Chapter 9

Assembly and Annotation of Kinetoplastid and Diplonemid Mitochondrial Genomes

Evgeny S. Gerasimov, Matus Valach, Gertraud Burger, Kristína Záhonová, and Vyacheslav Yurchenko

Abstract

This chapter outlines the assembly and annotation of mitochondrial genomes and transcriptomes of kinetoplastids and diplonemids using high-throughput sequencing data. The mitochondrial genomes of these protists are highly atypical, characterized by multipartition and extensive RNA editing. These complexities pose significant challenges for genome assembly, gene prediction, and transcript reconstruction. Here, we describe analytical strategies developed in our laboratories that address these issues.

Key words Kinetoplastea, Trypanosomatids, Diplonemids, Mitochondrion, RNA editing, Next-generation sequencing

1 Introduction

As all euglenozoans, kinetoplastids and diplonemids share in the mitochondrial (mt) DNA alternative genetic code, in which UGA stop codon codes for tryptophan [1]. Although they are sister clades in phylogenetic trees, their mitochondrial genomes differ drastically from each other, calling for dedicated approaches in bioinformatics genome analysis.

1.1 Kinetoplast Mitochondrial Genomes and Kinetoplast (K) DNA

The apomorphy for all kinetoplastid protists (Euglenozoa: Kinetoplastea) is the presence of a large mass of mtDNA called kinetoplast (k)DNA [2, 3], which is (1) compacted and localized in the vicinity of the flagellar pocket (= eukinetoplast) as in trypanosomatids, (2) distributed throughout the mitochondrion in several nearly identical clusters (= polykinetoplast) as in bodonids, or (3) unevenly dispersed as a diffuse mass (= pankinetoplast) as in prokinetoplastids [4–6].

In trypanosomatids, the by-far best studied kinetoplastids because of their medical and economical importance [7], kDNA

Paul A. M. Michels et al. (eds.), *Euglenozoa: Methods and Protocols, Volume 1*, Methods in Molecular Biology, vol. 3013,
https://doi.org/10.1007/978-1-0716-5142-1_9,

is comprised of circular double-stranded DNA molecules of two functional types: maxicircles and minicircles [8, 9]. Of note, the situation in other kinetoplastid groups is less clear and warrants further investigation, as it has been recently speculated that parabodonids, for example, contain not circular but linear versions of minicircles [10]. The typical maxicircle of Trypanosomatidae is 20–50 kb in size and present in dozens of identical copies. It encodes mitochondrial genes and is, therefore, an analog of mtDNA of other organisms, but not without peculiarities. Maxicircle can be divided into two regions. The coding region (CR), which is tightly packed with compactly organized and typically very syntenic genes among trypanosomatid species, is typically defined as the sequence that extends from the 12S ribosomal RNA (*rRNA*) gene to the *ND5* gene [11–13]. These genes contain extremely short intergenic spacers or may even overlap in some species. For naming convention of kinetoplastids' mtDNA genes, *see* **Note 1**.

The remainder of the maxicircle sequence is termed the divergent region (DR) and comprises arrays of repeats of various length and composition, its sequence assembly and analysis being challenging [14]. The DR cannot be directly assembled from short sequencing reads, therefore most of the maxicircle sequences deposited in nucleotide databases contain only the CR with short flanks. However, recent advances in sequencing technologies (especially long-read platforms) have resolved this issue.

The bizarre molecular feature of trypanosomatid kinetoplast is U-insertion/deletion RNA editing that modifies many primary maxicircle transcripts [15–17]. Edited genes are termed cryptogenes. Some transcripts are edited so extensively that over a half of their translatable sequence is created by inserted Us. Examples of extensively (pan-) edited genes are *RPS12*, *ND3*, *ND8*, *ND9*, *G3*, and *G4* [18], but some genes tend to change the editing degree over the course of evolution [19]. Insertions and deletions carried out by the RNA editing machinery are guided by special small guide RNAs (gRNAs), which are typically encoded by minicircles [20–22]. Minicircles are typically small (ranging between 600 and 2000 bp in most species, but with some notable exceptions [9, 23]) and present in thousands of copies in the kDNA. The common characteristic of these molecules is the high heterogeneity within sequence classes, although minicircles often have characteristic common sequence motifs called conserved sequence blocks (CSBs) [24–28]. Sequence and copy-number heterogeneity of minicircles impose algorithmic difficulties on the kDNA repertoire assembly because many routinely used methods rely on uniformity of coverage and do not work properly when a common sequence block is present in different sequences.

