NanoOK Documentation

Release 1.27

Richard Leggett

Sep 18, 2018

Contents

CHAPTER 1

Download and installation

1.1 Overview

There are a number of dependency requirements in order to run NanoOK. You need to ensure these are installed before installing NanoOK (see below). If you would rather use a pre-installed image, we also provide a *VirtualBox image* or a *Docker image*.

1.2 Requirements

Please make sure you have installed the following before attempting to run NanoOK:

1. Java SE runtime environment - the main application is written in Java.
2. R statistical environment - for graph plotting. Make sure you have the necessary packages and see section below on installing missing ones.
   - Packages needed: `ggplot2`, `scales`, `gridExtra`, `reshape`
3. LaTeX environment that includes pdflatex - for generating PDF. Make sure you have the necessary packages installed.
   - Packages needed: `graphicx`, `url`, `multirow`, `rotating`, `color`, `titlesec`, `geometry`, `float`
4. HDF5 tools - are required for extracting FASTA/Q from FAST5 files.
5. LAST or other supported alignment tool - NanoOK also supports BLASR, BWA-MEM and MarginAlign. NanoOK assumes the aligner is available in one of the directories pointed to by the PATH variable.
6. Git - required for accessing the source code. Or you can download from the NanoOK github page.

1.3 Checking if you have the dependencies installed

1. Type ‘‘java -h‘‘ to see the help text for Java.

2. Type `R -h` to see if R is installed.
3. Type `pdflatex -v` to see if LaTeX is installed.
4. Type `h5dump -h` to see if HDF5 tools are installed.
5. Type `lastal -h` to see help text for LAST.

1.4 Installing missing R packages

First, start R - on Linux you may want to give sudo access to allow the installation, i.e.:

```
sudo R
```

Then, to install packages type:

```
install.packages(c("ggplot2", "scales", "gridExtra", "reshape"))
```

You may then have to select a 'mirror' site for the download. Once complete, you can type q() to exit R.

1.5 Installing NanoOK

NanoOK works best on Linux and MacOS. If you need to use Windows, some tips are provided below, but installation of the dependencies is not as straightforward as on Linux and Mac OS. NanoOK can be downloaded from TGAC's GitHub repository. You can either download the .zip file, or if you have git installed, you can type:

```
git clone https://github.com/TGAC/NanoOK.git
```

To install:

1. Copy the whole NanoOK directory somewhere appropriate on your system.
2. Set a NANOOK_DIR environment variable to point to this directory. This enables NanoOK to know where to find the Java JARs and R scripts it uses. On Linux, you would typically do this by adding the following command to your .bash_profile (or .profile on Ubuntu) file or 'source' script:

   ```
   export NANOOK_DIR=/path/to/NanoOK
   ```

   On MacOs, you can edit the file using:

   ```
   open -e ~/.bash_profile
   ```

3. Add the bin directory to your PATH variable. On Linux, you would typically do this by adding the following command to your .bash_profile (or .profile on Ubuntu) file or 'source' script:

   ```
   export PATH=/path/to/NanoOK/bin:$PATH
   ```

4. You may have to reset for the change to take effect.
5. Set Java heap size for the tool - the main JAVA program is packaged as a .jar file. A wrapper script called nanook (in the bin directory) enables this to be run without specifying the Java Virtual Machine parameters. By default, this script request 2Gb of RAM for the Java heap. If you wish to reduce/increase this, please edit this file.

1.6 Tips for Ubuntu users

Most of the dependencies can be installed with sudo apt-get install:

```
sudo apt-get update
sudo apt-get install r-base
sudo apt-get install r-cran-ggplot2
sudo apt-get install hdf5-tools
sudo apt-get install texlive
sudo apt-get install texlive-latex-extra
sudo apt-get install default-jre
sudo apt-get install git
```

LAST needs to be compiled from the source code, but this is very straightforward - see the website. NanoOK can then be installed as above.

1.7 Tips for Mac OS users

- Clicking on the links in the "Requirements" section above should take you to pages where you can download the dependencies.
- The HDF5 tools binary for Mac OS isn't completely obvious to find! Try this link: https://www.hdfgroup.org/ftp/HDF5/releases/hdf5-1.8.7/obtain5187.html
- Some users like to use Homebrew (http://brew.sh) to install some of the dependencies. This provides a package installation command similar to the Ubuntu apt-get install command.

