В биоинформатике огромное значение имеет определение различных молекул. Определение это обычно происходит посредством масс-спектрометрии. Масс-спектрометрия проходит в два этапа. В первом этапе, специальный прибор (масс-спектрометр) разбивает молекулы образца по m/z (отношение массы молекулы к её заряду) на различные группы. На втором этапе масс-спектрометр разбивает молекулы какой-то группы на маленькие части и считает количество (интенсивность) фрагментов каждой получившийся массы. Полученные данные записываются в файл, мы этот файл далее будем называть спектром.

1.1.2 Дерепликаторы

Полученные данные масс-спектрометрии нужно анализировать. В зависимости от типа исследуемых молекул используется разное программное обеспечение. Особенный класс важных для биоинформатики молекул — малые молекулы обрабатываются дерепликаторами. У них также имеется база данных с формулами молекул, одной из которых является исследуемая молекула.

То есть, можно сказать, назначением дерепликаторов по факту является выявление формулы исследуемой молекулы из предоставленных кандидатов по предоставленному спектру.

1.1.3 Термины

Начнем с определения базовых понятий:

Спектр — файл, в котором хранится информация о разбиении какой-то молекулы на масс-спектры.

База данных — файл, в котором хранится информация о формулах молекул, одной из которых является исследуемая молекула (там также может хранится и другая информация о молекулах, но пользоваться мы ей не будем).

Дерепликатор — программа, назначение которой определять молекулу по спектру среди множества данных молекул (задаётся файл с базой данных).

InChi-ключ — что-то вроде хэш-кода молекулы, то есть некоторый символьный отпчаток, который, если у двух молекул совпадает, то сами молекулы тоже совпадают с очень большой вероятностью.
Скор — мера уверенности дерепликатора в той или иной идентификации, чем меньше скор, тем сильнее дерепликатор уверен в правильности соответствующего ответа.
NPD-Quast — разработанная мной программа, которая выполняет указанную в отчёте задачу.

1.1.4 Проблема

На данный момент разработчики дерепликаторов и их пользователи испытывают сложности с определением качества их работы, а также сравнении их друг с другом. В качестве решения этой задачи, на текущий момент, периодически проводятся соревнования, где участниками являются разработчики дерепликаторов. Участники

отсылают свои дерепликаторы организаторам, те в свою очередь запускают эти дерепликаторы на некоторых данных, единых для всех участников, а потом анализируют и составляют метрики на их основе.
С целью упростить эту проблемму мной была разработана программа NPD-Quast (Natural Peptide Dereplicator - Quaslity Accessment Tool). Данная программа по факту делает тоже самое, что делали организаторы соревнований, только автоматически и локально, а именно запуск некоторых дерепликаторов на имеющихся данных и составление метрик на поученных данных.

1.2 Описание

1.2.1 Цель работы

Наша цель в том, чтобы любой разработчик дерепликатора мог бы добавить свой инструмент в NPD-Quast, чтобы на правильно выстроенных данных была возможность запустить его программу через NPD-Quast, а потом измерить качество её работы и сравнить с другими дерепликаторами.

В случае NPD-Quast качество работы измеряется различными метриками, указанными ниже. Все они основываются на сравнении работы дерепликаторов с правильными ответами. То есть, подразумевается, что про все подаваемые спектры известно, какой молекуле из базы данных он соответствует. Это важное замечание, так как получается, что для корректной работы NPD-Quast требуется файл с правильными ответами.

Также разработан единый для всех разработчиков и пользователей формат данных, в котором будут хранится спектры и базы данных, а также правильные ответы (то есть файл, в котором прописано какой спектр какой молекуле соответствует).

1.2.2 Требование к программе

- Запускать дерепликаторы на имеющихся данных и считать метрики,
- Сравнивать с дерепликаторы между собой,
- Дать возможность разработчикам дерепликаторов добавить свой дерепликатор в список поддерживаемых,
- Считать метрики на данных без правильных ответов,
- Считать метрики на уже имеющихся результатах работы не поддерживаемых дерепликаторов.

2. Методы

2.1 Формат данных

Первой проблемой, которую нужно было решить, стало отсутствие стандартизации формата входных данных для разных дерепликаторов. Нами был разработан единый формат данных, в котором хранятся спектры и базы данных с молекулами-кандидатами. Для преобразования в требуемый дерепликатором формат, в NPD-Quast включен специальный модуль. Мы реализовали модуль преобразования данных для дерепликаторов Sirius [3], Dereplicator+ [1] и MAGMa+ [4]. Благодаря модульной архитектуре, в NPD-Quast могут без труда быть включены модули преобразования данных и для других дерепликаторов.

2.1.1 Идея

Разбиваем все спектры на группы — в каждой общая база данных. Отдельно добавляем файл с правильными идентификациями. Во время работы, NPD-Quast будет запускать каждую группу отдельно. После анализирует работу, сохраняя промежуточные данные в отдельной папке (которую по умолчанию потом удаляет, но в некоторых режимах оставляет), а потом загружает полученный отчёт о качестве работы уже в другую отдельную папку.

