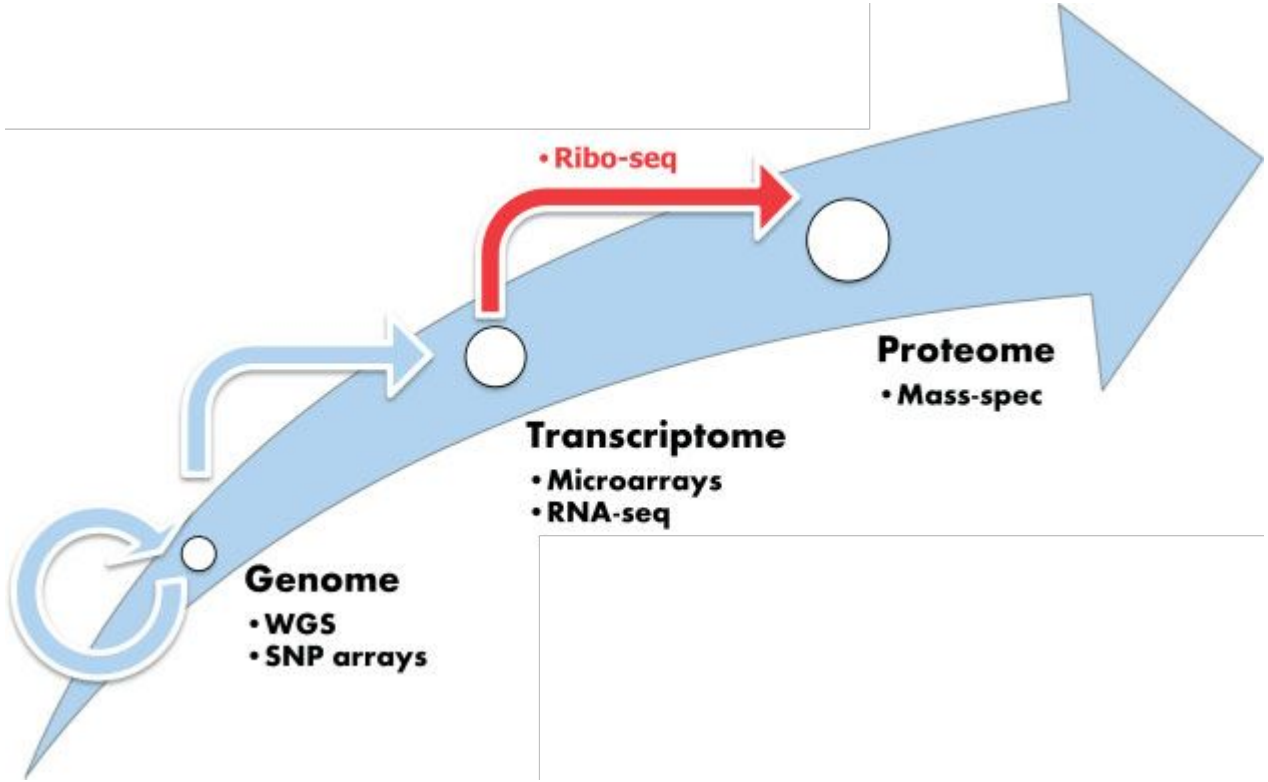
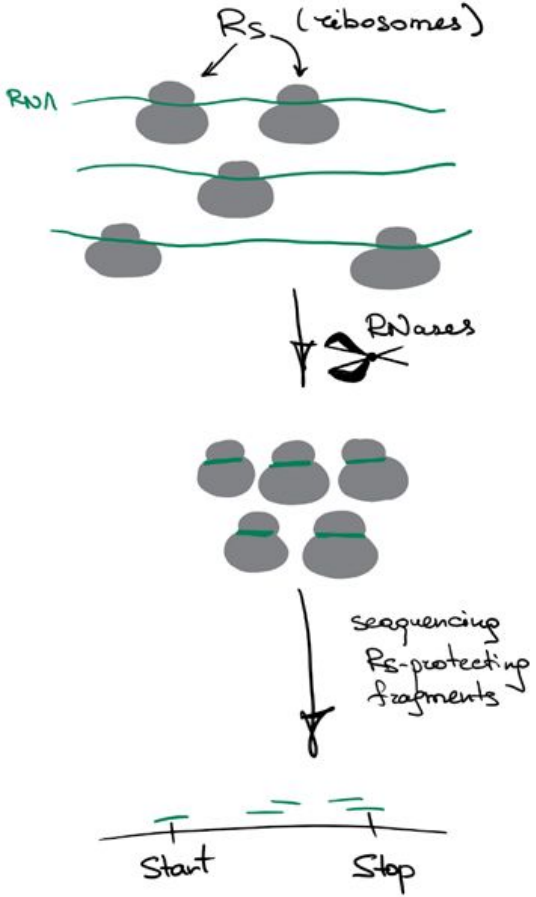
Анализ данных рибосомного профайлинга

Полезные инструменты для Ribo-Seq и других над-транскриптомных данных

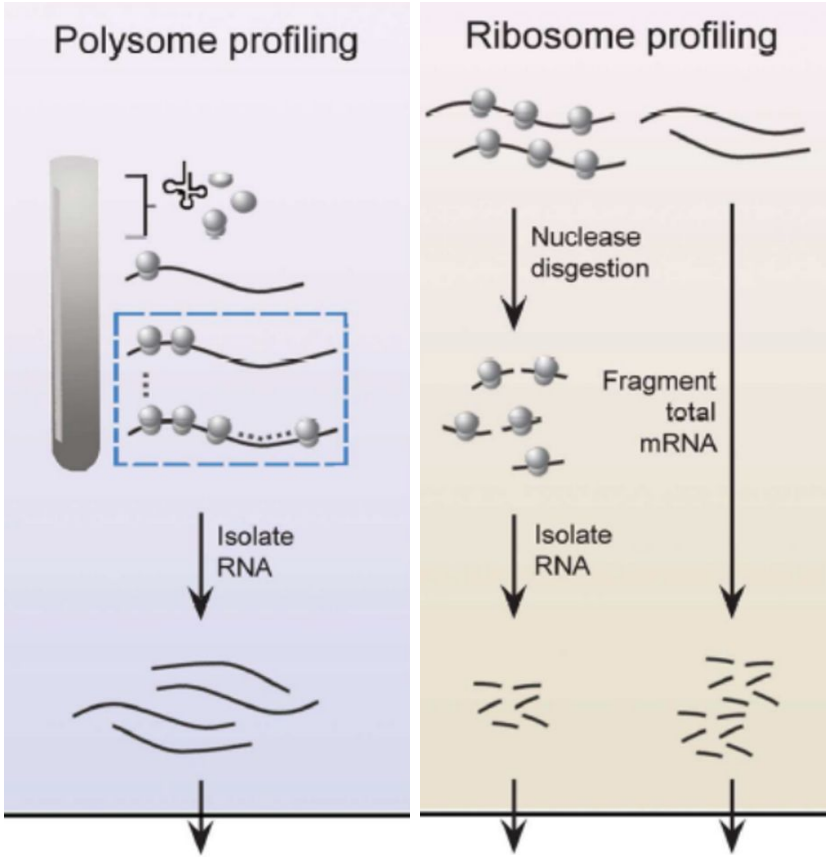
Где Ribo-Seq


• Ribo-seq
Proteome
• Mass-spec
Transcriptome
• Microarrays
• RNA-seq
Genome
• WGS
• SNP arrays

