Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики
Факультет информационных технологий и программирования

Кафедра информационных систем

Бутомов Артем Сергеевич

Разработка программного компонента для проведения сравнительного анализа биологических данный FAIRE-seq

Бакалаврская работа

Допущена к защите.
Зав. кафедрой:
профессор, д.т.н. Парфенов Владимир Глебович

Научный руководитель:
магистр Лебедев Сергей Андреевич

Рецензент:
аспирант Сергушичев Алексей Александрович

Санкт-Петербург
2015

Оглавление

Введение

ДНК (дезоксирибонуклеиновая кислота) — длинная двухцепочечная молекула, являющаяся носителем генетической информации в биологических организмах.

В клетке ДНК упакована, то есть убирается в маленькую по размеру клетку.

Изучать пространственную структуру ДНК организма важно для понимания механизмов регуляции жизнедеятельности клетки.

Формальдегидная изоляция регуляторных элементов с последующим секвенированием (Formaldehyde-Assisted Isolation of Regulatory Elements sequencing, FAIRE-Seq) — это биологический протокол, позволяющий находить участки, в которых ДНК доступна для связывания белками. Суть протокола заключается в том, что выделенную из клетки ДНК фиксируют на белки с помощью формальдегида. Затем ДНК фрагментируют с помощью ультразвука. После этого происходит разделение полученных фрагментов ДНК на две группы: участки связанные с белками и свободные участки. Далее свободные фрагменты читают с помощью секвенатора. И наконец, для каждого прочтения секвенатора определяют место в геноме исследуемого организма, откуда он был прочитан.

В контексте данной работы, геном разбивается на непересекающиеся отрезки фиксированной длины, называемые бинами. Подсчитывается количество прочтений, начинающихся внутри каждого отрезка. Таким образом, получатся вектор из неотрицательных целых чисел, именуемый вектором покрытия.

Из вектора покрытия можно сделать предположение о вероятности расплетения региона, чем больше значение элемента вектора, тем больше вероятность, что регион, соответствующий элементу, был расплетен.

Однако, рассматриваемый протокол не исключает возможности наличия ошибок в результатах биологического эксперемента. Неточности метода FAIRE-seq обусловлены следующими моментами.

1. Протокол работает с колонией клеток. Таким образом в результатах эксперимента мы видим некоторое среднее состояние по всем клеткам

2. Этап фиксации не обладает 100% КПД, то есть некоторые белки могут отвалиться

3. Этап разделения свободных и связанных фаз также неточен. Вместе со свободными вполне могут попасться и связанные фрагменты

Так как в результате эксперемента появляется шум, данные FAIRE-seq удобно анализировать с помощью вероятностых моделей.

Цель данной работы — разработать математическую модель для проведения сравнительного анализа нескольких эксперементов биологический данных FAIRE-seq и

научиться оценивать и контролировать число неверных предсказаний модели.

Для достижения цели были поставлены следующие задачи:

1. Изучить предметную область
2. Предложить несколько вероятностных моделей для сравнения экспериментов FAIRE-seq
3. Реализовать модели в виде программы на языке Python
4. Оценить эффективность полученной программы

1. Предлагаемые модели

1.1. Описание задачи

Пусть $\vec{x} = (x_1, .., x_N)$ — вектор прочтений, построенный из какого-то BAM файла. Сопоставим каждому наблюдению некоторую метку-состояние s_n из множества базовых состояний $s = \{+, --, \text{null}\}$, истинные значения которых не знаем. Каждое базовое состояние описывает ситуацию в одном образце.

Вероятностная модель позволяет найти наиболее правдоподобную последовательность состояний.

$$\hat{s}_{ML} = \arg\max_{i \in 1,..,S^N} \mathcal{P}(x, s; \theta)$$

1.2. Смесь многомерных распределений Пуассона

Будем моделировать количество прочтений вдоль генома с помощью смеси многомерных распределений Пуассона. Пусть $\pi = (\pi_1, .., \pi_S)$ — априорные вероятности компонент. Причем, $\sum \pi_i = 1$

$\lambda = (\lambda_1, .., \lambda_S)$ — параметры пуассоновских испусканий для каждой компоненты смеси. Тогда правдоподобие неполных данных записывается так:

$$p(x; \pi, \lambda) = \prod_{n=1}^{N} \sum_{i=1}^{S} \pi_i \mathcal{P}(x_n; \lambda_i) \tag{1}$$

Интерпретация в качестве порождающей модели следующая[3]: сначала случайным образом выбираем скрытое состояние, применяя распределение π, а затем используем выбранное скрытое состояние для порождения наблюдения.

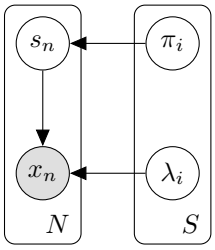

Рис. 1: Графическая диаграмма для пуассоновской смеси

Чтобы найти параметры модели с помощью оценки максимального правдоподобия, следует обратить в максимум совместное правдоподобие наблюдаемых и скрытых переменных. Методом максимального правдоподобия найти решение довольно сложно, поскольку для этого требуется решить систему, где оцениваемые параметры зависят от наблюдаемой выборки x и неизвестных значений скрытых состояний s. Асимптотическая сложность решения возрастет в S^N раз.

