UMI-linked nanopore consensus sequencing (UMIC-seq) of highly similar gene variants

Paul Jannis Zurek
University of Cambridge https://orcid.org/0000-0002-7049-0813
Philipp Knyphausen
Katharina Neufeld
Ahir Pushpanath
Johnson Matthey
Florian Hollfelder (fh111@cam.ac.uk)
University of Cambridge https://orcid.org/0000-0002-1367-6312

Method Article

Keywords: nanopore sequencing, consensus sequences, amplicon sequencing, directed evolution, protein engineering

Posted Date: November 30th, 2020

DOI: https://doi.org/10.21203/rs.3.pex-1177/v1

Abstract

Here we present a straightforward unique molecular identifier (UMI)-linked nanopore consensus sequencing workflow (UMIC-seq), resulting in cost-effective and accurate long-read sequencing of amplicons. Short random molecular barcodes (i.e. unique molecular identifiers, UMIs) are attached to a pool of gene variants prior to nanopore sequencing to enable reliable clustering and the generation of accurate consensus sequences, even when starting from highly similar gene variants (e.g. a library of point mutants in directed protein evolution) that could not be reliably distinguished in the ordinary nanopore sequencing output.

Introduction

Accurate consensus sequences of nanopore reads are commonly generated for genome assemblies, where every sequencing read represents a unique fragment that overlaps with many others, facilitating stacking for accurate consensus generation[1,2]. By contrast, libraries in directed protein evolution comprise rather closely related members. Here the sequence diversity in randomized gene libraries is typically low: protein variants with a few or even a single point mutation can be functionally different. An erroneous raw third-generation sequencing read cannot reliably be assigned to its template, when multiple template candidates differing only in few point mutations are possible. This inability to assign reads to any one parental molecule therefore renders consensus-based ‘polishing’ from unlinked reads impossible. Similar samples in which individual members differ merely in a few nucleotides can also be found in e.g. immune repertoires, metagenomic 16S amplicons or medical diagnostics, calling for a suitably high-quality sequencing solution.

Here, we introduce an accurate nanopore consensus sequencing workflow based on a bioinformatic sequence link between descendant reads and their origin molecule by introducing random DNA barcodes (i.e. unique molecular identifiers, UMIs) prior to amplification for sequencing. UMIs are random sequence tags most commonly used to retain information on true molecule numbers before PCR amplification. Specifically, the distinct template identity marked by the UMI enables confident identification and quantification of PCR duplicates[3]. This method has been applied to achieve unbiased counting of molecules for expression data in RNA-seq experiments[4,5] and the accurate quantification of rare mutations[6,7]. Here, we leverage UMI-tags to assign erroneous nanopore reads to their origin molecule, facilitating clustering for accurate consensus formation even when starting with a pool of highly similar sequences (e.g. a library of gene variants in protein evolution containing only point mutations) that could not be reliably distinguished in the ordinary nanopore sequencing output.

The UMIs have to be highly diverse to allow faithful assignment of raw, low-accuracy nanopore reads to the corresponding variant. Based on our experience, 50 bp UMIs are chosen to enable high efficiency clustering at comparatively high nanopore sequencing error rates. The UMIs are incorporated using primers in two PCR cycles, with deliberately low amplification so that potential PCR bias is reduced. UMI-variant complexity is constricted by a transformation step into cells, because an exact number of

molecules is represented in the number of transformant colonies, directly combining selective amplification and control over molecule count in a straightforward molecular biology step. Finally, DNA is straightforwardly isolated from individual colonies and sequenced using current standard nanopore amplicon protocols. UMI clustering is performed with a fast, 'greedy' agglomerative algorithm (see the scripts available at https://github.com/fhlab/UMIC-seq)[8] due to the immense number of sequence comparisons that need to be performed, limiting the use of most conventional all-vs-all comparisons for clustering. Potential mutations are identified for each cluster via signal-level analysis with nanopolish[1,9], using the parental gene sequence as a reference.

Reagents

Nuclease-free water (Ambion)

Plasmid DNA (Pool of variants in pASK-IBA63b+)

Acceptor plasmid (pRSFDuet-1)

Primers (Supplementary Table S1)

Q5 High-Fidelity 2X Master Mix (NEB)

Gibson Assembly Master Mix (NEB)

FastDigest *Kpn*I and *Bam*HI

Phosphate Buffered Saline (PBS)

AMPure XP SPRI beads (Beckman Coulter)

Equipment

Library preparation:

Zymo Clean & Concentrator-5 (Zymo Research)

High-efficiency electrocompetent *E. coli* cells (e.g. E. cloni 10G ELITE, Lucigen, #60051-1)

GeneJET Plasmid Miniprep Kit (ThermoFischer Scientific)

SQK-LSK109 sequencing kit and flow cell (Oxford Nanopore Technologies, ONT)

Generation of consensus sequences:

Unix-based operating system: MacOS, Windows Subsystem for Linux, Ubuntu or similar.

