MinION sequencing

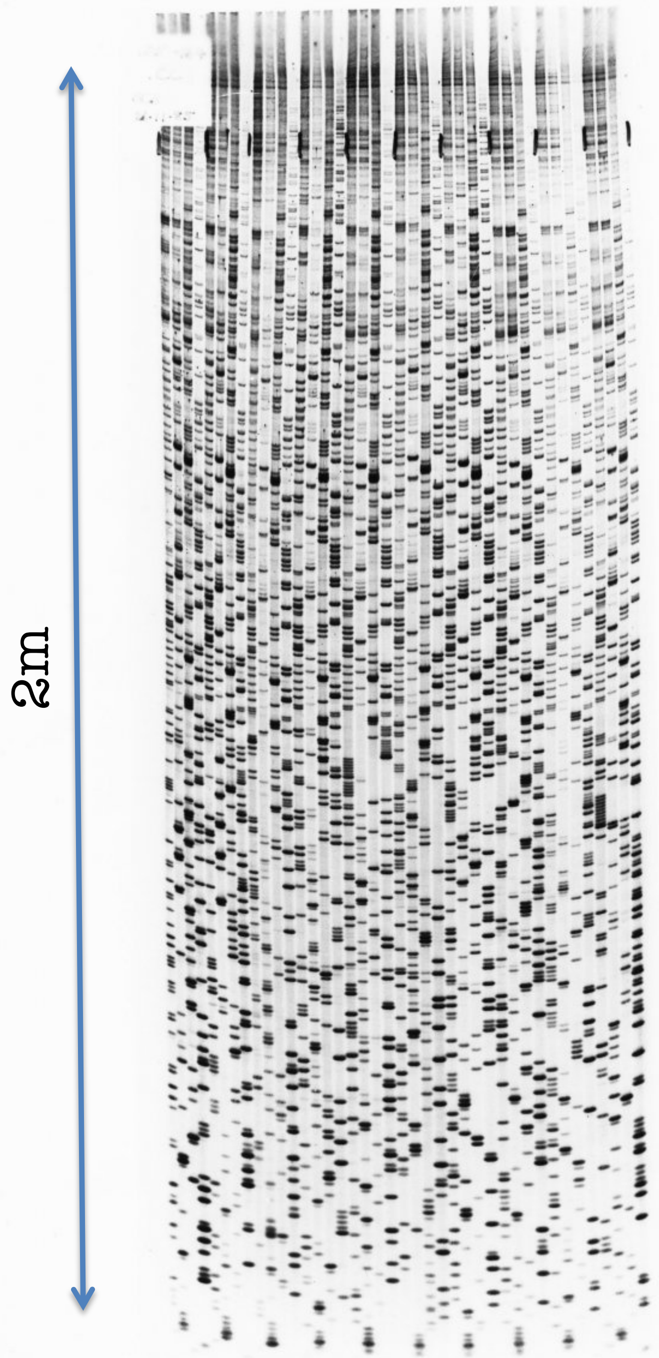
What’s next?

EvoPad Summer School 2019

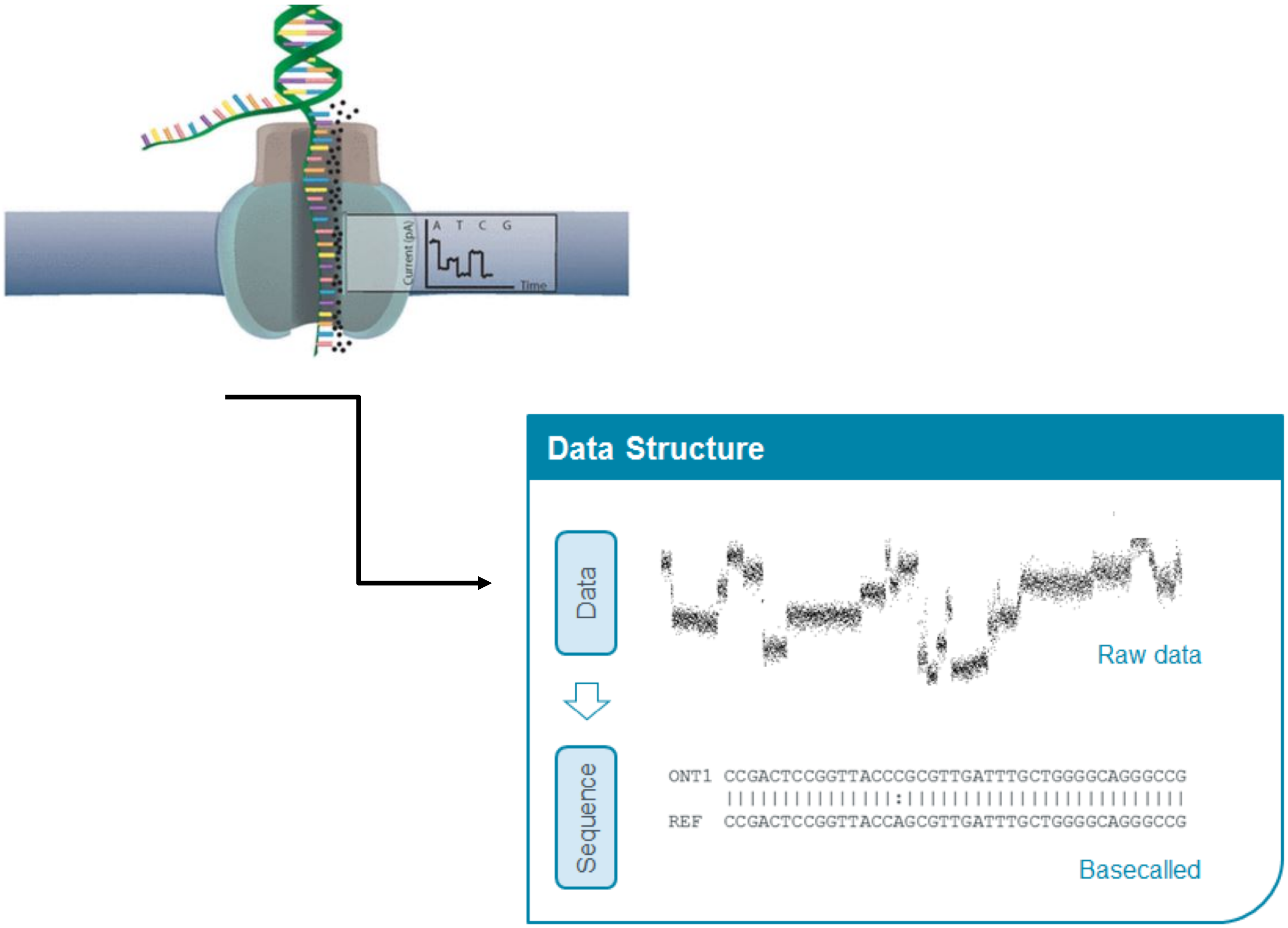
Victoria Shabardina

Elegance of the technical progress ;)

