



Big data

Rayan Chikhi

Institut Pasteur

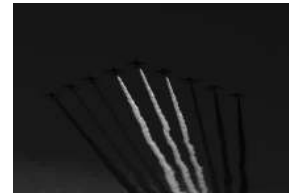
Workshop on Genomics 2024





Hello again again!

- I'm Rayan and I do bioinformatics!

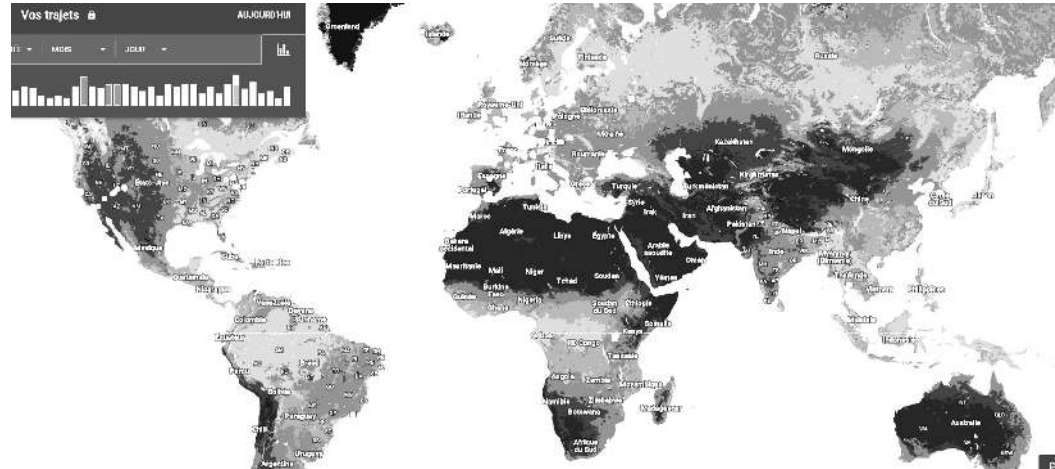
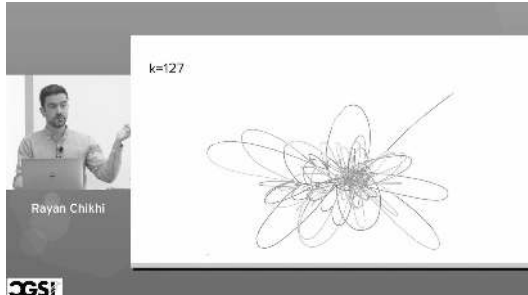


Sequence
Bioinformatics



 @RayanChikhi on Twitter

<http://rayan.chikhi.name>



CGSI

Pi



High expectations from last year - This won't be the greatest big data talk, just a tribute



Part 1: Intro

Founding members of biological big data

Early Eras of Bioinformatics, Representative Leaders

- » Generation -1: E.O. Wilson (compatibility aka perfect-phylogeny - 1965)
- » Generation 0: Margret Dayhoff, Russ Doolittle, Joe Felsenstein
- » Generation 1: Mike Waterman, David Sankoff (Era of algorithms, pre-data)
- » Generation 2: Gene Myers, Russ Altman, Richard Durbin, Sean Eddy

Dayhoff-Eck

- » Worked out the theoretical basis of "shotgun-sequencing" of protein (1970)
- » Published the first "Atlas of protein sequence and structure" (1966) with 65 sequences. Really the first comprehensive database in bioinformatics. Continued with several additional editions.

technologies to support advances in biology and medicine, most notably the creation of protein and nucleic acid databases and tools to interrogate the databases. She originated one of the first [substitution matrices](#), [point accepted mutations \(PAM\)](#). The [one-letter code](#) used for amino acids was developed by her, reflecting an attempt to reduce the size of the data files used to describe amino acid sequences in an era of punch-card computing.

Margaret Oakley Dayhoff



Born

Margaret Belle Oakley
March 11, 1925
[Philadelphia,](#)
[Pennsylvania](#)

Died

February 5, 1983

Big data is the natural flow of biology

1972: single gene sequenced

2000: 1 high-quality human genome

2013: many low-quality human genomes

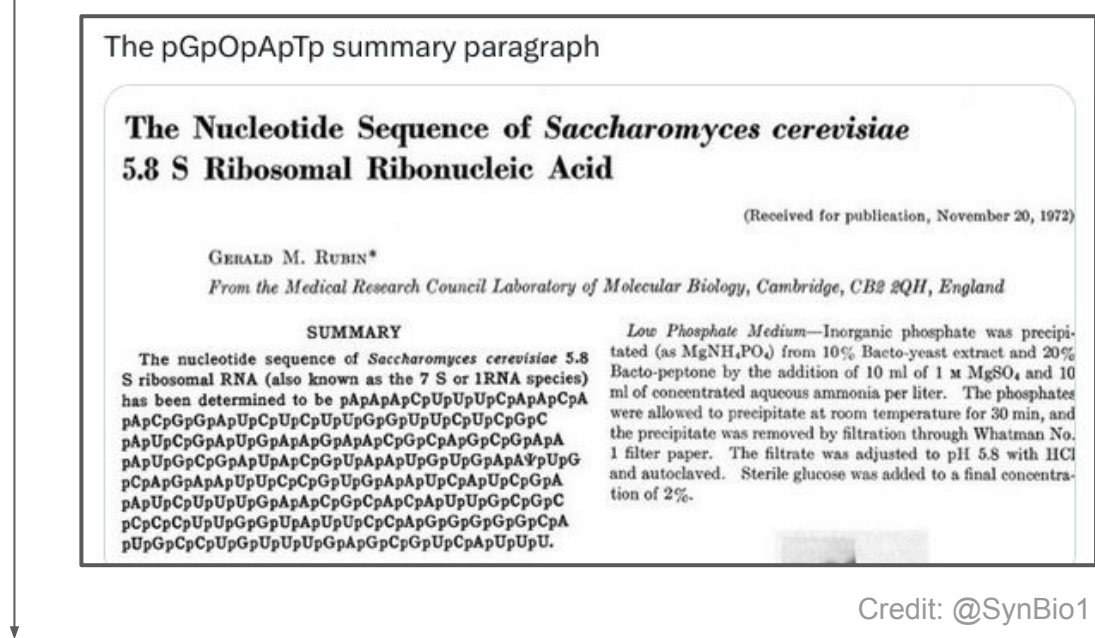
2021: 10 petabases of reads analyzed

2022: 1 million humans VCFs

2022: 50 high-quality human genomes

2024-: ?

Data size



Is big data just a *technical* matter?!

Information technologies scale exponentially

Sydney Brenner and Nathan Myhrvold, ~2005

		Base pairs
1995	Bacterium	2×10^6
2000/3	Mammal	3×10^9
2013	2500 humans	7.5×10^{12}
2021	~1M genomes	3×10^{15}

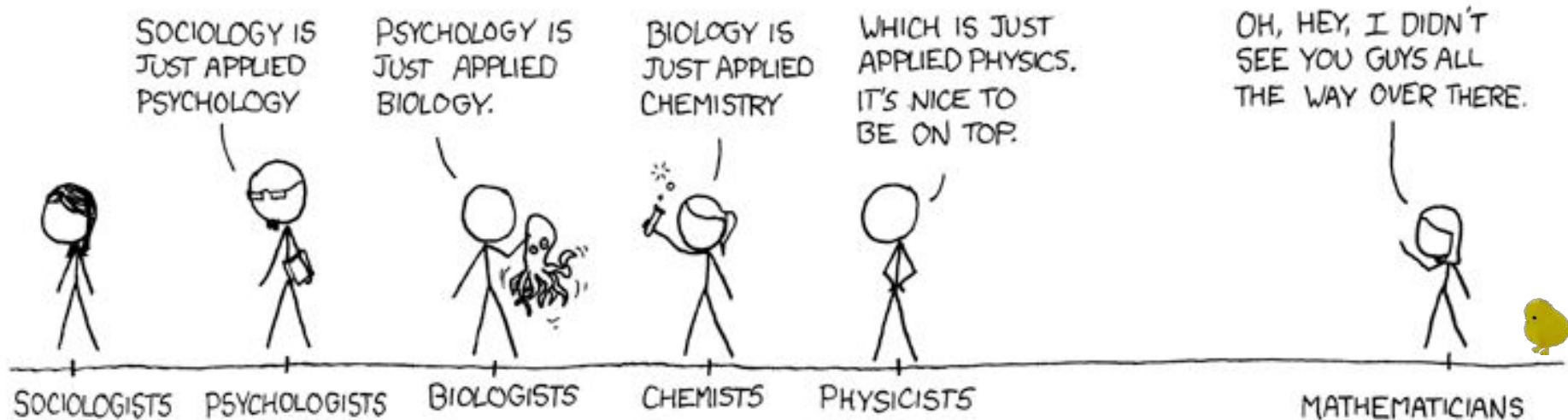
Cost drop from \$1/bp to \$10⁻⁷/bp

- Sustained increase in data at more than 2-fold per year over two decades
- Faster than Moore's law implies continual demand for computational improvements
- Interplay between
 - Analysis and understanding of gene function
 - Improved computational and mathematical methods
 - Evolutionary models

DNA sequence, genomes and computation together
Informatics is to biology what mathematics is to physics ?

*“Informatics is to biology,
what mathematics is to physics”*

Richard Durbin, RECOMB 2023 keynote



Informatics?

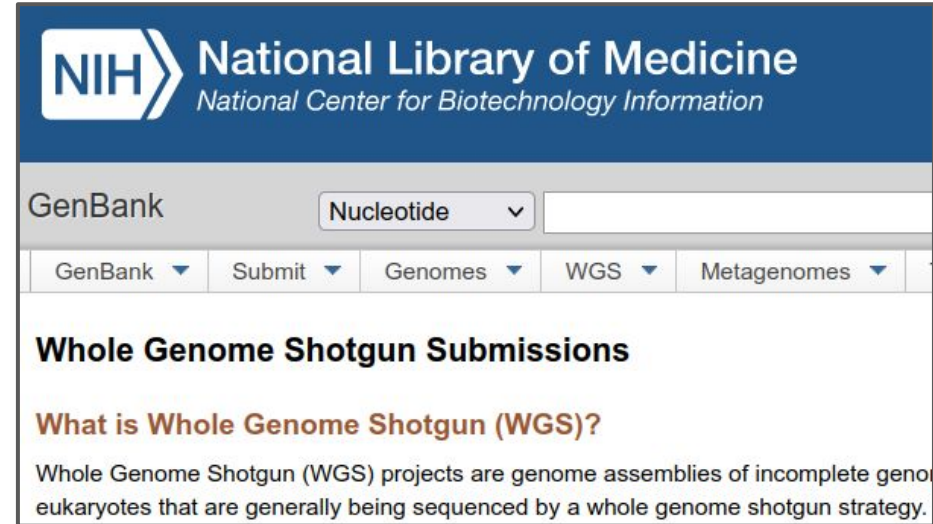
→
“purity”
←
“usefulness”

Big data in biology: NCBI GenBank & WGS



Type: genome assemblies of
>500,000 species
Size: 1.2 terabytes (TB) ([2022](#))

All sequences are *annotated*



Type: genome assemblies
Size: 16 TB ([2022](#))

Unannotated

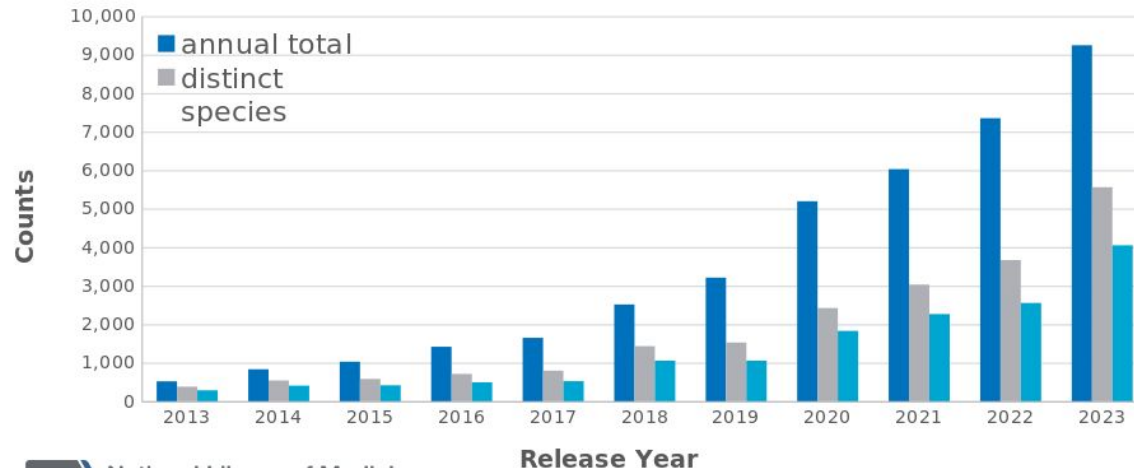
Genome issues for comparative analyses

ALL EUKARYOTIC GENOMES (Cumulative: Dec 2023):

GenBank genomes (all): 36,593 (15,453 species)

GenBank (with annotation): 6,817 (3,801 species)

Annual Growth in Sequenced Species and Genomes



GenBank eukaryotic genome submissions (2021):

- 55% are contaminated
- 80% lack annotation
- 20% have annotation
 - 58% have >50% proteins annotated as "*HYPOTHETICAL*"

NCBI SRA

Type: reads
Size: 50 PB

SRA

SRA

Advanced

Search

Help



SRA

Sequence Read Archive (SRA) makes biological sequence data available to the research community to enhance reproducibility and allow for new discoveries by comparing data sets. The SRA stores raw sequencing data and alignment information from high-throughput sequencing platforms, including Roche 454 GS System®, Illumina Genome Analyzer®, Applied Biosystems SOLiD System®, Helicos Heliscope®, Complete Genomics®, and Pacific Biosciences SMRT®.