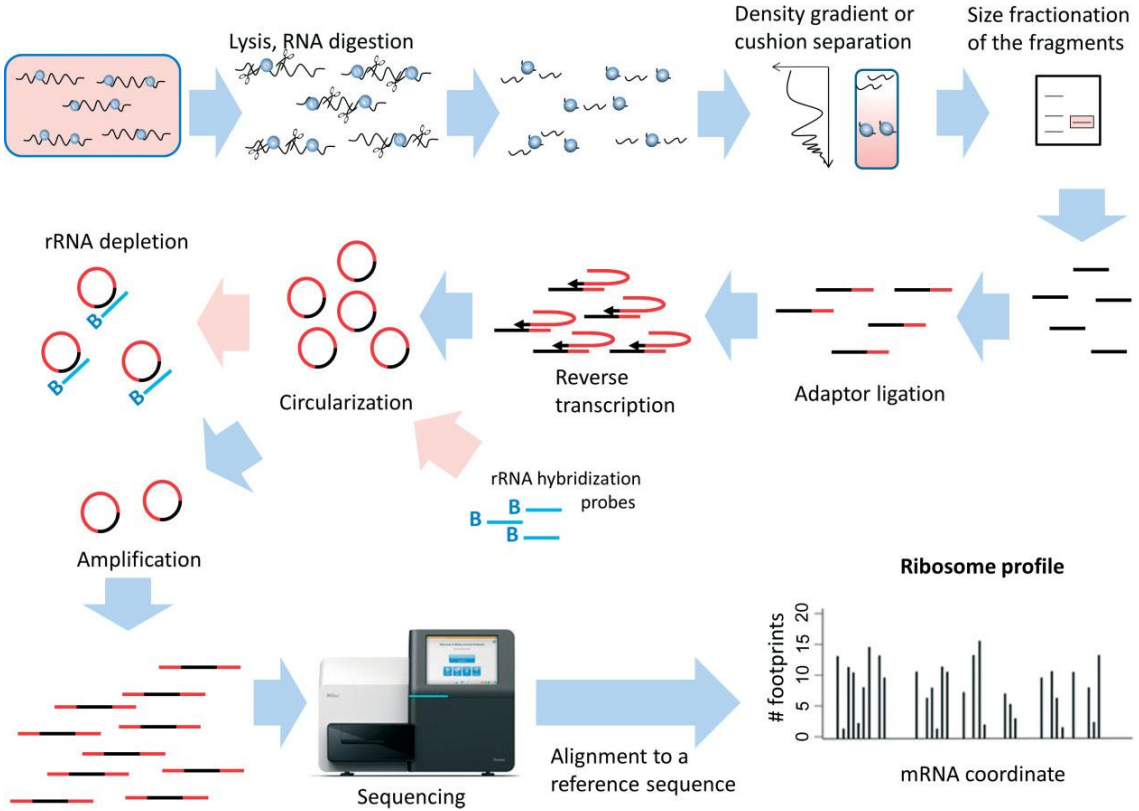
Что Ribo-Seq


Rs (ribosomes)
RNA
RNases
sequencing
Rs-protecting
fragments
Start
Stop

Что Ribo-Seq


Polysome profiling
Isolate
RNA
Ribosome profiling
Nuclease
disgestion
Fragment
total
mRNA
Isolate
RNA

Что Ribo-Seq


Lysis, RNA digestion
Density gradient or cushion separation
Size fractionation of the fragments
rRNA depletion
Circularization
Reverse transcription
Adaptor ligation
rRNA hybridization probes
Amplification
Sequencing
Alignment to a reference sequence
Ribosome profile
footprints
0 5 10 15 20
mRNA coordinate