1977

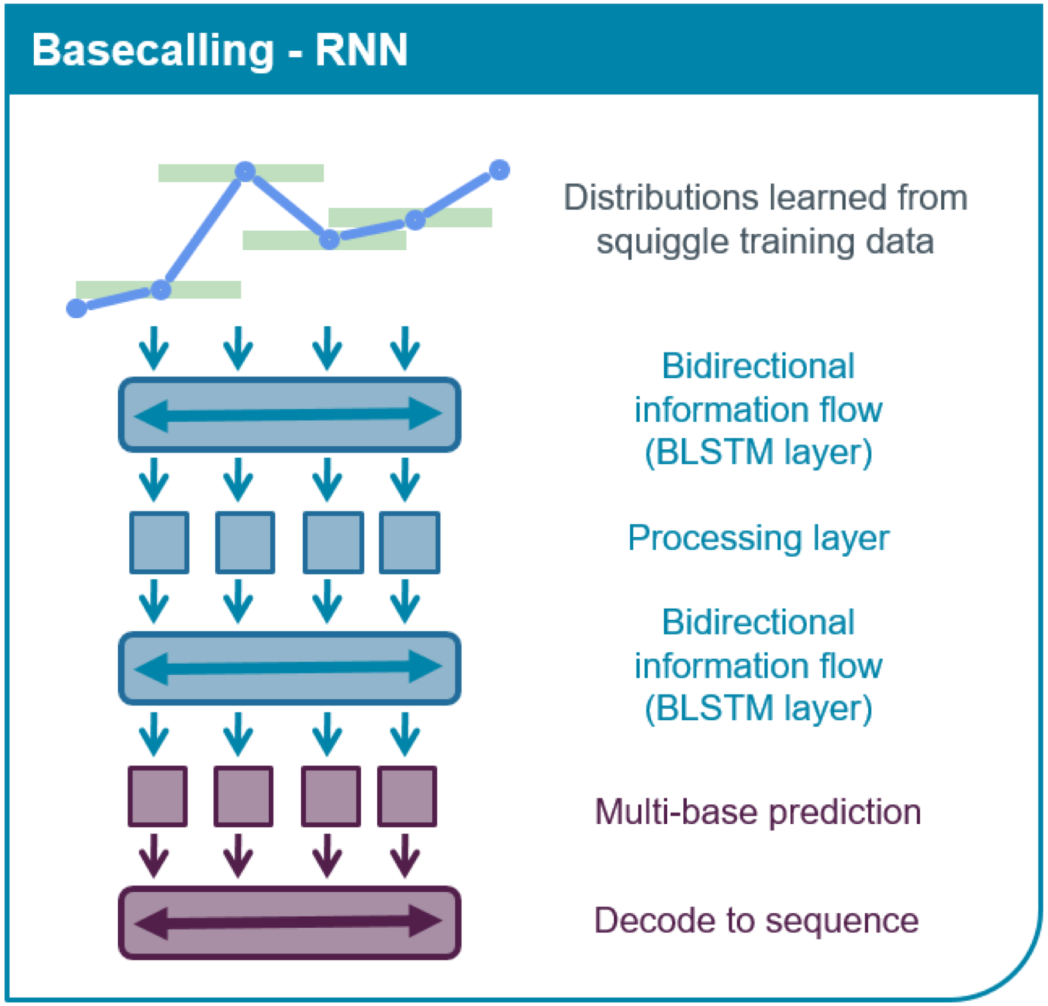


2017

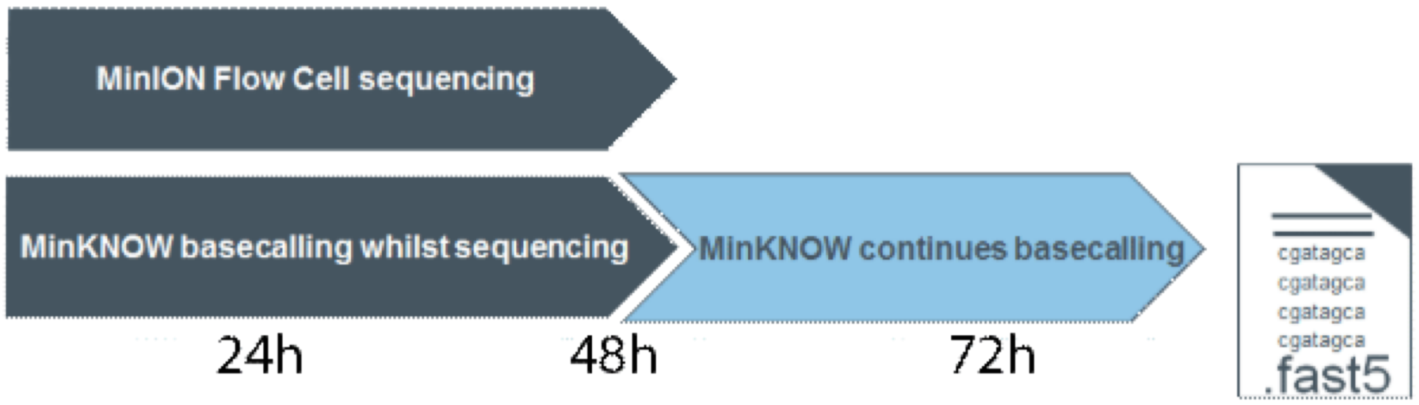


```
ONT1 CCGACTCCGGTTACCCGCGTTGATTTGCTGGGGCAGGGCCG
     |||||||||||||||:|||||||||||||||||||||||||
REF  CCGACTCCGGTTACCAGCGTTGATTTGCTGGGGCAGGGCCG
```

Reccurent Neural Network (RNN) – works like your brain! It can learn on the previous data and improve its performance on new data


Basecalling - RNN
Distributions learned from squiggle training data
Bidirectional information flow (BLSTM layer)
Processing layer
Bidirectional information flow (BLSTM layer)
Multi-base prediction
Decode to sequence

Nanopore basecallers are trained on many sequenced data, so you can run it on your data even if you are sequencing first time


MinION Flow Cell sequencing
MinKNOW basecalling whilst sequencing
MinKNOW continues basecalling
24h
48h
72h
cgatagca
cgatagca
cgatagca
cgatagca
.fast5

$$\begin{tikzcd}
\text{MinION sequencing is controlled by MinKNOW software} \arrow[r] & \text{MinKNOW runs basecalling in parralel} \arrow[r] \arrow[dr] & \text{If you stop sequencing MinKNOW stops basecalling !!!} \arrow[d] \\
& & \text{Alternative: Guppy software (command line)}
\end{tikzcd}$$

FAST5 file format

FAST5 is a type of the Hierarchical Data Format (HDF) and designed for the storage of big datasets

It is binary – not readable by human

HDFview – the tool to see HDF files

FASTQ file format

Each entry consists of 4 lines:

1 @header_name_of_sequence/read
2 sequence
3 +
4 quality (coded by ASCII symbols: https://en.wikipedia.org/wiki/FASTQ_format#Encoding)

Poretools is the tool developed for MinION to convert .fasta5 files to .fastq or .fasta files

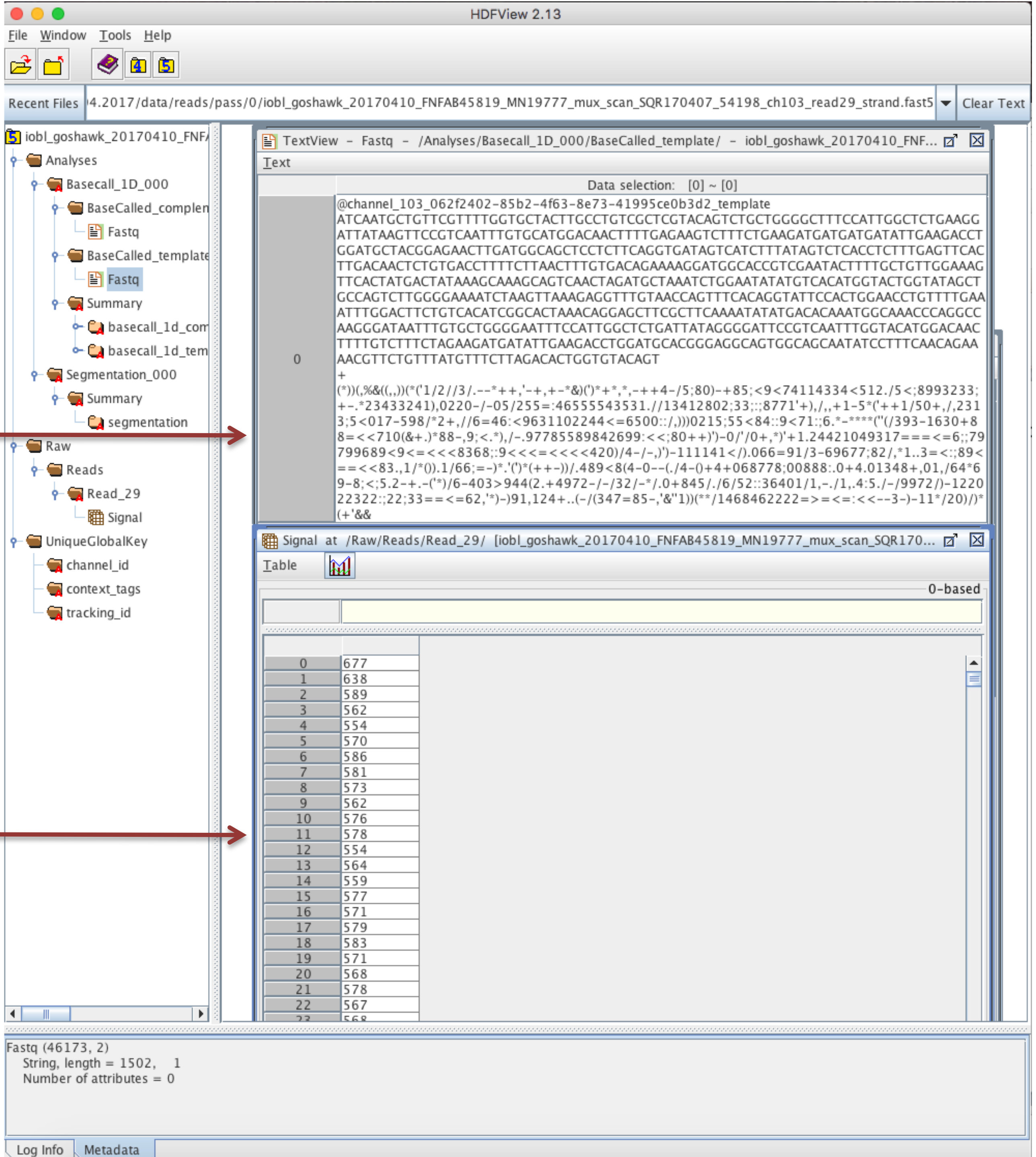
Guppy basecaller allows generating directly .fast5 or .fastq files ☺