Поэтому будем искать приближенное решение с помощью EM-алгоритма, в котором нижняя оценка правдоподобия оптимизируется до сходимости. Последовательность действий формируется следующим образом[2]:

1. Инициализировать начальные значения параметров
2. Присвоение ожидаемых значений скрытым переменным при условии текущих оценок параметров и нахождение математического ожидания правдоподобия.

$$\mathrm{Q}(\theta|\theta^{\mathrm{old}}) = \mathrm{E}[\log p(x, s; \theta)] \leq \log p(x; \theta)$$

3. Переоценка параметров с учетом обновленных ожидаемых значений скрытых переменных

$$\theta^{\mathrm{new}} = \arg\max_{\theta \in \Theta} \mathrm{Q}(\theta|\theta^{\mathrm{old}})$$

4. Вычислить логарифм правдоподобия и проверить на сходимость

Выведем EM-алгоритм для смеси многомерных Пуассоновских испусканий.
Запишем логарифм функции правдоподобия:

$$\ln p(x; \pi, \lambda) = \sum_{n=1}^{N} \ln \sum_{i=1}^{S} \pi_i \mathcal{P}(x_n; \lambda_i) \tag{2}$$

Шаг E

$$\gamma(s_{ni}) = \frac{\pi_i \mathcal{P}(x_n; \lambda_i)}{\sum_{j=1}^{S} \pi_j \mathcal{P}(x_n; \lambda_j)} \tag{3}$$

Шаг M

Чтобы максимизировать логарифмическое правдоподобие относительно параметров, необходимо взять частные производные и приравнять их к нулю.

$$
\begin{aligned}
&\frac{\partial}{\partial \lambda_i} \mathrm{E}[\log p(x; s, \theta)] \\
&\qquad = \frac{\partial}{\partial \lambda_i} \sum_{n=1}^{N} \sum_{s=1}^{S} \mathrm{E}[s_{ni}] \{\log \pi_i + \log \mathcal{P}(x_n|\theta)\} \\
&\qquad = \sum_{n=1}^{N} \mathrm{E}[s_{ni}] \frac{\partial}{\partial \lambda_i} \log \mathcal{P}(x_n|\theta) \\
&\qquad = \sum_{n=1}^{N} \gamma(s_{nk}) \frac{\partial}{\partial \lambda_i} \log \frac{\lambda^{x_n} e^{-\lambda}}{x_n!} \\
&\qquad = \sum_{n=1}^{N} \gamma(s_{nk}) \frac{\partial}{\partial \lambda_i} (\log {\lambda_i}^{x_n} + \log e^{-\lambda_k} - \log x_n!) \\
&\qquad = \sum_{n=1}^{N} \gamma(s_{ni}) \frac{\partial}{\partial \lambda_i} (x_n \log \lambda_i - \lambda_i - \log x_n!) \\
&\qquad = \sum_{n=1}^{N} \gamma(s_{ni}) (\frac{x_n}{\lambda_S} - 1) \\
&\qquad = 0
\end{aligned}
$$

получаем

$$
\lambda_i^* = \frac{1}{N_i} \sum_{n=1}^{N} \gamma(s_{ni}) x_n \tag{4}
$$

где

$$
N_i = \sum_{n=1}^{N} \gamma(s_n i) \tag{5}
$$

Найдем новые значения априорных вероятностей. Воспользуемся методом множителей Лагранжа для учета ограничений на вектор априорных вероятностей.

$$\frac{\partial}{\partial \pi_i}(\mathrm{E}[\log p(x; s, \theta)] + \lambda(\sum_{j=1}^{S} \pi_j - 1))$$

$$= \frac{\partial}{\partial \pi_i} \sum_{n=1}^{N} \sum_{i=1}^{S} E[s_{ni}]\{\log \pi_i + \log \mathcal{P}(x_n; \theta)\} + \frac{\partial}{\partial \pi_i}\lambda(\sum_{j=1}^{S} \pi_j - 1)$$

$$= \sum_{n=1}^{N} \gamma(s_{ni}) \frac{\partial}{\partial \pi_i}\{\log \pi_i + \log \mathcal{P}(x_n|\theta)\} + \frac{\partial}{\partial \pi_i}\lambda(\sum_{j=1}^{S} \pi_j - 1)$$

$$= \sum_{n=1}^{N} \gamma(s_{ni}) \frac{\partial}{\partial \pi_i}\{\log \pi_i\} + \frac{\partial}{\partial \pi_i}\lambda(\sum_{j=1}^{S} \pi_j - 1)$$

$$= \sum_{n=1}^{N} \gamma(s_{ni}) \frac{1}{\pi_i} + \lambda$$

$$= 0$$

Домножив на π_k, получаем

$$\sum_{n=1}^{N} \gamma(s_{ni}) + \pi_i \lambda = 0 \tag{6}$$

Просуммируем вдоль i

$$\sum_{n=1}^{N} \sum_{i=1}^{S} \gamma(s_{ni}) + \sum_{i=1}^{S} \pi_i \lambda = 0 \tag{7}$$

Применяя

$$\sum_{i=1}^{S} \gamma(s_{ni}) = 1$$

и

$$\sum_{i=1}^{S} \pi_i = 1$$

Из выражения (7) получаем

$$N + \lambda = 0$$

$$\lambda = -N$$

Далее, подставляя найденное λ в выражение (6), получаем

$$\pi_i = -\frac{\sum_{n=1}^{N} \gamma(s_{ni})}{\lambda} = \frac{N_i}{N}$$

Наконец,

$$\pi_i^* = \frac{N_i}{N} \tag{8}$$

Примечание. Чтобы обучить наши данные, следует проинициализировать начальные значения входных параметров модели. Для этой задачи можно использовать алгоритм кластеризации KMeans++[4], который разбивает наши наблюдения на S кластеров. Работа алгоритма основана на методе максимального правдоподобия. На шаге E мы определяем для каждого наблюдения ближайший кластер. На шаге M Вычисляем новое значение кластера, которое принимаем за среднее выборочное наблюдений, относящихся к данному кластеру. Алгоритм итерируется до тех пор, пока изменения логарифма правдоподобия не станет меньше константы. В результате, начальные значений параметров Пуассоновского распределения можно принять за значения кластеров.