Search results

Items: 1 to 20 of 19964

NextSeq 500 paired end sequencing (ERR3407135)

Metadata Analysis (alpha) Reads Download

☐ [NextSeq 500 paire](#)

1. 1 ILLUMINA (Illumina)
Accession: ERX34307

☐ [NextSeq 500 paire](#)

2. 1 ILLUMINA (Illumina)
Accession: ERX34307

☐ [NextSeq 500 paire](#)

3. 1 ILLUMINA (Illumina)
Accession: ERX34307

Filter: Find Filtered Download [What does it do?](#)

[What can the filter be applied to?](#)

< 1 1 346553 >

View: ☒ biological reads ☐ technical reads

Reads (separated)

```
1. ERR3407135.1 ERS3549882
name: NB551234.144:HL523AFXY:1.11101:5421:
member: default
<
>
>gnl|SRA|ERR3407135.1.1 NB551234.144:HL523AFXY:1.11101:5421:1076 F (Biological)
ACCTGAGCGCGCAGCTCCAGTAAATCAAACGCGGCGCGGAATTTGGGATGTTCCATCAGT
TTCCAGGCGCGTTTGCCCTGACGTGCGGACATGCGTAACGAAAGCTGCCAAATATCAGCG
GTAAGCGTGGTAAGCGCTTTTCGGATCGCCA
2. ERR3407135.2 ERS3549882
name: NB551234.144:HL523AFXY:1.11101:2248:
member: default
<
>
>gnl|SRA|ERR3407135.1.2 NB551234.144:HL523AFXY:1.11101:5421:1076 R (Biological)
ATCAACAACAGCGGGAATACCACTCTTCCAGCCGTTGTTTCCAAACCAATACGCGTTAAT
TCACCGAAACCGGACAGCGCAATGGAACGCATCATTCGCGCAGGTGTTCGAGAATACGGA
AAACCGCATCCGAAACGAGATGCGCGTTAAT
3. ERR3407135.3 ERS3549882
name: NB551234.144:HL523AFXY:1.11101:2566:
member: default
<
>
4. ERR3407135.4 ERS3549882
name: NB551234.144:HL523AFXY:1.11101:2119:
member: default
<
>
5. ERR3407135.5 ERS3549882
name: NB551234.144:HL523AFXY:1.11101:2350:
member: default
<
>
```

Units

yotta [Y] $10^{24} = 1\,000\,000\,000\,000\,000\,000\,000\,000$

zetta [Z] $10^{21} = 1\,000\,000\,000\,000\,000\,000\,000\,000$

exa [E] $10^{18} = 1\,000\,000\,000\,000\,000\,000\,000$

peta [P] $10^{15} = 1\,000\,000\,000\,000\,000\,000$

tera [T] $10^{12} = 1\,000\,000\,000\,000\,000$

giga [G] $10^9 = 1\,000\,000\,000$

mega [M] $10^6 = 1\,000\,000$

kilo [k] $10^3 = 1\,000$

hecto [h] $10^2 = 100$

deca [da] $10^1 = 10$

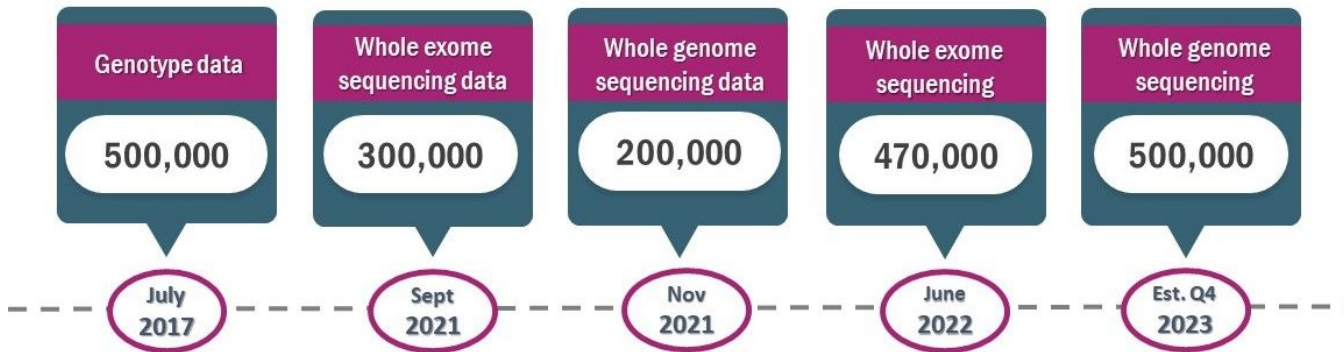
Size: 25+ PB

source:
https://twitter.com/uk_biobank/status/1578023831578427393

Type: reads*

* but many use just the SNPs

Genetic data release timeline



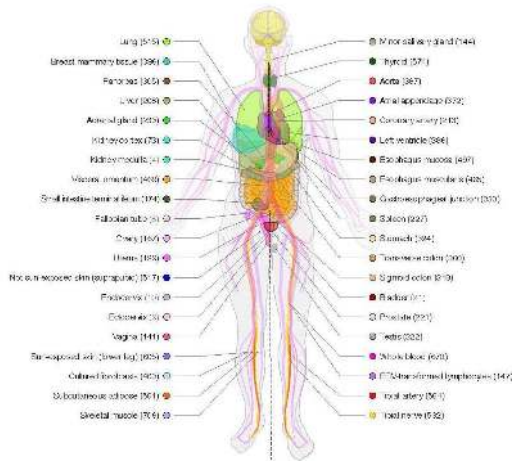
GTE_x

Size: 150 TB

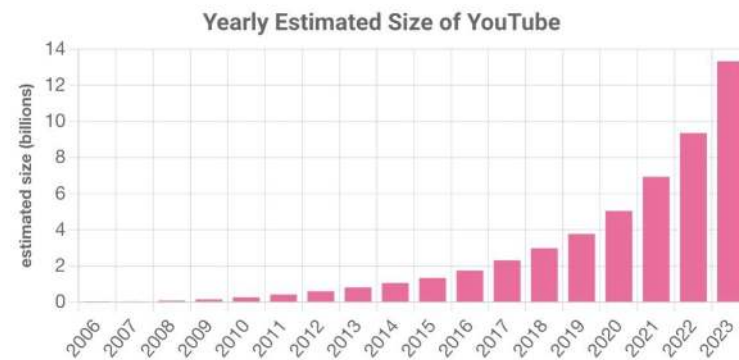
from:
<https://www.genomeweb.com/informatics/anvil-platform-makes-popular-nhgri-qtex-database-free-download>

Type: reads*

- * but many use just the expression data



(Youtube: 300 PB)



Institut Pasteur: 10 PB

Your laptop: 0.001 PB



State of Data Archives (2024):





With big data and big computers, one could perform wonderful, ground-breaking genomics



... But how?

*People at the leading edge
of a rapidly changing field
"live in the future."*

- Paul Buchheit (GMail creator)



“Living in the future” in biology?

- ❑ Have a lab technique only a few know
- ❑ Have data that will only be public later
- ❑ Hold a belief that isn't established yet
- ❑ Discover for the first time that [*some phenomenon*] happens
- ❑ Work on “sci-fi” projects (e.g. create a cell from scratch, genome editing, ..)
 - ❑ ...

“Living in the future” in ~~biology~~ bioinformatics

- ❑ Have a ~~lab~~ *computational* technique only a few know
- ❑ Have data that will only be public later
- ❑ Work on “sci-fi” projects (e.g. quantum computing, AI, big data, ..)

Some people living in the future

- George Church, Craig Venter
- Karen Miga & T2T team*
- Evan Eichler, Erik Garrison
- **All researchers****

* While the rest of the world still used GRCh38/hg19

** Generally ~months ahead, with your papers to be published



The human genome is *finally* complete

The cover of the April 14, 2022 issue of the journal Science. The main headline is "FILLING THE GAPS" with the sub-headline "Closing in on a complete human genome v.1.0". The cover features a circular diagram representing the human genome with various colored segments. Other headlines on the cover include "Earth's last of its kind: a new species of... 100,000 years old" and "PacBio HiFi + ultra-long data (ONT)".

- Introduced nearly **200 Mb** of new sequence (vs the GRCh38)
- Finally resolved with combination of high-coverage long accurate reads (PacBio HiFi) + ultra-long data (ONT)

The logo for the T2T Genome-to-Genome Consortium, featuring the letters "T2T" in a stylized font with a blue and yellow color scheme.A group photograph of the T2T team members, consisting of a large group of people of various ages and ethnicities standing outdoors in a grassy field, many with their arms raised in celebration.

Nanopore Community Meeting 2022 | @NanoporeConf | #NanoporeConf

© 2022 Oxford Nanopore Technologies plc. Oxford Nanopore Technologies products are not intended for use for health assessment or diagnosis. Read, analyze, copy, or report any sequence as evidence.

The Oxford Nanopore Technologies logo, featuring a circular icon and the text "Oxford NANOPORE Technologies".

Part 2: Big Data Toolbox

Computation

- Big computers, Cloud, Cluster
- Storage management
- Galaxy
- Knowledge of scaling limits
- Knowledge of cloud costs
- GNU parallel

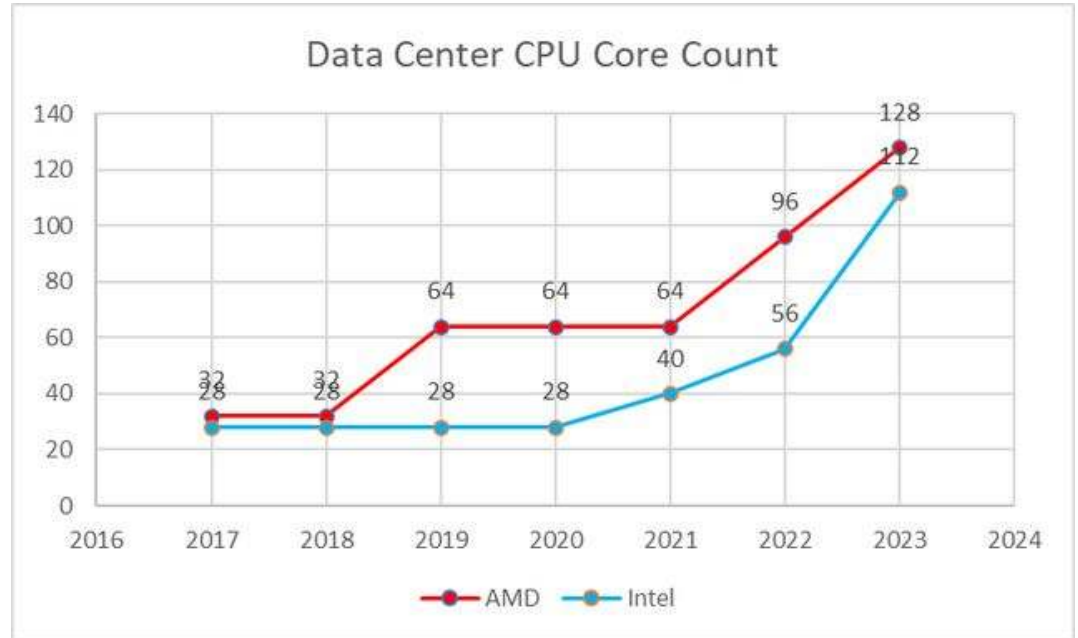
Data mining

- Pebblescout, branchwater
- ORA
- deCOM
- SRA metadata



Future genomics, today?

*No such thing as 'big data',
only 'small computers'*



Cloud

= A collection of computers owned by a single organization and accessible from the Internet



@SR111606871.1 1
length=4250
CCGGGATGTGCTTGGC
TTTCGGCACCATGTAC
CTGGATGCCAAAGAA
ACGGTGCCGTATCCCT
TACCGCTCATGAAGTC
ACGGGGCTGA

```
M:worker_pipeline::114.476*64.47] mapped 24103 sequences
M:worker_pipeline::116.688*64.99] mapped 24151 sequences
M:worker_pipeline::118.523*65.75] mapped 24157 sequences
M:worker_pipeline::119.921*66.67] mapped 24213 sequences
M:worker_pipeline::121.669*67.42] mapped 24135 sequences
M:worker_pipeline::123.241*68.13] mapped 24129 sequences
M:worker_pipeline::124.775*68.95] mapped 24160 sequences
M:worker_pipeline::126.667*69.41] mapped 22755 sequences
M:main] Version: 2.26-r1175
M:main] CMD: minimap2 -xmap-hifi -t192 chm13v2.0.fa m64062.190806.063919.fastq.1
M:main] Real time: 127.332 sec; CPU: 8792.248 sec; Peak RSS: 12.868 GB
8547.99user 244.33system 2:07.42elapsed 6900%CPU (0avgtext+0avgdata 13493472maxresident)k
0inputs+344088outputs (0major+7316184minor)pagefaults 0swaps
6a.48xlarge:~$
[1] 0:time* 1:htop- "ip-172-31-65-227.ec2." 23:39 17-Jul-
```

```
6a.48xlarge:~$ aws s3 cp s3://sra-pub-src-2/SRR11292120/m64062_190806_063919.fastq.1
--no-sign-request
Completed 4.6 GiB/39.1 GiB (278.0 MiB/s) with 1 file(s) remaining
```