2.1.2 Общая структура

Все данные храним в директории следующего формата: $NPDQuast\ folder/$

$challenges/$
- $challenge\ 1/$
  - $spectra/$
    - $specter\_1.mgf$
    - $specter\_2.mgf$
    - $specter\_3.mgf$
    - ...
  - $database.csv$
- $challenge\_2/$
- $challenge\_3/$
- ...

$reports/$
- $some\_data\_in.html$
- $report\_name\_1/$
  - $tool\_answers.txt$
  - $some\_data\_in.html$
- $report\_name\_2/$
- $report\_name\_3/$
- ...

$true\_answers.txt$ Далее расмотрим составляющие этой директории.

2.1.3 Директории

Директория challenges

Данная директория содержит исходные данные для дерепликаторов. Как уже было написано, данные делятся на группы. Эти группы я назвал челленджами. Каждая группа состоит из множества спектров (поддиректория $spectra$ с файлами спектров) и файла с базой данных.

Директория reports

Данная директория содержит данные для html сайта с отчётами со всеми проведёнными исследованиями. Мы не будем подробно рассамтривать, какие файлы нужны для этого, всё создаётся автоматически (включая саму директорию). Только обратим внимание, что для каждого отдельного отчёта используется поддиректория, в которой, в частности, есть файл с ответами исследуемого дерепликатора.

2.1.4 Файлы

Файл true answers.txt

Файл, в котором содержатся правильные ответы (см. Рис 2 в Приложении). Устроен следующим образом. Каждая идентификация — это строка вида:

$$challenge\_name\ spectra\ specter\_file\ answer\_inchi\_key$$

где:

- $challenge\_name\ spectra\ specter\_name$ — путь к файлу с настоящим спектром,
- $answer\_inchi\_key$ — InChi-ключ правильный молекулы.

Файл tool answers.txt

Файл, в котором содержатся ответы дерепликатора. Устроен почти также, но с некоторыми отличиями. Идентификаций на один спектр может приходится несколько.

Каждая идентификация — это строка вида:

$$challenge\_name \quad spectra \quad specter\_file \quad answer\_inchi\_key \quad score$$

Или

$$challenge\_name \quad decoys \quad specter\_file \quad answer\_inchi\_key \quad score$$

Где:

- $challenge\_name\ spectra\ specter\_name$ — путь к файлу с настоящим спектром,
- $challenge\_name \quad decoys \quad specter\_name$ — путь к файлу со спектром-декоем,
- $answer\_inchi\_key$ — InChi-ключ выданной дерепликатором молекулы,
- $score$ — скор ответа.

Файл specter.mgf

Формат, в котором должны находится все спектры (см. Рис. 1 в Приложении). Количество пиков может быть любое. Структура всегда такая:

$$
\begin{aligned}
&BEGIN\ IONS\\
&MSLEVEL = 2\\
&PEPMASS = \{pepmass\}\\
&CHARGE = 1+\\
&SCANS = \{scan\}\\
&\{peak\_mass\_1\} \quad \{peak\_intensity\_1\}\\
&\{peak\_mass\_2\} \quad \{peak\_intensity\_2\}\\
&\{peak\_mass\_3\} \quad \{peak\_intensity\_3\}\\
&\dots\\
&END\ IONS
\end{aligned}
$$

Здесь:

- $pepmass$ — настоящая масса исследуемой молекулы,
- $scan$ — код спектра,
- $peak\_mass\_i$ — масса пика,
- $peak\_intensity\_i$ — интенсивность пика.

Файл database.csv

Csv-таблица со столбцами:

- $Identifier$ — id-строки,
- $CompoundName$ — имя молекулы,
- $MonoisotopicMass$ — масса,
- $MolecularFormula$ — формула,
- $SMILES$ — $SMILES$ молекулы,
- $MolInChI$ — InChi молекулы,
- $InChIKey$ — InChi-ключ молекулы.

2.2 Оценка качества

В NPD-Quast реализовано два режима работы. Первый режим позволяет оценить качество работы дереплика-тора на аннотированных данных (т.е. когда для каждого спектра известен правильный ответ). Второй режим оценивает качество работы дерепликатора на реальных данных, когда правильный ответ неизвестен.

2.2.1 Оценка качества на аннотированных данных

При запуске на аннотированных данных NPD-Quast сверяет идентификации, представленные дерепликаторами, с правильными ответами. В результате

Первый вариант (и самый очевидный) — держать дополнительно файл с правильными идентификациями. Сверяя полученные ответы с ними, составлять различные метрики, описанные в соответствующем разделе. Такой подход привлекателен своей простотой и понятностью. Единственный недостаток — не всегда имеется информация о правильных идентификациях.

2.2.2 Метрики

Каждому спектру соответствует некоторый правильный InChi-ключ, но дерепликатор, не зная правильного ответа, предлагает несколько возможных InChi-ключей (кандидатов) в порядке своей уверенности в ответе. Каждому кандидату дерепликатор сопоставляет некоторое значение – скор – характеризующее меру уверенности дерепликатора в правильности ответа. Поскольку само это значение редко бывает несмещенным, на практике как правило используют позицию кандидата в списке ответов отсортированном в порядке убывания скора. У правильных идентификаций есть какая-то позиция в этих списках (или, если дерепликатор вообще её не нашёл, то назначается некоторая дефолтная позиция, например максимально возможная).