Python (version > 3.6)

Python packages

scikit-bio, scikit-allel, biopython

Nanopolish (version > 10.1)

NanoPlot and NanoFilt

Porechop

Porechop can be used to demultiplex reads. As it is no longer officially maintained, demultiplexing was also incorporated into the UMIC-seq scripts.

Procedure

Library preparation:

Any plasmid can be used as template, as the variants are tagged via PCR. In this case, we used pASK-IBA63b+. After UMI-tagging, it is useful to switch to a different acceptor plasmid that has an alternative selection marker to eliminate the transfer of non-tagged variants. We used pRSFDuet-1 with kanamycin resistance for this purpose. To prepare the sequencing input, the tagged variants are excised from the plasmid with restriction sites that were introduced on the primers. In this protocol, the PCR annealing temperature is given according to the primers used (see below) and the PCR extension time is given according to the gene length in this study (~1.1 kb). Both parameters can be adjusted to suit other experiments.

Primers used here:

Name Sequence

• F_BC1 **ACCATCATCACCACAGCCAGGATCC GATAC AAGAAAGTTGTCGGTGTCTTTGTG**

CTAGAAATAATTTTGTTTAACTTTAAGAAGGAGATATACCATG

• F_BC2 **ACCATCATCACCACAGCCAGGATCC GATAC TCGATTCCGTTTGTAGTCGTCTGT**

CTAGAAATAATTTTGTTTAACTTTAAGAAGGAGATATACCATG

• F_BC3 **ACCATCATCACCACAGCCAGGATCC GATAC GAGTCTTGTGTCCCAGTTACCAGG**

CTAGAAATAATTTTGTTTAACTTTAAGAAGGAGATATACCATG

• R_UMI **GTTTCTTTACCAGACTCGAGGGTACC GATAC NNNNNNNNNNNNNNNNNNNNNNNNN GATAC NNNNNNNNNNNNNNNNNNNNNNNNN GCTCCAAGCGCTCTCGAG**

• F_gibson **ACCATCATCACCACAGCCAG**

• R_gibson **GTTTCTTTACCAGACTCGAGGGTAC**

All steps are performed in parallel for each experiment-specific reaction until sequencing. In this case, the pools of variants from three rounds of evolution are treated in parallel with experiment-specific barcodes.

1) Attaching the UMI and an experiment-specific barcode to the pooled variants

• Prepare UMI tagging reaction in one PCR tube per experiment (e.g. each round of evolution):

Reaction:

25 µl	Q5 High-Fidelity 2X Master Mix
2.5 µl	Forward primer: F_BC
2.5 µl	Reverse primer: R_UMI
500 ng	Template DNA (Variant plasmid pool)
to 50 µl	Nuclease-free water

PCR Program:

98 °C	1 min	
98 °C	10 s	| 2 cycles
60 °C	30 s	|
72 °C	1 min	|
72 °C	5 min	
4 °C	Hold	

• Purify PCR reaction with the Zymo Clean & Concentrator-5 according to the manufacturer's instructions. Elute DNA in 10 µl elution buffer.

• Limited amplification PCR

Reaction:

25 µl	Q5 High-Fidelity 2X Master Mix
2.5 µl	Forward primer: F_gibson
2.5 µl	Reverse primer: R_gibson
250 ng	Template DNA (purified PCR reaction from previous step)
to 50 µl	Nuclease-free water

PCR Program:

98 °C	1 min	
98 °C	10 s	5-15 cycles
60 °C	30 s	
72 °C	1 min	
72 °C	5 min	
4 °C	Hold	

• Purify each reaction via agarose gel extraction and clean-up with the Zymo Clean & Concentrator-5. Elute in 10 µl elution buffer.

2) Sub-cloning to restrict molecule diversity via Gibson assembly and transformation.

• Prepare acceptor plasmid via digestion

Reaction:

3 µl	10X FastDigest Green Buffer
1 µg	Acceptor plasmid (pRSFDuet-1)
1 µl	FastDigest *Bam*HI

1 µl	FastDigest *Kpn*I
to 30 µl	Nuclease-free water

PCR Program:

37 °C	60 min
80 °C	5 min

• Purify linearized acceptor plasmid via agarose gel extraction and clean-up with the Zymo Clean & Concentrator-5

• Prepare the Gibson assembly of the tagged variants from step 1 and the acceptor plasmid for each reaction.