HDFviewer window:

(https://support.hdfgroup.org)

FASTQ file

Raw signal
(current from a MinION channel)



```
4.2017/data/reads/pass/0/iobl_goshawk_20170410_FNFAB45819_MN19777_mux_scan_SQR170407_54198_ch103_read29_strand.fast5
```

```
iobl_goshawk_20170410_FNF
  Analyses
    Basecall_1D_000
      BaseCalled_complen
        Fastq
      BaseCalled_template
        Fastq
      Summary
        basecall_1d_com
        basecall_1d_tem
    Segmentation_000
      Summary
        segmentation
  Raw
    Reads
      Read_29
        Signal
  UniqueGlobalKey
    channel_id
    context_tags
    tracking_id
```

TextView – Fastq – /Analyses/Basecall_1D_000/BaseCalled_template/ – iobl_goshawk_20170410_FNF...

```
Data selection:   [0] ~ [0]
0  @channel_103_062f2402-85b2-4f63-8e73-41995ce0b3d2_template
   ATCAATGCTGTTCGTTTTGGTGCTACTTGCCTGTCGCTCGTACAGTCTGCTGGGGCTTTCCATTGGCTCTGAAGG
   ATTATAAGTTCCGTCAATTTGTGCATGGACAACTTTTGAGAAGTCTTTCTGAAGATGATGATGATATTGAAGACCT
   GGATGCTACGGAGAACTTGATGGCAGCTCCTCTTCAGGTGATAGTCATCTTTATAGTCTCACCTCTTTGAGTTCAC
   TTGACAACTCTGTGACCTTTTCTTAACTTTGTGACAGAAAAGGATGGCACCGTCGAATACTTTTGCTGTTGGAAAG
   TTCACTATGACTATAAAGCAAAGCAGTCAACTAGATGCTAAATCTGGAATATATGTCACATGGTACTGGTATAGCT
   GCCAGTCTTGGGGAAAATCTAAGTTAAAGAGGTTTGTAACCAGTTTCACAGGTATTCCACTGGAACCTGTTTTGAA
   ATTTGGACTTCTGTCACATCGGCACTAAACAGGAGCTTCGCTTCAAAATATATGACACAAATGGCAAACCCAGGCC
   AAGGGATAATTTGTGCTGGGGAATTTCCATTGGCTCTGATTATAGGGGATTCCGTCAATTTGGTACATGGACAAC
   TTTTGTCTTTCTAGAAGATGATATTGAAGACCTGGATGCACGGGAGGCAGTGGCAGCAATATCCTTTCAACAGAA
   AACGTTCTGTTTATGTTTCTTAGACACTGGTGTACAGT
   +
   (*))(,%&((,,))(*('1/2//3/.--*++,'-+,+-*&)(')*+*,*,-++4-/5;80)-+85;<9<74114334<512./5<;8993233;
   +-.*23433241),0220-/-05/255=:46555543531.//13412802;33;:;8771'+),/,,+1-5*('++1/50+,/,231
   3;5<017-598/*2+,//6=46:<9631102244<=6500::/,)))0215;55<84::9<71:;6.*-****("(/393-1630+8
   8=<<710(&+.)*88-,9;<.*),/-.97785589842699:<<;80++)')-0/'/0+,*)'+1.24421049317===<=6;;79
   799689<9<=<<<8368;:9<<<=<<<<420)/4-/-,)')-111141</).066=91/3-69677;82/,*1..3=<:;89<
   ==<<83.,1/*()).1/66;=-)*.'(')*(++-))/.489<8(4-0--(./4-()+4+068778;00888:.0+4.01348+,01,/64*6
   9-8;<;5.2-+.-('*)/6-403>944(2.+4972-/-/32/-*/.0+845/./6/52::36401/1,-./1,.4:5./-/9972/)-1220
   22322:;22;33==<=62,'*)-)91,124+..(-/(347=85-,'&"1))(**/1468462222=>=<=:<<--3-)-11*/20)/)*
   (+'&&
```

Signal at /Raw/Reads/Read_29/ [iobl_goshawk_20170410_FNFAB45819_MN19777_mux_scan_SQR170...

```
 0  677
 1  638
 2  589
 3  562
 4  554
 5  570
 6  586
 7  581
 8  573
 9  562
10  576
11  578
12  554
13  564
14  559
15  577
16  571
17  579
18  583
19  571
20  568
21  578
22  567
```

```
Fastq (46173, 2)
    String, length = 1502,    1
    Number of attributes = 0
```

FASTA file format

contains 2 lines:
1 >header
2 sequence

```
>ff6c98dd-bce9-4a2b-bf13-081841413c94_Basecall_2D_minion_20170511_Ae_aegypti
GCGCTGGTTCAGTTACATATTGCTAGGGTTAAGCAGTGGTGACCACAGATTTTTATGATTTATGGATT
CTTTTCTTCTGGCTACATTACTGGAACAGAGCCTGCTTCTCAACAGTGTTCTTATGAACGCTTCAGCTTA
GTATAAAGGC
>4c8a2487-0e13-41a6-ac7b-6cd2fbf5eb95_Basecall_2D_minion_20170511_Ae_aegypti
TACGCGGTGACAAAAACGTGCGTACCGGCAACCGCATGTTGAAACAGGAAAACGTACAAAGGACCC
TCGCAAAATGCGCGACAAAATCTGCAACGTACAACATGCGATAAACGTGCGTGAGGAGATC
>…..
………..
```

> sign is convenient marker to browse through the FASTA files

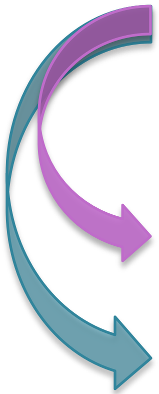
Mind_gaps_in_command_line and file_names!

Basecalling with Guppy, an ONT produced tool-kit

Guppy can do 4 different jobs:	Commands:
1 1D basecalling	*guppy_basecaller*
2 1D2 basecalling	*guppy_basecaller_1d2*
3 Debarcoding (demultiplexing)	*guppy_barcoder*
4 Alignment	*guppy_aligner*

Guppy can be used on Windows, Mac OS, and Linux

Basecalling with Guppy, an ONT produced tool-kit



Guppy can do 4 different jobs:	Commands:
1 1D basecalling	*guppy_basecaller*
2 1D2 basecalling	*guppy_basecaller_1d2*
3 Debarcoding (demultiplexing)	*guppy_barcoder*
4 Alignment	*guppy_aligner*

Workflow:

guppy_basecaller / MinKNOW → *guppy_barcoder*

guppy_basecaller / MinKNOW → *guppy_aligner*

Basecalling with Guppy

How to use your computer efficiently?

Concider: RAM (random-access memory) and number of CPUs (central processing unit).