Пусть

- ans_{spc} — это бинарный вектор ответа дерепликатора на спектр $spc.mgf$, где 0 означает неправильный ответ, а 1 — правильный.

- $score_{spc}$ — это вектор уверенности ответов дерепликатора на спектр $spc.mgf$. То есть, если дерепликатор выдал пять разных ответов и в каждом из них он уверен по разному, то вектор уверенностей будет выглядеть так:

  $$0, 1, 2, 3, 4.$$

  Однако, если например во втором и третьем ответах он будет уверен одинаково, то вектор будет

  $$0, 1.5, 1.5, 3, 4.$$

- $Spectra$ — множество всех спектров.

Скалярное произведение векторов будем обозначать через $*$.

Ранг правильного ответа

Для каждого спектра вычисляется позиция в списке идентификаций, на которой стоит правильный ответ. После этого вычисляется среднее или медиана этих позиций по всему множеству спектров по формулам:

Медиана:

$$median(\{ans_{spc} * score_{spc} | spc \in Spectra\})$$

Среднее:

$$mean(\{ans_{spc} * score_{spc} | spc \in Spectra\})$$

Квантиль уровня k

Помимо медианы, NPD-Quast вычисляет квантили полученных ответов,

$$quantile(k, \{ans_{spc} * score_{spc} | spc \in Spectra\})$$

и строит график этой функции при k от 0 до 100%.

Позиции правильных идентификаций

Для вычисления данной метрики мы объединяем идентификации дерепликатора для всех спектров в один общий список. Этот список сортируется по скору и вычисляется метрика $Top(x)$ как количество правильных идентификаций среди первых x идентификаций.

NPD-Quast строит график $Top(x)$ для всех x.

Доля ложных идентификаций

Тоже самое, только считается не количество правильных идентификаций, а их доля среди первых x ответов.

$$FDR(x) = \frac{Top(x)}{x}$$

Зачёт по медалям

Данная метрика используется при запуске нескольких дерепликаторов для сравнения их работы [2]. Она представляет из себя сумму баллов за соревнования между измеряемыми дерепликаторами. Для каждого спектра проходит отдельное соревнование.

Каждое соревнование выглядит следующим образом — у каждого дерепликатора есть позиция правильной идентификации для данного спектра. Победитель — тот, у кого эта позиция самая меньшая. Второй, у кого эта позиция самая меньшая после победителя и тд.

Баллы, которые дерепликатор получает от позиции в соревновании, зависят от метода подсчета:

$Classic : [5, 3, 1]$,
$F1 : [24, 18, 15, 12, 10, 8, 6, 4, 2, 1]$,
$Gold : [1]$,
$All : [1, 1, 1]$.

Результатом является суммарное количество набранных баллов для всех спектров.

Относительная ранговая позиция

Каждому спектру можно сопоставить следующее число:

$$RRP(spc) = \frac{1}{2}\Big(1 - \frac{UpCorrect(spc) - BelowCorrect(spc)}{Total(spc) - 1}\Big)$$

где:

- $UpCorrect$ — количество неправильных идентификаций, в которых исследуемый дерепликатор уверен больше, чем в правильном, то есть

$$UpCorrect(spc) = \#\{j : score_{spc}[j] < score_{spc}[i]\}$$

где $i : ans_{spc}[i] = 1$,

- $BelowCorrect$ — количество неправильных ответов, в которых исследуемый дерепликатор уверен меньше, чем в правильном, то есть

$$BelowCorrect(spc) = \#\{j : score_{spc}[j] > score_{spc}[i]\}$$

где $i : ans_{spc}[i] = 1$.

Далее среди всех этих чисел можно взять среднее или медиану.

$$Median(\{RRP(spc)|spc \in Spectra\}),\ Mean(\{RRP(spc)|spc \in Spectra\})$$

Взвешенная относительная ранговая позиция

Тоже самое, что и в предыдущей метрике, только теперь каждому спектру сопоставляется число:

$$wRRP(spc) = 1 - UpNormalized(spc) - SameNormalized(spc),$$

где:

- $UpNormalized(spc)$ — доля неправильных ответов, в которых исследуемый дерепликатор уверен больше, чем в правильном, среди всех ответов дерепликатора (на данных спектр).

$$UpNormalized(spc) = \frac{\#\{j : score_{spc}[j] < score_{spc}[i]\}}{\#score_{spc}}$$

где $i : ans_{spc}[i] = 1$

- $SameNormalized(spc)$ — доля неправильных ответов, в которых исследуемый дерепликатор уверен на столько же, насколько и в правильных, среди всех ответов дерепликатора (на данных спектр).

$$SameNormalized(spc) = \frac{\#\{j : score_{spc}[j] = score_{spc}[i]\}}{\#score_{spc}}$$

где $i : ans_{spc}[i] = 1$.

2.2.3 Оценка качества на неаннотированных данных

Иногда пользователь имеет данные масс-спектрометрии и базы данных молекул, но не знает что чему соответствует. Тогда для того, чтобы понять как хорошо дерепликатор справился со своей работой используются подход *target-decoy* [5]. Этот подход был разработан в протеомике, где для оценки доли ложных идентификаций создается база данных заведомо ложных пептидов. В метаболомике создание такой базы данных затруднено разнообразием классов исследуемых химических соединений. Поэтому применяется другой подход основанный на создании ложных спектров [6].