1.3. Скрытая Марковская Модель

На практике предположение о независимости состояний между соседними наблюдениями в предыдущей модели не выполняется.

Поэтому, перейдем к Скрытой Марковской Модели второго порядка, чтобы учесть зависимость между состояниями соседних наблюдений.

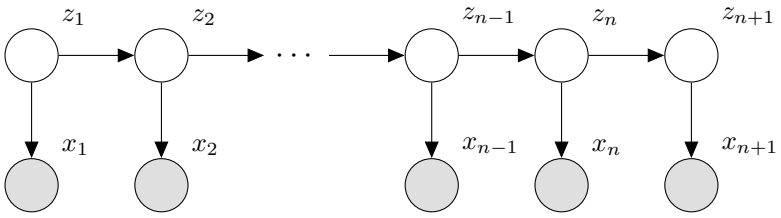

Рис. 2: Графическая диаграмма для марковской цепи

$z_{(n+1),i}$ зависит от $z_{n,i}$

Введем понятие базовых состояний: $S \in \{+, -, \text{null}\}$. Каждое из базовых состояний описывает ситуацию в одном образце.

Семантика обозначений следующая:
(+) - сигнал есть, (−) - шумовый сигнал, (null) - сигнала нет.

Замечание. Отличие (null) от (−) заключаются в полном отсутствии сигнала.

Для задачи сравнения нам нужно множество состояний, описывающее, что происходит в каждом из образцов, то есть S^2.

Состояние s с одинаковым базовыми состояниями $(+,+), (-,-), (\text{null}, \text{null})$ означает, что данные сравниваемых образцов наблюдения x_n похожи между собой.

Чтобы задать модель, нужно определить распределения испусканий, то есть $p(x_n; s_{ni} = 1)$, где i — индекс состояния из S^2, а n — индекс наблюдения. Будем считать, что x_n — это наблюдение из многомерного распределения Пуассона с независимыми компонентами, то есть:

$$p(x_n; s_{ni}) = \prod_{d=1}^{2} p(x_{nd}; \lambda_{id}) \tag{9}$$

На данном этапе для каждого состояния и каждого образца есть свой параметр распределения Пуассона, что является неверной параметизацией. Рассмотрим два со-

стояния $i = (+,+)$ и $j = (+,-)$ и выпишем для них функцию вероятности распределения Пуассона:

$$p(x_n|s_{ni}) = \prod_{d=1}^{2} p(x_{nd}|\lambda_{id}) = p(x_{n1}|\lambda_{i1})p(x_{n2}|\lambda_{i2})$$

$$p(x_n|s_{nj}) = \prod_{d=1}^{2} p(x_{nd}|\lambda_{jd}) = p(x_{n1}|\lambda_{j1})p(x_{n2}|\lambda_{j2})$$

Первый множитель в обоих выражениях соответствует наблюдению, порождённому базовым состоянием (+) в первом образце. Логично положить, что $\lambda_{i1} = \lambda_{j1}$, потому что в обратном случае испусканиям для одного и того же базового состояния будут соответствовать разные параметры распределения Пуассона. Таким образом, различных параметров у нас не $2 * |S^2| = 8$, а $2 * |S| = 4$.

Для реализации удобно представлять параметры многомерного распределения Пуассона в виде вектора размерности $2*|S|$, а для состояния i из S^2 использовать матрицу трансляции D. Матрица трансляции — это двухмерная матрица размерности $2 \times |S^2|$, где D_{di} - индекс параметра из вектора:

$$\vec{\lambda} = \{\lambda_1, \lambda_2, \lambda_3, \lambda_4\, \lambda_5, \lambda_6\}$$

		(n,n)	(+,+)	(–,–)	(n,+)	(n, –)	(–, n)	(+,n)	(+, –)	(–, +)
D	d=1	3	4	5	6	7	8	9	10	11
	d=2	3	4	5	6	7	8	9	10	11

Пусть

$\pi = (\pi_1, .., \pi_{S^2})$ — априорные вероятности состояний.

$\mathcal{A}$ — матрица вероятностей перехода между состояниями.

Перепишем функцию вероятности распределения Пуассона в терминах D.

$$p(x_n; s_{ni}) = \prod_{d=1}^{2} \prod_{s=1}^{|S^2|} p(x_{nd}; \lambda_s)^{I[D_{di}=s]} \tag{10}$$

Функция правдоподобия определяется как

$$p(x, s; \theta) = p(s_1; \pi) \left[\prod_{n=2}^{N} p(s_n; s_{n-1}, A) \right] \prod_{m=1}^{N} p(x_n; s_m, \theta) \tag{11}$$

Подставив определение функции вероятности распределения (10) в функцию прав-

доподобия для СММ (11) можно убедиться, что М-шаг для вектора лямбд:

$$\lambda_s = \frac{\sum_{d=1}^{2} \sum_{i=1}^{|S^2|} I[D_{di} = s] \sum_{n=1}^{N} \gamma_{ni} x_{nd}}{\sum_{d=1}^{2} \sum_{i=1}^{|S^2|} I[D_{di} = s] \sum_{n=1}^{N} \gamma_{ni}} \tag{12}$$