Terminal

```

top
Tasks: 45, 263 thr; Load average: 84.83 25.05 8.99; Uptime: 01:29:46
PID    PPID    USER      COMMAND
1      0      root     /usr/sbin/init
2      0      root     [kthreadd]
3      0      root     [ksoftirqd/0]
4      0      root     [kworker/0:0]
5      0      root     [kworker/0:1]
6      0      root     [kworker/0:2]
7      0      root     [kworker/0:3]
8      0      root     [kworker/0:4]
9      0      root     [kworker/0:5]
10     0      root     [kworker/0:6]
11     0      root     [kworker/0:7]
12     0      root     [kworker/0:8]
13     0      root     [kworker/0:9]
14     0      root     [kworker/0:10]
15     0      root     [kworker/0:11]
16     0      root     [kworker/0:12]
17     0      root     [kworker/0:13]
18     0      root     [kworker/0:14]
19     0      root     [kworker/0:15]
20     0      root     [kworker/0:16]
21     0      root     [kworker/0:17]
22     0      root     [kworker/0:18]
23     0      root     [kworker/0:19]
24     0      root     [kworker/0:20]
25     0      root     [kworker/0:21]
26     0      root     [kworker/0:22]
27     0      root     [kworker/0:23]
28     0      root     [kworker/0:24]
29     0      root     [kworker/0:25]
30     0      root     [kworker/0:26]
31     0      root     [kworker/0:27]
32     0      root     [kworker/0:28]
33     0      root     [kworker/0:29]
34     0      root     [kworker/0:30]
35     0      root     [kworker/0:31]
36     0      root     [kworker/0:32]
37     0      root     [kworker/0:33]
38     0      root     [kworker/0:34]
39     0      root     [kworker/0:35]
40     0      root     [kworker/0:36]
41     0      root     [kworker/0:37]
42     0      root     [kworker/0:38]
43     0      root     [kworker/0:39]
44     0      root     [kworker/0:40]
45     0      root     [kworker/0:41]
46     0      root     [kworker/0:42]
47     0      root     [kworker/0:43]
48     0      root     [kworker/0:44]
49     0      root     [kworker/0:45]
50     0      root     [kworker/0:46]
51     0      root     [kworker/0:47]
52     0      root     [kworker/0:48]
53     0      root     [kworker/0:49]
54     0      root     [kworker/0:50]
55     0      root     [kworker/0:51]
56     0      root     [kworker/0:52]
57     0      root     [kworker/0:53]
58     0      root     [kworker/0:54]
59     0      root     [kworker/0:55]
60     0      root     [kworker/0:56]
61     0      root     [kworker/0:57]
62     0      root     [kworker/0:58]
63     0      root     [kworker/0:59]
64     0      root     [kworker/0:60]
65     0      root     [kworker/0:61]
66     0      root     [kworker/0:62]
67     0      root     [kworker/0:63]
68     0      root     [kworker/0:64]
69     0      root     [kworker/0:65]
70     0      root     [kworker/0:66]
71     0      root     [kworker/0:67]
72     0      root     [kworker/0:68]
73     0      root     [kworker/0:69]
74     0      root     [kworker/0:70]
75     0      root     [kworker/0:71]
76     0      root     [kworker/0:72]
77     0      root     [kworker/0:73]
78     0      root     [kworker/0:74]
79     0      root     [kworker/0:75]
80     0      root     [kworker/0:76]
81     0      root     [kworker/0:77]
82     0      root     [kworker/0:78]
83     0      root     [kworker/0:79]
84     0      root     [kworker/0:80]
85     0      root     [kworker/0:81]
86     0      root     [kworker/0:82]
87     0      root     [kworker/0:83]
88     0      root     [kworker/0:84]
89     0      root     [kworker/0:85]
90     0      root     [kworker/0:86]
91     0      root     [kworker/0:87]
92     0      root     [kworker/0:88]
93     0      root     [kworker/0:89]
94     0      root     [kworker/0:90]
95     0      root     [kworker/0:91]
96     0      root     [kworker/0:92]
97     0      root     [kworker/0:93]
98     0      root     [kworker/0:94]
99     0      root     [kworker/0:95]
100    0      root     [kworker/0:96]
101    0      root     [kworker/0:97]
102    0      root     [kworker/0:98]
103    0      root     [kworker/0:99]
104    0      root     [kworker/0:100]
105    0      root     [kworker/0:101]
106    0      root     [kworker/0:102]
107    0      root     [kworker/0:103]
108    0      root     [kworker/0:104]
109    0      root     [kworker/0:105]
110    0      root     [kworker/0:106]
111    0      root     [kworker/0:107]
112    0      root     [kworker/0:108]
113    0      root     [kworker/0:109]
114    0      root     [kworker/0:110]
115    0      root     [kworker/0:111]
116    0      root     [kworker/0:112]
117    0      root     [kworker/0:113]
118    0      root     [kworker/0:114]
119    0      root     [kworker/0:115]
120    0      root     [kworker/0:116]
121    0      root     [kworker/0:117]
122    0      root     [kworker/0:118]
123    0      root     [kworker/0:119]
124    0      root     [kworker/0:120]
125    0      root     [kworker/0:121]
126    0      root     [kworker/0:122]
127    0      root     [kworker/0:123]
128    0      root     [kworker/0:124]
129    0      root     [kworker/0:125]
130    0      root     [kworker/0:126]
131    0      root     [kworker/0:127]
132    0      root     [kworker/0:128]
133    0      root     [kworker/0:129]
134    0      root     [kworker/0:130]
135    0      root     [kworker/0:131]
136    0      root     [kworker/0:132]
137    0      root     [kworker/0:133]
138    0      root     [kworker/0:134]
139    0      root     [kworker/0:135]
140    0      root     [kworker/0:136]
141    0      root     [kworker/0:137]
142    0      root     [kworker/0:138]
143    0      root     [kworker/0:139]
144    0      root     [kworker/0:140]
145    0      root     [kworker/0:141]
146    0      root     [kworker/0:142]
147    0      root     [kworker/0:143]
148    0      root     [kworker/0:144]
149    0      root     [kworker/0:145]
150    0      root     [kworker/0:146]
151    0      root     [kworker/0:147]
152    0      root     [kworker/0:148]
153    0      root     [kworker/0:149]
154    0      root     [kworker/0:150]
155    0      root     [kworker/0:151]
156    0      root     [kworker/0:152]
157    0      root     [kworker/0:153]
158    0      root     [kworker/0:154]
159    0      root     [kworker/0:155]
160    0      root     [kworker/0:156]
161    0      root     [kworker/0:157]
162    0      root     [kworker/0:158]
163    0      root     [kworker/0:159]
164    0      root     [kworker/0:160]
165    0      root     [kworker/0:161]
166    0      root     [kworker/0:162]
167    0      root     [kworker/0:163]
168    0      root     [kworker/0:164]
169    0      root     [kworker/0:165]
170    0      root     [kworker/0:166]
171    0      root     [kworker/0:167]
172    0      root     [kworker/0:168]
173    0      root     [kworker/0:169]
174    0      root     [kworker/0:170]
175    0      root     [kworker/0:171]
176    0      root     [kworker/0:172]
177    0      root     [kworker/0:173]
178    0      root     [kworker/0:174]
179    0      root     [kworker/0:175]
180    0      root     [kworker/0:176]
181    0      root     [kworker/0:177]
182    0      root     [kworker/0:178]
183    0      root     [kworker/0:179]
184    0      root     [kworker/0:180]
185    0      root     [kworker/0:181]
186    0      root     [kworker/0:182]
187    0      root     [kworker/0:183]
188    0      root     [kworker/0:184]
189    0      root     [kworker/0:185]
190    0      root     [kworker/0:186]
191    0      root     [kworker/0:187]
192    0      root     [kworker/0:188]
193    0      root     [kworker/0:189]
194    0      root     [kworker/0:190]
195    0      root     [kworker/0:191]
196    0      root     [kworker/0:192]
197    0      root     [kworker/0:193]
198    0      root     [kworker/0:194]
199    0      root     [kworker/0:195]
200    0      root     [kworker/0:196]
201    0      root     [kworker/0:197]
202    0      root     [kworker/0:198]
203    0      root     [kworker/0:199]
204    0      root     [kworker/0:200]
205    0      root     [kworker/0:201]
206    0      root     [kworker/0:202]
207    0      root     [kworker/0:203]
208    0      root     [kworker/0:204]
2
```

"ip-172-31-65-227.ec2." 23:28 17-Jul-23

Need to
Mapquik

Live: Demo of mapping human
10x coverage HiFi reads using
mapquik in <20 seconds,
including FASTA conversion
using seqkit and chatgpt

Galaxy Project



Data Intensive *analysis* for everyone

- Versatile and reproducible workflows
- **Web** platform
- **Open source** under [Academic Free License](#)

- If you do not have a cluster
- ..or the will to install tools..
- Galaxy offers free computation on pre-installed workflows

Main Galaxy interface

The screenshot displays the Galaxy web interface. At the top, a navigation bar includes links for 'Workflow', 'Visualize', 'Shared Data', 'Help', 'User', and a grid icon. The main content area is divided into three panels. The left panel, titled 'Tools', contains a search bar and a list of tool categories such as 'Get Data', 'Collection Operations', 'GENERAL TEXT TOOLS', 'Text Manipulation', 'Filter and Sort', 'Join, Subtract and Group', 'Datamash', 'GENOMIC FILE MANIPULATION', 'FASTA/FASTQ', 'FASTQ Quality Control', 'SAM/BAM', 'BED', 'VCF/BCF', 'Nanopore', 'Convert Formats', 'Lift-Over', 'COMMON GENOMICS TOOLS', 'Interactive tools', 'Operate on Genomic Intervals', 'Fetch Sequences/Alignments', 'GENOMICS ANALYSIS', and 'Assembly'. The central panel features a large blue banner for the 'James P. Taylor Foundation for Open Science' with a quote from Jim Taylor and a 'Learn More' button. Below the banner is a notification about SARS-CoV-2 data analysis and logos for PennState, Johns Hopkins University, and Oregon Health & Science University. The right panel, titled 'History', shows a search bar and a list of datasets, including 'Galaxy 101 History' with details like '2 shown' and '7.48 MB', and '2: SNPs' and '1: Exons'.

Cluster

Acquire knowledge about it:

- Queues:
 - How many CPUs/RAM per job, what timelimit
 - Can your group access any ✨*special* queue✨
- Storage:
 - Your quota
 - Is “scratch” quota-free? Do files expire?



My scripts:

```
srun -q seqbio -p seqbio --mem 100G -c 10 --pty bash
```

Quickly allocates a terminal on any machine

```
squeue -o "%.18i %.9P %.8j %.8u %.2t %.10M %.6D %R cores:%c mem:%m cmd:%o " | grep seqbio
```

See what machines are currently being used

Storage management

- How to never run out of storage space:
 - Have 2 folders:
 - ~/archive
 - ~/scratch
 - Rules:
 - Archive is backed up, contains command lines and final results
 - Scratch is fast, but may be deleted at any time
 - Keep the list of files for both, somewhere
 - Keep a dummy 100 GB file ready to be deleted
- Data compression
 - BAM => CRAM => delete it
 - FASTQ => gzip => delete it
 - VCF => BCF
 - GFF/GTF => don't annotate

Part 2: Big Data Toolbox

Computation

- Big computers, Cloud, Cluster
- Galaxy
- Storage management
- Knowledge of scaling limits
- Knowledge of cloud costs
- GNU parallel

Data mining

- Pebblescout, branchwater
- ORA
- deCOM
- SRA metadata



Knowledge of scaling limits

In order of difficulty:

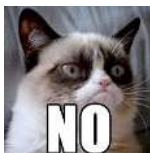
1. **Estimate** how long an analysis will take
2. Reasons **why** some analyses are slower than expected
3. **How** to reduce that time



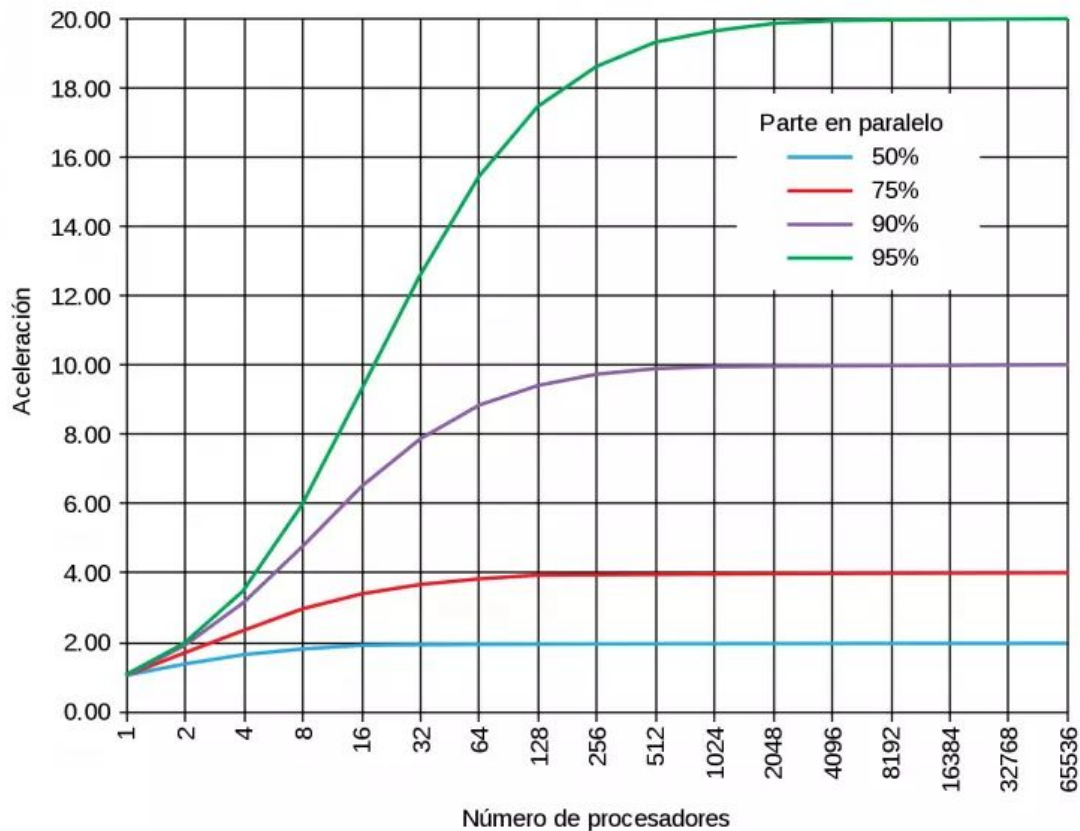
Do 200 CPUs always go 200x faster?



Amdahl's law:



Ley de Amdahl



Connect the dots from left to right

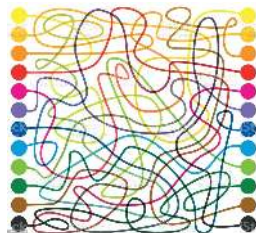
1) Access data from a SSD disk

2) Access data in memory

3) Access <http://www.evomics.org> in Australia

4) Human cell cycle

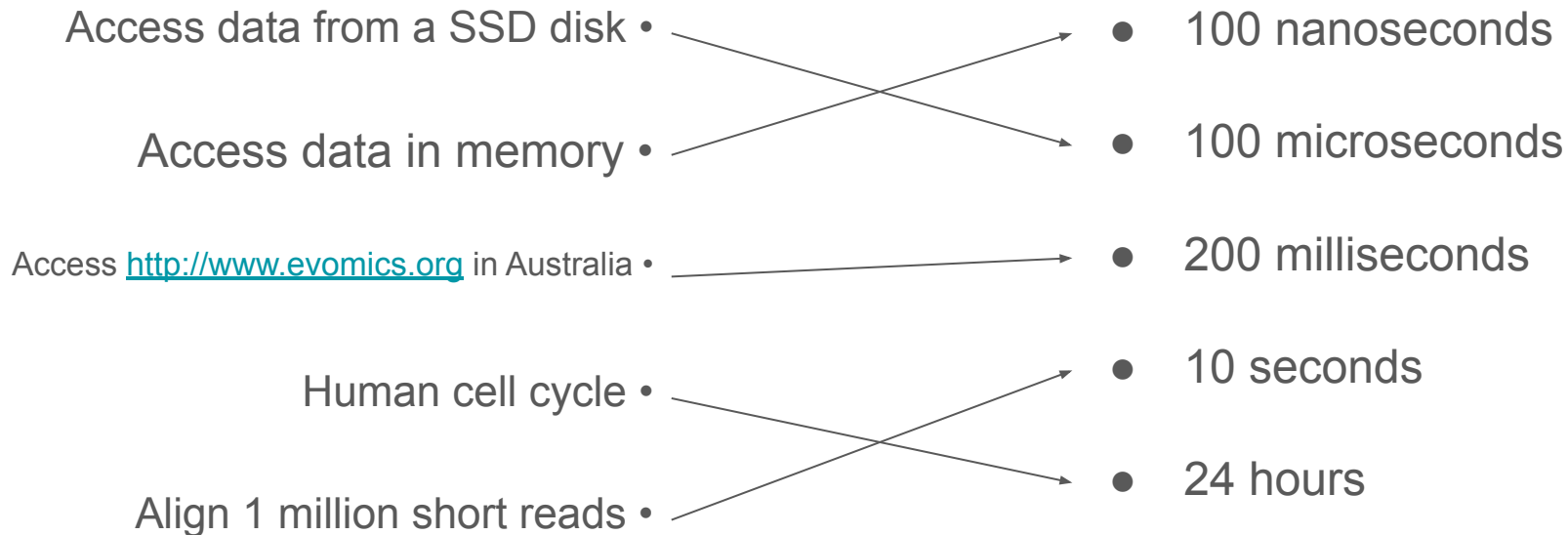
5) Align 1 million short reads



- 100 nanoseconds
- 100 microseconds
- 200 milliseconds
- 10 seconds
- 24 hours

n	nano	10^{-9}
μ	micro	10^{-6}
m	milli	10^{-3}

Connect the dots from left to right



n	nano	10^{-9}
μ	micro	10^{-6}
m	milli	10^{-3}

Knowledge of scaling limits

In order of difficulty:

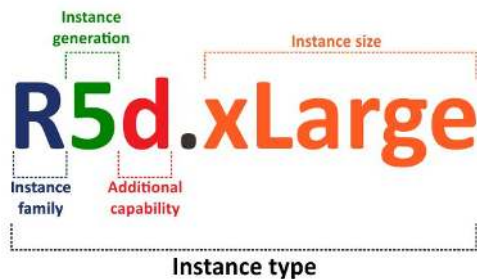
1. **Estimate** how long an analysis will take
 - Look at performance table in tool paper
 - Try on smaller data and extrapolate
2. **Reasons why** some analyses are slower than expected
 - Limited number of CPUs
 - Limited RAM
 - Slow disk (HDD < Cluster network drives < SSD < NVMe)
3. **How** to reduce that time
 - Most analyses go fast enough on a big cloud/cluster and the right tools



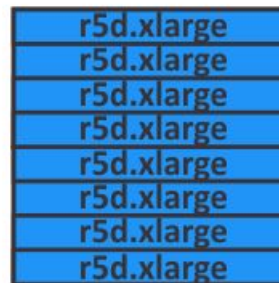
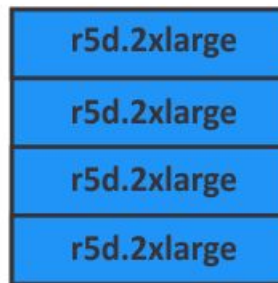
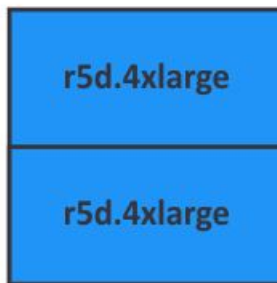
Knowledge of cloud costs

Your workshop instance: `t3a.large` : 2 CPU cores, 8 GB memory
15 cents per hour, 3\$/day

AWS EC2 instance naming



AWS EC2 instance sizes



💖 `c6a.48xlarge` 💖 : 192 cores, 384 GB mem, 7\$/hour

All costs: <https://instances.vantage.sh/>

Knowledge of cloud costs

Storage costs!