Зачем Ribo-Seq

Трансляция в различных внешних условиях

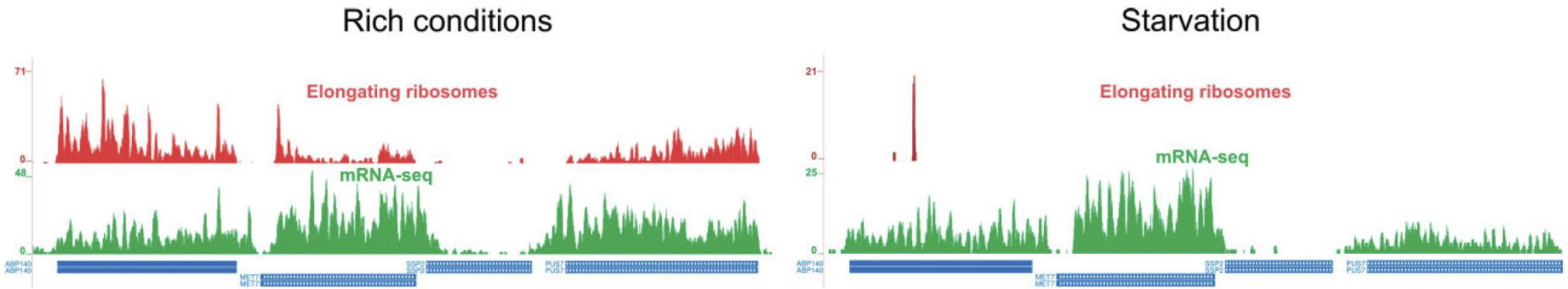
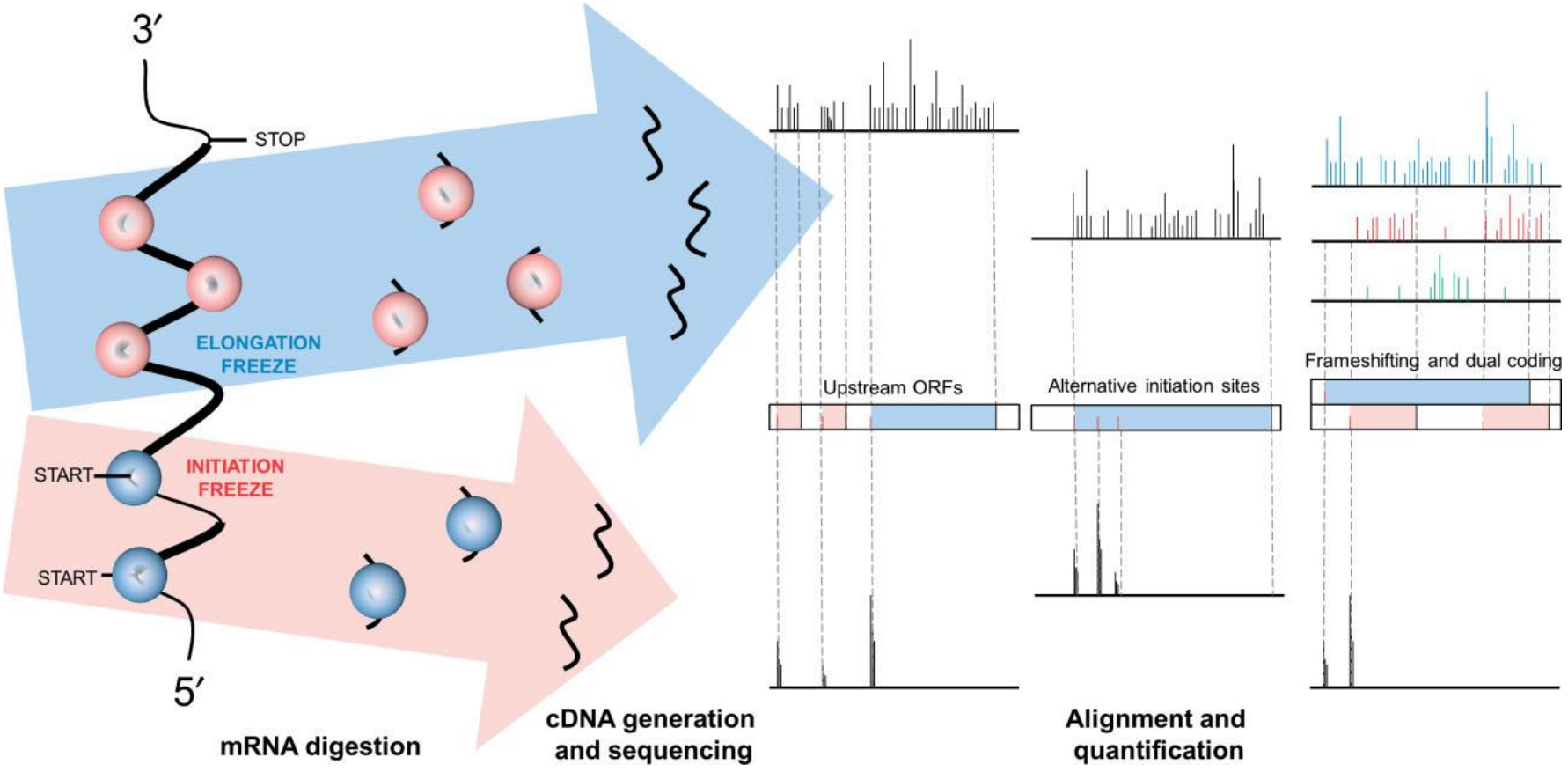
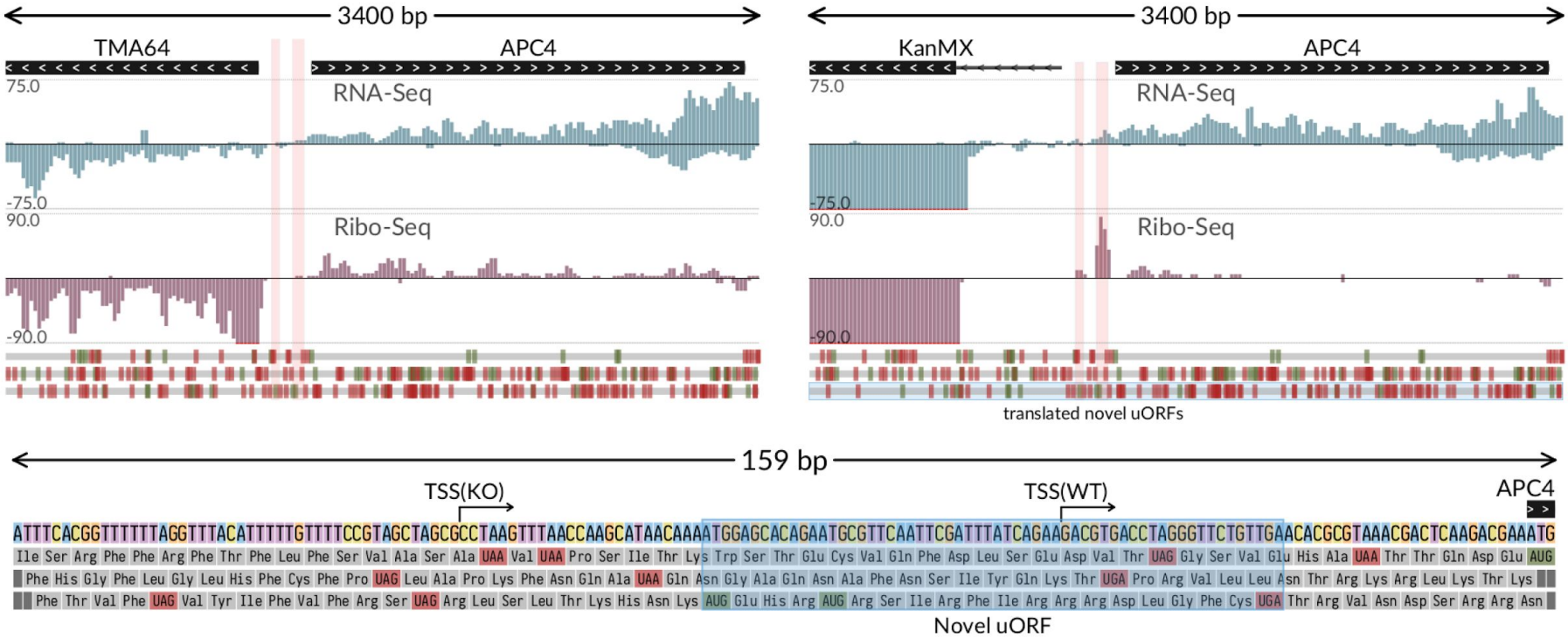


FIGURE 4 | Ribo-seq (red) and mRNA-seq (green) coverage plots for the *S. cerevisiae* genome locus containing *ABP140*, *MET7*, *SSP2*, and *PUS7* genes obtained with GWIPS-viz (http://gwips.ucc.ie/) using data from Ref 1. Under starvation conditions (right), *ABP140*, *MET7* and *PUS7* are transcribed, but not translated.