Reaction:

100 ng	Linearized acceptor plasmid
100 ng	Tagged variants (~3-fold molar excess over plasmid)
10 µl	Gibson Assembly Master Mix (2X)
to 20 µl	Nuclease-free water

PCR Program:

50 °C	60 min

• Purify with Zymo Clean & Concentrator-5. Elute in 10 µl nuclease-free water.

• Transform 25 µl electrocompetent E. cloni 10G ELITE with 5 µl of the purified Gibson assembly for each reaction. Plate a dilution series.

3) Isolation of linear, amplified and tagged molecules

• Select number of colonies:

This determines the number of final consensus sequences that will be generated. Here, 500-1000 variants were obtained during each round of evolution. To achieve at least 3-fold oversampling, 3000 colonies were selected.

• Scrape the selected number of colonies off the transformation plates with the help of ~2 ml PBS, pellet the cells and perform plasmid isolation (GeneJET Miniprep Kit).

• Digest the isolated plasmids to receive the tagged variant sequences for each reaction

Reaction:

5 µl	10X FastDigest Green Buffer
3 µg	Plasmid isolation from the previous step
2 µl	FastDigest *Bam*HI
2 µl	FastDigest *Kpn*I
to 50 µl	Nuclease-free water

PCR Program:

37 °C	60 min
80 °C	5 min

• Purify the tagged insert via agarose gel extraction and clean-up with the Zymo Clean & Concentrator-5. Elute in 100 µl elution buffer.

• Purify again with AMPure XP SPRI beads. Elute in 25 µl nuclease-free water.

4) Nanopore sequencing

• Pool purified tagged fragments equimolarly per barcoded-reaction to a final content of 200 fmol DNA in 47 µl nuclease-free water.

• Follow instructions for further library preparation and sequencing with ONT's Ligation Sequencing Kit (SQK-LSK109).

• Run the sequencing until 100X* expected coverage is reached. Here, 3000 colonies were selected in three barcoded experiments, which thus corresponds to 9000 unique fragments after isolation. The tagged fragment length is ~1250 bp, thus sequencing is run until at least 9000 * 100 = 900,000 reads or 9000 * 100 * 1250 ≈ 1.1 Gb are sequenced.

Generation of accurate consensus sequences:

Tools made for this study are made available as Python scripts, hosted on GitHub (https://github.com/fhlab/UMIC-seq). Additionally, an example is provided on GitHub, showing the core steps based on a small dataset in detail. Some settings, such as the length filtering, have to be adjusted from experiment to experiment. Here, the expected read length was ~1250 bp.

Requirements:

- Raw and basecalled nanopore reads *(fast5 / fastq)*
- List of barcodes for demultiplexing *(fasta)*
- Probe sequence for UMI extraction *(fasta)*

1) Quality control and read filtering

• Plot sequencing metrics for quality control: Visualize read quality and length distribution to find suitable thresholds for filtering.

$NanoPlot -summary sequencing_summary.txt -o nanoplot_folder --drop_outliers

• Filter sequences based on length and average read quality: In our case, the quality filtering was stringent – approximately 25% of all reads were discarded during filtering.

$cat merged.fastq | NanoFilt -q 10 --length 1200 --maxlength 1300 > filtered.fastq

2) Demultiplex reads belonging to different experiments

• Demultiplex the experiment specific barcodes, here to separate the three rounds of evolution. The barcode sequences were taken from the PCR Barcoding Expansion Pack 1-96 (EXP-PBC096).

$porechop -i filtered.fastq -b demultiplexed_folder --barcode_threshold 65 --barcode_diff 10 --end_threshold 100 --no_split -v 1 --untrimmed

Alternative: As porechop is officially unsupported as of October 2018, demultiplexing was integrated into a helper script. A list of barcodes for demultiplexing must be supplied *(fasta)*, as well as the basecalled reads *(fastq)* and alignment thresholds.

$python UMIC-seq_helper.py demultiplex --barcodes barcodes.fasta --input filtered.fastq --threshs 22 29 --output demultiplexed_folder

3) Extraction and clustering of UMIs

• UMI extraction: To extract the UMI, a probe sequence must be supplied (*fasta*). This probe sequence should be a short constant region (e.g. 50 bp) next to the UMI, i.e. a part of the reference gene. Extraction will copy the sequence next to the probe into a new file. The following steps are run individually for each of the demultiplexed experiments.

$python UMIC-seq.py UMIextract –input demultiplexed.fastq –probe probe.fasta –umi_loc down –umi_len 65 –output ExtractedUMIs.fasta

• UMI clustering threshold approximation: To estimate a suitable clustering threshold, a threshold approximation can be run. The thresholds to test can be specified, e.g. as –steps 20 70 10, which will sample thresholds from 20 to 70 with a step width of 10. The script will output similarity histograms and a clustering test plot. In the similarity histogram, a randomly chosen UMI is aligned to all other UMI sequences and the resulting alignment scores are plotted as a histogram. Here, a lot of low alignment scores as well as few sequences with high alignment scores (a potential cluster) are expected. A suitable clustering threshold separates the two. The clustering test plot will provide clustering information (cluster size and similarity of sequences in a cluster) for clusters with each of the sampled thresholds. A suitable threshold is found when both metrics begin to saturate. In our experience, the clustering threshold will be close to the length of the UMI.