Декои — это спектры, похожие на спектры настоящих молекул, однако на самом деле ими не являющиеся. В реальности нет никакой молекулы, которой соответствует этот спектр. Но задумка в том, что дерепликатор может ошибиться и сопоставить ему какую-то молекулу из соответствующей базы данных. Тогда, такую идентификацию мы будем считать неправильной. А все идентификации с настоящими спектрами — правильными (это грубое упрощение, однако допустимое). Таким образом, мы получим некоторое подобие правильных и неправильных ответов.

В NPD-Quast реализовано несколько стратегий создания декоев. В данном разделе будем считать, что у нас есть структура данных $Peak$ — это просто два числа $mass$ и $intensivity$ — масса и интенсивность пика. А также структура данных $MassSpecter$ — которая содержит список пиков ($peaks$), массу ($pepmass$) и номер спектра ($scan$).

Случайная генерация

В данном методе всё просто — генератор просто получает файл реального спектра, который хочет подделать ($target\_specter$), а потом случайно генерирует массы пиков из списка настоящих спектров ($real\_spectra$), сохраняя интенсивность как у подделываемого спектра (Алгоритм 1).

Algorithm 1 Random generation

```
all_peak_masses ← empty list
for real_specter in real_spectra do
    for peak in real_specter.peaks do
        all_peaks_masses.append(peak.mass)
    end for
end for
result_decoy ← MassSpecter()
result_decoy.scan ← target_specter.scan
result_decoy.pepmass ← target_specter.pepmass
decoy_length ← len(target_specter.peaks)
i ← 0
while i < decoy_length do
    new_peak ← Peak()
    new_peak.mass ← random(all_peaks_masses)
    new_peak.intensivity ← target_specter.peaks[i].intensivity
    result_decoy.peaks.append(new_peak)
    i ← i + 1
end while
return result_decoy
```

Метод главных пиков

В этом методе всё почти также, только массы пиков выбираются не из всех настоящих спектров, а только самые частые (сколько самых частых пиков будет рассмотрено зависит от параметра $peaks\_count$) и с весами их частоты встречаемости (чем чаще пик встречается, тем с большей вероятностью его возьмут). Этот метод описан в Алгоритме 2.

Пошаговый метод

В данном методе поиск создание декоя происходит итеративно. На i-й итерации добавляется пик с интенсивностью как у i-го пика целевого спектра ($target\_specter$) и массой, выбранной случайно среди масс пиков тех спектров, где есть масса предыдущего пика (Алгоритм 3).

Algorithm 2 Top x peaks generation

```
all_peak_masses ← default dict with 0 default value
for real_specter in real_spectra do
    for peak in real_specter.peaks do
        all_peaks_masses[int(peak.mass * 100)] ← all_peaks_masses[int(peak.mass * 100)] + 1
    end for
end for
leave only the peaks_count largest values among all_peak_masses keys
normalize all_peak_masses values
result_decoy ← MassSpecter()
result_decoy.scan ← target_specter.scan
result_decoy.pepmass ← target_specter.pepmass
decoy_length ← len(target_specter.peaks)
i ← 0
while i < decoy_length do
    new_peak ← Peak()
    new_peak.mass ← random(all_peaks_masses.keys(), weight = all_peaks_masses.values())
    new_peak.intensivity ← target_specter.peaks[i].intensivity
    result_decoy.peaks.append(new_peak)
    i ← i + 1
end while
return result_decoy
```

Algorithm 3 Step by step generation

```
suitable_peak_masses ← empty list
for real_specter in real_spectra do
    for peak in real_specter.peaks do
        all_peaks_masses.append(peak.mass)
    end for
end for
result_decoy ← MassSpecter()
result_decoy.scan ← target_specter.scan
result_decoy.pepmass ← target_specter.pepmass
decoy_length ← len(target_specter.peaks)
i ← 0
while i < decoy_length do
    new_peak ← Peak()
    new_peak.mass ← random(suitable_peaks_masses)
    new_peak.intensivity ← target_specter.peaks[i].intensivity
    result_decoy.peaks.append(new_peak)
    suitable_peak_masses ← empty list
    for real_specter in real_spectra do
        if new_peak.mass in real_specter.peaks.masses then
            for peak in real_specter.peaks do
                suitable_peaks_masses.append(peak.mass)
            end for
        end if
    end for
    i ← i + 1
end while
return result_decoy
```

2.2.4 Использование декоев

Как уже было описано, использование декоев позволяет получить некоторое подобие правильных и неправильных идентификаций, даже если на деле неизвестно что чему соответствует. Однако не все метрики будут точны, поскольку это приближение. В данной работе, мы используем метрику FDR.

2.3 Схема работы

2.3.1 Анализ качества работы дерепликатора (классический вариант)

Преобразуем последовательно полученные данные, запускаем дерепликатор, собираем результаты, формируем отчёт.

Инициализация

В самом начале создаётся поддиректория $report\_name$ в поддиректории $reports$ (если самой поддиректории $reports$ ещё не добавлено, то она сама тоже создаётся). Далее создаётся поддиректория $temp$, в которую будут записываться все промежуточные файлы, необходимые для составления отчёта.