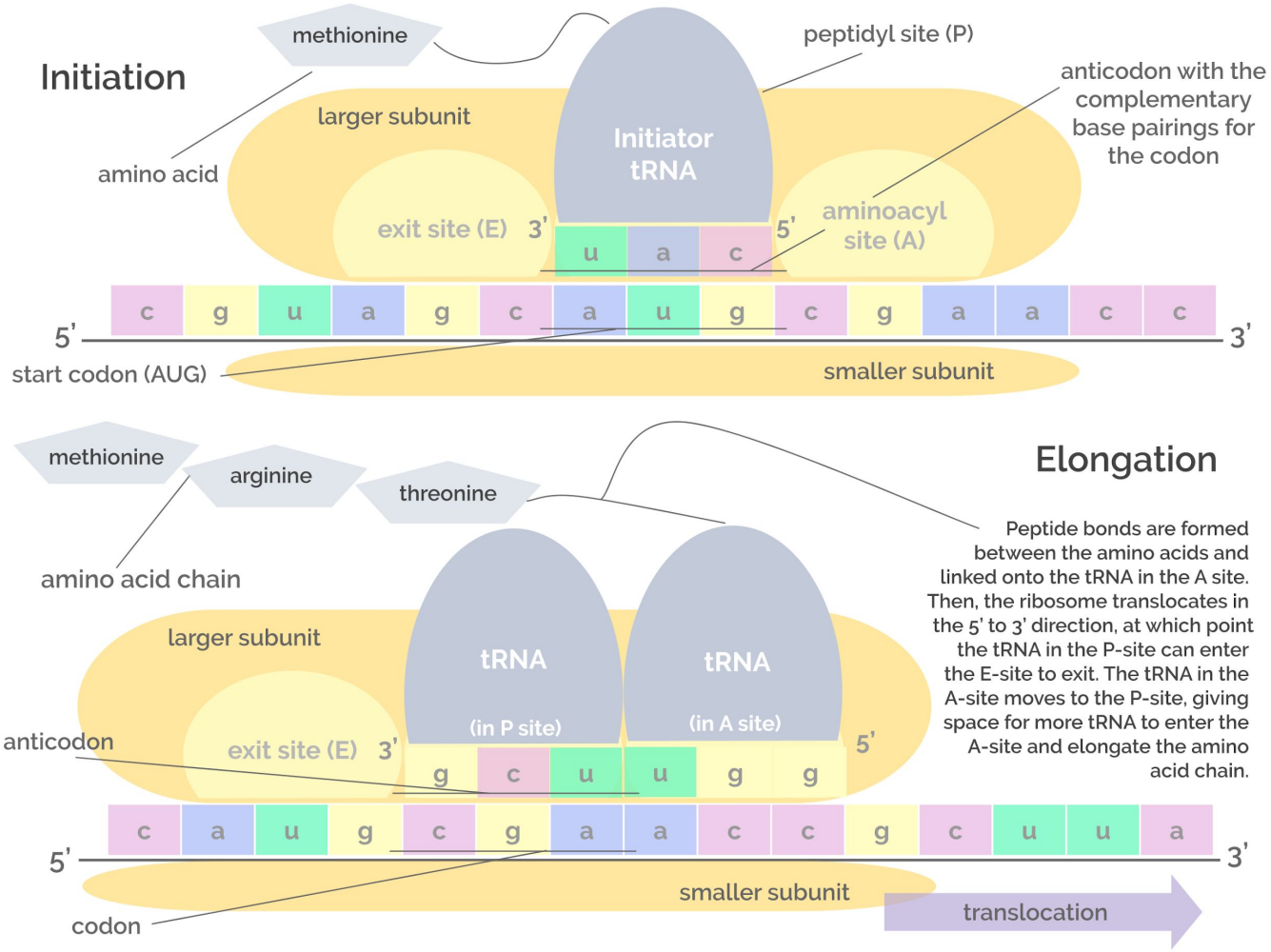
Зачем Ribo-Seq



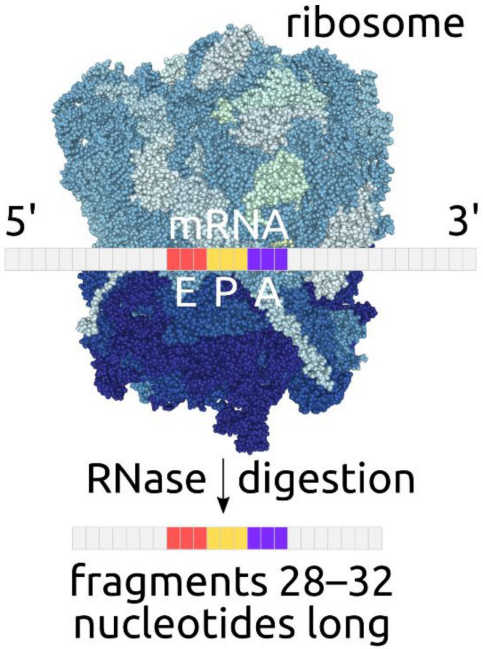
Нокаут кассетой KanMX изменяет трансляцию соседних генов

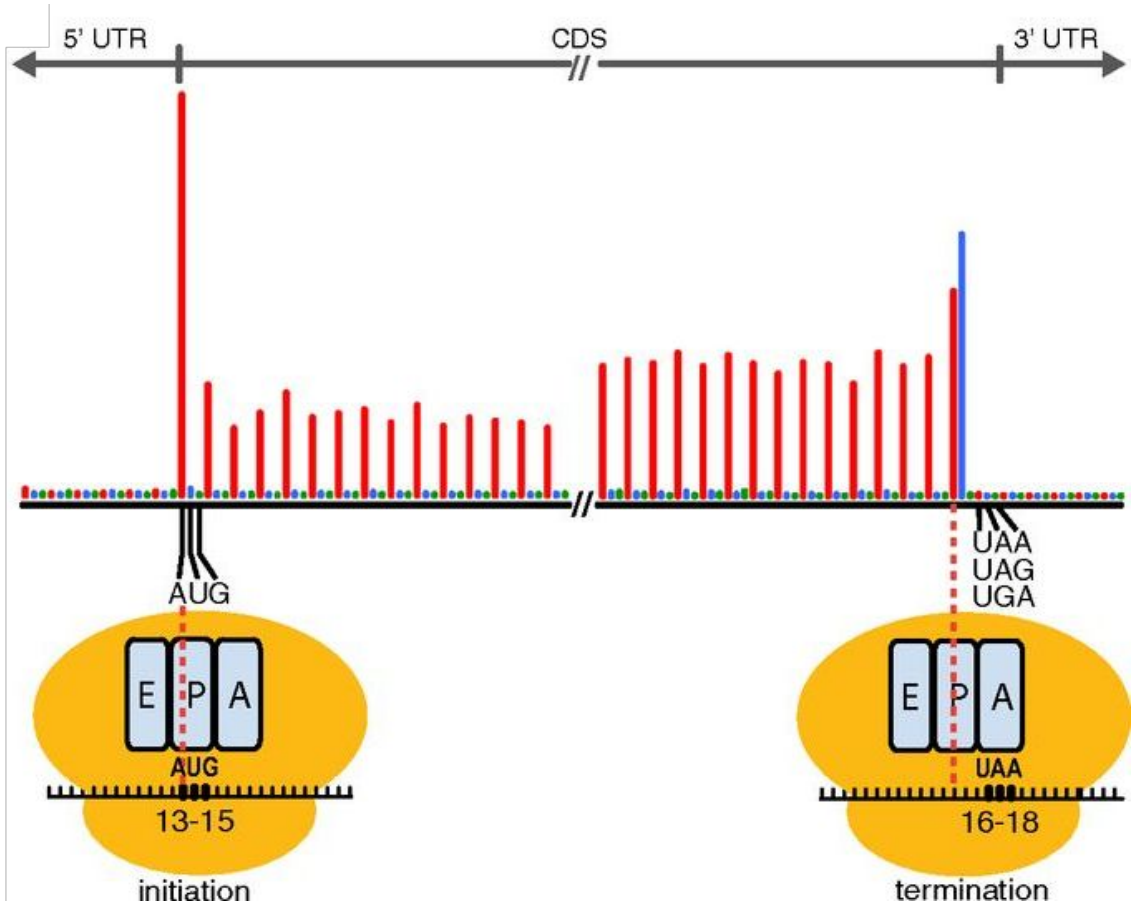


*plotted with svist4get


Initiation
methionine
peptidyl site (P)
anticodon with the complementary base pairings for the codon
larger subunit
amino acid
Initiator tRNA
exit site (E)
3'
5'
aminoacyl site (A)
u a c
c g u a g c a u g c g a a c c
5'
3'
start codon (AUG)
smaller subunit
Elongation
methionine
arginine
threonine
amino acid chain
Peptide bonds are formed between the amino acids and linked onto the tRNA in the A site. Then, the ribosome translocates in the 5' to 3' direction, at which point the tRNA in the P-site can enter the E-site to exit. The tRNA in the A-site moves to the P-site, giving space for more tRNA to enter the A-site and elongate the amino acid chain.
larger subunit
tRNA
(in P site)
tRNA
(in A site)
anticodon
exit site (E)
3'
5'
g c u u g g
c a u g c g a a c c g c u u a
5'
3'
smaller subunit
translocation
codon