CHAPTER 2

Using the Docker image

A docker image of NanoOK is provided on Docker Hub which includes all the dependencies needed to run. First, you need to have installed the Docker Engine. Then you can pull the NanoOK image:

```
docker pull richardmleggett/nanook
```

To run NanoOK, the easiest way is to run a shell in the NanoOK image using:

```
docker run -i -t -v /path/to/your/data:/usr/nanopore richardmleggett/nanook bash
```

From here you will get a prompt from which you can run your NanoOK commands, for example:

```
nanook extract -s /usr/nanopore/YourSample
```

When you have finished, type `exit` to end Docker.

Notes:

- In the docker run command, you need to map your data directory to the Docker image. This is done with the `-v` option. In the above example, the data on our local machine is in `/path/to/your/data` and this appears in the Docker image as `/usr/nanopore`, which is why we specify `/usr/nanopore/YourSample` as the sample directory to the nanook command.
- If you get an error from the docker command, it may be because you haven't sudo'd it, or added your user to the docker group - see How can I use docker without sudo?

CHAPTER 3

Using the VirtualBox image

You can download a VirtualBox image from TGAC's opendata site. This is a complete virtual computer system that includes an Ubuntu operating system, NanoOK, all the dependencies and an example data set.

NOTE: Currently, this image only includes the LAST aligner. If you wish to use an alternative aligner, you will need to download that to the VM. In future releases, we hope to include other alignment tools.

- To run it, you will need to have installed the free VirtualBox software - it's available from virtualbox.org.
- When you start the image, it will boot into an Ubuntu desktop. You can click on the Terminal icon on the left hand side and start typing commands.
- The screen display is quite small - this is because Oracle (owners of VirtualBox) do not license users to distribute additional components known as the "Guest Additions". However, you can download this for free and install it yourself - see the VirtualBox manual for details.
- You also need to install the Guest Additions before you can share folders with the host operating system.
- There is one user setup called 'nanook' and the password is also 'nanook'.

To run the example dataset (consisting of only 500 reads) with NanoOK, do the following:

1. Start the VirtualBox image.
2. Open a terminal and type:

```
cd ~/Documents/NanoOK\_Example
```

3. To index the reference, type:

```
cd references
lastdb -Q 0 ecoli\_dh10b\_cs ecoli\_dh10b\_cs.fasta
cd ..
```

4. To extract, type:

```
nanook extract -s N79596\_dh10b\_8kb\_11022015 -t 2
```

5. To align, type:

```
nanook align -s N79596\_dh10b\_8kb\_11022015 -r references/ecoli\_dh10b\_cs.fasta␣
↪-t 2
```

6. To analyse, type:

```
nanook analyse -s N79596\_dh10b\_8kb\_11022015 -r references/ecoli\_dh10b\_cs.
↪fasta -passonly -t 2
```

7. To view the PDF, use the "Files" icon in the left-hand toolbar of the Ubuntu desktop and browse to Documents->NanoOK_Example->N79596_dh10b_8kb_11022015->latex_last_passonly and double-click on the PDF file. Note, that as this example consists of such a small number of reads, the graphs will not be quite as informative as for a full dataset.

When new versions of NanoOK are released, you don't need to download another VM image. Instead type the following inside a Terminal window to install the latest version:

```
cd ~/Documents/NanoOK
git pull
```

CHAPTER 4

Running NanoOK

Note about Albacore FASTQ output: We are currently working on support for the Albacore FASTQ output structure, but this is not yet ready. In the meantime, the easiest approach is to use Albacore fast5 output, followed by nanook extract. This also lets you filter your reads according to a minimum quality threshold. Alternatively, you can run nanook_split_reads (see *nanook_split_reads*) in order to split to separate FASTQ/FASTA files before running nanook align.

4.1 Overview

NanoOK understands the concept of sample directories/folders. Within a sample directory will be a set of subdirectories:

- for FAST5 files (though these can exist outside of the subdirectory if you want)
- for FASTA/Q files
- for alignments, named after the aligner - e.g. "bwa" or "last"
- for logs
- for graphs
- for analysis
- for the LaTeX report generated

With the exception of the FAST5 subdirectory, NanoOK will create directories within the sample directory as it needs them.

NanoOK expects FAST5 files to be arranged in one of the following standard ways:

Metrichor structure	Albacore structure
• N79681_1stLambda_8kb (the sample directory) – downloads ∗ pass · BCXX or barcodeXXX (if barcoded) · batch_XXXX · xxxx.fast5 ∗ fail · unaligned (if barcoded) · batch_XXXX · xxx.fast5	• N79681_1stLambda_8kb (the sample directory) – workspace ∗ barcodeXXXX (if barcoded) · 0 · xxx.fast5

You can use the -f parameter of nanook extract to tell the program where your fast5 files are, e.g. -f workspace (for a relative path within the sample directory) or -f /path/to/workspace (for an absolute path).