EBS (instances hard drive): \$0.08/GB/month

S3 (“Dropbox”): \$0.023/GB/month

- If an instance is stopped: EBS costs occur
- If you create an instance snapshot: EBS costs occur too

How to avoid these costs? Terminate instances, delete snapshots, don't store too much on your S3

General scaling considerations

- Alignment
 - Highly parallel, low memory, scales well with number of CPUs
- Assembly
 - Moderately parallel, high memory, typically requires a single big machine
- Annotation
 - Don't! (jk), but moderately parallel. Single machine too?
- Phylogenomics
 - Can be made parallel (RAxML, Iq-Tree)

GNU parallel



Allows to run the same job on multiple files, simultaneously. Circumvents SLURM.

To count number of lines across many FASTQ files:

```
find . -name *.fastq | parallel -j10 "wc -l {} > {}.nb_lines"
```

To run many jobs defined by CSV data:

```
cat data.csv | parallel --colsep ',' "./myprogram {1} {2}"
```

Part 2: Big Data Toolbox

Computation

- Big computers, Cloud, Cluster
- Storage management
- Galaxy
- Knowledge of scaling limits
- Knowledge of cloud costs
- GNU parallel

Data mining

- Pebblescout, branchwater
- ORA
- deCOM
- SRA metadata



Exploring metagenomes: Pebblescout and Branchwater

- Cutting-edge sequence database search tools
- Think BLAST, but the database is no longer “nr”; it’s all metagenomes.





Pebblescout pre-indexes nucleotide resources and searches them. The index contains at least one 25-mer from every 42-mer for all subjects in the database. Search has three modes: profile, summary, and detailed. Summary search ranks matching subjects using Pebblescout score. Search generates hashes from given user queries using the same scheme as used for indexing. This guarantees that every 42 bp match between the user query and any subject in the database is found.

Seven databases currently available are as follows:

1. **Metagenomic:** All metagenomic and metatranscriptomic runs released in public SRA before the end of 2021
2. **WGS:** All assemblies for the Whole Genome Shotgun sequencing projects available as of Feb 14, 2022
3. **RefSeq:** All assemblies available in the Reference Sequence collection as of April 22, 2022
4. **PH2HS_Runs:** Runs from Phase 3 of the 1000 Genomes project
5. **PH3HS_Biosample:** Runs from Phase 3 of the 1000 Genomes project where all runs for the same BioSample are considered as one subject
6. **Human RNAseq 2021:** All Human RNAseq runs released in public SRA in the year 2021
7. **Virus PacBio HiFi:** Viral samples sequenced with the PacBio SMRT technology defined in [PMC9528980](#)

[Documentation](#) provides additional information. A preprint for the [Pebblescout manuscript](#) is available at [biorxiv](#).

Please provide nucleotide queries, choose database and type of search to be performed, change parameters, as needed, and click View or Download. Please re-click View or Download if you change inputs.

Type FASTA Lines or GenBank Accessions Separated by Commas

Type FASTA lines here (sequence length must be at least 42 bases) or comma separated list or GenBank accessions



or Upload FASTA File

- All metagenomes, all assemblies (WGS), all human RNAseq, RefSeq
- Search for any sequence > 42 nt

Pebblescout usage example



Collaborator needs to search SRA for all samples containing Wolbachia



We did exactly this in our paper!

- 36 host species were known for Wolbachia
 - Found by searching SRA metadata (2,545 runs)
- Pebblescout search for 3 genes (ftsZ, groE, wsp)
 - Found 16 more hosts (35 runs)

Branchwater Metagenome Query

Real-time search for a genome within metagenomes in the SRA.

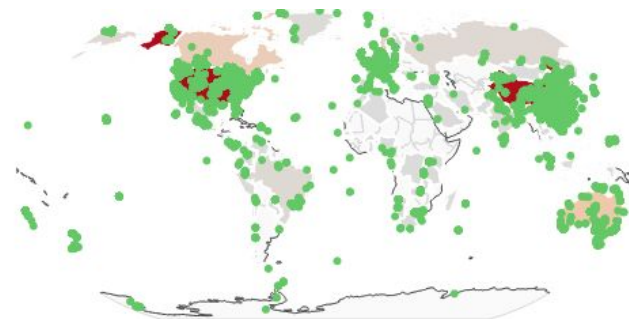
Your query returned 11100 accession IDs. The returned metadata can be pre-filtered prior to .CSV download and plotting with the table below. Your filtered table contains 11100 accession IDs

Download CSV

acc	assay_type	bioproject	biosample_link	cANI	collection_date_...	containment	geo_loc_name_c...	lat_lon	organism
				Min Max		Min Max			
SRR14986175	WGA	PRJNA742226	https://www.ncbi.nlm.nih.gov/sra/SRR14986175	0.9	2017-06-14	0.12	Germany	49.61,10.28	soil metagenome
SRR6958475	WGS	PRJNA444974	https://www.ncbi.nlm.nih.gov/sra/SRR6958475	0.95	2012-05-01	0.37	USA	33.5944,-109.1397	soil metagenome
SRR3501856	WGS	PRJNA320780	https://www.ncbi.nlm.nih.gov/sra/SRR3501856	0.9	2015-07-03	0.11	Singapore	1.33,103.75	activated sludge met...
SRR8925775	WGS	PRJNA681092	https://www.ncbi.nlm.nih.gov/sra/SRR8925775	0.9	2017-10-23	0.12	China	36.19,111.59	bioreactor metagen...

Compared to Pebblescout:

- Only support long queries (> 10 kbp)
- More verbose output/visualizations





kmindex and ORA: indexing and real-time user-friendly queries in terabyte-sized complex genomic datasets

Lemane et al,
2023 (BioRxiv)
2024 (Nat Comp Biol)

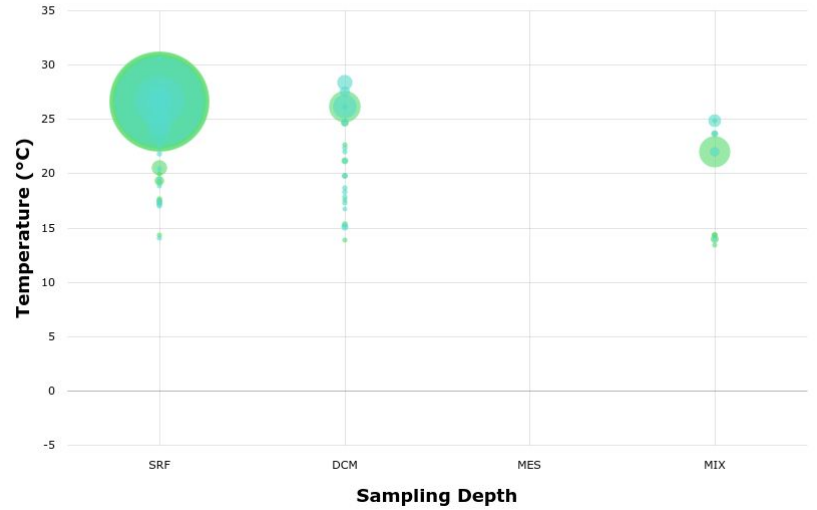
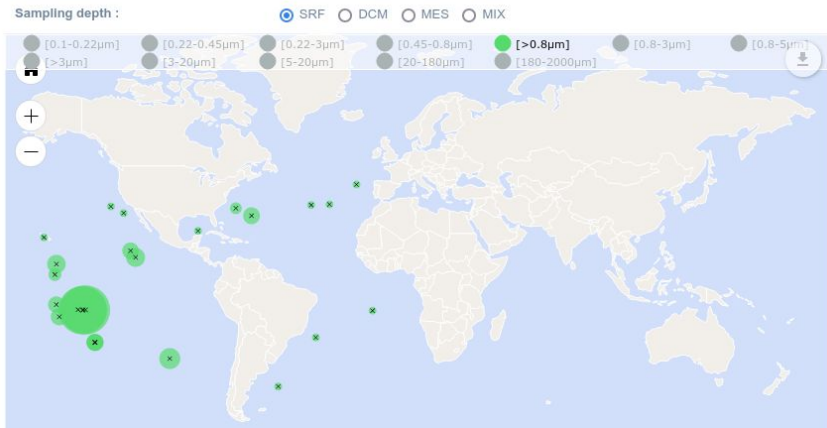
All TARA data,
Supports short queries,
Instant results

Dataset: TARA ?
Job title: nifH_gene_example ?

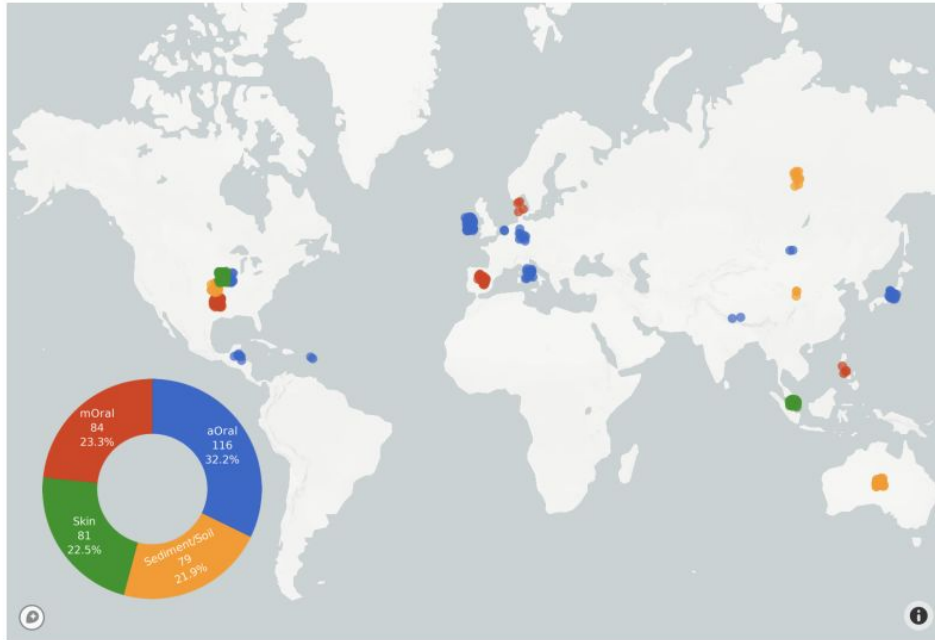
Query sequence:
>nifH_gene LT907975.1:3538795..3539625 [Pseudodesulfovibrio profundus]
atgagaaaagtagcaattacggaaggaacattaaaaatccaccaccactcaaac
actgtcgccggttggcggaaatggccgc
gccgactccaccgcctgtgctcggtgtct
cgtgaagagggcgaggatgtgaactcga

- [0.1-0.22µm] [0.22-0.45µm] [0.45-0.8µm] [0.8-3µm] [0.8-5µm] [0.8-10µm] [10-20µm] [20-180µm] [180-2000µm]

Geographic distribution of k-mer ratios ?



deCOM: integrating all ancient oral metagenomes



We gathered a collection of 360 samples (including contaminants and non contaminants) and obtained a k-mer matrix



	S1	S2	S3	S4	S5	S6	S7	S8
AATCG	1	0	0	0	0	1	1	0
GGGCT	0	0	0	0	0	1	1	0
TTCGA	0	0	1	1	0	1	1	0
AAACG	0	0	0	0	0	1	1	0
GGGCT	0	0	0	1	1	0	0	1
AATTT	0	0	0	0	0	0	0	0
ATCCC	0	0	1	0	0	0	0	0
GGGGT	1	1	1	1	1	1	1	1

New Results

 [Follow this preprint](#)

deCOM: Similarity-based microbial source tracking of ancient oral samples using k-mer-based methods

Camila Duitama González, Riccardo Vicedomini, Téó Lemane, Nicolas Rascovan, Hugues Richard,  Rayan Chikhi

doi: <https://doi.org/10.1101/2023.01.26.525439>

This article is a preprint and has not been certified by peer review [what does this mean?].

SRA metadata

.. will be presented in the next part

Wrapping up of Part 2: Big Data Toolbox

Computation

- Big computers, Cloud, Cluster
- Galaxy
- Storage management
- Knowledge of scaling limits
- Knowledge of cloud costs
- GNU parallel

Data mining

- Pebblescout, branchwater
- ORA
- deCOM
- SRA metadata





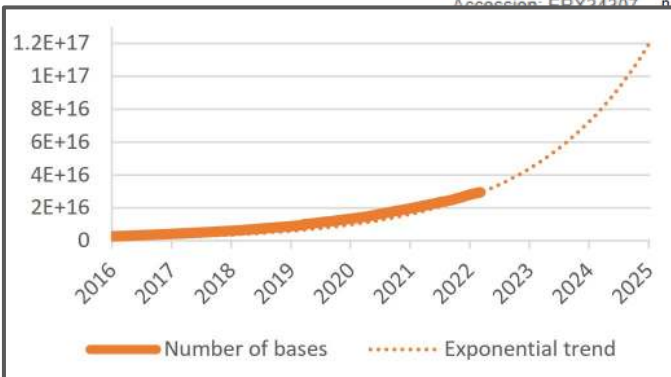
Part 3

SRA-scale sequence exploration

NCBI SRA

All public
sequencing reads

Size: 47 PB
as of late 2023



SRA

SRA

Sequence Read Archive (SRA) makes biological sequence data available to the research community to enhance reproducibility and allow for new discoveries by comparing data sets. The SRA stores raw sequencing data and alignment information from high-throughput sequencing platforms, including Roche 454 GS System®, Illumina Genome Analyzer®, Applied Biosystems SOLID System®, Helicos Heliscope®, Complete Genomics®, and Pacific Biosciences SMRT®.

