Supplementary data: NanoSpring: reference-free lossless compression of nanopore sequencing reads using an approximate assembly approach

Qingxi Meng, Shubham Chandak, Yifan Zhu, Tsachy Weissman
Stanford University

GitHub: `https://github.com/qm2/NanoSpring`

Contents

1 Datasets

In this section, we provide instructions for obtaining the datasets used for evaluating NanoSpring and the other tools. While NanoSpring supports gzipped input FASTQ during compression, we decompressed to the FASTQ file for a fair comparison with the other tools. The Zymo dataset is from the R10.3 pore while the other datasets are from the R9.4.1 pore.

1.1 *S. aureus*

The *S. aureus* dataset was obtained from the work in Wick *et al.* (2019) where it was used for evaluating basecalling tools. We downloaded the fast5 dataset with the raw signal and basecalled it with three basecaller modes for Guppy 5.0.7. We also calculated the mean error rate using alignment. Note that we use the basecall with the *hac* mode as the dataset for the main results (*sa* dataset).

Downloading the fast5 files and the reference sequence

```
mkdir Staphylococcus_aureus_CAS38_02/
cd Staphylococcus_aureus_CAS38_02/
mkdir fast5/ && cd fast5/
wget -O Staphylococcus_aureus_CAS38_02_fast5s.tar.gz \
https://bridges.monash.edu/ndownloader/files/14260568
tar -xzvf Staphylococcus_aureus_CAS38_02_fast5s.tar.gz
cd ../
wget -O Staphylococcus_aureus_CAS38_02_reference.fasta.gz \
https://bridges.monash.edu/ndownloader/files/14260241
gunzip Staphylococcus_aureus_CAS38_02_reference.fasta.gz
rm fast5/Staphylococcus_aureus_CAS38_02_fast5s.tar.gz
cd ../
```

Basecalling

```
wget https://mirror.oxfordnanoportal.com/software/analysis/ont-guppy_5.0.7_linux64.tar.gz
tar -xzvf ont-guppy_5.0.7_linux64.tar.gz
rm ont-guppy_5.0.7_linux64.tar.gz
for config in "fast" "hac" "sup"; do
    ./ont-guppy/bin/guppy_basecaller -i Staphylococcus_aureus_CAS38_02/fast5/ \
    -s tmp_out_$config/ -c dna_r9.4.1_450bps_$config.cfg -x cuda:all
    cat tmp_out_$config/pass/*.fastq tmp_out_$config/fail/*.fastq >\
    Staphylococcus_aureus_CAS38_02/basecall_$config.fastq
    rm -r tmp_out_$config/
done
```

Alignment and error calculation

We use minimap2 (Li, 2018) and Samtools (`http://www.htslib.org/`) for calculating the mean error rates for the three basecaller modes.

```
for config in "fast" "hac" "sup"; do
    minimap2 -ax map-ont \
    Staphylococcus_aureus_CAS38_02/Staphylococcus_aureus_CAS38_02_reference.fasta \
    Staphylococcus_aureus_CAS38_02/basecall_$config.fastq -t 20 | samtools stats - > \
    Staphylococcus_aureus_CAS38_02/basecall_$config.stats.txt
done
```

We use the `error rate` entry in the generated statistics file as the mean error rate.

1.2 Zymo

The Zymo dataset (Nicholls *et al.*, 2019) is a metagenomic dataset and we use data from the R10.3 run described at `https://lomanlab.github.io/mockcommunity/r10.html`.

```
wget https://nanopore.s3.climb.ac.uk/mock/Zymo-GridION-EVEN-3Peaks-R103-merged.fq.gz
```

1.3 Snail

The snail dataset (Sun *et al.*, 2021) was obtained from `https://www.ebi.ac.uk/ena/browser/view/SRR12763791`.

```
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR127/091/SRR12763791/SRR12763791_1.fastq.gz
```

According to the reference, the dataset was basecalled using Guppy 3.6.0 (high accuracy mode).

1.4 Banana

The banana dataset (Belser *et al.*, 2021) was obtained from `https://www.ebi.ac.uk/ena/browser/view/ERX5238183`.

```
wget ftp://ftp.sra.ebi.ac.uk/vol1/fastq/ERR545/008/ERR5455028/ERR5455028.fastq.gz
```

According to the reference, the dataset was basecalled using Guppy 4.0.1.