U-insertion/deletion RNA editing also complicates the analysis of the mitochondrial transcriptome. From the algorithmic point of view, the edited RNA is very different from its genomic cryptogene sequence so that "general purpose" read alignment algorithms fail to build accurate read alignments. Another problem is that a variety of partially and alternatively edited transcripts (or isoforms) are found, which are very difficult to distinguish [29]. This led to the development of several dedicated analysis toolkits (i.e., T-Aligner [30] or TREAT [31]), which implement editing-aware algorithms and address problems that arise from extreme transcriptome complexity and the presence of multitude erroneous editing products. Hereafter, we will focus on usage of T-Aligner for RNA editing analysis of trypanosomatids.

1.2 Diplonemid Mitochondrial Genomes

The mtDNAs of diplonemids are also multi-partite but organized very differently compared to kinetoplastids. The mitochondrial genome of the type species, *Paradiplonema papillatum* (originally referred to as *Diplonema papillatum* [32]), comprises 81 distinct circular molecules that are 6 and 7 kb long. Both circle classes mostly contain repeats with ~80% sequence identity across the groups [33]. In addition, the copy numbers of the various classes can differ by more than 100-fold [34]. As with kinetoplastid mini-circles discussed above, the assembly of diplonemid mtDNA from shotgun and short-read data is virtually impossible. It can only be achieved in certain species on rare occasions when circle classes are unique and lack longer repeats [35]. The diplonemid mtDNA circles also include coding sequence, albeit extraordinarily short, as diplonemid mitochondrial genes are fragmented in an unprecedented way. For example, in *P. papillatum*, mitochondrial genes are split into up to 11 pieces, with sizes between ~40 and 540 bp, each encoded on a separate molecule. These gene pieces (termed modules) are transcribed separately and then assembled by RNA ligation into mature mRNAs and rRNAs. Due to their small size, identifying coding sequence in mtDNA can be difficult, but the major hurdle of gene identification is the massive RNA editing operating in diplonemid mitochondria. *P. papillatum* has two types of RNA editing. One is substitutions of As by Is and Cs by Us, i.e., deaminations, with densely packed clusters of up to 50% edited positions over a 100 bp stretch. The other type is U-appendage editing, where up to ~50 Us are added to the 3′-end of module transcripts, prior to their assemblage into mature RNAs [36]. Gene identification, therefore, requires transcriptome information. As mentioned in the case of kinetoplastids, the sometimes-considerable DNA-RNA differences complicate the alignment of RNA reads to genome sequence, and, thus, the assignment of transcripts to their coding regions.

The above-given examples of genome multipartition, gene fragmentation, and RNA editing come from *P. papillatum*, a species with relatively moderate deviations in mitochondrial genome architecture, gene structure, and encryption. Other diplonemids, especially hemistasiids, show far more extreme features in all three aspects, further complicating in silico analyses of their mtDNA [35].

The methodological approaches described below will be divided into two parts because of the fundamental differences between kinetoplastid and diplonemid mtDNA. Yet, we believe that it is beneficial to present them within one chapter devoted to the analysis of mt genome of Euglenozoa.

2 Materials

2.1 Software

BBTools (e.g., BBDuk, BBMap) [37], BEDtools [38], BLAST+ [39], Bowtie2 [40], BWA [41, 42], exonerate [43], FastQC [44], fastp [45], Flye [46], FreeBayes [47], Geneious [48], MAFFT [49], SAMtools [50], SPAdes [51], T-Aligner [30], Trinity [52], VCFtools [53]. Kinetoplastids' RNA editing pipeline is better run under native Linux with python 3.10+ and packages "numpy", "matplotlib", "pandas", and "scipy" installed.

2.2 Sequencing Data

For the analysis of kinetoplastid mtDNA, short sequencing Illumina paired-end HiSeq 150 nt or MiSeq 250 nt reads are convenient. Isolation of DNA material from the mitochondria-enriched fraction is highly recommended, but total DNA sequenced to a median nuclear genome coverage of 200× (assuming a size of 20–40 Mb) is usually sufficient to obtain a quasi-complete mitochondrial genome sequence. For species with large (longer than 2 kb) minicircles or for full-length maxicircle assembly, long sequencing reads are necessary. For RNA sequencing, Illumina paired-end HiSeq 150 nt reads or MiSeq 250 nt reads, poly(A)-enriched reads or reads from mitochondria-enriched fraction are preferred. Total RNA sequencing without enrichment and selection usually contains only a few edited reads, which limits the downstream analyses.