Search results

Items: 1 to 20 of 19964

NextSeq 500 paired end sequencing (ERR3407135)

Filter: [What does it do?](#)

☐ [NextSeq 500 paire](#)
1. 1 ILLUMINA (Illumina)
Accession: ERX34307

☐ [NextSeq 500 paire](#)
2. 1 ILLUMINA (Illumina)
Accession: ERX34307

1. ERR3407135.1 ERS3549882
name: NB551234.144:HL523AFX:1.11101:5421:1076 F (Biological)
member: default

ERR3407135.2 ERS3549882
name: NB551234.144:HL523AFX:1.11101:2248:1076 R (Biological)
member: default

ERR3407135.3 ERS3549882
name: NB551234.144:HL523AFX:1.11101:2566:1076 R (Biological)
member: default

ERR3407135.4 ERS3549882
name: NB551234.144:HL523AFX:1.11101:2119:1076 R (Biological)
member: default

ERR3407135.5 ERS3549882
name: NB551234.144:HL523AFX:1.11101:2350:1076 R (Biological)
member: default

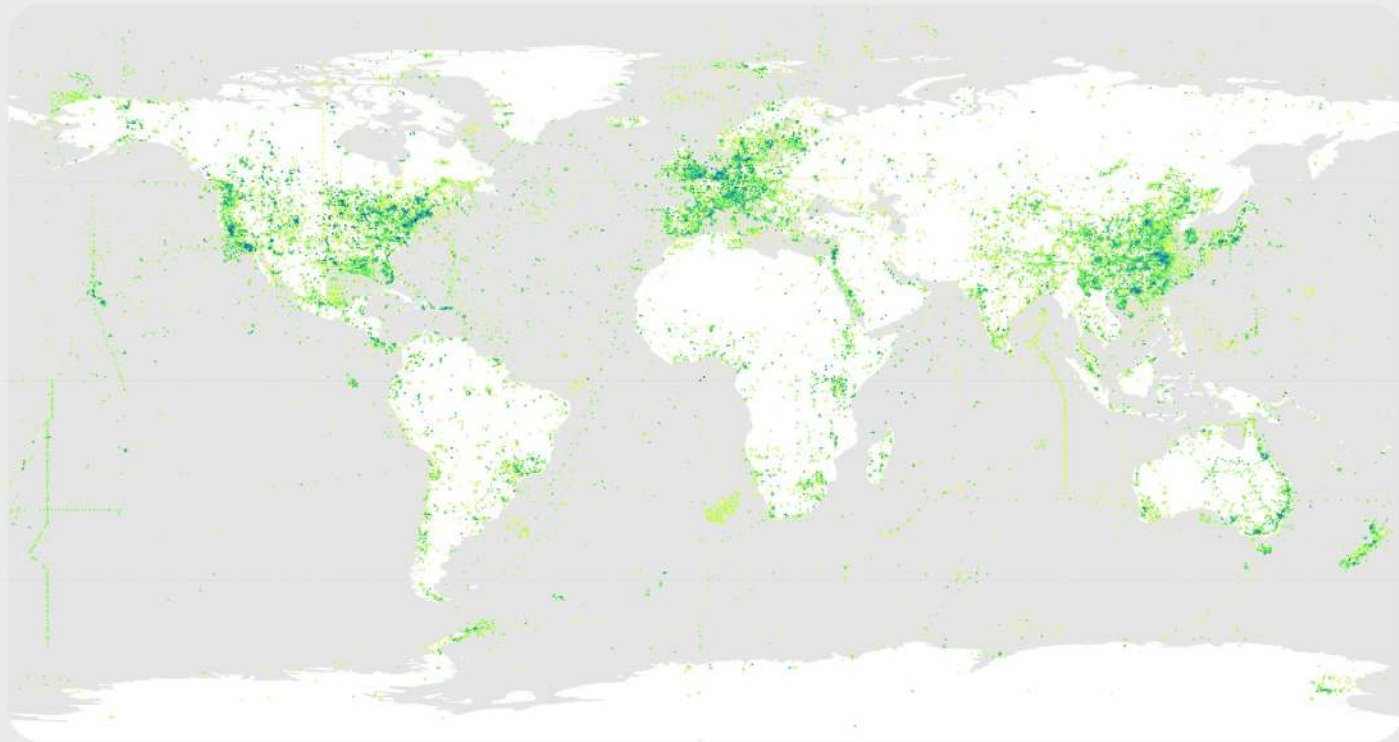
Reads (separated)

```

>gnl|SRA|ERR3407135.1.1 NB551234.144:HL523AFX:1.11101:5421:1076 F (Biological)
ACCTGAGCGCGCAGCTCCAGTAAATCAAACGCGCGCGGAATTGGGATGTTCCATCAGT
TTCCAGGCGCGTTTGCCCTGACGTCGCGACATCGCTAAGTGAAGCTGCCAAATATCAGCG
GTAAAGCGTGGTAAGGCGTTTCGGGATCGCCA
>gnl|SRA|ERR3407135.1.2 NB551234.144:HL523AFX:1.11101:5421:1076 R (Biological)
ATCAACAACAGCGGGAATACCCTCTTCCAGCGCTGTGTTCCAACCAATACGCGTTAAT
TCACCGAAACCGCGACAGCGCAATGGAACGCATCATTTGCGCAGGTGTGTCAGAAATACGGA
AAACCGCATCCGAAACGAGATCGCGCTTAAT

```

Geography of SRA samples



1 20 400 8000
Sequencing density (datasets)

Planetary DNA/RNA sequencing



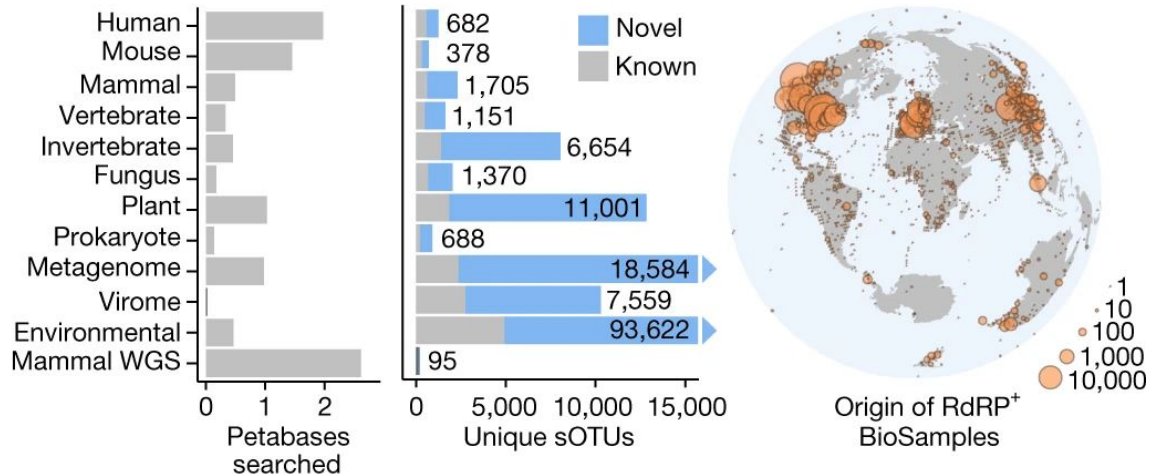
Credit: A. Babian

What to do with the entire SRA?

Serratus: all public RNA-seqs analyzed for viral discovery



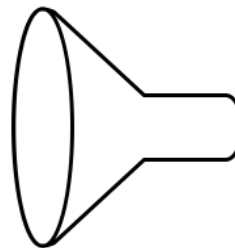
Discovery of 130,000 new RNA viral species. One-off analysis, 20,000 CPUs (Nature, 2022)





All RNA-seqs
pre-2020
(10 petabases)

**Serratus download &
align (bowtie2) to all
viral reference
genomes**

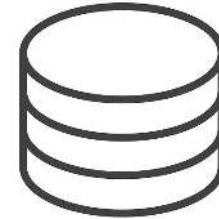
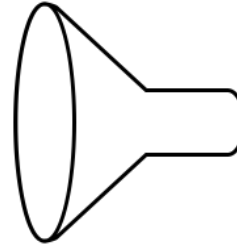


56,000 CoV+ samples
including 9 novel
coronavirus species
discovered



All RNA-seqs
pre-2020

**Serratus download &
sensitive align
(DIAMOND2)
to all known versions of
RNA virus universal gene**



**aligned reads
(.bam files)**
130k novel species
discovered

Toolbox used in Serratus

Part 2: Big Data Toolbox

Computation

- Big computers, Cloud
- Galaxy
- Knowledge of scaling limits
- Knowledge of cloud costs
- GNU parallel

Data mining

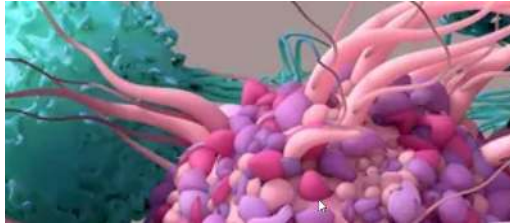
- Pebblescout, branchwater
- ORA
- deCOM
- SRA metadata

Didn't exist



Some follow-ups to Serratus

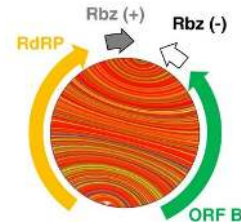
Viral reactivation (Nature 2023)



Discovered HHV-6 reactivation in CAR-T cells

Independent use of Serratus data

Ambiviruses (Nat Comm 2023)



50 known => 20,000 discovered viroids

Analysis of **circular contigs** in Serratus assemblies

Diving into SRA's data

What's SRA metadata?

All this information



[SRX8451857](#): Resequencing of *Vicugna vicugna* V_ss18

1 ILLUMINA (HiSeq X Ten) run: 111.2M spots, 33.4G bases, 11.8Gb downloads

Design: Resequencing

Submitted by: Universidad Austral de Chile

Study: Resequencing of Genomes of South American Camelids

[PRJNA612032](#) • [SRP265528](#) • [All experiments](#) • [All runs](#)

Sample: V_ss18

[SAMN14360346](#) • SRS6753932 • [All experiments](#) • [All runs](#)

Organism: [Vicugna vicugna mensalis](#)

Library:

Name: Vss18

Instrument: HiSeq X Ten

Strategy: WGS

Source: GENOMIC

Selection: RANDOM

Layout: PAIRED

Runs: 1 run, 111.2M spots, 33.4G bases, [11.8Gb](#)

Run	# of Spots	# of Bases	Size	Published
SRR11905265	111,191,160	33.4G	11.8Gb	2020-06-08

Accessing SRA metadata

~~0. NCBI website~~

1. NCBI FTP metadata

<https://trace.ncbi.nlm.nih.gov/Traces/index.html?view=mirroring>

2. SRA metadata on cloud SQL database
(AWS Athena, GCP BigQuery)

```
1 SELECT acc, mbases, mbytes, avgspotlen, librarylayout, instrument
2 FROM sra.metadata as s
3 WHERE consent = 'public' and avgspotlen >= 31
```

SQL Ln 1, Col 1

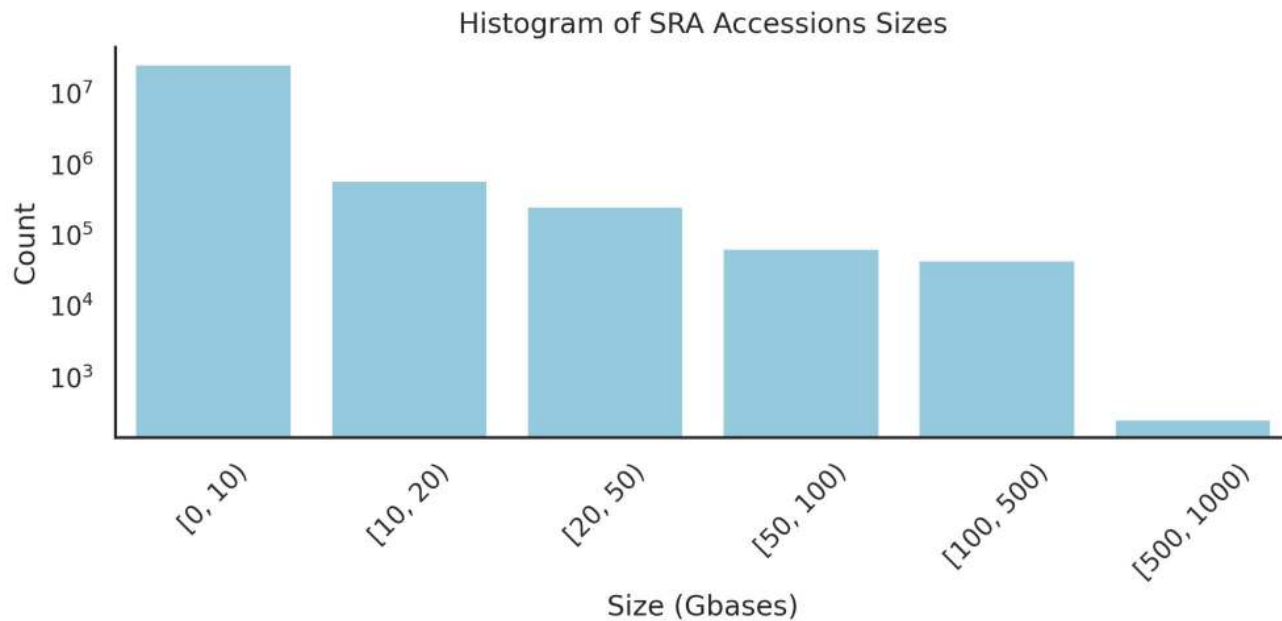
Run Explain [↗] Cancel Clear Create ▼

SRA metadata

[-] tax_analysis	:
— acc	string :
— tax_id	int :
— rank	string :
— name	string :
— total_count	bigint :
— self_count	bigint :
— ilevel	int :
— ileft	int :
— iright	int :

[-] metadata	:	— organism	string :
— acc	string :	— sra_study	string :
— assay_type	string :	— releasedate	date :
— center_name	string :	— bioproject	string :
— consent	string :	— mbytes	int :
— experiment	string :	— loaddate	timestamp :
— sample_name	string :	— avgspotlen	int :
— instrument	string :	— mbases	int :
— librarylayout	string :	— insertsize	int :
— libraryselection	string :	— library_name	string :
— librarysource	string :	— biosamplemodel_sam	array<string> :
— platform	string :	— collection_date_sam	array<string> :
— sample_acc	string :	— geo_loc_name_country_calc	string :
— biosample	string :	— geo_loc_name_country_continent_calc	:

SRA accessions sizes (2023)





Jonathan Jacobs    
@bioinformmer

Wondering: how many individual sequencing reads are in SRA?