$python UMIC-seq.py clustertest –input ExtractedUMIs.fasta –steps 20 70 10 –output UMIclustertest

• UMI clustering: Full clustering is performed with the previously determined threshold. The size threshold specifies the minimal size of a cluster to be written to file.

$python UMIC-seq.py clusterfull –input ExtractedUMIs.fasta –reads demultiplexed.fastq –aln_thresh 60 –size_thresh 50 –output cluster_folder

3) Sequence polishing and calling of mutations with Nanopolish

• Each cluster file can now be used for signal level analysis with Nanopolish. Nanopolish will be run on each cluster file to generate the mutations of that cluster (*.vcf*). See the nanopolish documentation for details.

#Example nanopolish commands

$nanopolish index -d read_folder -s sequencing_summary.txt clusterfile.fasta

$minimap2 -ax map-ont reference.fasta clusterfile.fasta | samtools sort -o clustermap.bam -T map.tmp

$samtools index clustermap.bam

$nanopolish variants –consensus -q dam,dcm -o mutations.vcf -r clusterfile.fasta -b clustermap.bam -g reference.fasta

To automate the execution of nanopolish on each cluster file, the UMIC-seq_helper script can generate a shell script performing the command on all cluster files. A list of cluster filenames needs to be provided, it can be generated e.g. with *$ls *.fasta > clusterfiles_list.txt*. The commands that should be run on each cluster are provided as a text file, with a keyword to be replaced by each cluster file name, e.g. clusterfile.fasta replaced by keyword in the example commands above.

#Generate the shell script

$python UMIC-seq_helper.py generateSH –input clusterfiles_list.txt –output run_polish.sh –arguments arguments.txt –keyword keyword

#Execute the generated shell script

$chmod +x run_polish.sh

$./run_polish.sh

• To combine all mutations in all the individual vcf files into full length sequences in one multi-fasta file, the helper script can be run again. Additionally, filtering of mutations by support fraction can be performed.

#A list of filenames needs to be provided

*$ls *.vcf > filenames.txt*

#Run the helper script to extract and filter the mutations

$python UMIC-seq_helper.py vcf2fasta –input filenames.txt –output consensus_sequences.fasta –min_suppfrac 0.6 –reference reference.fasta

• Alternative: Run a quicker, but potentially less accurate consensus generation with medaka instead of nanopolish. Run the consensus mode on each cluster file.

$medaka_consensus -i clusterfile.fasta -d reference.fasta -o outdir -t threads -m r941_min_high

Troubleshooting

Time Taken

Anticipated Results

References

1. Loman, N. J., Quick, J. & Simpson, J. T. A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nat. Methods* **12**, 733–735 (2015).

2. Vaser, R., Sovic, I., Nagarajan, N. & Sikic, M. Fast and accurate de novo genome assembly from long uncorrected reads. *Genome Res.* gr.214270.116 (2017) doi:10.1101/gr.214270.116.

3. McCloskey, M. L., Stöger, R., Hansen, R. S. & Laird, C. D. Encoding PCR Products with Batch-stamps and Barcodes. *Biochem. Genet.* **45**, 761–767 (2007).

4. Kivioja, T. *et al.* Counting absolute numbers of molecules using unique molecular identifiers. *Nat. Methods* **9**, 72–74 (2012).

5. Islam, S. *et al.* Quantitative single-cell RNA-seq with unique molecular identifiers. *Nat. Methods* **11**, 163–166 (2014).

6. Kinde, I., Wu, J., Papadopoulos, N., Kinzler, K. W. & Vogelstein, B. Detection and quantification of rare mutations with massively parallel sequencing. *Proc. Natl. Acad. Sci.* **108**, 9530–9535 (2011).

7. Schmitt, M. W. *et al.* Detection of ultra-rare mutations by next-generation sequencing. *Proc. Natl. Acad. Sci. U. S. A.* **109**, 14508–14513 (2012).

8. Zurek, P. J., Knyphausen, P., Neufeld, K., Pushpanath, A. & Hollfelder, F. UMI-linked consensus sequencing enables phylogenetic analysis of directed evolution. *Github Fhlab UMIC-Seq* (2020) doi:10.5281/zenodo.4055319.

9. Quick, J. *et al.* Real-time, portable genome sequencing for Ebola surveillance. *Nature* **530**, 228–232 (2016).