Guppy_basecaller (1D) uses 1GB per 1 CPU + 4 GB

4 CPUs: 1x4 + 4 = 8 GB of RAM

Guppy_basecaller_1d2 uses 2GB per 1 CPU + 4GB

Basecalling with Guppy

```
guppy_basecaller --help
```

One line:

```
guppy_basecaller –i input/reads.fast5  -s output/reads.fastq --flowcell FLO-MIN107
--kit SQK-LSK108 --qscore_filtering –q 0 --num_callers 1 --cpu_threads_per_caller 1 -r
```

Basecalling with Guppy

One line:

```
guppy_basecaller –i input/reads.fast5  -s output/reads.fastq --flowcell FLO-MIN107
--kit SQK-LSK108 --qscore_filtering  –q 0  --num_callers 1  --cpu_threads_per_caller 1  -r
```

-i (where is your input files)
-s (where you want to save the output)
--flowcell
--kit

```
guppy_basecaller --print_workflows
```

--qscore_filtering (sorts reads into ‘pass‘ and ‘fail‘ folders, --min_qscore is 7 by default)
-q 0 (writes all reads per run in one FASTQ file, default is 4000 reads per file)
-r – recursive (will go through all files in the folder)
--num_callers and *--cpu_threads_per_caller* tell how much of your computer power to use

Basecalling with Guppy

Other options:

GPU run possible

--fast5_out (output FAST5 and FASTQ files, default – only FASTQ)
--compress_fastq (generates gzip output file)

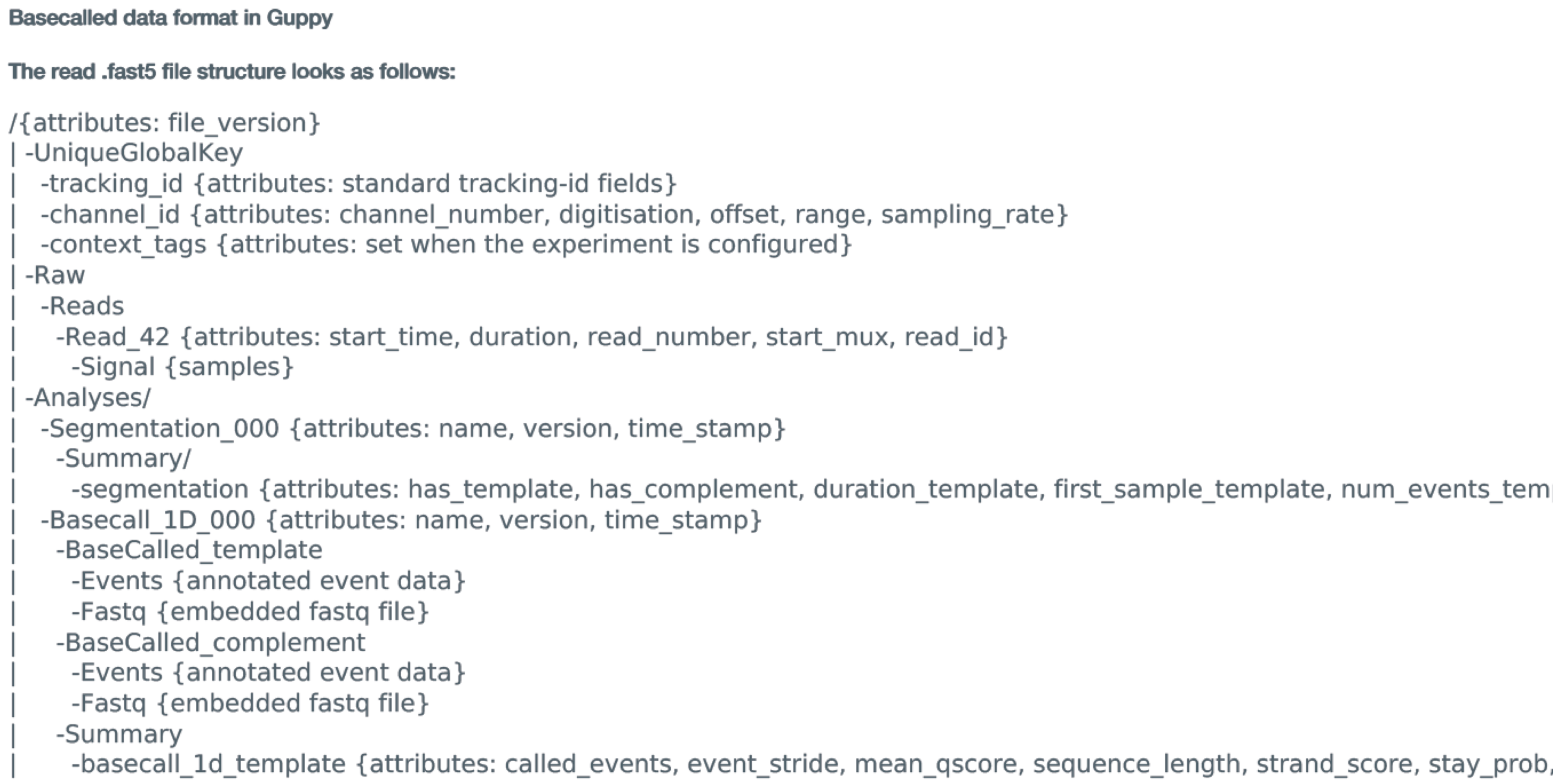
RNAseq:
--reverse_sequence (RNA strain goes through the pore backwards)
--u_substitution (T → U)

--resume – useful if basecalling was interrupted

--calib_detect - callibration strand detection

Adapter trimming is by default

FAST5 file structure of a basecalled read



Basecalled data format in Guppy

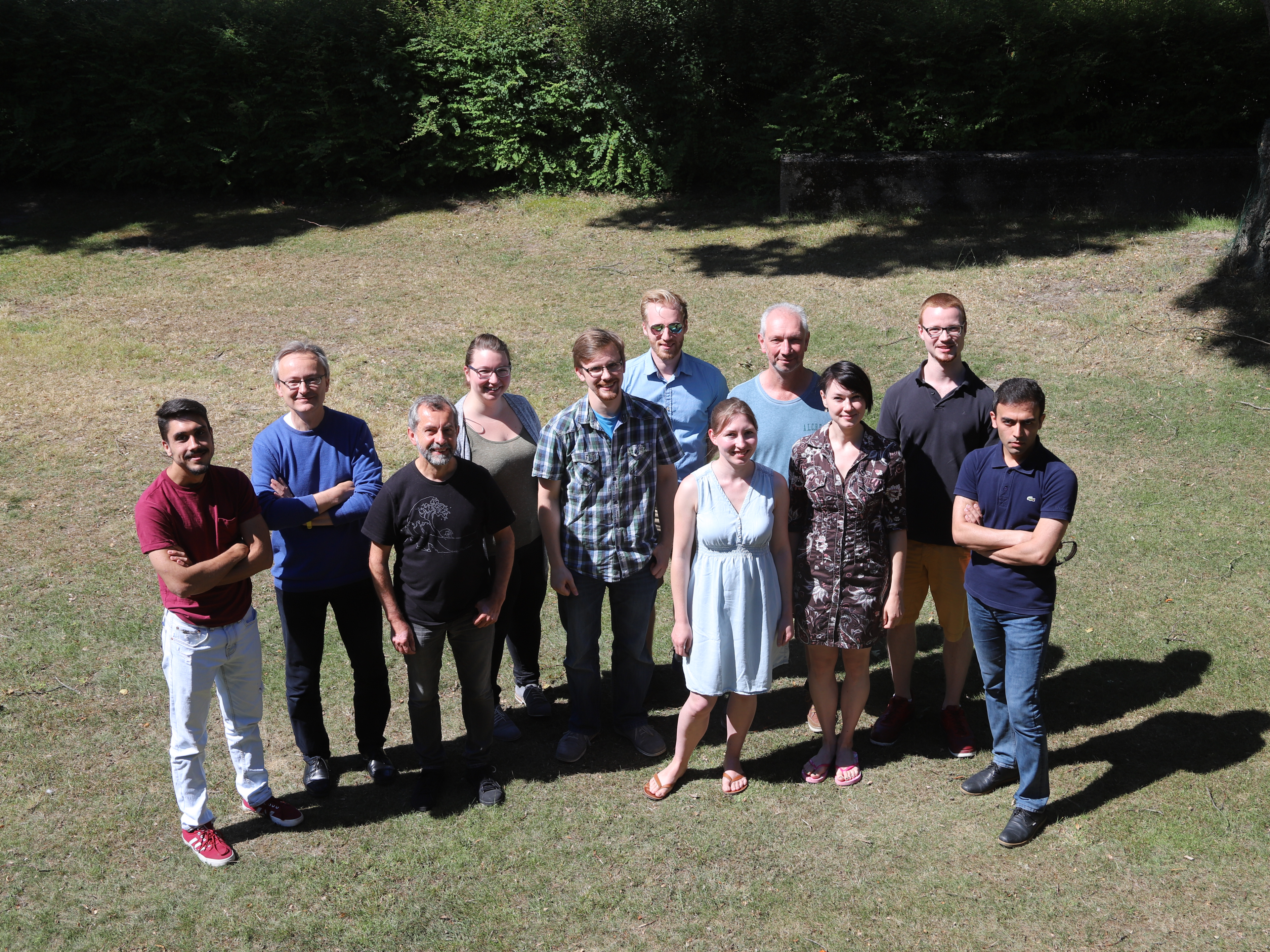
The read .fast5 file structure looks as follows:

```
/{attributes: file_version}
| -UniqueGlobalKey
|   -tracking_id {attributes: standard tracking-id fields}
|   -channel_id {attributes: channel_number, digitisation, offset, range, sampling_rate}
|   -context_tags {attributes: set when the experiment is configured}
| -Raw
|   -Reads
|     -Read_42 {attributes: start_time, duration, read_number, start_mux, read_id}
|       -Signal {samples}
| -Analyses/
|   -Segmentation_000 {attributes: name, version, time_stamp}
|     -Summary/
|       -segmentation {attributes: has_template, has_complement, duration_template, first_sample_template, num_events_tem
|   -Basecall_1D_000 {attributes: name, version, time_stamp}
|     -BaseCalled_template
|       -Events {annotated event data}
|       -Fastq {embedded fastq file}
|     -BaseCalled_complement
|       -Events {annotated event data}
|       -Fastq {embedded fastq file}
|     -Summary
|       -basecall_1d_template {attributes: called_events, event_stride, mean_qscore, sequence_length, strand_score, stay_prob,
```