If your files are not like this, then NanoOK will probably struggle to process them.

The next few sections assume the use of the LAST aligner - our currently preferred choice. Later sections detail changes for other aligners.

4.2 For the impatient

```
lastdb -Q 0 referencename referencename.fasta
nanook extract -s SampleDir -f path/to/fast5
nanook align -s SampleDir -r referencename.fasta
nanook analyse -s SampleDir -r referencename.fasta -passonly
```

4.3 Extracting FASTA files

Use the nanook extract option:

```
nanook extract -s SampleDir
```

where:

- `-s` or `-sample` specifies the sample name (ie. same as directory name)
- `-a` or `-fasta` (optional, set by default) specifies FASTA output
- `-q` or `-fastq` specifies FASTQ output
- `-f` or `-reads` allows you to specify the location of the FAST5 reads. This needs to be either absolute (beginning with a '/' - e.g. `-f /Users/leggett/examplerun/workspace`) or relative to the sample directory (e.g. `-f workspace` if the sample directory contains the albacore workspace directory).
- `-minquality` allows you to set the minimum quality for a pass read. Without specifying this, your reads will be treated as per the basecaller's criteria. If the basecaller doesn't separate into pass/fail, all reads are considered pass.

4.4 Preparing references

NanoOK supports multiple reference sequences - however, these currently need to be located in a single FASTA file.

References first need to be indexed with the aligner, e.g. with LAST:

```
lastdb -Q 0 referencename referencename.fasta
```

The -Q 0 tells LAST to expect a FASTA file, the next parameter is the output prefix and the final parameter is the reference file.

Note: the output prefix must be the same name as the FASTA file, apart from the .fa or .fasta extension. This is because NanoOK needs to use the original FASTA file.

NanoOK also produces its own index file. It will do this when you first run alignments and will generate a file called `referencename.fasta.sizes`, which will look similar to:

```
gi|556503834|ref|NC_000913.3| 4641652 Escherichia_coli
```

This is a three column format - the first column gives sequence IDs in the FASTA file, the second the size of the sequence and the third the display name used by NanoOK. The display name is also used for filenames for various intermediate files.

NanoOK will extract meaningful display names from NCBI format sequences, but it cannot cope with every format of sequence ID. It is recommended that every time you generate a new .sizes file, you check that you are happy with the display names and that they do not contain characters that can cause problems with filenames and LaTeX - e.g. | (bar). You can do this before running nanook analyse (below).

If you make changes to the reference file, you will need to delete the current .sizes file and re-generate it, otherwise there will be contigs missing from it.

4.5 Running alignments

To run alignments for all reads, type:

```
nanook align -s SampleDir -r referencename.fasta
```

where:

- `-s` or `-sample` specifies the sample name (ie. same as directory name)
- `-r`` or ``-reference` specifies the name of the reference file to use (including the .fasta or .fa extension)
- `-aligner` (optional) specifies the name of the aligner, which defaults to 'last'. Valid options are:
  - `last` - for LAST
  - `bwa` - for BWA-MEM
  - `blasr` - for BLASR
  - `marginalign` - for MarginAlign
  - `graphmap` - for GraphMap

4.6 Running NanoOK analysis

NanoOK can be run from the command line as follows:

```
nanook analyse -s SampleDir -r referencename.fasta -passonly
```

where:

- `-s` or `-sample` specifies the sample name (ie. same as directory name).
- `-r` or `-reference` specifies the name of the reference file to use.
- ‘‘-passonly‘‘ tells NanoOK only to process the ‘pass’ directory. You can leave this out to analyse both pass and fail, or even specify a -failonly parameter if you just want to analyse the ‘fail’ reads .
- `-aligner` specifies the aligner (default ‘last’). Valid options are the same as for `nanook align`.
- ‘‘-2donly‘‘ will generate a report that contains only 2D data.
- `-bitmaps` will generate PNG format graphs instead of the default PDF format. This can result in faster rendering of PDFs for reports with lots of reads.

This will generate a LaTeX file (with a .tex extension) and a corresponding PDF within a latex subdirectory of the run directory. The naming of the latex subdirectory depends on the aligner and options used - e.g. latex_last_passonly for passonly alignments with LAST. This naming is designed so that you can generate multiple reports with different alignment tools or options.