The original 177x dataset was used as the *banana* dataset in the main results. In addition we obtained several subsampled versions for studying the impact of coverage on the performance of NanoSpring. We used seqtk (`https://github.com/lh3/seqtk`) for generating the subsampled dataset. The command used is

```
seqtk sample -s100 ERR5455028.fastq.gz $fraction_reads > ERR5455028_$fraction_reads.fastq
```

where the variable `fraction_reads` was set to 0.0705, 0.141, 0.282 and 0.565 to get subsampled datasets of coverage approximately 12x, 25x, 50x and 100x, respectively.

1.5 Human: NA12878

The human NA12878 dataset (Jain *et al.*, 2018) was obtained from `https://github.com/nanopore-wgs-consortium/NA12878/blob/master/Genome.md`, where we use the latest rel6 version basecalled with Guppy 2.3.8.

```
wget http://s3.amazonaws.com/nanopore-human-wgs/rel6/rel_6.fastq.gz
```

1.6 Human: GM24385

The human GM24385 dataset was obtained from `https://labs.epi2me.io/gm24385_2020.09/` and is part of a series of high quality open source datasets released by Oxford Nanopore Technologies (ONT) at `https://labs.epi2me.io/dataindex`. We use part of the dataset from September 2020 basecalled using Guppy 4.0.11 (high accuracy mode), downloaded using the following command.

```
aws s3 cp --no-sign-request \
s3://ont-open-data/gm24385_2020.09/analysis/r9.4.1/\
20200914_1357_1-E11-H11_PAF27462_d3c9678e/guppy_v4.0.11_r9.4.1_hac_prom/basecalls.fastq.gz\
 .
```

1.7 Human: CHM13

The human CHM13 dataset (Nurk *et al.*, 2021) is part of the telomere-to-telomere (T2T) consortium attempting to sequence and assemble the human genome without any gaps. The dataset was obtained from `https://github.com/marbl/CHM13`, where we use the latest rel7 version basecalled with Bonito 0.3.1.

```
wget https://s3-us-west-2.amazonaws.com/human-pangenomics/T2T/CHM13/nanopore/rel7/rel7.fastq.gz
```

The original 126x dataset was used to obtain several subsampled versions for studying the impact of coverage on the performance of NanoSpring. We used seqtk (`https://github.com/lh3/seqtk`) for generating the subsampled dataset. The command used is

```
seqtk sample -s100 rel7.fastq.gz $fraction_reads > rel7_$fraction_reads.fastq
```

where the variable `fraction_reads` was set to 0.099, 0.198, 0.397 and 0.595 to get subsampled datasets of coverage approximately 12x, 23x, 46x and 69x, respectively. The 23x and 46x datasets correspond to the *hs3* and *hs4* datasets in the main result.

1.8 N50 calculation

The script for calculating the N50 read length for a FASTQ file is available at `https://github.com/qm2/NanoSpring/util/sctipts/get_fastq_stats.sh`.
The command used to calculate the N50 for `file.fastq` is

```
cat file.fastq | sed -n '2~4p' | awk '{ print length }' | ./get_fastq_stats.sh
```

The command used to calculate the N50 for `file.fastq.gz` (gzipped FASTQ file) is

```
zcat file.fastq.gz | sed -n '2~4p' | awk '{ print length }' | ./get_fastq_stats.sh
```

2 Installing and running tools

In this section, we provide instructions for running the various tools used in the experiments. The scripts used for running the main experiments and the parameter evaluation are available at `https://github.com/qm2/NanoSpring/tree/master/util/scripts`. The logs for the experiments are available at `https://github.com/qm2/NanoSpring/tree/master/logs`. Further information on installing the tools in various environments can be found on their respective GitHub pages.

2.1 NanoSpring

2.1.1 Download and install

Commit:
`https://github.com/qm2/NanoSpring/commit/9e9db9fa984727e55b4d8ca8a47d844eebcdffa7`

```
git clone --recursive https://github.com/qm2/NanoSpring.git
cd NanoSpring
mkdir build
cd build
cmake ..
make -j
```

2.1.2 Usage

General usage:

```
Allowed options:
  -h [ --help ]                   produce help message
  -c [ --compress ]               compress
  -d [ --decompress ]             decompress
  -i [ --input-file ] arg         input file name
  -o [ --output-file ] arg        output file name
  -t [ --num-threads ] arg (=20) number of threads (default 20)
  -k [ --kmer ] arg (=23)         kmer size for the minhash (default 23)
  -n [ --num-hash ] arg (=60)     number of hash functions for minhash (default
                                  60)
  --overlap-sketch-thr arg (=6)   the overlap sketch threshold for minhash
                                  (default 6)
  --minimap-k arg (=20)           kmer size for the minimap2 (default 20)
  --minimap-w arg (=50)           window size for the minimap2 (default 50)
  --max-chain-iter arg (=400)     the max number of partial chains during
                                  chaining for minimap2 (default 400)
  --edge-thr arg (=4000000)       the max number of edges allowed in a consensus
                                  graph (default 4000000)
  -w [ --working-dir ] arg (=.)   directory to create temporary files (default
                                  current directory)
```