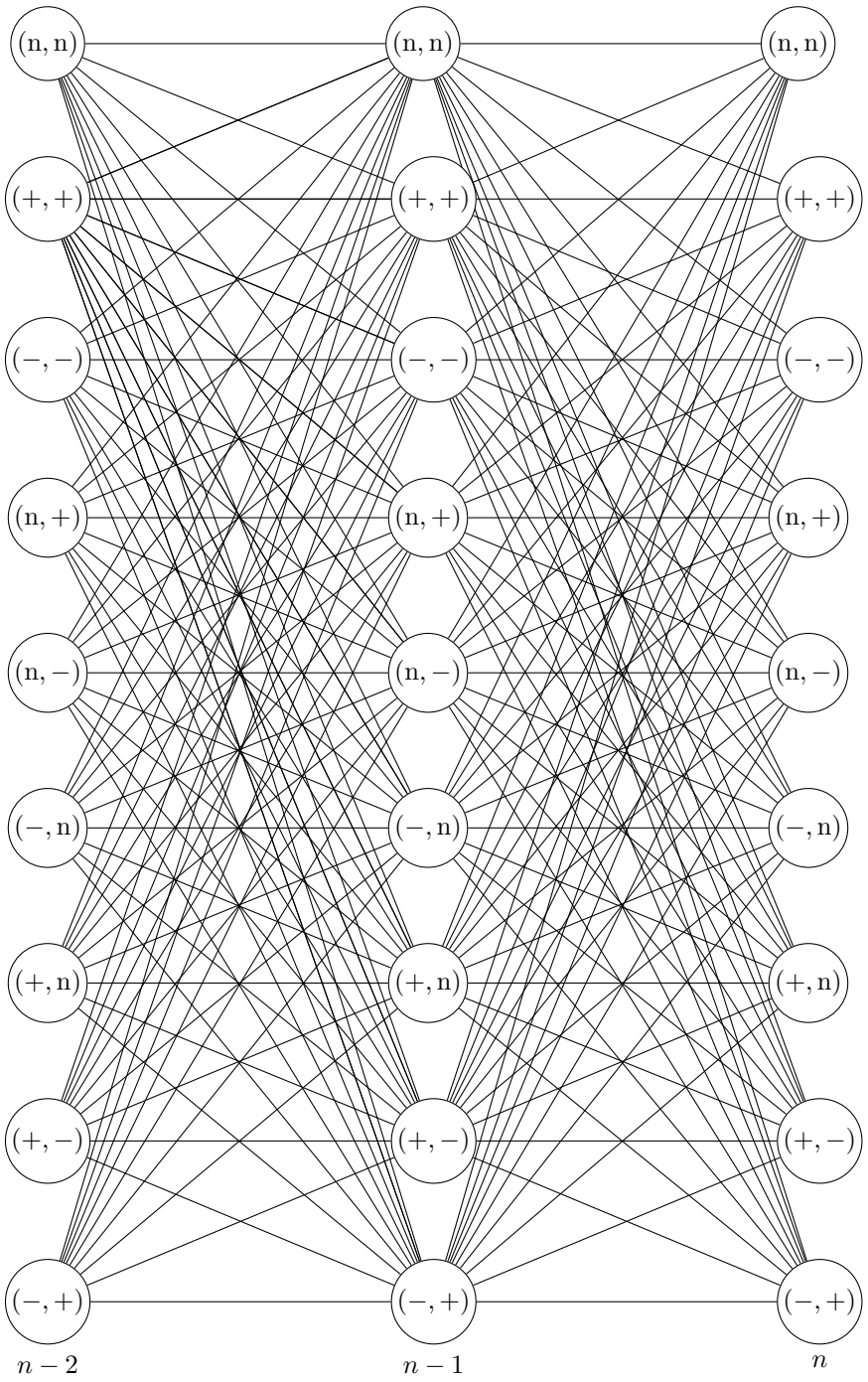

Рис. 3: Решетка перехода состояний

Решетка представляет собой диаграмму переходов между скрытыми состояниями модели. Красным светом выделены состояния, в котором базовые состояния похожи, то есть $(+,+), (-,-), (\text{null}, \text{null})$

Шаг E

$$\gamma_{ni} = \frac{\alpha_{ni}\beta_{ni}}{\sum_{j=1}^{S^2} \alpha_{nj}\beta_{nj}} \tag{13}$$

$$\xi_{nij} = \frac{\alpha_{(n-1),i}\mathcal{A}_{ij}\mathcal{P}(x_n;\lambda_j)\beta_{nj}}{\sum_{i'=1}^{S^2}\sum_{j'=1}^{S^2}\alpha_{(n-1),i'}A_{i'j'}\mathcal{P}(x_n;\lambda_{j'})\beta_{nj'}} \tag{14}$$

Где,

$\alpha_{ni} = p(s_{ni} = 1, x_1, x_2, ..x_n; \theta)$

$\beta_{ni} = p(x_{n+1}, .., x_N; s_{ni} = 1, \theta)$

Вычисления α и β производится с помощью алгоритма прямого-обратного хода[5]:

$$\alpha_{1i} = \pi_i\mathcal{P}$$

$$\beta_{Ni} = 1$$

$$\alpha_{ni} = \mathcal{P}(x_n;\lambda_i)\sum_{j=1}^{S}\alpha_{(n-1),j}A_{ji}$$

$$\beta_{ni} = \sum_{j=1}^{S}A_{ij}\mathcal{P}(x_{n+1};\lambda_j)\beta_{(n+1),j}$$

Шаг M

Новые значения параметров модели вычисляются так:

$$\pi_i^* = \gamma(s_{1i}) \tag{15}$$

$$A_{ij}^* = \frac{\sum_{n=2}^{N}\xi_{nij}}{\sum_{j'=1}^{S}\sum_{n=2}^{N}\xi_{nij}\xi_{nij'}} \tag{16}$$

Замечание. В алгоритме прямого-обратного хода может быть underflow - это значит, что не хватает точности чисел с плавающей точкой. Поэтому удобно проводить вычисления в логарифмах[1].