@chris_osulliva is there an estimate of this someplace? 🤖

7:46 PM · Nov 10, 2023

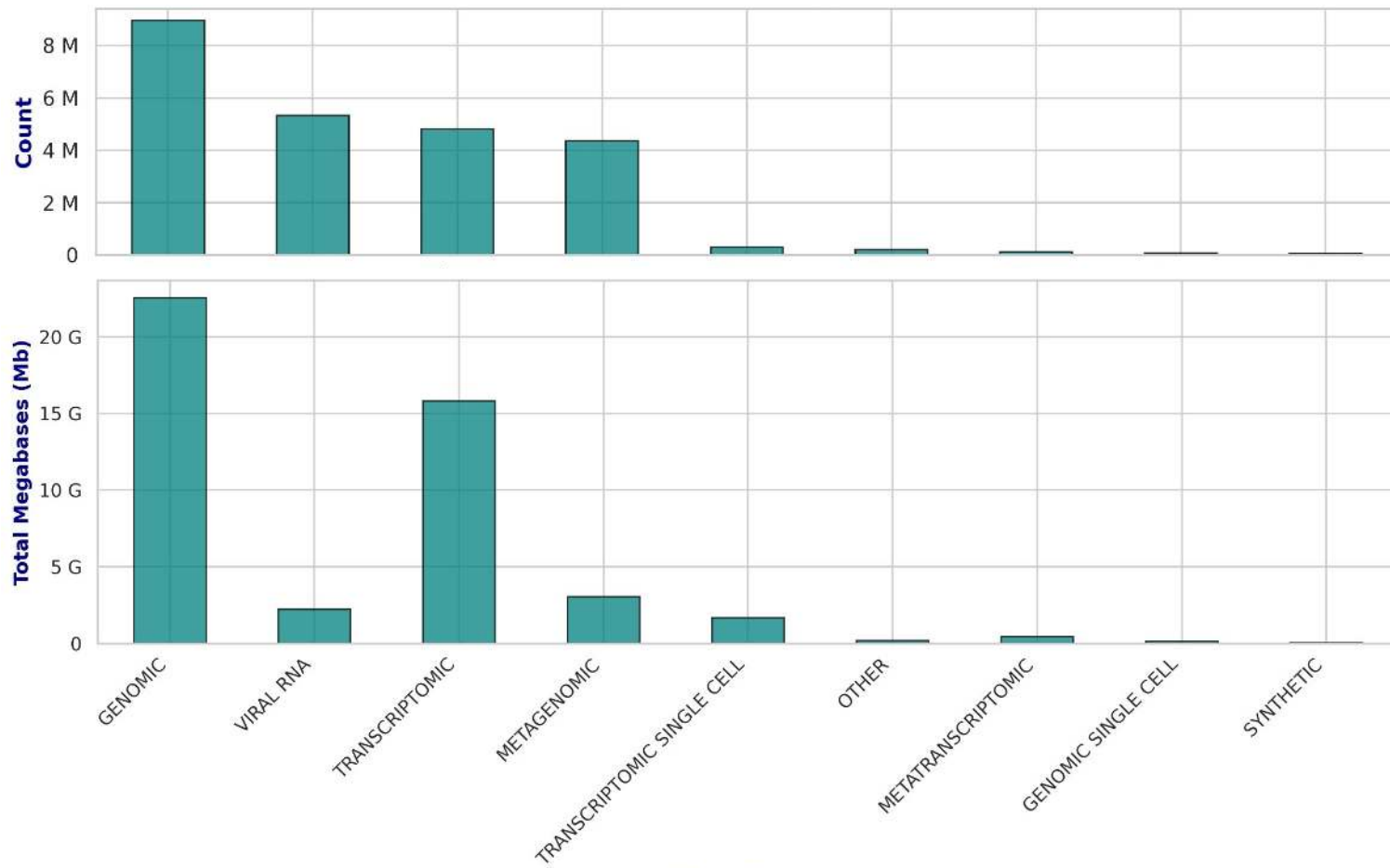


Rayan Chikhi @RayanChikhi · 3m
About 387 trillion as of today.

..

```
SELECT SUM(  
    CASE  
        WHEN avgspotlen > 0 THEN (CAST(mbases AS BIGINT) *1000000) / avgspotlen  
        ELSE 0  
    END  
    ) AS total_number_of_reads  
FROM metadata;
```

SRA accessions types (2023)



SRA taxonomy analysis

Method | Open Access | Published: 20 September 2021

STAT: a fast, scalable, MinHash-based k -mer tool to assess Sequence Read Archive next-generation sequence submissions

Kenneth S. Katz , Oleg Shutov, Richard Lapoint, Michael Kimelman, J. Rodney Brister & Christopher O'Sullivan

Genome Biology **22**, Article number: 270 (2021) | [Cite this article](#)

"we have processed more than 27.9 Peta base pairs from runs"

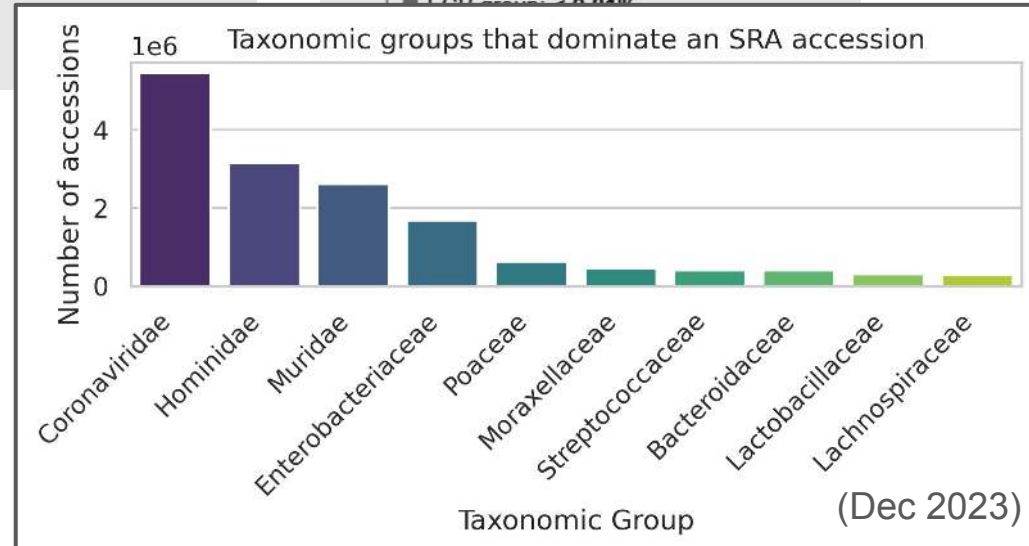
Example STAT output:

Taxonomy Analysis

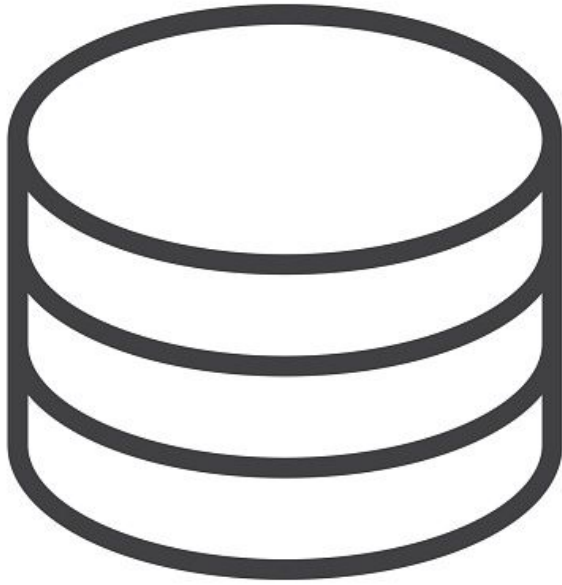
Unidentified reads: 40.04%

Identified reads: 59.96%

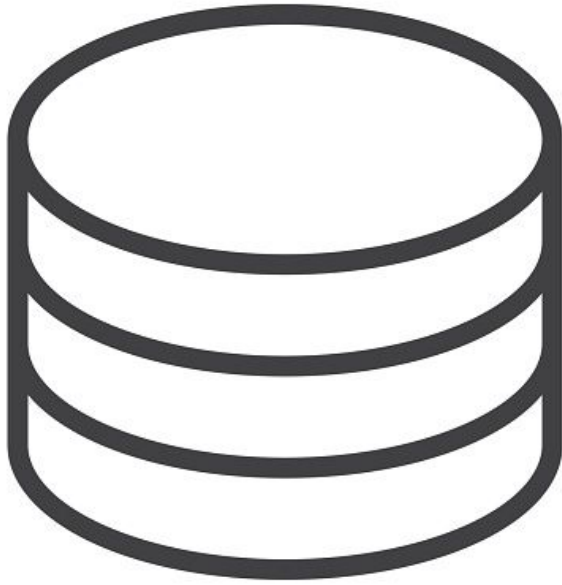
- ▢ Viruses: 50.55%
 - ▢ ssRNA viruses: 50.55%
 - ▢ Measles morbillivirus: 50.55%
 - ▢ dsDNA viruses, no RNA stage: < 0.01%
 - ▢ ssDNA viruses: < 0.01%
 - ▢ Ortervirales: < 0.01%
- ▢ cellular organisms: 9.4%
 - ▢ Bacteria: 6.44%
 - ▢ Proteobacteria: 1.76%
 - ▢ Terrabacteria group: 0.48%
 - ▢ FCB group: 0.01%



How to analyze the entire SRA?



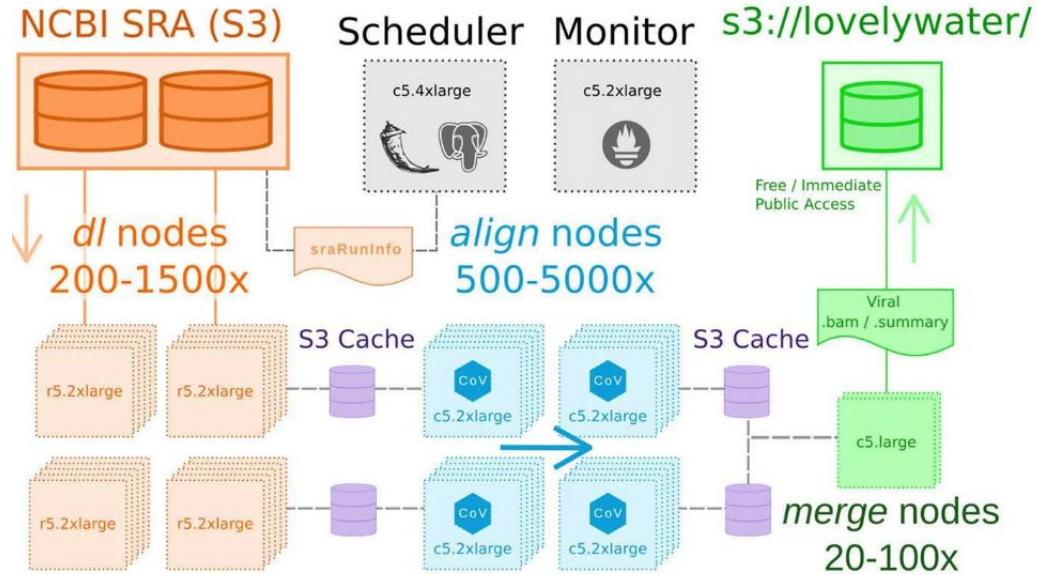
- How much time to download 40 petabytes at 200 MB/sec?



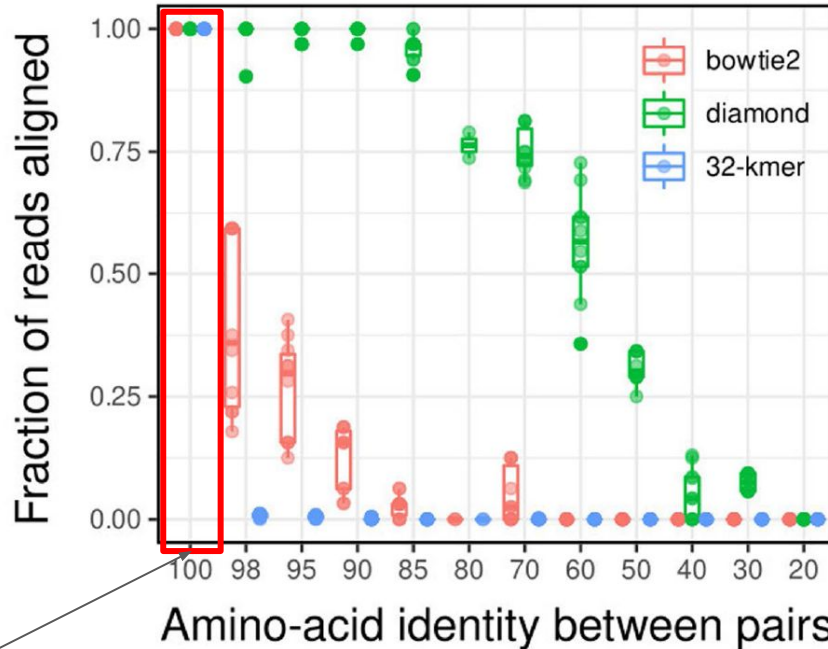
- How much time to download 40 petabytes at 200 MB/sec?

~ 6 years

Serratus infrastructure



Alignment: high **speed** or high **sensitivity**, choose one



Credit: RC Edgar

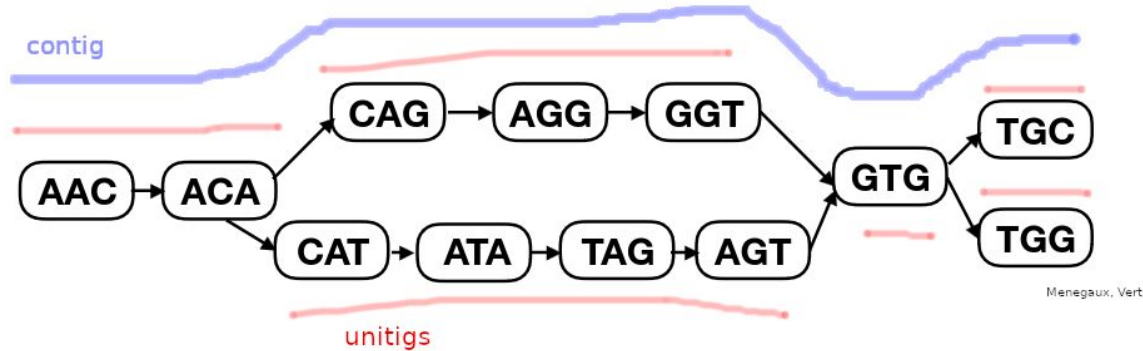
Human reads alignment

SRA-scale alignment

State of the art (ordered by sensitivity/speed):

1. **Sourmash branchwater** (sketches)
 - Metagenomes, long sequences
2. **NCBI Pebblescout** (k-mers, no alignment)
 - Metagenomes, > 42 bp sequences
3. **Bowtie2, STAR** (k-mers, alignment)
 - Serratus1 (all RNAseqs)
 - Recount3 (750k human/mouse RNAseqs)
4. **DIAMOND** (AA-mers)
 - Serratus1.5 (all RNAseqs)
5. **HMMs?** (profile)

An aparté on unitigs



Many dedicated construction methods:

- **BCALM** (2014), **BCALM2** (2016), .., **Cuttlefish2** (2022), **GGCAT** (2023)

Summary

- Exploring all of Life's sequencing data
- Tools:
 - SRA metadata
 - SRA data on cloud
 - Alignment algorithms (fast+sensitive)
 - Short read assembly (fast+lowmem+contiguous)
 - Indexing algorithms (fast+sensitive)

Part 4: pangenomics into the wild



Species:
Gallus gallus

Vocabulary

Kmer:



A “fun” experiment..