For the analysis of diplonemid mtDNA, we recommend Illumina MiSeq paired-end reads, but for systematic full-length chromosome assembly, long sequencing reads (Oxford Nanopore Technologies or alternatively, PacBio) are required. For RNA sequencing, we recommend Illumina HiSeq paired-end strand-specific reads produced from poly-A RNA fraction, but total RNA libraries may be necessary to identify mitochondrial rRNA genes.

3 Methods

3.1 Analysis of mtDNA of Kinetoplastids

3.1.1 Quality Control

1. Check read base quality and adapter presence with FastQC:

```
mkdir fastqc_reports
fastqc -t 4 -o fastqc_reports <read_files_*fastq>
```

2. Trim low-quality bases and technical adapter sequences with fastp.

```
fastp -a <adapter_seq> -l 100 -q 25 -o <trimmed_R1.fastq> -O
<trimmed_R2.fastq> -i <totrim_R1.fastq> -I <totrim_R2.fastq>
```

Use adapter sequence **<adapter_seq>** from the sequencing kit's manual. Alternatively, do not use **-a** option, instead, rely on fastp's feature of automatic adapter detection or provide adapters file with **--adapter_fasta <file_with_adapters.fasta>** option.

3. Estimate the relative abundance of mtDNA reads by counting CSB3 sequence (common for most minicircles, but can slightly diverge from consensus in different species) with:

```
grep -cP "GGGGTTGGTGTA" <read_R1.fastq>
```

Since DNA sequencing is not strand-specific, you can probe only forward sequence to get a rough estimate.

3.1.2 Maxicircle Assembly

1. Assemble a mitochondrial maxicircle using one of the following strategies depending on what type of input data you have.

   1.1. Assemble the coding region of maxicircle using short sequencing (e.g., Illumina) reads with SPAdes:

```
spades.py -t 4 --pe1-1 <read_R1.fastq> --pe1-2 <read_R2.fastq>
-o <spades_out> --disable-gzip-output [--isolate]
```

   Adding the **--isolate** option can usually give slightly better results. Full maxicircle assembly is only possible with long PacBio or Oxford Nanopore sequencing reads (see next step).

   1.2. Assemble full-size maxicircle from long PacBio sequencing reads:

```
flye --genome-size <gs> --pacbio-raw <pacbio_reads.fastq> --
out-dir <flye_out> --threads 8
```

   or long Oxford Nanopore reads:

```
flye --genome-size <gs> --nanopore <nanopore_reads.fastq> --
out-dir <flye_out> --threads 8
```

   See **Note 2** on how to set the **<gs>** parameter.

   1.3. Perform hybrid assembly of maxicircle using short and long sequencing reads in single command:

```
spades.py --threads 10 --disable-gzip-output --pacbio <pac-
bio_reads.fastq> --pe1-1 <illumina_R1.fastq> --pe1-2 <illumi-
na_R2.fastq> -o <spades_hybrid>
```

Further steps will be performed using the output of **step 1.1** but can be easily adjusted for two other cases.

2. Locate maxicircle contig using 12S and ND5 homologues from other species (*Leishmania tarentolae* or *Trypanosoma brucei*). Firstly, prepare nucleotide blast database from assembly scaffolds file:

```
makeblastdb -in <spades_out>/scaffolds.fasta -out <spades_-
out>/scaffolds.db -dbtype nucl
```

3. Search for marker genes in assembly scaffolds with BLASTn and figure out the scaffold entry name, which corresponds to a potential maxicircle (usually both searches hit the same contig/scaffold):

```
blastn -query <ND5_sequences.fasta> -db <spades_out>/scaf-
folds.db -evalue 1e-6 -word_size 5 -max_target_seqs 5 -outfmt
6
blastn -query <12S_sequences.fasta> -db <spades_out>/scaf-
folds.db -evalue 1e-6 -word_size 5 -max_target_seqs 5 -outfmt
6
```

4. Extract the assembled maxicircle scaffold:

```
samtools faidx <spades_out>/scaffolds.fasta <NODE_XXX_ful-
l_name> > <maxicircle.fasta>
```

5. (If needed) Reverse-complement the maxicircle sequence to put 12S rRNA gene on the forward strand:

```
seqtk seq -r <maxicircle.fasta> > <maxicircle_final.fasta>
```

3.1.3 Maxicircle Annotation

Use SnapGene or similar software to annotate the ~18 mitochondrial genes. SnapGene allows viewing the sequence and in silico translations of six frames.