1.3.1. Предсказание модели

В результате работы алгоритма, необходимо предсказать наиболее вероятную последовательность скрытых состояний, породивших наши наблюдения. Для каждого наблюдения выбирается состояние, соответствующее наибольшей апостериорной

вероятности[7].

$$s_n = \arg\max_{i \in 1,..,S^N} \gamma(s_{ni})$$

1.4. Выбор модели

Для задачи сравнения двух образцов была выбрана Скрытая Марковская Модель. В частном случае, для анализа одного биологического образца СММ более правдоподобнее, чем смесь Пуассоновских испусканий. Это значит, что модель, в которой предполагается зависимость между состояниями соседних наблюдений, более правдоподобная.

1.5. Актуальности разработки

Существуют инструменты для анализа одного эксперимента с использованием FAIRE-seq:

- ChromHMM
- ZINBA
- Fseq
- ChIPOTle Peak Finder
- и другие ...

Примечание. Косвенный аналог ChromHMM моделирует многомерную последовательность из $\{0, 1\}$. Данный инструмент, как и другие аналоги, используется для анализа одного биологического образца. Стало быть, для задачи сравнения он не подходит. Также ChromHMM не умеет моделировать репликаты(образцы).

2. Оценка модели

2.1. Оценка качества работы модели

При оценке качества работы модели мы хотели бы получать число неверных предсказаний FDR[6] среди всех предсказаний модели. Для того чтобы ввести FDR, сформулируем гипотезы, которые будем проверять с помощью модели:

- H_0 - разницы между состояниями эксперементальных данных в одном наблюдении нет
- H_1 - разница есть

Рассмотрим некоторое наблюдение с индексом n в нашей выборке, сформируем критерий

- $P(\text{отличий в n нет}; x, \theta) := p_0$
- $P(\text{отличия в n есть}; x, \theta) := p_1$

H_0 отвергается если $p_0 < p_1$ и не отвергается в обратном случае.

Примечание. Вспомним, что $p_0 + p_1 = 1$, поэтому наш критерий можно записать как:

$$p_0 \leq 0.5$$

Таким образом, применив сформулированный выше критерий ко всем бинам $n = 1, ..., N$ мы получим N результатов. FDR $= a \in [0, 1]$ означает, что среди N результатов a N — неверны.

В общем виде FDR записывается так:

$\text{FDR} = \text{E}[\text{FP} / (\text{TP} + \text{FP})]$,

где FP — количество неверно отвергнутых нулевых гипотез, TP — количество верно отвергнутых нулевых гипотез.

Для удобства будем использовать следующую разновидность FDR:

$$\text{mFDR} = \text{E}[\text{FP}] / \text{E}[\text{TP} + \text{FP}] \tag{17}$$

Числитель можно представить ввиду суммы индикаторных случайных величин FP_n

$$\text{E}[\text{FP}] = \sum_{n=1}^{N} \text{E}[\text{FP}_n]$$

Если n-й учаток не содержит отличий, то $\text{FP}_n = 0$. Если n-й участок по мнению модели содержит отличия, то $\text{FP}_n = 1$ в случае, когда отличия действительно есть и 0 в обратном случае.

Разумеется, правильного ответа для каждого n мы не знаем, но у нас есть вероятность ошибки, а именно $P(\text{отличий в n нет}; x, \theta)$, которую мы обозначили за p_1. Посчитаем математическое ожидание:

$$\mathrm{E}[\mathrm{FP}] = \sum_{n=1}^{N} I[p_1 \leq 0.5] p_1$$

Здесь мы воспользовались тем, что математическое ожидание индикаторной случайной величины с параметром p равняется p.

Перейдём к знаменателю. Выражение для E[FP] у нас уже есть, построим аналогичное выражение для E[TP]. $\mathrm{TP}_n = 1$ если отличия в n-м участке есть, и модель с этим солидарна, и 0 — во всех остальных случаях. Заметим, что TP_n независимы потому же почему и FP_n.

$$\mathrm{E}[\mathrm{TP}] = \sum_{n=1}^{N} I[p_1 <= 0.5] p_0$$

Выпишем то, что получилось:

$$\mathrm{E}[\mathrm{TP} + \mathrm{FP}] = \sum_{n=1}^{N} (I[p_1 \leq 0.5] p_1 + I[p_1 \leq 0.5] p_0) = \sum_{n=1}^{N} I[p_1 \leq 0.5]$$

Здесь мы воспользовались тем, что $p_0 + p_1 = 1$

И всё вместе:

$$\widehat{\mathrm{mFDR}} = \frac{\sum\limits_{n=1}^{N} I[p_1 \leq 0.5] p_1}{\sum\limits_{n=1}^{N} I[p_1 \leq 0.5]}$$

Таким образом, оценив апостериорные вероятности мы можем также оценить и mFDR. Важно понимать, что по выборке (точнее по апостериорным вероятностям для выборки) мы получили оценку mFDR, а не его истинное значение.