NanoPipe – interactive tool for MinION sequencing analysis

Developed in the Institute of Bioinformatics, University of Münster, Germany

http://bioinformatics.uni-muenster.de/tools/nanopipe

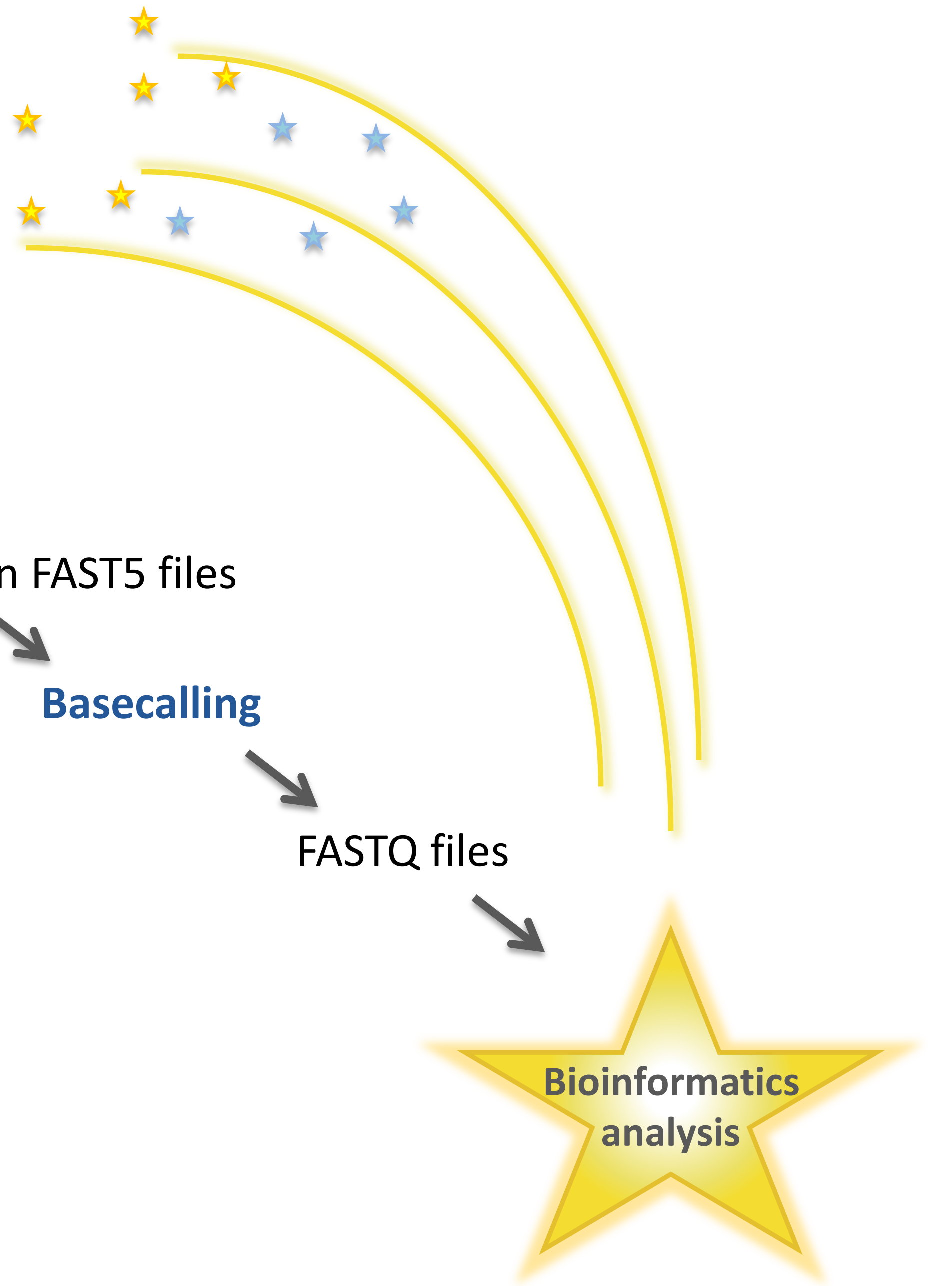


Sequencing

Electric signal in FAST5 files

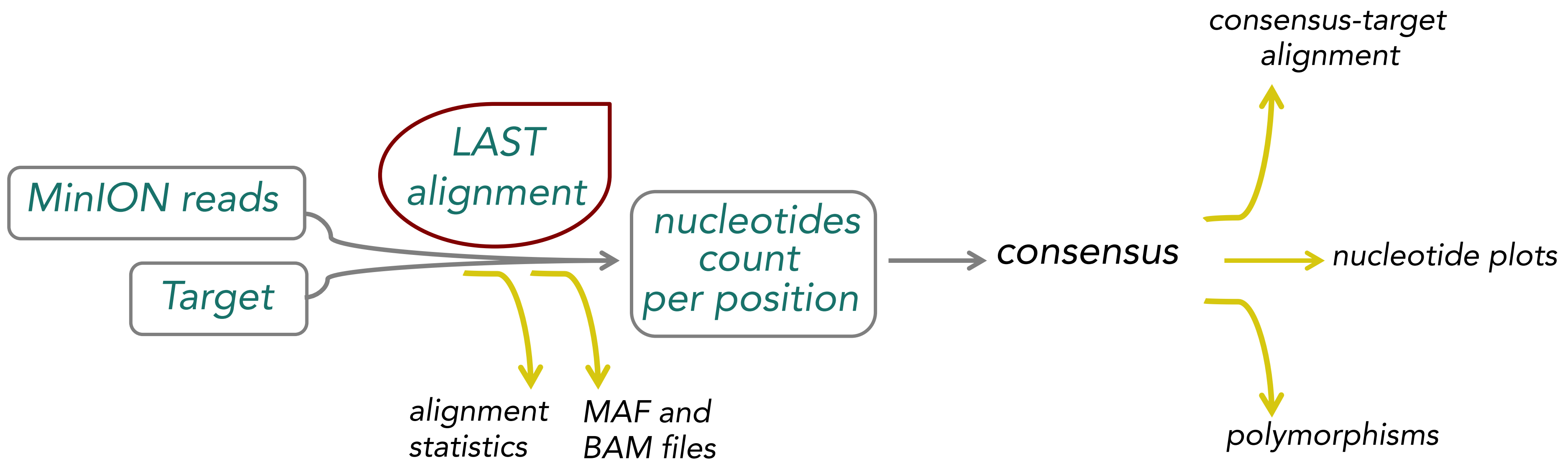
Basecalling

FASTQ files



Bioinformatics analysis

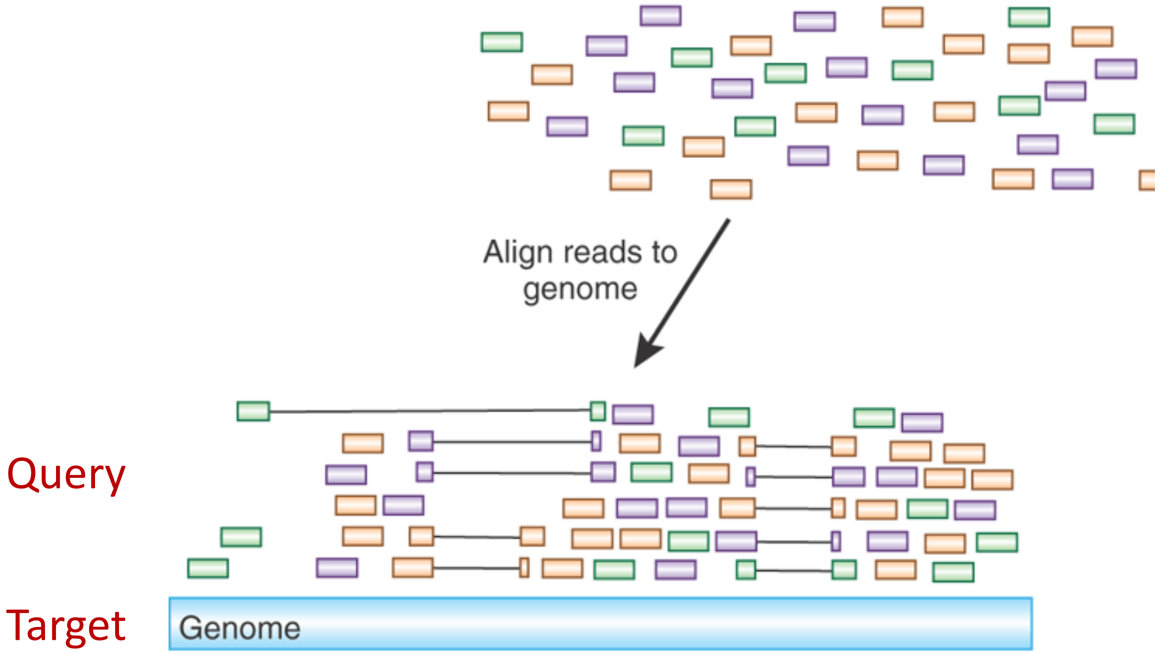
NanoPipe workflow: what does “*pipe*” mean?



Key steps of NanoPipe: LAST

LAST sequence aligner maps MinIon-produced reads to a target (selected region, exon, gene, genome)

Martin Frith
University of Tokyo,
Division of Biosciences

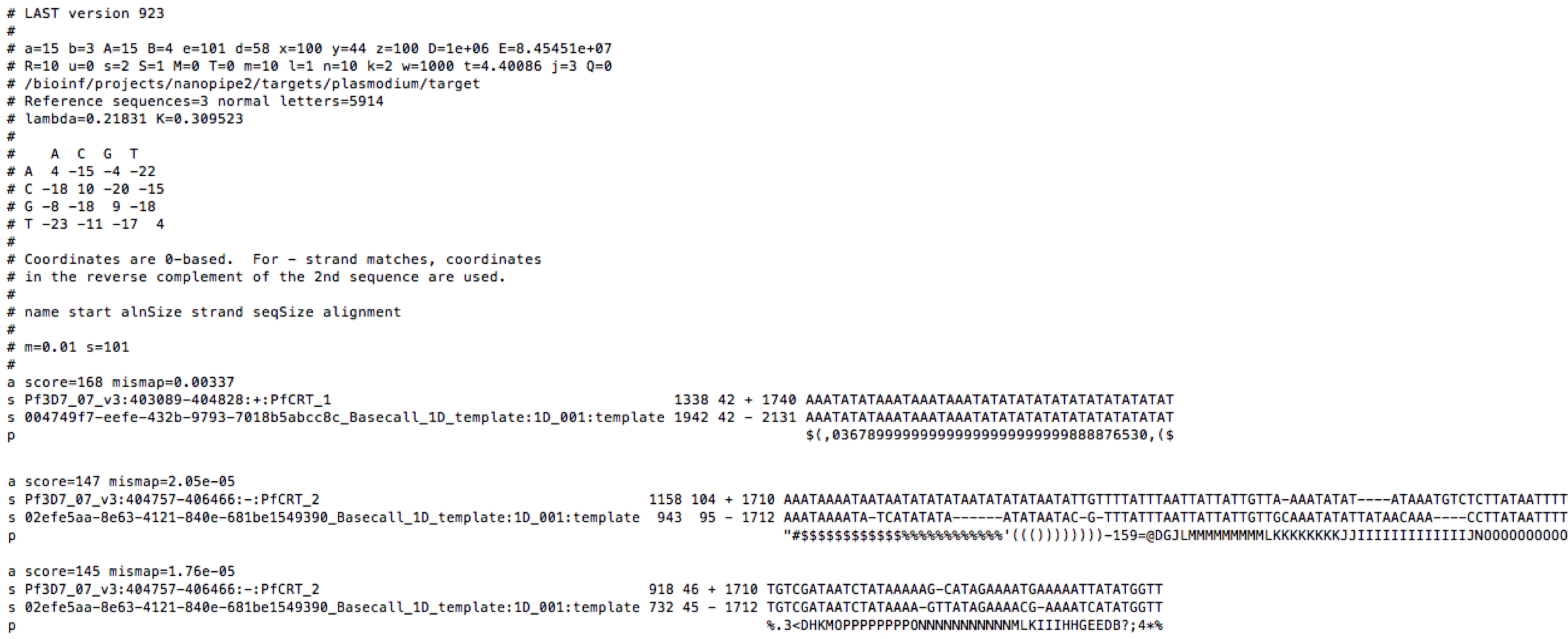


Other aligners: BLAST (psi-BLAST, delta-BLAST), HMMER, MUSCLE, MAFFT…

File formats that are important to know when working with mapped reads:

- .maf
- .sam
- .bam

.maf file format – shows pairs of aligned sequences with the coordinates

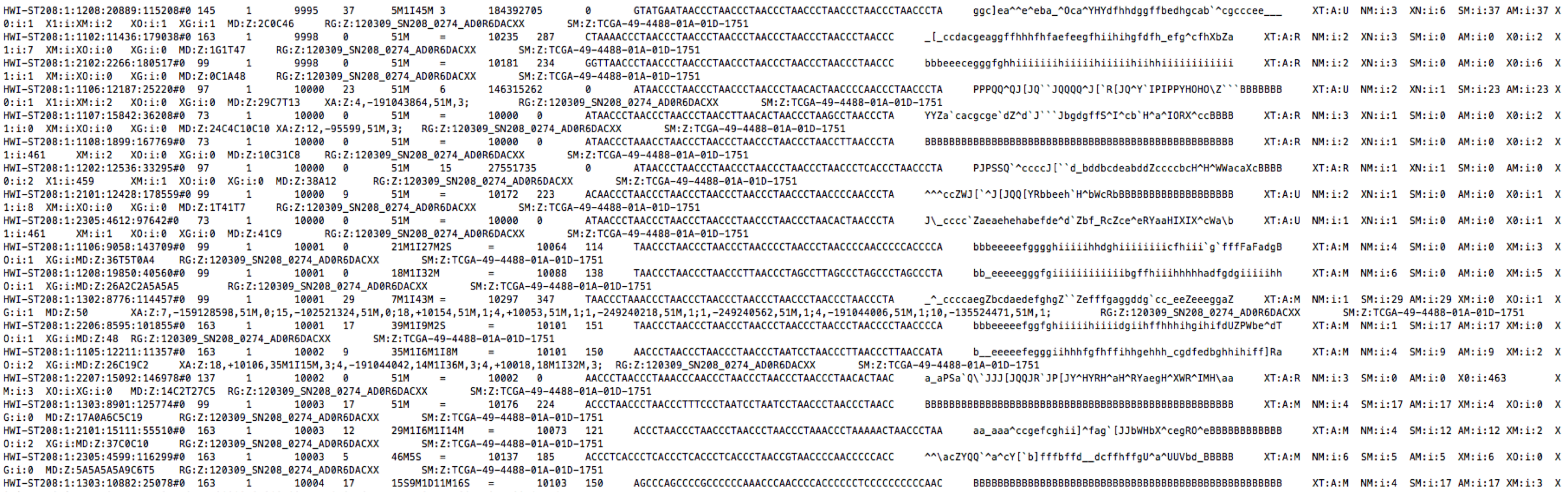


```
# LAST version 923
#
# a=15 b=3 A=15 B=4 e=101 d=58 x=100 y=44 z=100 D=1e+06 E=8.45451e+07
# R=10 u=0 s=2 S=1 M=0 T=0 m=10 l=1 n=10 k=2 w=1000 t=4.40086 j=3 Q=0
# /bioinf/projects/nanopipe2/targets/plasmodium/target
# Reference sequences=3 normal letters=5914
# lambda=0.21831 K=0.309523
#
#    A   C   G   T
# A  4 -15  -4 -22
# C -18  10 -20 -15
# G -8 -18   9 -18
# T -23 -11 -17   4
#
# Coordinates are 0-based.  For - strand matches, coordinates
# in the reverse complement of the 2nd sequence are used.
#
# name start alnSize strand seqSize alignment
#
# m=0.01 s=101
#
a score=168 mismap=0.00337
s Pf3D7_07_v3:403089-404828:+:PfCRT_1                                        1338 42 + 1740 AAATATATAAATAAATAAATATATATATATATATATATATAT
s 004749f7-eefe-432b-9793-7018b5abcc8c_Basecall_1D_template:1D_001:template 1942 42 - 2131 AAATATATAAATAAATAAATATATATATATATATATATATAT
p                                                                                            $(,03678999999999999999999999888876530,($


a score=147 mismap=2.05e-05
s Pf3D7_07_v3:404757-406466:-:PfCRT_2                                     1158 104 + 1710 AAATAAAATAATAATATATATAATATATATATAATATTGTTTTATTTAATTATTATTGTTA-AAATATAT----ATAAATGTCTCTTATAATTTT
s 02efe5aa-8e63-4121-840e-681be1549390_Basecall_1D_template:1D_001:template  943  95 - 1712 AAATAAAATA-TCATATATA------ATATAATAC-G-TTTATTTAATTATTATTGTTGCAAATATATTATAACAAA----CCTTATAATTTT
p                                                                                          "#$$$$$$$$$$$%%%%%%%%%%%%%'((()))))))))-159=@DGJLMMMMMMMMMLKKKKKKKKJJIIIIIIIIIIIIIJNOOOOOOOOOO

a score=145 mismap=1.76e-05
s Pf3D7_07_v3:404757-406466:-:PfCRT_2                                     918 46 + 1710 TGTCGATAATCTATAAAAAG-CATAGAAAATGAAAAATTATATGGTT
s 02efe5aa-8e63-4121-840e-681be1549390_Basecall_1D_template:1D_001:template 732 45 - 1712 TGTCGATAATCTATAAAA-GTTATAGAAAACG-AAAATCATATGGTT
p                                                                                        %.3<DHKMOPPPPPPPPPONNNNNNNNNNNMLKIIIHHGEEDB?;4*%
```

.bam and .sam file format – informative about alignment of sequences to a target

.bam is a binary file (humans can not read), it contains a lot of information about the alignment. Important part is the HEADER and FLAGs specifications (https://broadinstitute.github.io/picard/explain-flags.html)

.sam format is the readable version of **.bam** (some info is casted away):



```
HWI-ST208:1:1208:20889:115208#0 145     1       9995    37      5M1I45M 3       184392705       0       GTATGAATAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTA     ggc]ea^^e^eba_^Oca^YHYdfhhdggffbedhgcab`^cgcccee___     XT:A:U  NM:i:3  XN:i:6  SM:i:37 AM:i:37 X
0:i:1  X1:i:XM:i:2   XO:i:1  XG:i:1  MD:Z:2C0C46     RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1102:11436:179038#0 163     1       9998    0       51M     =       10235   287     CTAAAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCC     _[_ccdacgeaggffhhhfhfaefeegfhiihihgfdfh_efg^cfhXbZa     XT:A:R  NM:i:2  XN:i:3  SM:i:0  AM:i:0  X0:i:2  X
1:i:7  XM:i:XO:i:0   XG:i:0  MD:Z:1G1T47     RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2102:2266:180517#0  99      1       9998    0       51M     =       10181   234     GGTTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCC     bbbeeecegggfghhiiiiiiihiiiiihiiiiihiihhiiiiiiiiiiii     XT:A:R  NM:i:2  XN:i:3  SM:i:0  AM:i:0  X0:i:6  X
1:i:1  XM:i:XO:i:0   XG:i:0  MD:Z:0C1A48     RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1106:12187:25220#0  97      1       10000   23      51M     6       146315262       0       ATAACCCTAACCCTAACCCTAACCCTAACACTAACCCCAACCCTAACCCTA     PPPQQ^QJ[JQ``JQQQQ^J[`R[JQ^Y`IPIPPYHOHO\Z```BBBBBBB     XT:A:U  NM:i:2  XN:i:1  SM:i:23 AM:i:23 X
0:i:1  X1:i:XM:i:2   XO:i:0  XG:i:0  MD:Z:29C7T13    XA:Z:4,-191043864,51M,3;        RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1107:15842:36208#0  73      1       10000   0       51M     =       10000   0       ATAACCCTAACCCTAACCCTAACCTTAACACTAACCCTAAGCCTAACCCTA     YYZa`cacgcge`dZ^d`J```JbgdgffS^I^cb`H^a^IORX^ccBBBB     XT:A:R  NM:i:3  XN:i:1  SM:i:0  AM:i:0  X0:i:2  X
1:i:0  XM:i:XO:i:0   XG:i:0  MD:Z:24C4C10C10 XA:Z:12,-95599,51M,3;   RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1108:1899:167769#0  73      1       10000   0       51M     =       10000   0       ATAACCCTAAACCTAACCCTAACCCTAACCCTAACCCTAACCTTAACCCTA     BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB     XT:A:R  NM:i:2  XN:i:1  SM:i:0  AM:i:0  X0:i:2  X