4.7 Comparison reports

NanoOK comparison let you compare NanoOK analyses for multiple runs. This enables comparison of, for example, chemistry versions, software versions, alignment tools. The comparison option can be run as follows:

```
nanook compare -l samples.txt -o outdir -type 2D
```

where:

- `-l or -samplelist` specifies a list of samples to compare (see below for format).
- ``-o`` or ``-outputdir`` specifies an output directory to write analyses, graphs and report to.
- `-type` specifies the type of data to compare - either 2D, Template or Complement.

This will generate a LaTeX file and a PDF file within a latex subdirectory of the output directory. The sample list file is a two column tab-separated file as follows:

```
SampleDir      SampleName       dirname1         sample_1         dirname2         sample_2
```

The SampleDIr column is the same name you would specify to the -s parameter of extract/align/analyse. The SampleName column is the display name that will be used in graphs.

4.8 Multi-threading

You can control the maximum number of threads used by nanook by specifying the `-t` or `-numthreads` parameter.

4.9 Barcoding

As of NanoOK 1.15, barcoding directory structures should be auto-detected.

4.10 1D data

To avoid creating 2D and Complement directories when running with 1D data, specify the `-templateonly` option.

4.11 Using BWA-MEM for alignments

You will need to index your reference with BWA:

```
bwa index referencename.fasta
```

When running `nanook align` and `nanook analyse`, make sure you specify the `-aligner bwa` option.

4.12 Using BLASR for alignments

You do not need to index your reference separately with BLASR.

When running `nanook align` and `nanook analyse`, make sure you specify the `-aligner blasr` option.

4.13 Using marginAlign for alignments

marginAlign works from FASTQ files, so you will need to extract these with the `-q` flag to `nanook extract`:

```
nanook extract -s <sample> -q
```

References do not need to be indexed with marginAlign.

When running `nanook align` and `nanook analyse`, make sure you specify the `-a marginalign` option.

4.14 Changing default aligner parameters

You can use the -alignerparams option to change the default tuning parameters used for the aligners. To use, you must enclose the parameters in speech marks, for example:

```
nanook align -s SampleDir -r referencename.fasta -alignerparams "-s 2 -T 0 -Q 0 -a 1"
```

The table below shows the default parameters used by NanoOK for the supported aligners:

Aligner	Parameters
LAST	"-s 2 -T 0 -Q 0 -a 1"
BWA MEM	"-x ont2d"
BLASR	""
marginAlign	""
GraphMap	""

4.15 nanook_split_reads

If you don't have individual read files, but they are merged into a single FASTA/Q, NanoOK currently cannot process them. However, you can use `nanook_split_reads.pl` to split them into separate files, for example:

```
nanook_split_reads.pl -i input.fasta -o outputdir
```

Additionally, you will need to place your files within the directory structure expected by NanoOK, e.g.:

```
mkdir -p fasta/pass/2D
nanook_split_reads.pl -i all_2D.fasta -o fasta/pass/2D
```

If at all possible, use NanoOK to do the extraction as well - you are far less likely to run into problems with the align and analyse steps.

CHAPTER 5

NanoOK Report Explanation

5.1 Pass and fail counts

This section simply shows the count of Template, Complement and 2D reads classified in the "pass" and "fail" directories by the base caller.

5.2 Read lengths

This section includes a table of read length statistics for Template, Complement and 2D reads, as well as read length histograms.

5.3 Template/Complement/2D alignments

One section for each read type. A summary table gives the number of reads of each type, the number (and percentage) with alignments and the number without alignments. There then follows a table with a line for each reference, summarising the number of reads that align to that reference, coverage and longest perfect kmer (longest stretch of perfectly mapping sequence).

5.4 Reference error analysis

For each reference, there is an error analysis section.

The first two rows in the table relate to query identity, excluding indels:

- **Overall identity (minus indels)** - the mean percentage of perfectly matching bases in the read set. Or specifically 100 * perfectly matching bases in read set / total bases in reads.
- **Aligned identity (minus indels)** - the mean percentage of perfectly matching bases in alignments. Or specifically 100 * perfectly matching bases / (perfectly matching bases in alignment + substituted bases in alignment).

The next four rows in the table are based on mean values per 100 bases of alignment, including indels - and so the four values in each column should add up to 100 (give or take a small rounding error).

- **Identical bases per 100 aligned bases** - the mean number of identical bases per 100 bases of aligned sequence.
- **Inserted bases per 100 aligned bases** - the mean number of inserted bases per 100 bases of aligned sequence.
- **Deleted bases per 100 aligned bases** - the mean number of deleted bases per 100 bases of aligned sequence.
- **Substitutions per 100 aligned bases** - the mean number of substituted bases per 100 bases of aligned sequence.

Then there are two rows with mean insertion and deletion sizes.