Let's study why Kmer has this bright yellow color.



- 1) First, what is the gene responsible for feather color in *Gallus gallus*?
(let's ask Kmer itself)
- 2) Then we'll gather sequencing data from chickens

MC1R

Where does this gene appear in the wild?

First we need to get its sequence, or a chunk of it:

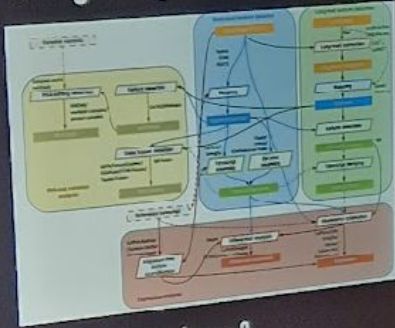
<https://www.ncbi.nlm.nih.gov/gene/427562>

Gathering MC1R genes

I don't have a farm of chicken, but I have big public data



Doing many analyses ? analysis paralysis is common



Which is the right way?

- Just start by get through a single pipeline, start to end
- Then try different approach to assess your first results

Used published data & code, then try additional approaches

"If you don't have data, download it"



Genomics is big

Bonus slide

Searching for MC1R

>MC1R

```
CTTCCCCATACCGCCGCTGAGCCCTCTCTGAGCTCCGGATACGCCGGGCAGTGCCGGTGGGGAGGGCGGCCGAGACAGCGAGTCCCCGCGCTGCTGCCAGAGGGCTCCCGGT  
GGGGGACCGCTTCCCCATCCTTGTGCTGGGGTGCAGAGGTGCCACATCCCTCTGCCTCGTGACCGCGTGCTGCGGGAGCACTGGTGGGGCTGGTTGGGCGCACGGGGGCTTTG  
TAGGTGCTGCAGTTGTGCTCGGGGGCCAGGCCCTCCAGCCAGGGGGTCCCTGGGGGCTGAGGCCGGGGCCATGTCGATGCTGGCCCCCTGCGCCTGCTGCGCGAGCCCTGGAACGC  
CAGTGAGGGCAACCAGAGCAATGCCACGGCCGGGGCCGGAGGTGCCTGGTGCCAGGGGCTGGACATCCCCAATGAGCTCTTCTGACGCTGGGGCTGGTGAGCCTGGTGAGAAC  
TGCTGGTGGTGGCCGCCATCCTCAAGAACAGGAATCTGCACTCGCCACGTACTACTTCATCTGCTGCCTGGCCGTCTCCGACATGCTGGTGAGCGTCAGCAACCTGGCCAAGACGCTC  
TTCATGCTGCTGATGGAGCAGCGCGTGCTGGTGATCCGCGCCAGCATCGTCCGCCACATGGACAATGTCATCGACATGCTCATCTGCAGCTCCGTCTGTCTCCCTCTCCTTCTCGG  
GGTCATCGCGTGGACCGCTACATCACCATCTTCTATGCGCTGCGCTACCAAGCATCATGACGCTGCAGCGCGCGTGGTCACCATGGCCAGCGCTGGCTGGCCAGCACCGTCTCC  
AGCACCGTCTTAATACCTACTACCGCAACAACGCCATCTGCTCTGCTCATTGGCTTCTTCTCTTCATGCTGGTCCTCATGCTGGTGCTCTACATTACATGTTGCGCTGGCGTGCC  
ACCAAGTGGCGAGCATCTCCAGCCAGCAGAAGCAGCCACCATCTACCGCACCAAGCAGCCTGAAGGGAGCCGTACGCTCACCATCCTGCTGGGAGTCTTCTTCATCTGCTGGGGGCC  
CTTCTTCTCCACCTATCCTCATGTCACCTGCCCCACCAACCCCTTCTGCACTGCTTCTCAGCTATTTCAACCTCTTCCATCCTCATCATCTGCAATTCAAGTGGTCGATCCCTGA  
TCTATGCTTCCGGAGCCAGGAGCTCCGCGGAGCTGCGGGAGGTGGTCTGTGCTCCTGGTAGGAGCGGCACAGACAGGAGGATGGATGGATGGATGGACGGATGGACG  
GATGGATGGATGGACAAACAGATGGGTGGATGGACAGATGGGTGGATGGACAGACAGACGACCGCGGGGTGTCCCTGGGTGCCCCAGTGACAGCTGGGGTTGGGCTGCCTGGCCT  
CGCGCTCCCAATAAAGGCTCTTTGCAAGTA
```

Three routes:

- 1) <https://pebblescout.ncbi.nlm.nih.gov/>
- 2) SRA metadata query
- 3) SRA taxonomy query

Pebblescout query

Finds 2000+ hits.

Some of doubtful quality.
“chicken” stops appearing in titles after hit number 1750.

Metagenomic Summary

#	QueryID	SubjectID	RawScore	%coverage	PBSscore	BioSample	Title
1	MC1R	ERR2241657	149.21	100.00	100.00	SAMEA104467160	As part of the EFFORT project, we sampled feces from pig and poultry livestock in nine European countries (BE, BG, DK, FR, ES, GE, NL, PL, SP). More than 9000 animals were sampled, across 181 pig and 178 poultry herds to generate herd-level composite fecal samples. Using shotgun metagenomics, we have quantified and characterized the antimicrobial resistance gene pools (resistomes) in Europe's two most intensively raised livestock species.
2	MC1R	ERR3340873	149.21	100.00	100.00	SAMEA5662822	Trial B 2019
3	MC1R	ERR3340883	149.21	100.00	100.00	SAMEA5662832	Trial B 2019
4	MC1R	ERR4832135	149.21	100.00	100.00	SAMEA7556319	CP562
5	MC1R	ERR4832949	149.21	100.00	100.00	SAMEA7556328	CP562
6	MC1R	ERR4833354	149.21	100.00	100.00	SAMEA7556341	CP562
7	MC1R	SRR12730716	149.21	100.00	100.00	SAMN16282411	Chicken Cecal Metagenome Sequencing
8	MC1R	SRR12730718	149.21	100.00	100.00	SAMN16282445	Chicken Cecal Metagenome Sequencing
9	MC1R	SRR12730726	149.21	100.00	100.00	SAMN16282437	Chicken Cecal Metagenome Sequencing
10	MC1R	SRR12730731	149.21	100.00	100.00	SAMN16282433	Chicken Cecal Metagenome Sequencing

SRA metadata query

<https://www.ncbi.nlm.nih.gov/sra/?term=%22yellow+chicken%22>

SRA Run Selector	Select	Runs	Bytes	Bases	Download	
	Total	324	1.23 Tb	3.43 T	Metadata	or Accession List

Overlap with Pebblescout: 0 🤖

[https://www.ncbi.nlm.nih.gov/sra/SRX4478521\[acn\]](https://www.ncbi.nlm.nih.gov/sra/SRX4478521[acn])

SRX4478521: DNA-seq of Gallus gallus: Wuhua yellow chicken

1 ILLUMINA (HiSeq X Ten) run: 38M spots, 11G bases, 3.9Gb downloads

SRA Athena STAT query

```
SELECT *  
FROM "sra"."tax_analysis"  
WHERE name = 'Gallus gallus' AND total_count > 100
```

In retrospect this is probably way too low, many false hits

Results (317,949)



Contains 72% of the Pebblescout hits

Contains 83% of the “yellow chicken” metadata hits

Getting data from the SRA

TL;DR: state of the art is **prefetch + fasterq-dump**

prefetch: downloads .sra file locally

fasterq-dump: transforms .sra to .fastq or .fasta

Example:

```
prefetch [accession] && fasterq-dump [accession].sra
```

Getting data from the SRA, easily

```
aws s3 cp s3://sra-pub-run-odp/sra/{accession}/{accession} \  
    {accession}.sra \  
    --no-sign-request
```

```
fasterq-dump --fasta-unsorted --stdout {accession}.sra
```

NIH NCBI Sequence Read Archive (SRA) on AWS

[bam](#) [cram](#) [fastq](#) [genetic](#) [genomic](#) [life sciences](#) [STRIDES](#) [transcriptomics](#) [whole exome sequencing](#) [whole genome sequencing](#)

Description

The Sequence Read Archive (SRA), produced by the [National Center for Biotechnology Information \(NCBI\)](#) at the [National Library of Medicine \(NLM\)](#) at the [National Institutes of Health \(NIH\)](#), stores raw DNA sequencing data and alignment information from high-throughput sequencing platforms. The SRA provides open access to these biological sequence data to support the research community's efforts to enhance reproducibility and make new discoveries by comparing data sets. Buckets in this registry contain public SRA data in the original (user submitted) format from select high value and newly-released studies as well as all public-access SRA formatted ETL+BQS data. Also included is all SRA metadata that can be leveraged for attribute-based data discovery.

Update Frequency

Daily

License

Resources on AWS

Description

.bam, .cram, and .fastq files in a public S3 bucket. This is the first of two S3 buckets for source submissions from sequencing methodologies such as PacBio, Oxford Nanopore Technologies, and 10X Genomics.

Resource type

S3 Bucket

Amazon Resource Name (ARN)

```
arn:aws:s3:::sra-pub-src-1
```

AWS Region

```
us-east-1
```

AWS CLI Access (No AWS account required)

```
aws s3 ls --no-sign-request s3://sra-pub-src-1/
```

Big data genomics:)

```
$ cat download_and_map_accession.sh
```

```
set -e
```

```
accession=$1
```

```
aws s3 cp s3://sra-pub-run-odp/sra/$accession/$accession \  
    $accession.sra --no-sign-request
```

```
minimap2 -t20 -x sr mclr.fa <(fasterq-dump --fasta-unsorted $accession.sra) \  
    -o mapping/$accession.minimap2_output
```

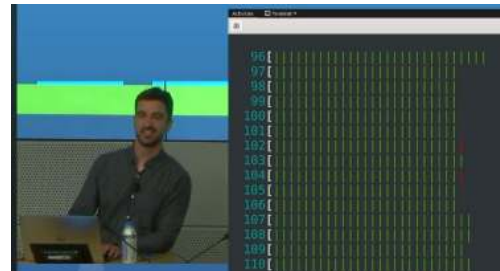
```
rm -f $accession.sra
```

Parallelize processing:

```
cat accessions.txt | parallel -j 10 "./download_and_map_accession.sh {}"
```

Running.. (htop)

On c6a.32xlarge (128 threads, 256 GB mem):



Main	I/O												
PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command		
70507	ec2-user	20		1672M	235M	2544		889.2	0.1	1:14.58	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
71216	ec2-user	20	0	1906M	274M	2524	S	882.6	0.1	0:43.80	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
70747	ec2-user	20		1866M	273M	2264		856.8	0.1	0:56.76	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
70544	ec2-user	20		1842M	242M	2276		856.1	0.1	1:07.39	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
70904	ec2-user	20		1736M	266M	2264		838.2	0.1	0:51.07	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
71073	ec2-user	20		1672M	242M	2524		833.0	0.1	0:46.12	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
73054	ec2-user	20		1842M	249M	2312		815.8	0.1	0:24.28	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
70627	ec2-user	20		1906M	267M	2492		793.3	0.1	0:58.48	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
70656	ec2-user	20		1906M	274M	2544		781.4	0.1	0:58.39	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
71433	ec2-user	20		1736M	266M	2284		773.4	0.1	0:40.49	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63		
71522	ec2-user	20		1900M	826M	4876		303.0	0.3	0:17.61	fasterq-dump --fasta-unsorted --stdout SRR10315070.sra		
70706	ec2-user	20		1900M	829M	4880		300.4	0.3	0:23.55	fasterq-dump --fasta-unsorted --stdout SRR10058581.sra		
70536	ec2-user	20		1900M	829M	4872		298.4	0.3	0:28.64	fasterq-dump --fasta-unsorted --stdout SRR10058584.sra		
71305	ec2-user	20		1900M	826M	4876		298.4	0.3	0:18.30	fasterq-dump --fasta-unsorted --stdout SRR10315066.sra		
70685	ec2-user	20		1900M	829M	4880		296.4	0.3	0:23.57	fasterq-dump --fasta-unsorted --stdout SRR10058582.sra		
70973	ec2-user	20		1900M	827M	4888		296.4	0.3	0:20.93	fasterq-dump --fasta-unsorted --stdout SRR10315069.sra		

But then a bit later..



```
0[25] 8[6] 16[ ] 24[1] 32[6] 40[2] 48[5] 56[ ] 64[6] 72[1] 80[6] 88[3] 96[2]
1[4] 9[0] 17[3] 25[3] 33[41] 41[48] 49[3] 57[ ] 65[1] 73[1] 81[ ] 89[1]
2[1] 10[ ] 18[ ] 26[1] 34[1] 42[9] 50[3] 58[ ] 66[6] 74[1] 82[2] 90[7]
3[1] 11[3] 19[6] 27[1] 35[5] 43[9] 51[7] 59[1] 67[6] 75[3] 83[ ] 91[6]
4[1] 12[8] 20[1] 28[3] 36[1] 44[ ] 52[ ] 60[ ] 68[6] 76[19] 84[ ] 92[6]
5[7] 13[ ] 21[2] 29[3] 37[12] 45[2] 53[3] 61[ ] 69[1] 77[1] 85[ ] 93[3]
6[3] 14[7] 22[11] 30[ ] 38[ ] 46[ ] 54[6] 62[35] 70[ ] 78[1] 86[ ] 94[6]
7[1] 15[7] 23[1] 31[6] 39[0] 47[3] 55[3] 63[2] 71[ ] 79[3] 87[3] 95[7] 103[58] 111[3] 119[1] 127[9]

Mem[|||||]
Swp[|||||]

Tasks: 66, 185 thr ; 15 running
Load average: 24.14 21.70 29.99
Uptime: 02:23:01
```

Main	I/O										
PID	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
802888	ec2-user	20	0	1736M	259M	2508	S	944.7	0.1	2:46.23	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63
802917	ec2-user	20		1900M	828M	4868		305.6	0.3	0:56.21	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802914	ec2-user	20		1736M	259M	2508		73.6	0.1	0:11.68	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63
802915	ec2-user	20		1736M	259M	2508	R	59.0	0.1	0:10.75	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63
802916	ec2-user	20		1736M	259M	2508	R	57.0	0.1	0:11.10	minimap2 -t 20 -a -x sr mclr.fa /dev/fd/63
802919	ec2-user	20		1900M	828M	4868		51.0	0.3	0:08.86	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802924	ec2-user	20		1900M	828M	4868		51.0	0.3	0:08.85	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802922	ec2-user	20		1900M	828M	4868	R	50.4	0.3	0:08.99	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802889	ec2-user	20		5340	1312	1200	R	49.7		0:08.56	samtools view -hF4 -
802923	ec2-user	20		1900M	828M	4868	R	49.7	0.3	0:08.57	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802920	ec2-user	20		1900M	828M	4868		45.1	0.3	0:08.77	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802921	ec2-user	20		1900M	828M	4868	R	38.5	0.3	0:08.55	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
802918	ec2-user	20		1900M	828M	4868	R	14.6	0.3	0:02.51	fasterq-dump --fasta-unsorted --stdout SRR6490215.sra
686759	ec2-user	20		1338M	370M	16052		8.0	0.1	1:55.79	/usr/bin/python3 -s /usr/bin/aws s3 cp s3://sra-pub-run-odp/sra/SRR4897316/SRR4
686408	ec2-user	20		1338M	370M	16052		8.0	0.1	1:55.79	/usr/bin/python3 -s /usr/bin/aws s3 cp s3://sra-pub-run-odp/sra/SRR4897316/SRR4

Remember Part 2..