1:i:461      XM:i:2   XO:i:0  XG:i:0  MD:Z:10C31C8    RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1202:12536:33295#0  97      1       10000   0       51M     15      27551735        0       ATAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTCACCCTAACCCTA     PJPSSQ`^ccccJ[``d_bddbcdeabddZccccbcH^H^WWacaXcBBBB     XT:A:R  NM:i:1  XN:i:1  SM:i:0  AM:i:0  X
0:i:2  X1:i:459      XM:i:1  XO:i:0  XG:i:0  MD:Z:38A12      RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2101:12428:178559#0 99      1       10000   9       51M     =       10172   223     ACAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCCAACCCTA     ^^^ccZWJ[`^J[JQQ[YRbbeeh`H^bWcRbBBBBBBBBBBBBBBBBBBB     XT:A:U  NM:i:2  XN:i:1  SM:i:0  AM:i:0  X0:i:1  X
1:i:8  XM:i:XO:i:0   XG:i:0  MD:Z:1T41T7     RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2305:4612:97642#0   73      1       10000   0       51M     =       10000   0       ATAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACACTAACCCTA     J\_cccc`Zaeaehehabefde^d`Zbf_RcZce^eRYaaHIXIX^cWa\b     XT:A:U  NM:i:1  XN:i:1  SM:i:0  AM:i:0  X0:i:1  X
1:i:461      XM:i:1   XO:i:0  XG:i:0  MD:Z:41C9       RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1106:9058:143709#0  99      1       10001   0       21M1I27M2S      =       10064   114     TAACCCTAACCCTAACCCTAACCCCTAACCCTAACCCCAACCCCCACCCCA     bbbeeeeefggghiiiiihhdghiiiiiiiicfhiii`g`fffFaFadgB     XT:A:M  NM:i:4  SM:i:0  AM:i:0  XM:i:3  X
O:i:1  XG:i:MD:Z:36T5T0A4     RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1208:19850:40560#0  99      1       10001   0       18M1I32M        =       10088   138     TAACCCTAACCCTAACCCTTAACCCTAGCCTTAGCCCTAGCCCTAGCCCTA     bb_eeeeegggfgiiiiiiiiiiiibgffhiiihhhhhadfgdgiiiiihh     XT:A:M  NM:i:6  SM:i:0  AM:i:0  XM:i:5  X
O:i:1  XG:i:MD:Z:26A2C2A5A5A5          RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1302:8776:114457#0  99      1       10001   29      7M1I43M =       10297   347     TAACCCTAAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTA     _^_ccccaegZbcdaedefghgZ``Zefffgaggddg`cc_eeZeeeggaZ     XT:A:M  NM:i:1  SM:i:29 AM:i:29 XM:i:0  XO:i:1  X
G:i:1  MD:Z:50         XA:Z:7,-159128598,51M,0;15,-102521324,51M,0;18,+10154,51M,1;4,+10053,51M,1;1,-249240218,51M,1;1,-249240562,51M,1;4,-191044006,51M,1;10,-135524471,51M,1;          RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2206:8595:101855#0  163     1       10001   17      39M1I9M2S       =       10101   151     TAACCCTAACCCTAACCCTAACCCTAACCCTAACCCTAACCCCTAACCCCA     bbbeeeeefggfghiiiiihiiiidgiihffhhhihgihifdUZPWbe^dT     XT:A:M  NM:i:1  SM:i:17 AM:i:17 XM:i:0  X
O:i:1  XG:i:MD:Z:48  RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1105:12211:11357#0  163     1       10002   9       35M1I6M1I8M     =       10101   150     AACCCTAACCCTAACCCTAACCCTAATCCTAACCCTTAACCCTTAACCATA     b__eeeeefegggiihhhfgfhffihhgehhh_cgdfedbghhihiff]Ra     XT:A:M  NM:i:4  SM:i:9  AM:i:9  XM:i:2  X
O:i:2  XG:i:MD:Z:26C19C2       XA:Z:18,+10106,35M1I15M,3;4,-191044042,14M1I36M,3;4,+10018,18M1I32M,3;  RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2207:15092:146978#0 137     1       10002   0       51M     =       10002   0       AACCCTAACCCTAAACCCAACCCTAACCCTAACCCTAACCCTAACACTAAC     a_aPSa`Q\`JJJ[JQQJR`JP[JY^HYRH^aH^RYaegH^XWR^IMH\aa     XT:A:R  NM:i:3  SM:i:0  AM:i:0  X0:i:463        X
M:i:3  XO:i:XG:i:0    MD:Z:14C2T27C5  RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1303:8901:125774#0  99      1       10003   17      51M     =       10176   224     ACCCTAACCCTAACCCTTTCCCTAATCCTAATCCTAACCCTAACCCTAACC     BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB     XT:A:M  NM:i:4  SM:i:17 AM:i:17 XM:i:4  XO:i:0  X
G:i:0  MD:Z:17A0A6C5C19       RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2101:15111:55510#0  163     1       10003   12      29M1I6M1I14M    =       10073   121     ACCCTAACCCTAACCCTAACCCTAACCCTAAACCCTAAAAACTAACCCTAA     aa_aaa^ccgefcghii]^fag`[JJbWHbX^cegRO^eBBBBBBBBBBBB     XT:A:M  NM:i:4  SM:i:12 AM:i:12 XM:i:2  X
O:i:2  XG:i:MD:Z:37C0C10       RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:2305:4599:116299#0  163     1       10003   5       46M5S   =       10137   185     ACCCTCACCCTCACCCTCACCCTCACCCTAACCGTAACCCCAACCCCCACC     ^^\acZYQQ`^a^cY[`b]fffbffd__dcffhffgU^a^UUVbd_BBBBB     XT:A:M  NM:i:6  SM:i:5  AM:i:5  XM:i:6  XO:i:0  X
G:i:0  MD:Z:5A5A5A5A9C6T5      RG:Z:120309_SN208_0274_AD0R6DACXX       SM:Z:TCGA-49-4488-01A-01D-1751
HWI-ST208:1:1303:10882:25078#0  163     1       10004   17      15S9M1D11M16S   =       10103   150     AGCCCAGCCCCGCCCCCCAAACCCAACCCCACCCCCCTCCCCCCCCCCAAC     BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB     XT:A:M  NM:i:4  SM:i:17 AM:i:17 XM:i:3  X
```