2.2. Контроль FDR

Мы хотим выдавать вектор предсказаний, в котором гарантированно не более чем фиксированное число ошибок.

3. Проектирование

Системную архитектуру можно представить следующим образом. Мы наследуем и расширяем библиотечный класс, реализующий скрытую Марковскую модель.

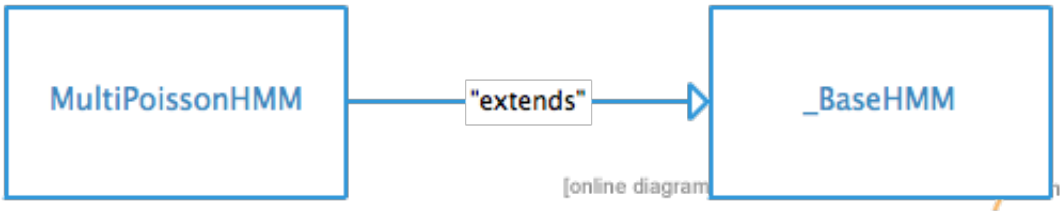

Рис. 4: системная архитектура

Программа разделена на три компоненты. Классы Reader и Writer выделены в отдельные классы для того, чтобы показать их независимость от класса MultiPoissonHMM, то есть абстрагируемся от деталей реализации этих классов, так как каждый класс выполняет независимую и отдельную задачу (см. Рис. 6). Класс Reader читает BAM файл и строит вектор покрытия. Класс MultiPoissonHMM обучает модель и предсказывает состояния, а также оценивает и контролирует статистику FDR. Для построения вектора предсказаний используется паттерн Strategy. Класс Writer содержит методы, которые записывает результаты модели в файлы.

На рис. 3 представлена диаграмма последовательности. В качестве обьектов выступают экземпляры классов. Reader передает данные сущности MiltiPoissonHMM, после некоторой обработки полученных данных MultiPoissonHMM отправляет синхронное сообщение обьекту Algorithm для получения вектора предсказаний(маски). И наконец, обработанные результаты передаются обьекту Writer для их записи.

В файле README можно найти описание программного компонента и примеры использования. Файлы __init__.py, __main__.py и setup.py предназначены для развертывания программы.

```
thesis
├── setup.py
├── fairy
│   ├── __init__.py
│   ├── __main__.py
│   ├── _speedups.c
│   ├── _speedups.pyx
│   └── faireanalysis.py
└── README.rst
```