Knowledge of scaling limits

In order of difficulty:

1. **Estimate** how long an analysis will take
 - Look at performance table in tool paper
 - Try on smaller data and extrapolate
2. Reasons **why** some analyses are slower than expected
 - Limited number of CPUs
 - Limited RAM
 - Slow disk (HDD < Cluster network drives < SSD < NVMe)
3. **How** to reduce that time
 - Most analyses go fast enough on a big cloud/cluster and the right tools



What's happening? see `iostat`

Total DISK READ:			0.00 B/s	Total DISK WRITE:			121.48 M/s
Current DISK READ:			0.00 B/s	Current DISK WRITE:			114.90 M/s
TID	PRI	USER	DISK READ	DISK WRITE	SWAPIN	IO>	COMMAND
618212	be/4	ec2-user	0.00 B/s	12.02 M/s	?unavailable?		python3 -s /usr/bi
618393	be/4	ec2-user	0.00 B/s	11.79 M/s	?unavailable?		python3 -s /usr/bi
687135	be/4	ec2-user	0.00 B/s	12.48 M/s	?unavailable?		python3 -s /usr/bi
688754	be/4	ec2-user	0.00 B/s	12.59 M/s	?unavailable?		python3 -s /usr/bi
778885	be/4	ec2-user	0.00 B/s	11.81 M/s	?unavailable?		python3 -s /usr/bi
791289	be/4	ec2-user	0.00 B/s	12.40 M/s	?unavailable?		python3 -s /usr/bi
798802	be/4	ec2-user	0.00 B/s	11.40 M/s	?unavailable?		python3 -s /usr/bi
802884	be/4	ec2-user	0.00 B/s	12.67 M/s	?unavailable?		python3 -s /usr/bi
807479	be/4	ec2-user	0.00 B/s	11.98 M/s	?unavailable?		python3 -s /usr/bi
809481	be/4	ec2-user	0.00 B/s	12.34 M/s	?unavailable?		python3 -s /usr/bi
1	be/4	root	0.00 B/s	0.00 B/s	?unavailable?		systemd --switched

Disk speed limit around 125 MB/sec

Workaround

-> Use more machines, smaller ones.

setup 10 machines, make a list of different accessions on each, then run:

```
cat ips.txt | parallel -j 10 \  
    ssh ec2-user@{} ./run_previous_parallel_cmd.sh
```

Didn't do that here 😬 (lazy instructor)

Analyzed so far

3 terabases from “yellow chicken” SRA accessions downloaded and mapped to MC1R

```
-rw-r--r--. 1 ec2-user ec2-user 154700 Jan 11 18:22 SRR11521907.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 174639 Jan 11 18:24 SRR11521908.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 150667 Jan 11 18:25 SRR11521909.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 135759 Jan 11 18:25 SRR11521910.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 194411 Jan 11 18:23 SRR11521911.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 149717 Jan 11 18:24 SRR11521912.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 149674 Jan 11 18:25 SRR11521913.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 204873 Jan 11 18:26 SRR11521914.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 180067 Jan 11 18:26 SRR11521915.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 139216 Jan 11 18:26 SRR11521916.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 113860 Jan 11 18:26 SRR11521917.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user 157065 Jan 11 18:27 SRR11521918.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user    6240 Jan 11 18:25 SRR11678145.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user   11665 Jan 11 18:25 SRR11678146.minimap2_output
-rw-r--r--. 1 ec2-user ec2-user   15025 Jan 11 18:25 SRR11678147.minimap2_output
```

Took around 1.5 hours, 6\$/hour

1:36:09elapsed 2026%CPU (0avgtext+0avgdata 1182952maxresident)k

Making mini-assemblies of MC1R

\$ sgpt --model gpt-4 "setup a variant calling script that takes a SAM file as input, mapped to a reference, and outputs a FASTA consensus of the reference"

```
samtools view -S -b $SAM_FILE > $BAM_FILE
```

```
samtools sort $BAM_FILE -o $SORTED_BAM_FILE
```

```
samtools index $SORTED_BAM_FILE
```

```
bcftools mpileup -O b -o $BCF_FILE -f reference.fasta $SORTED_BAM_FILE
```

```
bcftools call --ploidy 1 -m -v -o $VCF_FILE $BCF_FILE
```

```
cat reference.fasta | bcftools consensus $VCF_FILE > $FASTA_FILE
```

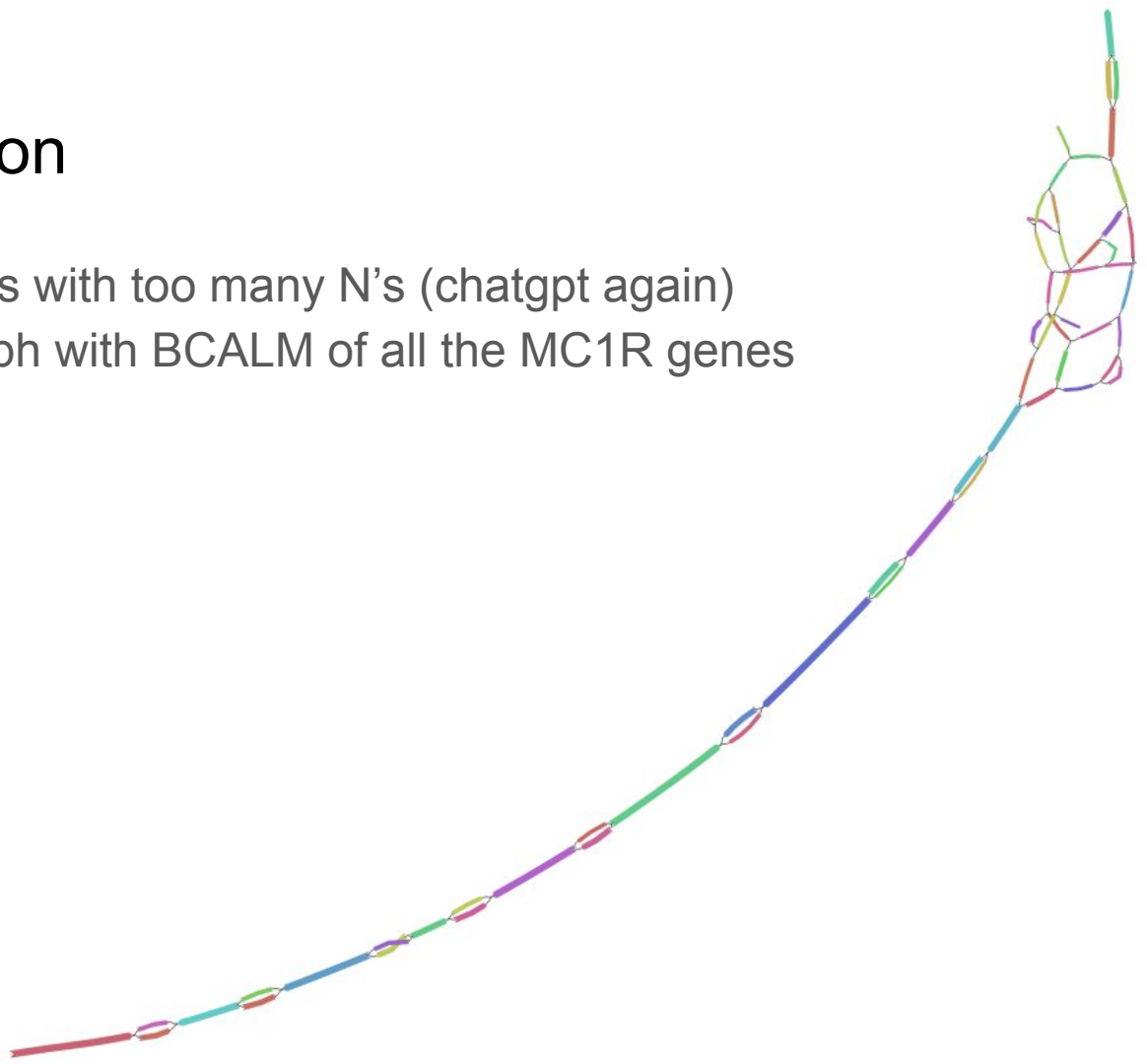
Making mini-assemblies of MC1R

\$ sgpt --model gpt-4 "can bcftools consensus mask regions where it doesn't have confidence in the base call? how to get mask.bed?"

```
samtools depth -a alignment.bam | awk '$3 < 5' | awk '{print  
$1"\t"$2"\t"$2+1}' > mask.bed
```

Analysis of variation

- Filtered consensuses with too many N's (chatgpt again)
- Constructed DB graph with BCALM of all the MC1R genes



So, where's the “yellow chicken” allele?

- Remember the Wuhua yellow chicken accession?
- BLASTed the consensus gene against the pangenome graph

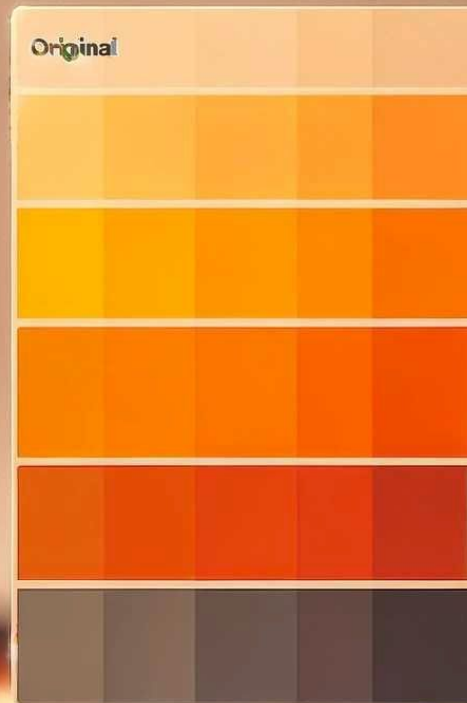


Several hits to low-frequency SNPs, could be any/some of those..



We need more data

Need color metadata



Back to SRA metadata

```
"SRR13606998","WGS","ZHEJIANG ACADEMY OF AGRICULTURAL SCIENCES","public","SRX10001135","ATC  
LLUMINA","SRS8170736","SAMN17734173","Gallus gallus","SRP304191","2022-02-01","PRJNA698651"  
al"],",,,,,,,,[fastq, run.zq, sra"],"[gs, ncbi, s3"],"[gs.US, ncbi.public, s3.us-east-1"],"[{  
es, v=3591984874}, {k=run_file_create_date, v=2021-02-01T21:25:00.000Z}, {k=breed_sam, v=Ti  
umber_sam_s_dpl45, v=Tibetan chicken ex situ in vivo conservation 1}, {k=tissue_sam_ss_dpl1  
y_search, v=698651}, {k=primary_search, v=ATC1}, {k=primary_search, v=ATC1_1.fq.gz}, {k=pri  
7734173}, {k=primary_search, v=SRP304191}, {k=primary_search, v=SRR13606998}, {k=primary_se  
{k=primary_search, v=bp0}]", "{""sex_calc"": ""female"", ""bases"": 11206718290, ""bytes"":  
25:00.000Z"", ""breed_sam"": [""Tibetan chicken""], ""dev_stage_sam"": [""adult""], ""sampl  
vo conservation 1"", ""tissue sam ss dpl145"": [""blood""], ""primary search"": ""17734173"
```

Breed information given for *some* of the chicken. How to extract?

Python script calling chatGPT

```
from openai import OpenAI

def determine_chicken_color(line):

    query = f"Determine the color of the chicken based on the  
following data: {line}. You may reply only: yellow, orange, or  
other. [...]. Do not guess."

    response = client.completions.create(

        model="gpt-3.5-turbo-instruct",

        prompt=query, max_tokens=50)

    return response.choices[0].text.strip()
```

Result of chicken coloring

```
$ tail chicken_color.txt
```

```
SRR2917304,other
```

```
SRR8490109,other
```

```
SRR25338401,Other
```

```
SRR13193600,other
```

```
ERR5036744,yellow
```

```
SRR24605477,other
```

```
SRR12228200,other
```

```
ERR4351384,other
```

```
ERR3505973,other
```

Those chatGPT color predictions..

```
$ tail chicken_color.txt
```

```
[..]
```

```
ERR5036744,yellow
```

```
[..]
```

```
$ grep ERR5036744 *.csv
```

```
[..], {k=common_name_sam, v=chicken}, [..],
```

```
{k=insdc_center_alias_sam, v=QUEEN MARY UNIVERSITY OF LONDON}, [..]
```

No breed information! ChatGPT hallucinated that yellow color.

This was a failed analysis

- This is OK
- Not all analyses are successes
- Move on to the next one
- We learned along the way, right?



Outro

What we've seen today

- Some elements of big data bioinformatics
- Toolbox for Big Data
 - Cloud, parallelism, storage handling, knowledge of limitations
- SRA primer
 - Mining metadata
 - Mining sequences
 - Serratus
- Chicken Pop-Pan
 - Mining 1 gene for 1 species across the SRA
 - Using metadata search and taxonomy search

bigger data

big data



WE'RE-GONNA-NEED



A BIGGER INSTANCE TYPE



Thirdly to **CGSI'23**



❤️ This talk was first dedicated to the
Workshop on Genomics 2023 in Cesky Krumlov:

Guy, Janina, Milos, Kartik, Alena, Madee, Joan, [Mercè](#), and Josie



.. and secondly dedicated to JOBIM'23 organizers..

Fourthly to Workshop on Genomics 2024!





Side note: all microbes can fit onto an SD,
carrier pigeons are faster than Internet



Testing this still on our todo list

K. Brinda



Sequence Bioinformatics

Institut Pasteur
Computational Biology Department



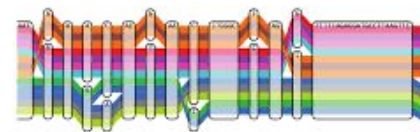
Genomes &
metagenomes
assembly



Algorithms and
data structures
on k-mers



Sequence
search in very
large datasets



Pangenomics

Thank you for your
attention!