Finally, there are histograms of insertion and deletion size for the three types of read.

5.5 Reference read identity

This section contains the following graphs, one for each read type:

- Histogram of read identity - showing the count of reads vs identity.
- Percent identical bases in query vs Length scatter plot.
- Alignment identity vs percentage of query sequence aligned scatter plot.

It is important to understand what we mean by alignment identity and read identity:

- **Read identity** is defined as the percentage of bases in the read which are perfectly matched to bases in the references - or 100 * matched bases / read length.
- **Alignment identity** is defined as the percentage of bases in the alignment string that are perfect matches.

5.5.1 Example

```
Reference: TGACACTA--TGCCTAGTTAGCTA-GC
Read:      TG--ACTATTCagCTAcTTAG--AGGCTGTGCTAC
```

In the above example, substitutions are shown in lower case for clarity.

- The read is 30 bases long, but only the first 24 have been included in the alignment.
- The alignment is 28 bases long.
- 16 bases match perfectly between the read and the reference.
- So the read identity is 100 * 16 / 30 = 53%.
- The alignment identity is 100 * 16 / 28 = 57%

5.6 Reference perfect kmers

By perfect kmer, we mean stretches of perfectly aligned sequence without any kind of error (substitution or indel).

- Cumulative perfect kmers - percentage of reads with a stretch of perfect sequence of at least this (x-axis) size.
- Best perfect kmer - histogram of percentage of reads for which this (x-axis) is the best stretch of perfect sequence.
- Scatter plot of longest perfect sequence vs read length.

5.7 Reference coverage

This section contains coverage plots for Template, Complement and 2D data, as well as a % GC plot for the reference.

5.8 Reference 5-mer analysis

Plots showing abundance of 5-mers in reads vs. references for Template, Complement and 2D data.

5.9 All reference 21mer analysis

Scatter plots showing the number of perfect 21mers vs read length for each read type.

5.10 All reference susbtitutions

A substitution table is provided for each read type - Template, Complement, 2D, showing the percentage of all base transition (e.g. a reference C that becomes an A).

5.11 Error motif analysis

NanoOK stores all 3-mers, 4-mers and 5-mers that occur before a substitution or indel. The error motif tables show the Top 10 most common and Bottom 10 least common kmers for each error type. The logo images are calculated based on the top 10 or bottom 10 kmers respectively.

CHAPTER 6

How NanoOK works

6.1 How NanoOK deals with alignments

If running with multiple reference sequences, a single query sequence may produce 1 or more alignments to 1 or more references. NanoOK adopts the following approach to assign reads to references:

1. Sort alignments in order of score. The read belongs to the reference with highest score.
2. Then merge any other alignments that align to the same reference as the highest scoring alignment, in order of score.
3. Any sections of these subsequent alignments that overlap with already merged alignments are discarded.

Where the highest score is shared by two or more identically scoring alignments, NanoOK choses one of them at random. **This can result in very slight changes in alignment figures reported**. If you wish deterministic behaviour, specify the `-deterministic` parameter.

CHAPTER 7

Alignment parser API

7.1 Overview

You can relatively easily add support to NanoOK for new parsers. The steps involved are:

- Implement the AlignmentFileParser interface - see below and look at the examples LastParser, BlasrParser, BWAParser, MarginAlignParser.
- Add a 'case' statement for the new aligner in the getParser method of NanoOKOptions.

7.2 The AlignmentFileParser interface

Your parser class will need to implement the following interface:

- **getProgramID** - returns a textual ID string for this aligner which should be lower case - e.g. "last". This is used as the command line option and also the directory name.
- **getAlignmentFileExtension** - returns the file extension of alignments, including . character - e.g. ".sam".
- **getReadFormat** - returns either NanoOKOptions.FASTA or NanoOKOptions.FASTQ to indicate the preferred input format.
- **setAlignmentParams** - passes through command line alignment parameters.
- **getRunCommand** - return a command line instruction to run the aligner.
- **parseFile** - parse an alignment file.
- **outputToStdout** - return true if the aligner only outputs to stdout and not to a file.
- **getHighestScoringSet** - return the highest scoring set of alignments (ie. highest scoring reference.
- **checkForIndex** - check presence of index files to warn before running alignments.

As you will see, the alignment parsers that come with NanoOK are very simple, with the harder work of parsing held within the SAMParser and MAFParser classes which they inherit from.

CHAPTER 8

NanoOK data files

While analysing alignments, NanoOK will write a number of tab delimited files to the 'analysis' subdirectory. These are used for graph plotting through R, but you may wish to use them in other applications or your own custom analyses. There are a number of global data files, plus subdirectories for each reference.