Рис. 5: структура файлов

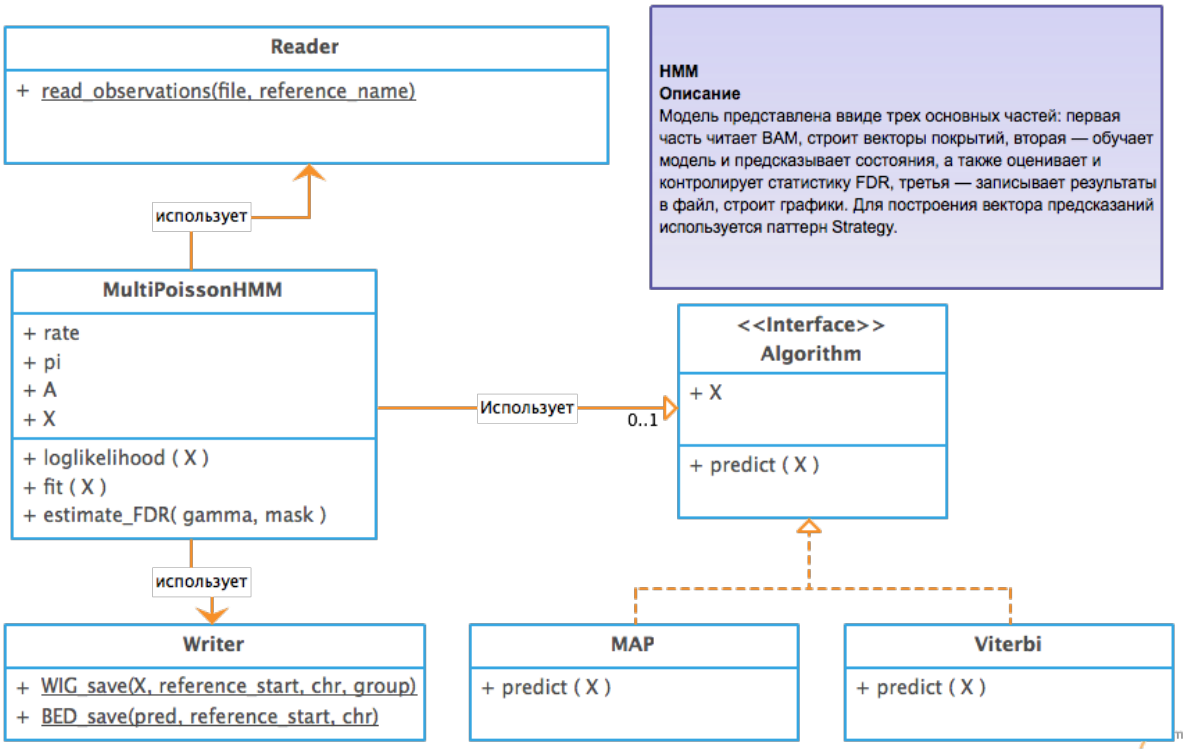

Рис. 6: диаграмма классов

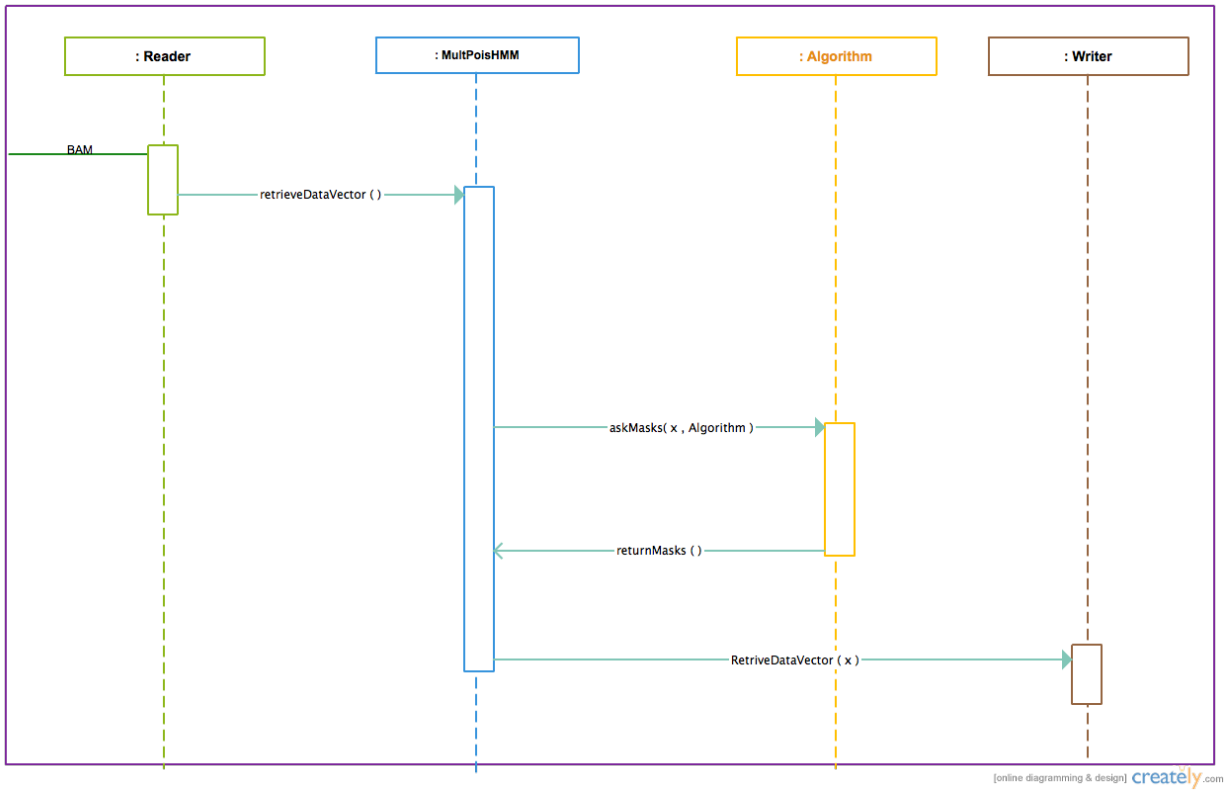

Рис. 7: диаграмма последовательности

4. Реализация

4.1. Особенности реализации

Для реализации CMM воспользовались расширением класса _BaseHMM библиотеки hmmlearn, перегрузив основные функции _compute_log_likelihood для вычисления нижней оценки правдоподобия, _initialize_sufficient_statistics - для инициализации начальных параметров модели, _accumulate_sufficient_statistics - для нахождения апостериорных вероятностей одного и двух последовательных скрытых переменных и _do_mstep - для нахождения новых значений параметров. Также были добавлены методы для моделирования репликатов и нахождения оценки FDR.

Основной технической проблемой кода является производительность, то есть время работы алгоритмов. В скрытой Марковской модели вычисление ξ на шаге E работает за $O(S^2T)$, где S — количество состояний, а T — размер выборки.

Язык Python не имеет строгой типизации, что сказывается на производительности. Для оптимизации кода были применены следующие подходы:

- Векторизация
- Cython
- Распараллеливание

Cython — это язык написания модулей C/C++ для Python. Применить его можно для ускорения работы с циклом. В нашем случае для перебора всех участков хромосомы, необходимый для построения вектора покрытия(см. код _speedups.c); Векторизация позволяет избегать явных циклов; Для этого можно воспользоваться пакетами научных вычислений NumPy/SciPy. Более того, применение векторизации делает код ясным и простым для понимания; И наконец, с помощью распараллеливания работы можно добиться двоекратного преимущества в производительности. Для этих целей применили модуль joblib.

4.2. Интерфейс командной строки

Программа реализована таким образом, что ее можно поставить как Python пакет. Для установки и запуска программы можно воспользоваться командной строкой. Сборка и установка производится следующей командой

```
$ python setup.py install
```

Программа может принимать три аргумента:

- --gr1 с указанием через запятую названий входных файлов первой группы образцов без указания расширения .bam

- --gr2 с указанием через запятую названий входных файлов второй группы образцов без указания расширения .bam
- --chr с перечислением названий хромосом, которые мы хотим проанализировать. В случае, если пропустить использование этого аргумента программа проанализирует все хромосомы.

Формат запуска выглядит следующим образом:

$$\begin{aligned} \$ \quad \text{python} \quad \text{-m} \quad \text{fairy} \quad & \text{--gr1} = \text{ENCFF000TJP}, \text{ENCFF000TJR} \\ & \text{--gr2} = \text{ENCFF000TJJ}, \text{ENCFF000TJK} \quad \text{--chr} = \text{chr2}, \text{chr3} \end{aligned}$$

5. Применение

Исходные данные были взяты из проекта ENCODE, который занимается изучением человеческого ДНК. В качестве исходных данных взяли образцы эмбриональных стволовых клеток H1-hESC и клеток рака груди (например, ENCFF000TJJ. bam, ENCFF000TJK. bam). Рассматриваемые образцы являются результатами работы FAIRE-seq протокола для одной колонии клеток, но при этом содержат отличия.