Summary: files' formats

Files' formats used for storing sequences:

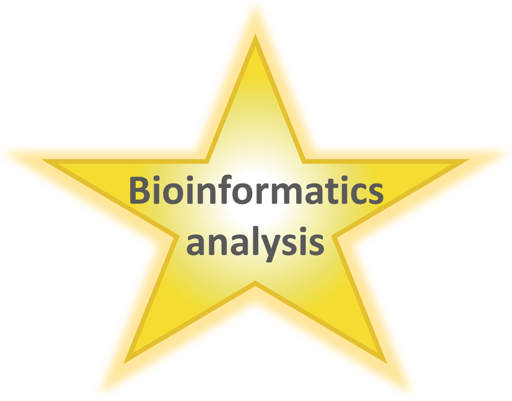
FAST5 – big, binary (cant read), contain a lot of metadata
FASTQ – readable by human, contains sequences and sequence quality
FASTA – readable, contains sequences

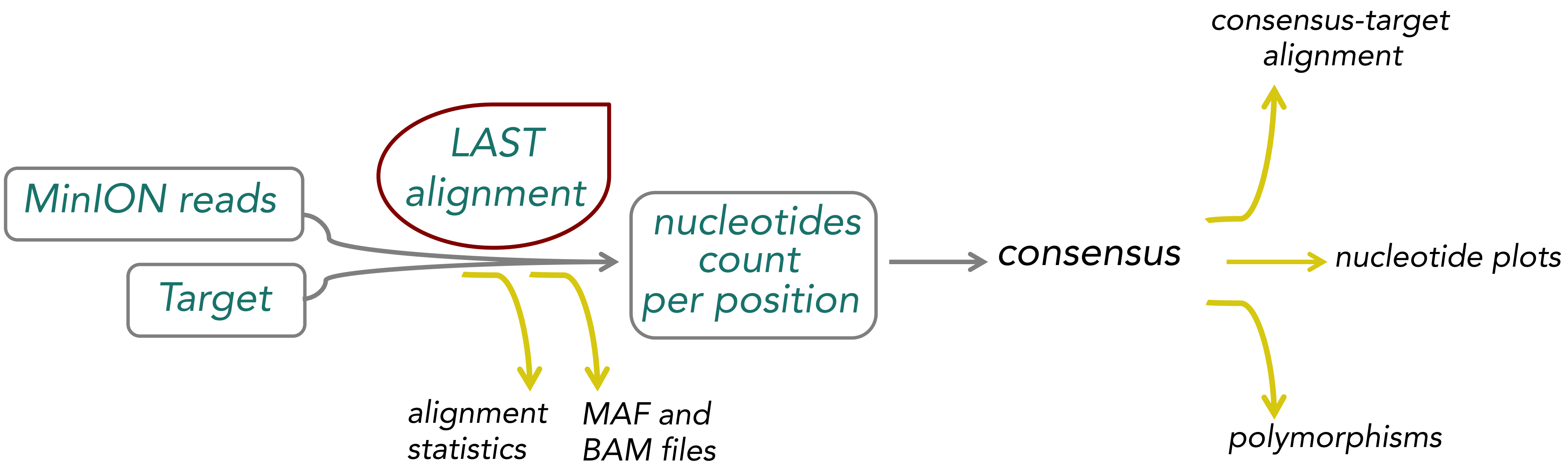
Files' formats used for storing results of sequence alignment:

maf – contains pairs of aligned sequences with the alignment's coordinates; for example, used by LAST
bam – binary format, includes aligned sequences, coordinates, information about bioinformatics processing, quality, …
sam – human readable version of .bam, much bigger in size

FASTQ, FASTA and bam files are widely used in all DNA/RNA bioinformatics analysis

NanoPipe workflow: what does "*pipe*" mean?





NanoPipe helps us to…

- See if our sequencing worked: how many reads were mapped to the target and where exactly, what part of each read mapped (Alignments length distribution)

- Detect insertions/deletions and single nucleotide variations

- Visualization of the experiment in NanoPipe and in IGV-viewer (bam and indexed bam files)

- FASTA file with the consensus sequence

Useful links:

Our lab ;p -
http://bioinformatics.uni-muenster.de

Genome viewers -
https://software.broadinstitute.org/software/igv/

https://genome.ucsc.edu/