In the analysis directory are the global data files:

- **length_summary.txt**
  - Column 1: Read type - Template, Complement or 2D
  - Column 2: Number of reads
  - Column 3: Mean read length
  - Column 4: Longest read length
  - Column 5: Shortest read length
  - Column 6: N50 length
  - Column 7: Number of reads covered by N50
  - Column 8: N90 length
  - Column 9: Number of reads covered by N90
- **all_summary.txt** - summary of number of reads and number of alignments
- **all_**[2D|Template|Complement]_**alignment_summary.txt** - summary of number of reads aligning to each reference for Template, Complement and 2D reads.
- **all_[2D|Template|Complement]_lengths.txt** - for each read of each type:
  - Column 1: read ID
  - Column 2: length of read
- **all_**[2D|Template|Complenent]_**kmers.txt** - for each read of each type:
  - Column 1: read ID
  - Column 2: length of read

    - Column 3: number of perfect 15mers
    - Column 4: number of perfect 17mers
    - Column 5: number of perfect 19mers
    - Column 6: number of perfect 21mers
    - Column 7: number of perfect 23mers
    - Column 8: number of perfect 25mers
- **all_[2D|Template|Complenent]_substitutions_percent.txt** - base substitution table.
- all**_[2D|Template|Complenent]_[deletion|insertion|substitution]_[n]mer_motifs.txt** - deletion/insertion/substitution kmer motifs:
    - Column 1: kmer
    - Column 2: percentage this kmer occurs before error

Within 'analysis', there will be a subdirectory for each reference. In each reference subdirectory is:

- **reference_[2D|Template|Complement]_alignments.txt** - multi-column files of read-by-read alignment data for each read type. Includes IDs, start and end positions of alignment, bases covered, longest perfect kmer, mean perfect kmer etc. Header line provides details.
- **reference_[2D|Template|Complement]_all_perfect_kmers.txt**
    - Column 1: kmer size
    - Column 2: number of perfect kmers of size across all reads
- **reference_[2D|Template|Complement]_best_perfect_kmers.txt**
    - Column 1: kmer size
    - Column 2: number of reads with best perfect kmer of size
    - Column 3: percentage of reads with best perfect kmer of size
- **reference_[2D|Template|Complement]_cumulative_perfect_kmers.txt**
    - Column 1: kmer size
    - Column 2: number of reads with best perfect kmer of size or greater
    - Column 3: percentage of reads with best perfect kmer of size or greater
- **reference_[2D|Template|Complement]_coverage.txt**
    - Column 1: position on reference
    - Column 2: mean coverage in bin
- **reference_[2D|Template|Complement]_deletions.txt**
    - Column 1: deletion size
    - Column 2: percentage of deletions that are this size
- **reference_[2D|Template|Complement]_insertions.txt**
    - Column 1: insertion size
    - Column 2: percentage of insertions that are this size
- **reference_gc.txt**
    - Column 1: position

  - Column 2: mean GC percentage for bin

- **reference_[2D|Template|Complement]_kmers.txt**
  - Column 1: kmer (5-mer)
  - Column 2: Number of times kmer occurs in reference
  - Column 3: Percentage of total kmers in reference represented by the kmer
  - Column 4: Number of times kmer occurs in the reads
  - Column 5: Percentage of total kmers in reads represented by the kmer

CHAPTER 9

Common errors

Error: unable to find any alignments to process

- Have you run the align stage? If not, run it.
- If you have run the align stage, have the alignments worked? Have a look inside sample/last/pass/2D or appropriate. If there are files there, but they are 0 bytes, see below.

All my LAST alignment files are empty

- Have you indexed your reference file?

lastal: invalid option – 'o' when aligning

- This means your version of LAST is too old. Older versions do not support the -o option to output to a file and only output to the screen.
- Solution: install latest version of LAST.

I don't get a PDF file in the latex directory

- If there is a .tex file, then something went wrong converting the LaTeX to PDF - probably missing LaTeX packages.
- Have a look at the .log file inside the latex directory. You will likely see an error such as: ' ' ! LaTeX Error: File *multirow.sty' not found. '*
- In this instance, you need to install the multirow package.

java.lang.OutOfMemoryError: Java heap space

- Try editing the line in the nanook file of the bin directory: JAVA_ARGS="-Xmx2048m"
- By default, this sets a maximum Java memory size of 2048 Mb - try increasing this according to what your memory you have available.