Фазирование





Зачем Ribo-Seq

Определение
фреймшифтов

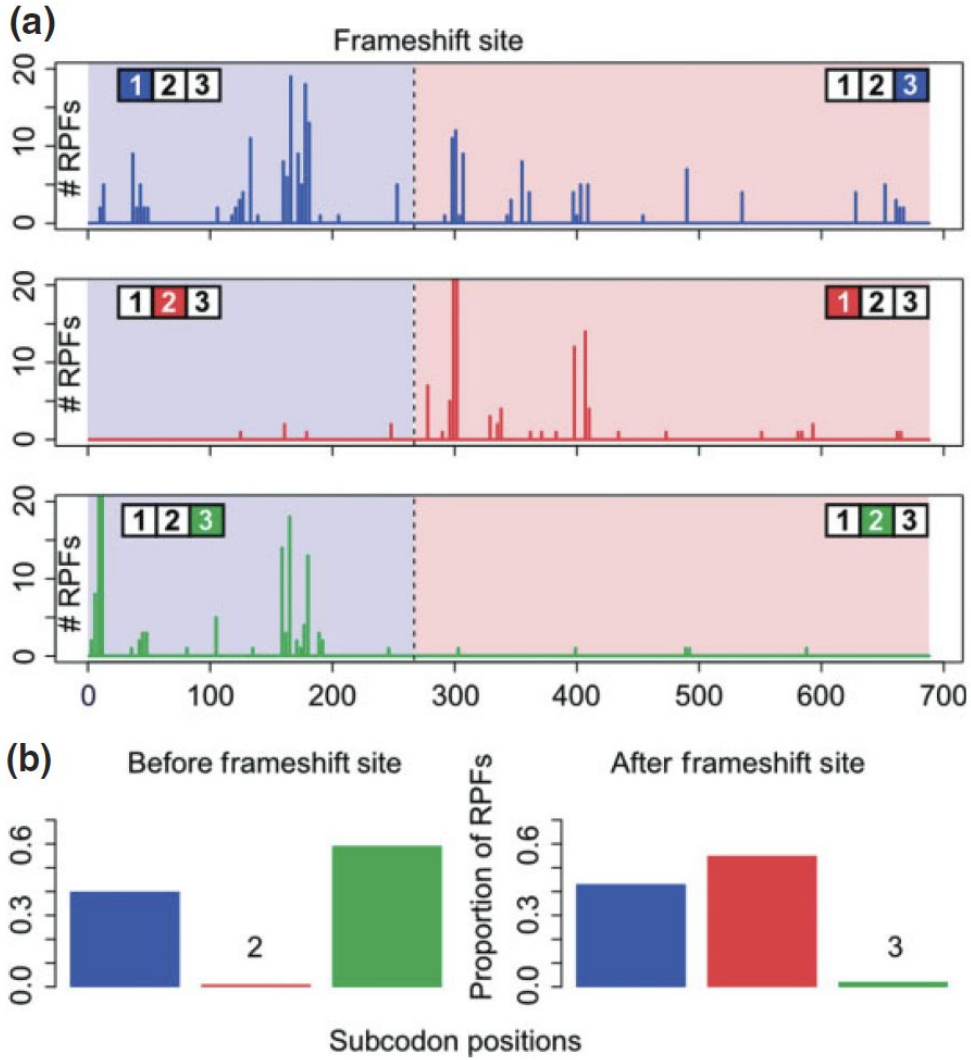
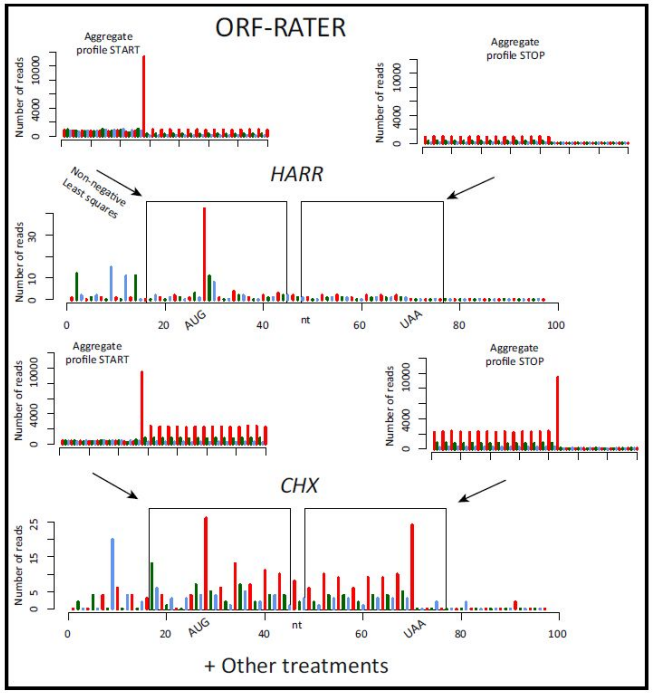
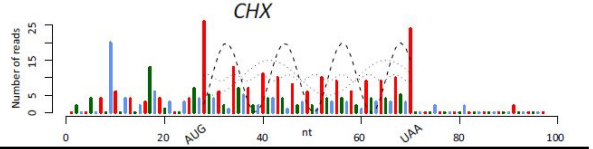


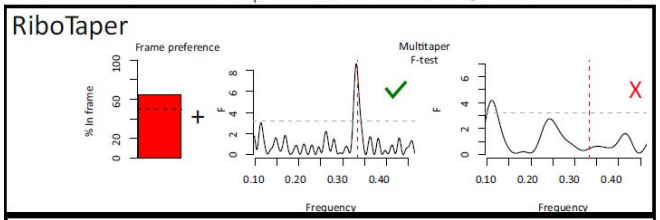
FIGURE 8 | The utilization of triplet periodicity for detecting transitions in translated reading frames. Panel (a) shows the absolute number of RPFs aligning to each subcodon position for the coding region of human antizyme 1 (*OAZ1*) mRNA.

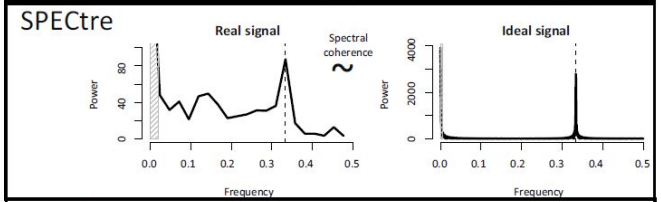
Зачем Ribo-Seq

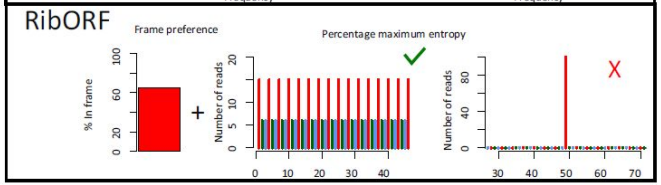
Неканонические рамки считывания

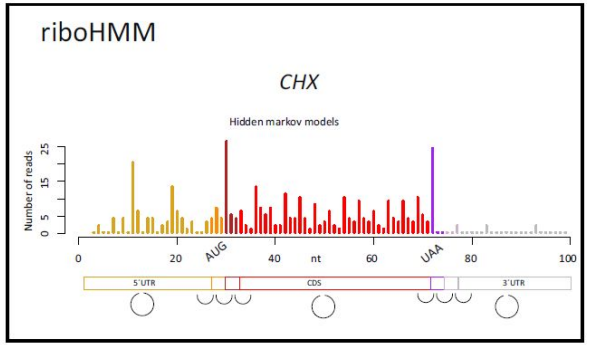


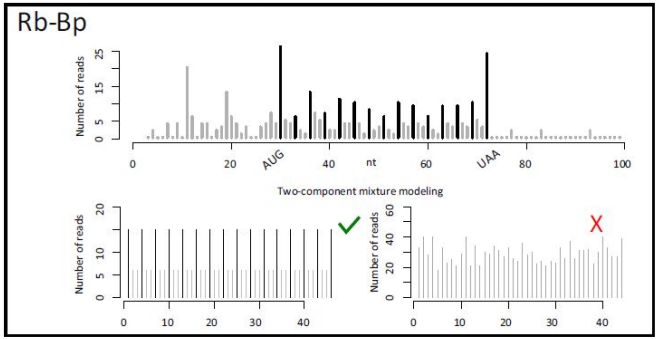












Intro based on

Ribosome profiling: a Hi-Def monitor for protein synthesis at the genome-wide scale