For compressing `file.fastq` using default 20 threads (used for the main experiments):

```
./NanoSpring -c -i file.fastq -o file.NanoSpring
```

For compressing `file.fastq.gz` (gzipped fastq file) using default 20 threads:

```
./NanoSpring -c -i file.fastq.gz -o file.NanoSpring
```

Decompressing the `file.NanoSpring` with default 20 threads to `file.reads` (used for the main experiments):

```
./NanoSpring -d -i file.NanoSpring -o file.reads
```

More examples are provided on the GitHub README.

2.2 pigz

2.2.1 Download and install

```
wget https://zlib.net/pigz/pigz-2.4.tar.gz
tar -xzvf pigz-2.4.tar.gz
cd pigz-2.4
make
```

2.2.2 Usage

Compression with 20 threads:

```
./pigz-2.4/pigz -k -p 20 file.fastq
```

Decompression with 20 threads:

```
./pigz-2.4/unpigz -k -p 20 file.fastq.gz
```

2.3 ENANO

2.3.1 Download and install

Commit:
`https://github.com/guilledufort/EnanoFASTQ/commit/d6119dbccd19485c8222ec6d72516114401d1a08`

```
git clone https://github.com/guilledufort/EnanoFASTQ.git
cd EnanoFASTQ/enano
make
```

2.3.2 Usage

Compression with 20 threads:

```
./enano -t 20 file.fastq file.enano
```

Decompression with 20 threads:

```
./enano -d -t 20 file.enano file_decompressed.fastq
```

2.4 RENANO

We compared NanoSpring to RENANO for two datasets. Since RENANO is a reference-based compressor, we also obtain the reference sequence for these datasets.

2.4.1 Download and install

minimap2 (for alignment):

```
wget https://github.com/lh3/minimap2/archive/v2.18.tar.gz
tar -xzvf v2.17.tar.gz
cd minimap2-2.17
make
cd ../
rm v2.18.tar.gz
```

RENANO (also available on GitHub at `https://github.com/guilledufort/RENANO`):

```
conda config --add channels defaults
conda config --add channels bioconda
conda config --add channels conda-forge
conda install renano=1.1
```

2.4.2 Download reference sequences

sa dataset:

```
wget -O Staphylococcus_aureus_CAS38_02_reference.fasta.gz \
https://bridges.monash.edu/ndownloader/files/14260241
gunzip Staphylococcus_aureus_CAS38_02_reference.fasta.gz
```

hs2 dataset:

```
curl https://ftp.ncbi.nlm.nih.gov/genomes/refseq/vertebrate_mammalian/Homo_sapiens/\
latest_assembly_versions/GCF_000001405.39_GRCh38.p13/\
GCF_000001405.39_GRCh38.p13_genomic.fna.gz -o hss_genome.fna.gz
gunzip hss_genome.fna.gz
```

2.4.3 Usage

Align reads to reference genome with minimap2 using 20 threads:

```
./minimap2-2.18/minimap2 -x map-ont --secondary=no --cs ref.fasta file.fastq -t 20\
> file.paf
```

Compression with 20 threads in mode that allows decompression without reference:

```
renano -t 20 -s ref.fasta file.paf file.fastq file.renano
```

Decompression with 20 threads:

```
renano -d -t 20 file.renano file_decompressed.fastq
```

References

Belser, C. *et al.* (2021). Telomere-to-telomere gapless chromosomes of banana using nanopore sequencing. *bioRxiv*.

Jain, M. *et al.* (2018). Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nature Biotechnology*, **36**(4), 338–345.

Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, **34**(18), 3094–3100.

Nicholls, S. M. *et al.* (2019). Ultra-deep, long-read nanopore sequencing of mock microbial community standards. *GigaScience*, **8**(5). giz043.

Nurk, S. *et al.* (2021). The complete sequence of a human genome. *bioRxiv*.

Sun, J. *et al.* (2021). Benchmarking oxford nanopore read assemblers for high-quality molluscan genomes. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, **376**(1825), 20200160.

Wick, R. R. *et al.* (2019). Performance of neural network basecalling tools for Oxford Nanopore sequencing. *Genome biology*, **20**(1), 1–10.