CHAPTER 10

NanoOK tutorial

1. Follow the installation instructions to install NanoOK on your system. Docker and a VM are also options.
2. Download the example E. coli dataset from http://opendata.earlham.ac.uk/nanook/ and uncompress:

```
tar -xvf nanook_ecoli_500.tar.gz
```

3. Change into the directory:

```
cd nanook_ecoli_500
```

4. Index the reference with LAST:

```
cd references
lastdb -Q 0 ecoli_dh10b_cs ecoli_dh10b_cs.fasta
cd ..
```

5. Extract FASTA:

```
nanook extract -s N79596_dh10b_8kb_11022015
```

6. Perform alignments:

```
nanook align -s N79596_dh10b_8kb_11022015 -r references/ecoli_dh10b_cs.fasta
```

7. Generate a report:

```
nanook analyse -s N79596_dh10b_8kb_11022015 -r references/ecoli_dh10b_cs.fasta -
↪passonly
```

8. View the PDF file inside `N79596_dh10b_8kb_11022015/latex`

CHAPTER 11

Change history

1.25 (7 June 2017)

- Added GraphMap support.
- Fix for trailing / on -f option.
- Fix for barcoding bug.

1.22 (5 May 2017)

- Fixes for comparison mode.
- Slurmit script added for NanoOK RT.

1.20 (13 Apr 2017)

- Better Albacore support.
- New -minquality option for filtering pass/fail reads.

1.17 (30 Mar 2017)

- Detection of albacore directory structure.

1.15 (17 Mar 2017)

- Auto-detection of directory structure - barcodes, batch_ etc.

1.14 (14 Mar 2017)

- Updates to support MinKNOW 1.4.2 directory structure.
- Fixed bug in R graph plotting.
- Better error checking in R scripts.
- Option to merge reads into single file.

0.95 (2 Nov 2016)

- Fixed issues with 1D report generation.
- Added warnings about .sizes file.

- New real-time watcher option for BAMBI project. Currently, this is not general purpose, but will be enabled in future release.
- Created new Dockerfile and re-built Docker images.

0.79 (7 Oct 2016)

- Added support for barcoded runs.

0.76 (7 Sep 2016)

- Fixed issue with badly formatted reference files.
- Fixed issue with grid.edit in R.
- More descriptive error messages.
- Option for 1D only processing for new rapid kit.
- Template only and complement only options.
- Updated help text.

0.72 (13 May 2016)

- New option to store original FAST5 path in FASTA output file (for nanopolish).
- Fixed issue with alignerparams not being passed through.
- Enabled extraction of 2D only reads.
- Better detection of old/new style directory structure.
- Various bug fixes.

0.62 (3 Dec 2015)

- Fixed extract bug that had been introduced with previous version.

0.61 (27 Nov 2015)

- Had to roll back from using HDF5 library due to cross-platform JNI issues.
- Otherwise, all functionality as of 0.60, including use of NANOOK_DIR.

0.60 (26 Nov 2015)

- Added support for Metrichor changes to FAST5 output format.
- Added support for multiple analyses in 1 file (i.e. /Analyses/Basecall_2D_XXX). New option -basecallindex to support it, but default behaviour is latest (highest numbered analysis).
- Moved from using HDF5 command line tool to using HDF5 Java library.
- Replaced the NANOOK_SCRIPT_DIR environment variable with a NANOOK_DIR one and slightly changed installation process.

CHAPTER 12

NanoOK RT

12.1 Introduction

NanoOK RT functionality is included within the core NanoOK software. However, NanoOK repoRTer is available as a separate tool.

The RT functionality should be considered alpha software. As such, it only supports the SLURM scheduler, via the supplied slurmit script. We plan to provide support for other schedulers in the near future.

12.2 Running NanoOK RT

To run NanoOK RT, you must specify the 'rt' option to nanook and specify a process file, for example:

```
nanook rt -t 8 -process processfile.txt -log log.txt -timeout 10000 -templateonly
```

where:

- `-t` specifies the number of execution threads to use.
- `-process` specifies the name of a process file (see below).
- `-log` optionally specifies a log file.
- `-timeout` sets the time after which NanoOK will exit if it hasn't seen a new read to process.
- `-templateonly` specifies only process template reads.