Переформатирование данных

В эту поддиректорию записываются сначала переформатированные спектры (если исследуемый дерепликатора требует входные данные иного формата, нежели записано в $NPD - Quastfolder$. Если формат спекторв для дерепликатора совпадает с тем, что требуется в NPD-Quast, тогда файлы просто копируются. Потом точно также переносится и база данных.

Запуск

Далее запускается дерекликатор, все результаты его работы сохраняются там же, в поддиректории $temp$. Потом эти данные переформатируются и записываются в $true_answers.txt$ файл поддиректории $report\_name$.

Постобработка

После этого проводится анализ полученных данных, считаются метрики, формируются html файлы с графиками и отчётами.

2.3.2 Анализ качества работы дерепликатора (через snakemake)

Параллельно запускаем исследуемую программу на каждом спектре.

Переформатирование данных

Если форматы спектров или баз данных у дерепликатора и у NPD-Quast отличаются, то тогда перед запуском дерепликаторов используется snakemake-файл, параллельно преобразующий входный данные в требуемые для дерепликатора форматы — $prepare\_databases.smk$. Если же требуемые форматы спектров и баз данных совпадают, то этот файл не требуется.

Запуск

Используем только один snakemake-файл, в котором происходит процесс запуска дерепликаторы параллельно для всех челленджей.

Постобработка

Анализ полученных данных (получение метрик, составление отчётов и тд) проходит таким же образом как и в предыдущей версии.

3. Результаты

3.1 Функционал

3.1.1 Запуск программы

Добавление отчёта

Итоговая программа способна добавлять отчёт о работе одного из доступных дерепликаторов к уже существующей папке формата $NPD - Quast\ folder$. Для этого нужно ввести команду:

```
$ smk_npd_quast.py run_n_report tool report_name folder
```

Или

```
$ npd_quast.py run_n_report tool report_name folder
```

В зависимости от того, хочет пользователь запускать программу через snakemake или не хочет (первый вариант — без snakemake). Здесь:

- tool — название исследуемого дерепликатора,
- report_name — имя отчёта,
- folder — путь к папке формата $NPD - Quast\ folder$.

Компиляция отчётов

Так же можно добавить свой файл с ответами какого-то не поддерживаемого дерепликатора и получить из него метрики.

```
$ smk_npd_quast.py compile_reports folder
```

Или

```
$ npd_quast.py compile_reports folder
```

В зависимости от того, хочет пользователь запускать программу через snakemake или не хочет (первый вариант — без snakemake). Здесь:

- folder — путь к папке формата $NPD - Quast\ folder$.

Создание декоев

Для того, чтобы генерировать декои, нужно иметь уже существующую папку формата $NPD - Quast\ folder$ с настоящими спектрами. После этого надо ввести команду:

```
$ decoys.py method folder
```

Здесь:

- method — метод генерации декоев,
- folder — путь к папке формата $NPD - Quast\ folder$.

3.1.2 Добавление своих дерепликаторов

Для того, чтобы добавить свои дерепликаторы нужно либо добавить snakemake-файлы для переформатирования файлов со спектрами и базами данных и для запуска нового дерепликатора.

3.2 Запуск на тестовых данных

В качестве тестовых данных мы взяли набор аннотированных тестовых данных из [2]. При помощи NPD-Quast мы запустили на этих данных Dereplicator+ [1] в обоих режимах работы. Результаты в режиме работы на аннотированных данных представлены на Рис. 3–5 в Приложении. Для второго режима мы сделали декои методом главных пиков. График оценки доли ложных идентификаций представлен на Рис. 6.

4. Заключение

В ходе проделанной работы была разработана программа, позволяющая запускать дерепликаторы на данных в определённом формате и мерить качество их работы, а также сравнивать между собой. Помимо этого разработчики получили возможность добавть свой дерепликатор в список поддерживаемых. Также было реализовано несколько несложных методов генерации декоев, которые позволяют измерять качетво работы дерепликатора даже без правильных ответов.

Далее можно усовершенствовать методы генерации декоев, а также добавлять новые дерепликаторы в список поддерживаемых.

Репозиторий проекта

`https://github.com/artemmilov/NPD-QUAST`

Список литературы

[1] Mohimani, H., Gurevich, A., Shlemov, A. et al. Dereplication of microbial metabolites through database search of mass spectra. Nat Commun 9, 4035 (2018).

[2] Schymanski, E.L., Ruttkies, C., Krauss, M. et al. Critical Assessment of Small Molecule Identification 2016: automated methods. J Cheminform 9, 22 (2017).

[3] Dührkop, K., Fleischauer, M., Ludwig, M. et al. SIRIUS 4: a rapid tool for turning tandem mass spectra into metabolite structure information. Nat Methods 16, 299–302 (2019).

[4] Verdegem, D., Lambrechts, D., Carmeliet, P. et al. Improved metabolite identification with MIDAS and MAGMa through MS/MS spectral dataset-driven parameter optimization. Metabolomics 12, 98 (2016).

[5] Elias, J.E., Gygi S.P. Target-decoy search strategy for mass spectrometry-based proteomics. Proteome bioinformatics (2010).

[6] Scheubert, K., Hufsky, F., Petras, D. et al. Significance estimation for large scale metabolomics annotations by spectral matching. Nat Commun 8, 1494 (2017).