Audrey M. Michel and Pavel V. Baranov*

How to cite this article:
WIREs RNA 2013, 4:473–490. doi: 10.1002/wrna.1172

riboseq.org

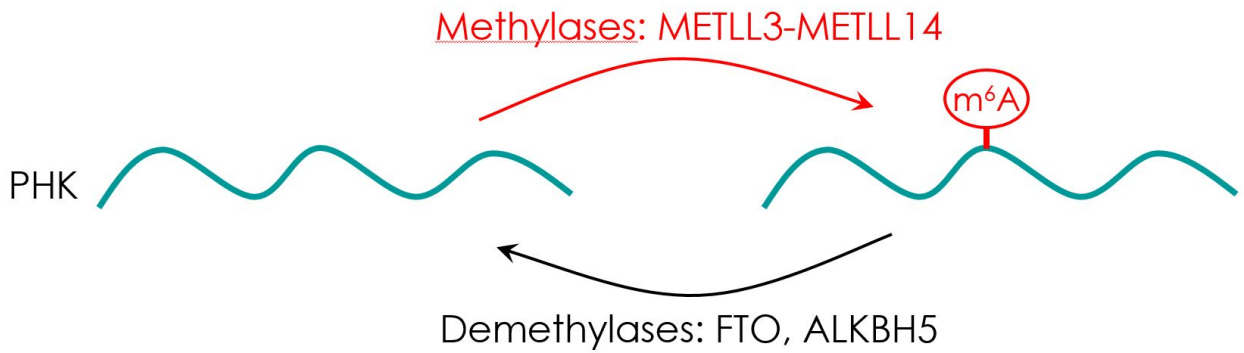
Практикум

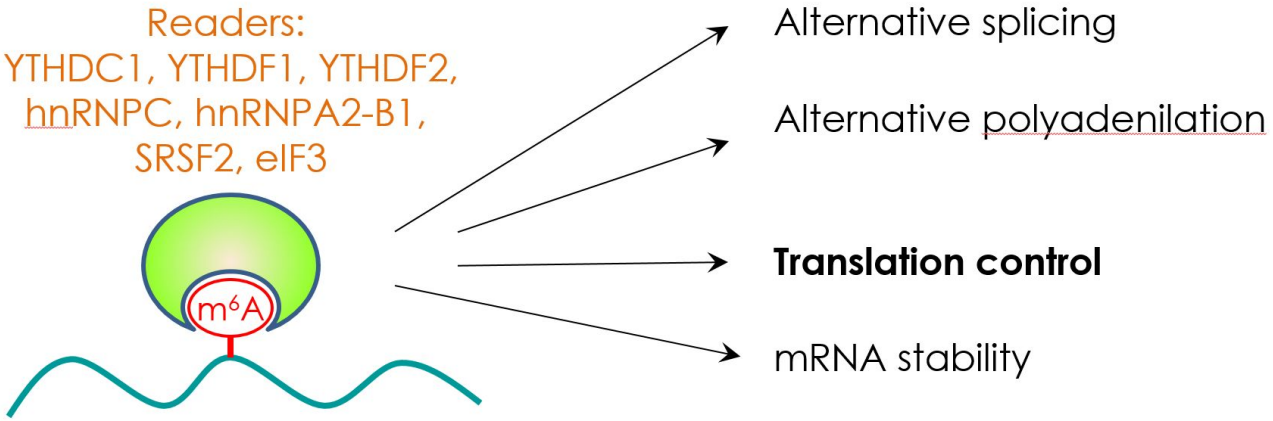
GEO+SRA = edirect + sratoolkit
Trimming+QC = cutadapt + FastQC
rRNA content = bowtie
[premade and filtered bam files]
Reads phasing+Metagene profiles = plastid
bedGraph+Visualization = plastid + svist4get

Что анализируем?

Эпитранскриптомный датасет по метилированию РНК m6A - нокдаун метилазы METTL3 и "читателя" метильной метки ABCF1

Идентификатор серии GEO:
GSE101865


Methylases: METLL3-METLL14
m⁶A
PHK
Demethylases: FTO, ALKBH5


Readers:
YTHDC1, YTHDF1, YTHDF2,
hnRNPC, hnRNPA2-B1,
SRSF2, eIF3
m⁶A
Alternative splicing
Alternative polyadenilation
Translation control
mRNA stability

Этап 0

Чтобы у вас все заработало

Используемые нами питоновые пакеты хотят свежий питон, в системном 3.4 работают криво.
Соответственно нам нужно поставить себе собственный питон, к счастью, теперь для этого не нужно sudo. Готовьтесь много раз жать **y** или печатать **yes** по ходу действия.

Будем пользоваться менеджером окружений conda:
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
chmod +x Miniconda3-latest-Linux-x86_64.sh
./Miniconda3-latest-Linux-x86_64.sh
./miniconda3/bin/conda init bash

Путь 1 (точно работает, длинный) {restart your bash session - перелогиньтесь\перезапустите сессию bash}

conda create --name snakes python=3.6
{snakes - имя создаваемого окружения}

conda activate snakes
{теперь мы внутри окружения - можно ставить свой софт}

pip install svist4get
pip install cutadapt
pip install --upgrade numpy pysam cython
{это были зависимости для plastid}
pip install plastid

conda install -c **conda**-forge **imagemagick** **{эта команда была забыта в прошлой версии слайдов}**

Этап 0

Чтобы у вас все заработало

Путь 2 - короткий. Не тестировался - но вроде должен сработать.

Клонировать готовое окружение:
conda env create -f ~/../kulakovskiy/conda_snakes.yml

Активировать окружение
conda activate snakes

В случае успеха - у вас внутри окружения (которое можно деактивировать **conda deactivate**)
будут доступны и не будут вылетать с ошибкой команды **cutadapt** и **svist4get**.

Этап 1

Тащим данные из GEO+SRA по нужному идентификатору.

Руководство к действию: вводная часть мануала, там есть ответы на 95% вопросов. В мануале можно вешать комментарии с вопросами. Также вопросы можно писать на ivan.kulakovskiy+hse@gmail.com (и туда же присылать отчеты о самостоятельной работе).