1. Find open reading frames (ORFs) corresponding to never-edited genes. These genes are encoded as continuous ORFs so they can be detected after in silico translation of all possible ORFs longer than 100 amino acids using “Mold, protozoan” genetic code table (usually equivalent of NCBI’s translation table #4). At this step, *ND1*, *ND4*, *ND5*, *CO1*, *CO3*, *MURF1*, and *MURF2* will be annotated for most trypanosomatid genomes.

2. Detect rRNA gene (*12S* and *9S*) boundaries by best local alignment with known *12S* and *9S* sequences and translate detected coordinates on the maxicircle:

```
exonerate --bestn 1 <maxicircle_final.fasta> <12S.fasta>
```

Exonerate will output the best alignment, giving the target and query ranges included into the alignment. Usually, the middle part of ribosomal RNA will be included in the alignment. Expand this region by adding the unaligned nucleotides of the query to get full gene length.

3. Inspect 5′-truncated ORFs of partially edited genes. Use synteny with known sequences of *L. tarentolae* (GenBank accession M10126.1) or *T. brucei* (GenBank accession M94286.1) as guidance. Most of these genes (e.g., *ND7*, *A6*, *CYb*) will have short edited domains on the 5′. Mark these incomplete ORFs with flanks of ~100 nt, which must include the edited domain.
4. Check and mark G-rich regions corresponding to heavily edited cryptogenes (usually, *ND3*, *ND8*, *ND9*, *G3*, *G4*, and *RPS12*). The typical size of the marked region will be ~250 nt.
5. Prepare a tabulated (gff/gtf-like) file <**annotation.gff**>, that will include exact boundaries and strand information ("+" or "–") of genes annotated in **steps 1** and **2**, and the approximate boundaries of cryptogenes annotated in **steps 3** and **4**.
6. Get annotated sequences in FASTA format:

```
bedtools getfasta -s -fi <maxicircle_final.fasta> -bed <annotation.gff> > <annotated_references.fasta>
```

This file will be used to reconstruct edited mRNAs and refine gene boundaries with T-Aligner.

3.1.4 Minicircles Assembly

1. Assemble an initial set of DNA contigs with SPAdes:

```
spades.py -t 4 --pe1-1 <read_R1.fastq> --pe1-2 <read_R2.fastq> -o <spades_minicircles> --disable-gzip-output --isolate -k 25,41,81,101,127
```

2. Assemble an initial set of DNA contigs with megahit:

```
megahit -1 <read_R1.fastq> -2 <read_R2.fastq> -t 8 -o <megahit_minicircles>
```

3. Extract minicircle contigs using marker sequence. Usually, the best is to use CSB3 sequence as marker:

```
cat <spades_minicircles>/scaffolds.fasta <megahit_minicir-
cles>/final.contigs.fa | awk 'BEGIN{data=""}{if(substr
($1,1,1) == ">") {print data; data=$1 "\n"} else {data=data
$1;}}END{print data}' | grep -B 1 -P "GGGGTTGGTGTA|TACAC-
CAACCCC" | grep -v -P "--" - > <possible_minicircles.fasta>
```

4. Filter <**possible_minicircles.fasta**> file by expected sequence length and/or by the presence of other marker minicircle sequences (for example, a typical size range of 600–800 and CSB1/2 block sequences can be used in case of *Leishmania* spp.).
5. (Alternative) Follow the KOMICS [54] package tutorial at https://github.com/FreBio/komics. KOMICS is a specialized package for minicircle repertoire assembly from short sequencing reads using megahit as an internal assembler. It also offers the rKOMICS R package for subsequent data analysis [55], which works particularly well with *Leishmania*-like monomeric minicircles (*see* **Note 3** about the assembly of large minicircles).