Приложение

```
BEGIN IONS
MSLEVEL=2
PEPMASS=167.104
CHARGE=1+
SCANS=1
56.0496         890015.8
58.0653         2683323
60.0557         19158144
68.0244         2064966.2
70.04   448485.8
81.0447         459195.2
83.0604         8710302
85.0508         32494888
100.0869        846321
108.0556        6498543.5
110.0461        1230449
112.0617        576228.1
125.0821        17335174
127.0727        814654.5
139.0725        2935127.5
140.093         1145310.1
150.0775        826622.3
167.1039        206684544
END IONS
```

Рис. 1: Пример файла со спектром

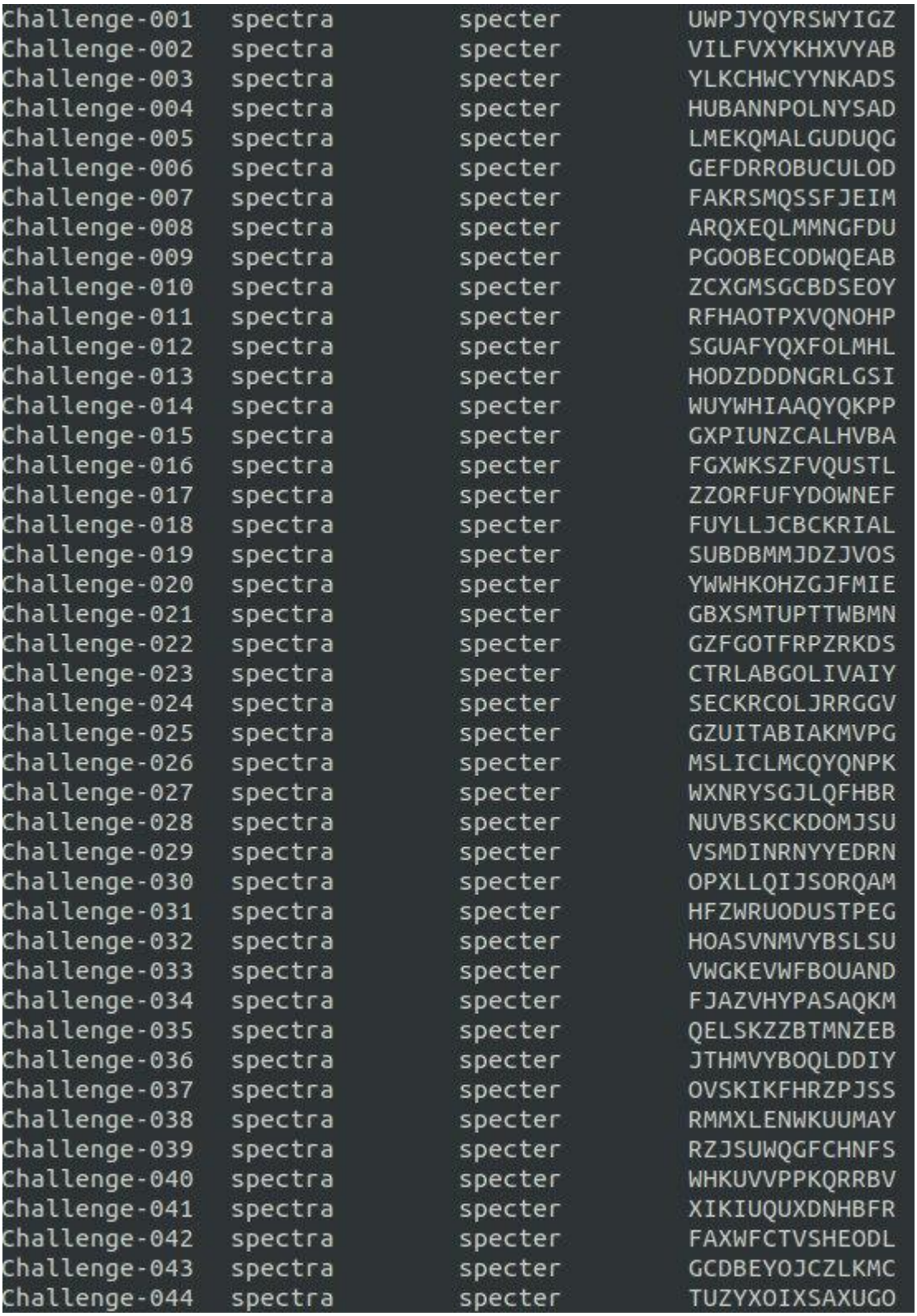

Рис. 2: Пример файла с правильными ответами

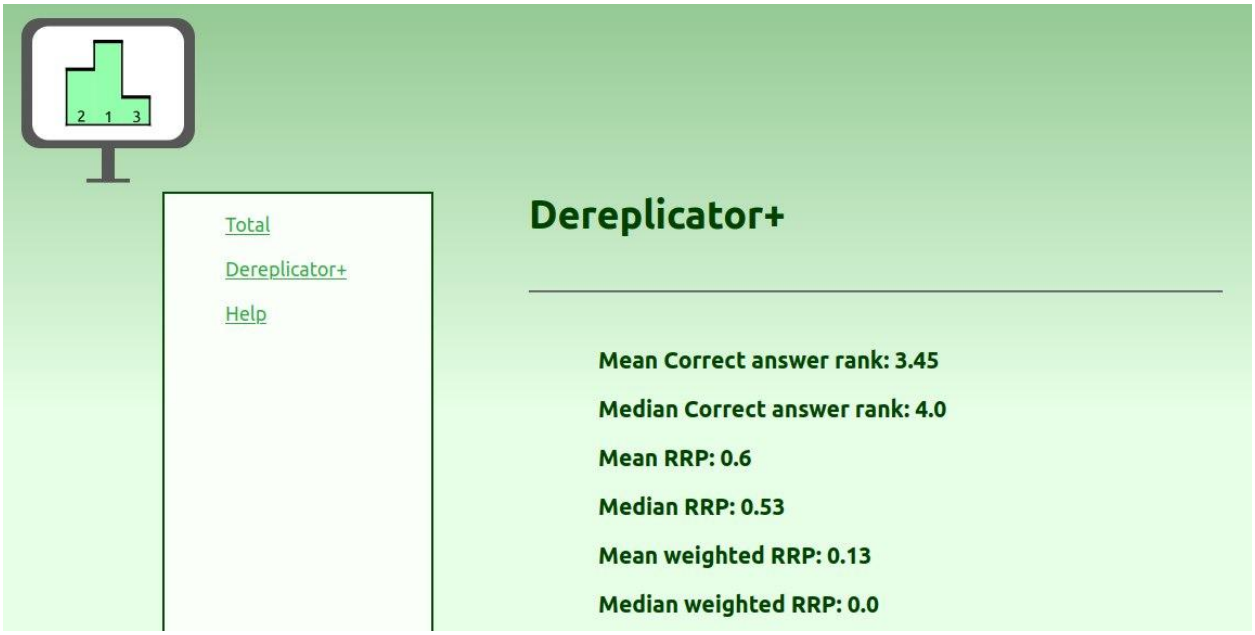

Рис. 3: Так выглядит отчёт одного из дерепликаторов

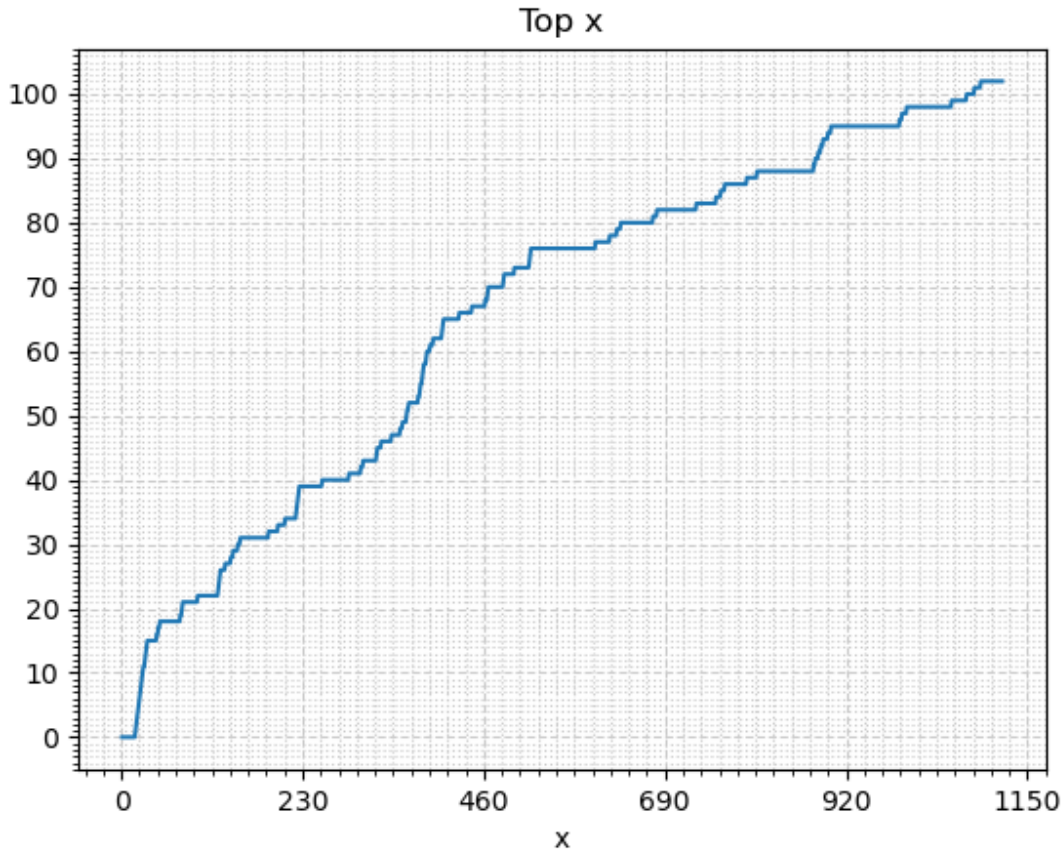

Рис. 4: График метрики "Позиции правильных идентификаций"