Статистику анализа двух образцов, соответсвующих одному биологическому состоянию, можно представить в следующей таблице.

	образцы	ENCFF000TJJ	ENCFF000TJK
chr1	похожесть	81,49%	
	mFDR	0,018%	
chr2	похожесть	93,52%	
	mFDR	0,261%	
chr3	похожесть	94,16%	
	mFDR	0,287%	
...		...	
chr20	похожесть	84,86%	
	mFDR	0,015%	
chr21	похожесть	95,18%	
	mFDR	0,00008%	

Рис. 8: Статистика №1.

Похожестью мы называем отношением числа участков, которые содержат одинаковые базовые состояния сравниваемых образцов, к общему количеству участков.

Статистика анализа групп образцов.

	образцы	ENCFF000TJJ	ENCFF000TJK	ENCFF000TJP	ENCFF000TJR
chr1	похожесть	91,48%			
	mFDR	0,0015%			
chr2	похожесть	82,99%			
	mFDR	0,002%			
chr3	похожесть	68,9%			
	mFDR	0,002%			
...		...			
chr20	похожесть	64,97%			
	mFDR	0,001%			
chr21	похожесть	81,47%			
	mFDR	0,00001%			

Рис. 9: Статистика №2.

Для визуализации результатов модели формируем три файла: один BED файл и WIG файлы для каждой группы образцов. Группой образцов мы называем объединением общих измерений разных клеток, имеющие схожую ДНК.

Формат вывода в BED файл представленный ниже, где chrom — название хромосомы; (chromStart, chromEnd) — участок, где модель обнаружила разницу.

chrom	chromStart	chromEnd
chr20	125869	128269
chr20	129669	130069
chr20	135469	136869
chr20	138669	140269
chr20	146069	147469
chr20	158869	160069
chr20	160869	162269
chr20	167269	168869
chr20	173269	173869

Рис. 10: Формат файла *.bed

Формат вывода WIG файла следующий.

fixedStep	chrom=chr20	start=80600	step=200
0.0			
0.0			
0.0			
1.0			
7.5			
3.5			
5.0			
0.0			
0.0			
1.5			
12.0			
14.5			

Рис. 11: Формат файла *.wig

В файл WIG экспортируется сигнал усредненный по всем образцам в каждой группе. Формат WIG, как и его родственник BED, предназначен для описания отрезков вдоль генома. Для описания мы воспользовались так называемым fixedStep вариантом, в котором расстояние между отрезками фиксировано и взято за 200 пар нуклеотидов.

Время работы программы на эмбриональных стволовых клетках H1-hESC и клетках рака груди ENCFF000TJJ. bam, ENCFF000TJK. bam составило 15 мин. и 31 с.

Заключение

В рамках работы были достигнуты следующие цели:

1. Разработана математическая модель для анализа и сравнения результатов нескольких FAIRE-seq эксперементов с помощью скрытой Марковской модели. Данная модель является улучшением смеси, так как позволяет учитывать зависимость между соседними участками. Для каждой модели были приведены описание и обоснование.

2. Проведено сравнение биологических образцов одной колонии клеток. Результаты показали 80-95 % похожести репликатов, что кажется адекватным.

3. Были оценены и проконтролированы число неверных предсказаний модели. Результаты для биологических репликатов составили 3 - 23 ошибочных предсказаний среди всех 10^5 предсказаний.

Список литературы

[1] Adams Ryan. Computing Log-Sum-Exp // Building Intelligent Probabilistic Systems Machine learning, statistics, neuroscience, everything... — 2012. — URL: https://hips.seas.harvard.edu/blog/2013/01/09/computing-log-sum-exp/ (online; accessed: 09.01.2013).

[2] Bishop Christopher. Pattern Recognition and Machine Learning (Information Science and Statistics). — Springer, 2007. — Ozon Books : http://www.ozon.ru/context/detail/id/2978313/.

[3] Flach Peter. Machine Learning The Art and Science of Algorithms that Make Sense of Data. Adaptive Computation and Machine Learning series. — Cambridge University Press, 2012. — Cambridge books : http://www.cambridge.org/us/academic/subjects/computer-science/pattern-recognition-and-machine-learning/machine-learning-art-and-science-algorithms-make-sense-data.

[4] Murphy Kevin P. Machine Learning: A Probabilistic Perspective. Adaptive Computation and Machine Learning series. — MIT Press, 2012. — Amazon Books : http://www.amazon.com/Machine-Learning-Probabilistic-Perspective-Computation/dp/0262018020.

[5] Rabiner Lawrence R. A tutorial on hidden Markov models and selected applications in Speech Recognition. — 2012. — URL: http://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/tutorialonhmmandapplications.pdf (online; accessed: 1989).

[6] Wikipedia. False discovery rate // Wikipedia, the free encyclopedia. — 2012. — URL: http://en.wikipedia.org/wiki/False_discovery_rate (online; accessed: 15.05.2015).

[7] К.В.Воронцов. Машинное обучение (курс лекций) // Байесовская теория классификации - Принцип максимума апостериорной вероятности. — 2009. — URL: http://www.machinelearning.ru/wiki (дата обращения: 13.04.2015).