Что нужно сделать: найти номер PRJNA (проекта в SRA) на странице в GEO; дернуть метаданные с помощью edirect, повтыкать в json/tsv-файлы.
(долго-опционально-не-в-дедлайн) попробовать запустить пред-скачивание prefetch на одном из SRR
(долго-опционально-не-в-дедлайн) попробовать запустить распаковку fastq-dump на одном из SRR
(быстро-вариант-для-ленивых) дернуть пару распакованных файлов fastq из директории **~/../kulakovskiy/fastqraw** и сделать симлинки на полные файлы у себя в папке **~/fastq**

ОБЩЕЕ ЗАМЕЧАНИЕ НА ЭТОТ и следующие этапы: НЕ ПОЛЬЗУЙТЕСЬ скриптами-помощниками, которые упоминаются в мануале (у вас мало данных и датасетов - все можно сделать по одному файлу; кроме того они могут бажить на вашем учебном сервере, т.к. рассчитаны на другую структуру папок). Автоматизация из ван-лайнеров на баше должна работать корректно.

ЦЕЛЬ ЭТАПА: получить хотя бы по 1 FASTQ-файлу для 2 образцов: RNA и Ribo для любого выбранного вами условия (нокдауна целевых белков или контроля). RNA можно будет посравнивать с Ribo на следующих этапах.

ПРЕДЛОЖЕНИЕ: если вы уперлись в дедлайн и все тормозит (= сервер занят другим счетом), **попробуйте насэмплировать себе 5-10 млн. ридов.** С меньшим числом ридов может не хватить статистики для след.этапов.

Этап 1

Тащим данные из GEO+SRA по нужному идентификатору.

Стартовый набор команд:

~/../kulakovskiy/edirect/esearch -db gds -query "GSE<тутномерGSE>[ACCN]" | ~/../kulakovskiy/edirect/efetch -format docsum -mode json > gse_details.json
~/../kulakovskiy/edirect/esearch -db sra -query PRJNA<тутномерPRJNA> | ~/../kulakovskiy/edirect/efetch -format runinfo > srx2srr.csv

Чтобы склеить понятную табличку из этой вермишели:
ruby ~/../kulakovskiy/h/h0_download_prep.rb gse_details.json srx2srr.csv > samples.tsv

Следующие шаги: см. мануал или лениво пропустить; рекомендуется позапускать и посмотреть как тормозит скачивание из GEO и распаковка. Не стоит делать близко к дедлайну.

Тем кто делает заранее и сделал - начисляется бонусы за этот этап. Бонус - возможность не делать\пропустить любой из последних этапов. Доказательство сделанного - сравнение размеров SRA (в результате prefetch) и распакованного FASTQ-файла.

Этап 2

Отрезка адаптеров и QC

Тут никакой магии - FastQC до и после обрезки адаптеров. Адаптеры режем cutadapt. Примеры есть в мануале.
Цель этапа: посмотреть на _распределение длин_ прочтений до и после обрезки. В Рибо- и РНК-секе.
В результате будет выяснено:
(а) какова характерная длина рибосомных футпринтов в рибосеке и
(б) обрабатывали ли РНК-сек по тому же протоколу (с нарезкой нуклеазами и отбором фрагментов по размеру) или нет.

Рекомендуемые параметры **cutadapt** смотрим в мануале. Адаптеры смотрим в
Здесь и далее число задействуемых потоков контролируем, чтобы не мешать другим людям на сервере (если вы там одни - можно занимать 2-4-8). Загрузку смотрим с помощью **htop** или **top**.

Последовательности адаптеров смотрим в GEO.

FastQC лежит в **~/../kulakovskiy/bin/fastqc/FastQC/fastqc** (но вероятно еще доступен и по старому адресу, про который вам рассказывали в лекции про QC). К сожалению на выделенном нам узле нет X-сервера, соответственно запускать GUI на сервере как это описано в мануале не получится. Выгружайте результаты в файлы.

Этап 1-2

Какие данные мне лучше взять?

Почитайте внимательно описания в GEO для конкретных образцов GSM. **Обратите внимание, в описаниях образцов можно найти последовательности адаптеров.**

Обратите двойное внимание: **реплика 2 проклята**, ее обработка потребует дополнительного анализа (про него упоминалось в лекции и рассказывается в мануале). Если хотите немножко страдать: вам понадобится установить seqkit в свою папку (скомпилированные версии доступны на https://bioinf.shenwei.me/seqkit/download/, должны ставиться локально без sudo просто распаковкой архива) и, помимо дедупликации, написать скрипт для выбрасывания баркодов после тримминга и дедупликации:
риды там имеют вид **RRRBBBSSSS...SSSSRRRR**
где
RRR random
BBB barcode
SSSS….SSSS sequence
RRRR random
BBB - константа (не влияет на дедупликацию), RRR - случайные UMI, SSSS...SSSS - целевая последовательность. Все это можно (если приглядеться) рассмотреть в сырых FASTQ.

"Ах да", до этого еще нужно отрезать адаптеры. **У реплики 1 и реплики 2 адаптеры отличаются.**

За решение задачи с **репликой 2** начисляются бонусы (= можно сделать это вместо 1-2 конечных этапов). Ответом является доля прочтений на самом деле являются PCR-дупликатами в случае Ribo и RNA образцов.

Этап 3

Проверяем долю рибосомной РНК

Единственное отличие от мануала - путь к программе bowtie и доступная память\потоки. Можно пожертвовать точностью, убрав ключ **--best**.
Обратите внимание на ключ **-p** (число потоков). Больше 16 на сервере нет, если кто-то кроме вас считает - не занимайте све.

~/bin/bowtie-1.2.3-linux-x86_64/bowtie --sam -p 16 ~/bin/bowtie-1.2.3-linux-x86_64/rRNA_euk/rRNA_euk cutadapt.fastq.gz > /dev/null

Результат этапа - оценка доли рРНК в оттримленных (иначе ничего не закартируется) прочтениях для РНК- и Рибо-сека.

Если вам не удается дождаться результата \ сервер слишком загружен - используйте сэмплированные файлы или сэмплируйте несколько миллионов чтений из уже оттримленных.

Шпаргалка

Частичная, для этапов 1-3

~/../kulakovskiy/edirect/esearch -db gds -query "GSE101865[ACCN]" | ~/../kulakovskiy/edirect/efetch -format docsum -mode json > gse_details.json

~/../kulakovskiy/edirect/esearch -db sra -query PRJNA395723 | ~/../kulakovskiy/edirect/efetch -format runinfo > srx2srr.csv

cutadapt -a AAAAAAAAAAAAA -q 20 --minimum-length 20 -j 15 ./fastq/Ribo_control_rep1_SRR5865792.fastq.gz -o cutadapt.fastq.gz

~/../kulakovskiy/bin/fastqc/FastQC/fastqc -o . cutadapt.fastq.gz

~/../kulakovskiy/bin/bowtie-1.2.3-linux-x86_64/bowtie --best --sam -p 20 ~/../kulakovskiy/bin/bowtie-1.2.3-linux-x86_64/rRNA_euk/rRNA_euk cutadapt.fastq.gz --chunkmbs 10000 > /dev/null

Этап 4

Картирование на реальный геном

Для тех, кто хочет самостоятельно закартировать прочтения, пользуясь опытом прошлых занятий - геном мыши с аннотацией доступен в **~/../kulakovskiy/genomes**
Кому интересно - некоторые подробности откуда и как подготовлена геномная аннотация есть в мануале.

Последовательности хромосом лежат в fasta-файле:
GRCm38.primary_assembly.genome.fa

Индекс, например, для hisat2, придется сделать самостоятельно.

ГОТОВЫЕ КАРТИРОВАНИЯ оттримленных чтений для следующих шагов лежат в **~/../kulakovskiy/bams**

Разумно взять для анализа ту же пару GSM-образцов (RNA+Ribo), которую вы использовали ранее.