12.3 Process files

Process files define the actions that NanoOK RT will carry out for each read. An example is shown below:

```
Sample:BAMBI\_P8\_2D\_Local\_070317 Extract:fasta
Fast5Dir:/tgac/workarea/group-si/BAMBI\_Pt4/BAMBI\_P8\_2D\_Local\_070317/basecalled
ReadsPerBlast:500
Blast:bacteria,blastn,/tgac/references/databases/blast/nt\_04022017/bacteria\_human\_
↪28022016,8G,tgac-medium
Blast:card,blastn,/tgac/workarea/group-si/BAMBI\_Pt1/CARD\_1.1.1\_Download\_17Oct16/
↪nucleotide\_fasta\_protein\_homolog\_model.fasta,8G,TempProject4
Blast:nt,blastn,/tgac/references/databases/blast/nt\_04022017/nt,16G,TempProject4
```

The line beginning ‘Sample:’ defines the sample directory - this can be relative from the current directory (this case) or absolute (beginning with a ‘/’).

The next section defines read extraction. The Extract line asks for FASTA extraction and the Fast5Dir specifies the location of the fast5 files (like the -f option of standard NanoOK).

Note: NanoOK RT looks at the fast5 directory structure in order to work out the origin of the basecalled .fast5 files (Metrichor, MinKNOW, Albacore), so you must either create this structure before running NanoOK RT, or only run NanoOK RT after some reads have been output.

The final section defines BLAST processes. The ReadsPerBlast defines the chunk size. Then each Blast process consists of the following:

```
Blast:identifier,blast_tool,/path/to/database,SLURM_memory_requirements,SLURM_
↪partition_name
```

CHAPTER 13

NanoOK Reporter

13.1 Introduction

NanoOK Reporter is a tool for real-time analysis of data generated by NanoOK RT. In the following text, we will guide you through analysis of data generated for the BAMBI project.

Please download NanoOK Reporter from https://github.com/richardmleggett/NanoOKReporter

Please download the BAMBI dataset from http://opendata.earlham.ac.uk/bambi/BAMBI_P8_2D_Local_070317.tar.gz

13.2 Running NanoOK Reporter

To clone from GitHub and run, type:

```
git clone https://github.com/richardmleggett/NanoOKReporter.git
cd NanoOKReporter
java -jar dist/NanoOKReporter.jar
```

13.3 Loading the BAMBI data

- Untar the BAMBI dataset:

  ```
  tar -xvzf BAMBI_P8_2D_Local_070317.tar.gz
  ```

- This is a NanoOK sample directory containing just the BLAST subdirectories - it is these that NanoOK Reporter uses.
- With the tar file uncompressed, go back to NanoOK Reporter and click the “Choose…” button next to the “Sample directory” field and browse to find the BAMBI_P8_2D_Local_070317 sample directory that you just untarred.

- Then select the ‘Template’ radio button and the ‘Pass’ radio button and click on “Load CARD” - this will load the CARD chunk data into NanoOK. You can follow it’s progress at the bottom of the window. When it has finished loading, click on the CARD tab in the set of tabs (“Stats/NT/Bacteria/CARD”).
- You can move the slider to move back in time through the data.
- You can click on “Load NT” to load the NT data. This will take slightly longer than for the CARD data.

13.4 Running with live sequencing

When running with live sequencing, you can force the latest data to be imported by clicking on “Load CARD” or “Load NT” again. This will not reload chunk files that have already been analysed.

NanoOK (pronounced na-nook) is a tool for extraction, alignment and analysis of Nanopore reads. NanoOK will extract reads as FASTA or FASTQ files, align them (with a choice of alignment tools), then generate a comprehensive multi-page PDF report containing yield, accuracy and quality analysis. Along the way, it generates plain text files which can be used for further analysis, as well as graphs suitable for inclusion in presentations and papers.

NanoOK has a number of dependencies - Perl, LaTeX, R and an alignment tool - which means it works best on Linux and Mac OS platforms.

Further information:

- To find out how to install NanoOK, see the *Download and installation page*.
- For further information on NanoOK RT, see *these comments*.
- To find out how to run NanoOK, see the *Running NanoOK page* or the *NanoOK tutorial page*.
- Source code is on GitHub.
- Here’s some information about the other Nanook.

CHAPTER 14

Paper

Leggett RM, Heavens D, Caccamo M, Clark MD, Davey RP (2016). NanoOK: multi-reference alignment analysis of nanopore sequencing data, quality and error profiles. Bioinformatics 32(1):142–144.

CHAPTER 15

Talks and posters

- Richard Leggett presented a poster at AGBT 2016 - here it is.
- Richard Leggett spoke at Genome Science 2015 - here are the slides.
- Richard Leggett spoke at the London Calling Nanopore conference - here are his slides.
- Robert Davey presented a poster at AGBT 2015 - here is the PDF.

CHAPTER 16

Follow us

You can follow NanoOK updates on twitter @NanoOK_Software.

Or if you would like to be on a NanoOK mailing list to receive information about updates, please email richard.leggett@earlham.ac.uk.