3.1.5 Transcriptome Assembly

1. Join all sequencing reads in a single file:

```
cat <read_R1.fastq> <read_R2.fastq> > <joined_rnaseq.fastq>
```

If you have two or more RNAseq libraries, join all of them.

2. Extract reference cryptogene sequence with flanks using coordinates obtained in Subheading 3.1.3:

```
echo -e"<Maxicircle_contig_name>\t<gene_start-50>\t<gene_end
+50>\t.\t.\t<gene_strand+/->" | bedtools getfasta -s -fi <max-
icircle_final.fasta> -bed - > <cryptogene_ref.fasta>
```

3. Align RNAseq data on the reference sequence with T-Aligner's `alignlib` tool:

```
~/t-aligner/alignlib --in_ref <cryptogene_ref.fasta> --in_lib
<joined_rnaseq.fastq> --out_prefix <t_aligner_out_>
```

The output of this step will include mapped reads in ".fastq" format and in T-Aligner-specific ".taf" format, which can be used to visualize cryptogene coverage and editing states with auxiliary scripts and perform other downstream analyses.

4. Visualize editing states and read coverage depth for the reference cryptogene:

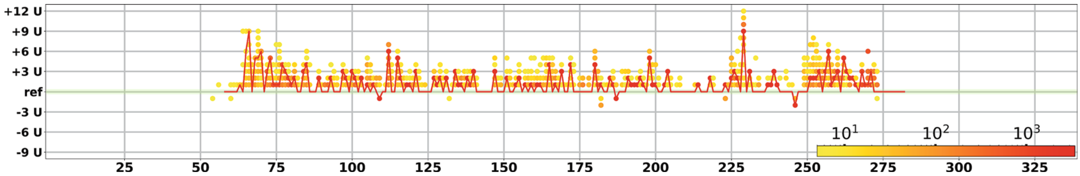

Fig. 1 A dot matrix plot for the *Vickermania spadyakhi ND8* cryptogene produced by T-Aligner's coverage.py script. The dot matrix is a heatmap, where dot intensities reflect read support and dots reflect editing events (termed "editing states"). The unedited cryptogene is drawn by green baseline ($Y = 0$), the editing states connected with U insertion are depicted by dots above this line ($Y > 0$), and deletions of U are shown below the line ($Y < 0$). The Y scale is the number of inserted/deleted Us, the X scale is the number of A/G/C nucleotide of analyzed reference sequence. As editing can modify the number of Us in each position of the reference, the default T-Aligner's approach is to ignore each T in a given DNA reference sequence and operate with a so-called T-less profile; the same approach is good for plotting the results. The solid red line depicts the path that reflects the editing events needed to produce a canonical fully edited *ND8* mRNA

```
python ~/t-aligner/coverage.py --r <cryptogene_ref.fasta> --t
<t_aligner_out_mapped_reads.taf> --o <dotmatrix_plot.png> --g
dots
python ~/t-aligner/coverage.py --r <cryptogene_ref.fasta> --t
<t_aligner_out_mapped_reads.taf> --o <coverage_plot.png> --g
bars
```

These plots give an overall view of the transcription and editing events for the studied cryptogene. Most plotting tools from T-Aligner package rely on ".taf" format mappings. The dot matrix visualization of editing events (Fig. 1) is a powerful feature of the package that gives an overview of all editing events (with their support and interrelations), which is extremely useful for studies of complex transcripts pools generated by RNA editing.

5. Assemble transcripts with arbitrary long ORFs with `findorfs` command:

```
~/t-aligner/findorfs --in_lib <t_aligner_out_mapped_reads.
fastq> --in_ref <cryptogene_ref.fasta> --out_prefix <assemble-
d_orfs_> --orf_tracing_mode extension --orf_min_orf_aa 80 --
orf_filter_esd 10
```

This step will assemble all transcripts for a given cryptogene, including multitude of isoforms and partially edited products. Canonical edited mRNA will be usually assembled too. *See* **Notes 4** and **5** for more details on `findorfs` command and how to locate canonical edited mRNA sequence.