Этап 5

Фазирование прочтений и метагенные профили

Этот этап должен у вас получиться по мануалу (см. мануал пункт 5.5.2.1. Оценка положения P-сайта рибосомы в прочтениях различной длины), с важным отличием: мы работаем с мышью, запуск команд plastid нужно проводить в созданном вами с помощью miniconda окружении, готовые срезы геномной аннотации для plastid лежат в **~/../kulakovskiy/genomes/plastidmetagen**

Результат подэтапа - файл с фазированием прочтений различной длины и картинка фазирования (ее сгенерирует plastid). Можно оценить сфазируются ли прочтения РНК-сека.

Следующая задача - построение метагенных профилей (см. мануал пункт 5.5.3.1. Построение спуленных профилей по всем образцам). Здесь вам не нужен сложный ванлайнер bash из мануала - достаточно в **--count_files** указать конкретный bam-файл с картированием, например вот так:

```
metagene count --countfile_format BAM --count_files one.file.bam --fiveprime --min_length 25 --max_length 32 --min_count 10
--use_mean --landmark Start ~/../kulakovskiy/genomes/plastidmetagen/mouse_start_rois.txt metagene_count_test
```

Результат этапа: картинка с метагенными профилями. Сравните РНК-сек и Рибо-сек; или фазированные и нефазированные профили.

Этап 6

Получение bedGraph файлов и визуализация

Задача этого этапа - сделать из картирования bedGraph-файл - либо с покрытием либо с 5' концами либо с фазированными чтениями - P-сайтами рибосом. Любой вариант подойдет, хорошо сделать Рибо-сек и РНК-сек.

```
make_wiggle -o output --count_files input.bam --normalize --min_length 25 --max_length 31 --fiveprime_variable --offset
psite_test_p_offsets.txt
```

Для РНК-сека заменяем **--fiveprime_variable** и **--offset** на **--center** (чтобы получить профили покрытия).

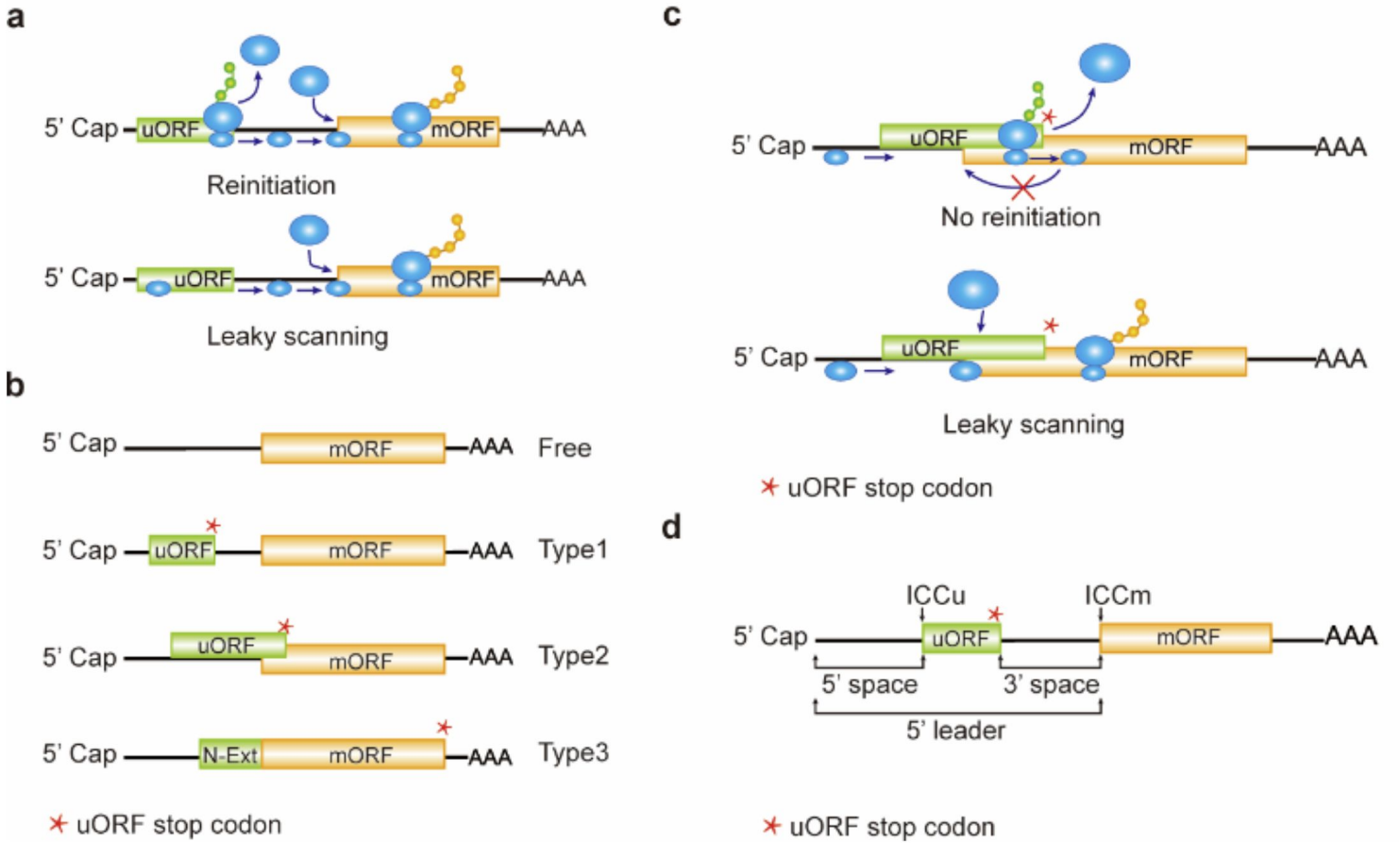
Профили bedGraph можно затем смотреть в геномных броузерах либо визуализировать из командной строки с помощью svist4get.

```
svist4get -gtf ~/genomes/ensembl.mouse.filtered.gtf -fa ~/genomes/GRCm38.primary_assembly.genome.fa -bg
control_ribo_Coots2017_m_r1.bedGraph METTL3_ribo_Coots2017_m_r1.bedGraph -g ENSMUSG00000022160 -hi -rc
```

Пути к gtf-аннотации, fa-файлу с геномной сборкой, bedGraph-файлы - вставьте свои.
Выберите какой-нибудь свой любимый ген (нужно указать его ENSG). Результат подъэтапа - картинка с выбранным геном.
Желающим поиграть с визуализацией документация тут: https://bitbucket.org/artegorov/svist4get/src/master/

```
НЕ ЗАБУДЬТЕ ЧТО ВСЕ КОМАНДЫ ЗАПУСКАЕМ ВНУТРИ ОКРУЖЕНИЯ CONDA (установленные системно plastid и svist4get не работают).
```

About uORF


a
5' Cap
uORF
mORF
AAA
Reinitiation
5' Cap
uORF
mORF
AAA
Leaky scanning
b
5' Cap
mORF
AAA
Free
5' Cap
uORF
mORF
AAA
Type1
5' Cap
uORF
mORF
AAA
Type2
5' Cap
N-Ext
mORF
AAA
Type3
uORF stop codon
c
5' Cap
uORF
mORF
AAA
No reinitiation
5' Cap
uORF
mORF
AAA
Leaky scanning
uORF stop codon
d
ICCu
ICCm
5' Cap
uORF
mORF
AAA
5' space
3' space
5' leader
uORF stop codon