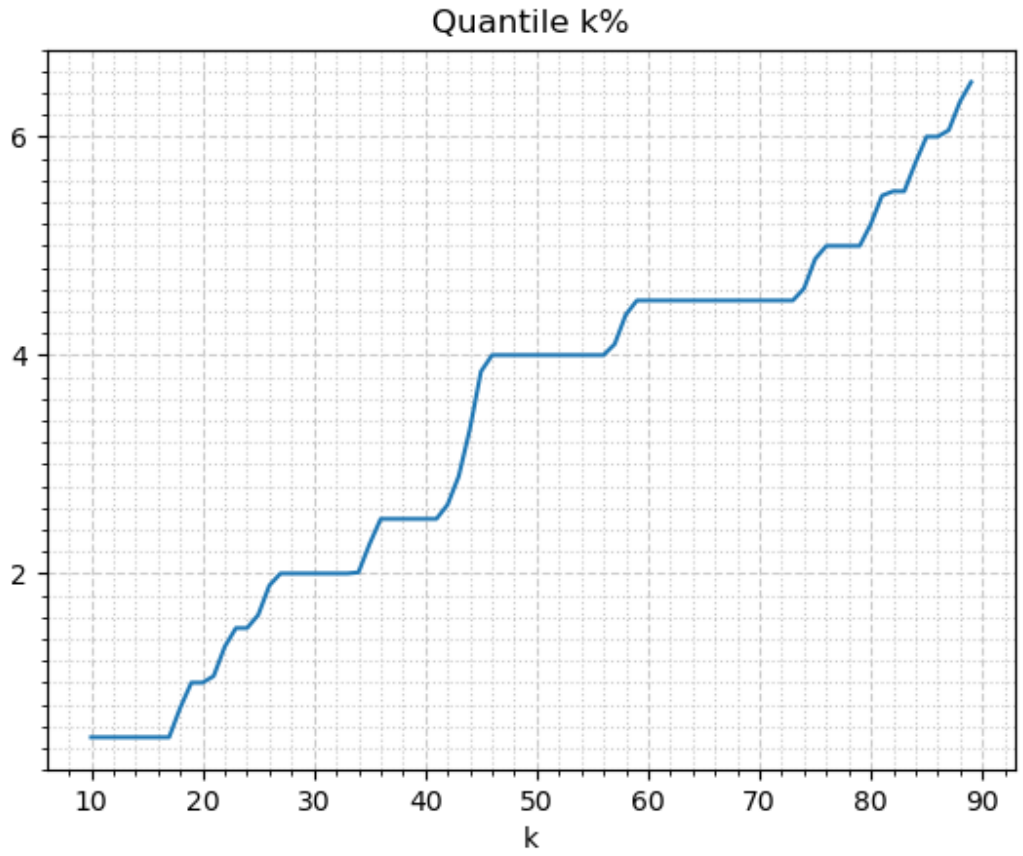

Рис. 5: График метрики "квантилей уровня k"

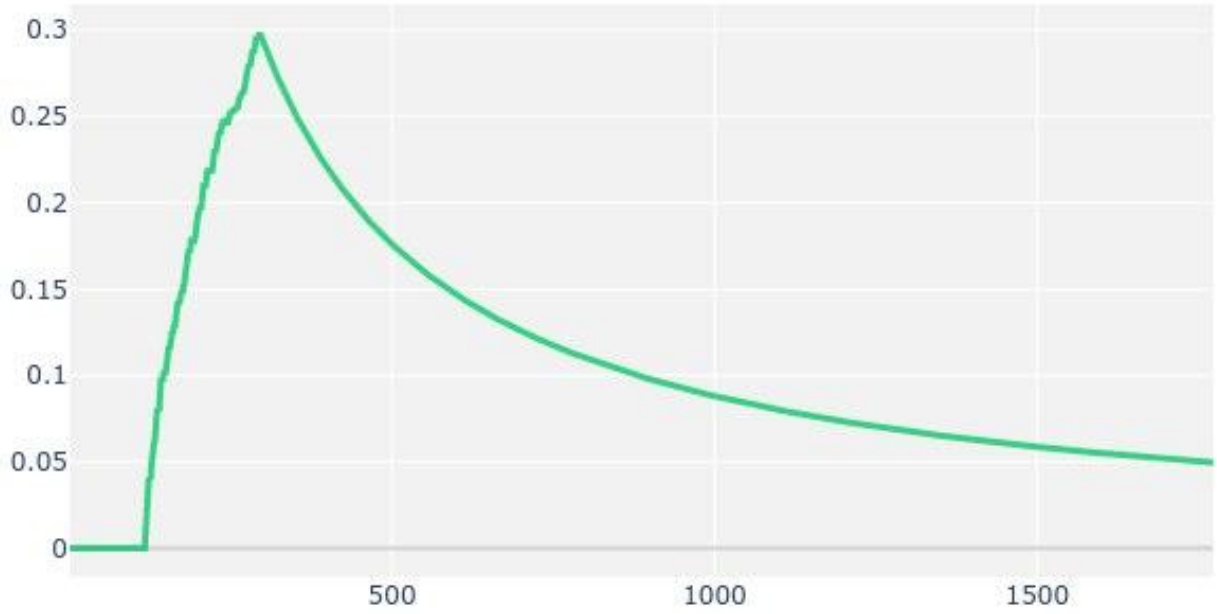

Рис. 6: График метрики "Доля ложных идентификаций"для сгенерированных декоев