3.1.6 gRNA Annotation

1. Predict gRNA-coding loci by alignment using `find_grna` tool from T-Aligner package and edited mRNA sequence(s) from Subheading 3.1.5.

```
find_grna --mrna <canonical_mRNA.fasta> --grna <minicircles.
fasta> --seed_score 20 --seed_length 16 --length 21 --score
25 --gu 15 --mm 4 --anchor 5 > <fow.out> 2> /dev/null
```

   The command predicts gRNAs by alignment on the given strand of minicircles. Allowed number of mismatches (--**mm**), G:U pairs (--**gu**), minimal alignment length (--**length**), anchoring region of gRNA (--**anchor**), minimal alignment score (--**score**), seed alignment length (--**seed_length**), and score (--**seed_score**) are controlled in this step. *See* **Notes 6** and 7 for more information.

2. Validate alignments with small RNA sequencing data (if available). Trim and merge small RNA sequencing reads:

```
fastp -i <small_rna1.fastq> -I <small_rna2.fastq> -q 25 -l
15 --merge --merged_out <merged.fq> --out1 <unmerged_1.fq> --
out2 <unmerged_2.fq> --thread 10
```

3. Index minicircles assembly:

```
bwa index -p <minicircles.bwa_index> <minicircles.fasta>
```

4. Map trimmed and merged reads onto a minicircle assembly:

```
bwa mem -t 10 <minicircles.bwa_index> <merged.fq> | samtools
sort -@ 4 -o <mapped.bam>
```

5. Process alignments and convert to FASTA format:

```
samtools index <mapped.bam>
samtools view <mapped.bam> | awk '{print ">" $1 "\n" $10; }' >
<mapped.fasta>
```

6. Cut polyA/U tails from the mapped reads:

```
for i in {0..10}; do echo Trimming gRNA ends cycle ${i}; sed -i
-e 's/^AAAA/A/1' <mapped.fasta>; sed -i -e 's/TTTT$/T/1'
<mapped.fasta>; done
```

7. Cluster sequences with `cd-hit` to produce clusters, which correspond to gRNAs:

```
cd-hit-est -i <mapped.fasta> -o <clustered.out> -c 0.90 -s
0.80 -d 40 -M 90000 -T 30 -n 8
```

From the clustered output file <**clustered.out**> pick gRNA sequences as clusters with 10+ sequences and length of 20–70 nt.

8. Compare resulting set of gRNAs predicted from alignment (**step 1**) with set of gRNAs obtained from small RNA sequencing data (**step 7**). Note that not all gRNAs predicted by alignment must be necessarily supported by small RNA sequencing for a variety of reasons (e.g., coverage and representation problems of small RNA library).

3.2 Analysis of mtDNA of Diplonemids

3.2.1 Read Processing

1. Check the quality of DNA- and RNA-Seq reads by FastQC as in **step 1** of Subheading 3.1.1.
2. Trim the reads with BBDuk to remove adaptors and low-quality reads. It is not necessary to unpack the read files beforehand. The typical command for trimming of paired-end short reads is:

```
bbduk in1=<forward_1.fastq.gz> in2=<reverse_2.fastq.gz>
out1=<trimmed_1.fastq.gz> out2=<trimmed_2.fastq.gz> ref=a-
dapters.fa usejni=t qtrim=rl trimq=20 ktrim=r k=22 mink=11
hdist=2 tpe tbo t=10 2> <bbduk_report.log>
```

The exact description of parameters can be found in the BBDuk manual.

3. Check the quality of trimmed reads by FastQC using the command from **step 1** of Subheading 3.1.1.

3.2.2 Genome and Transcriptome Assembly

1. Assemble DNA-Seq trimmed reads using SPAdes:

```
spades.py --pe1-1 <trimmed_DNA_1.fastq.gz> --pe1-2 <trimmed_D-
NA_2.fastq.gz> --careful -t 40 -o <spades_output_directory>
```

If using Nanopore or Pacbio data, refer to **step 1.2** of Subheading 3.1.2 (and also *see* **Note 2**).

2. Assemble RNA-Seq trimmed reads by Trinity:

```
Trinity --seqType fq --left <trimmed_RNA_1.fastq.gz> --right
<trimmed_RNA_1.fastq.gz> --output <trinity_output_directory>
--max_memory 100G --CPU 30
```

3.2.3 Identification of Mature Transcripts and Genomic Modules

1. Prepare nucleotide databases from the assembled genome and transcriptome:

```
makeblastdb -in <spades_output_directory>/scaffolds.fasta
-dbtype nucl
makeblastdb -in <trinity_output_directory>/Trinity.fasta
-dbtype nucl
```

2. Identify mature transcripts by BLASTx searches using previously identified diplonemids' proteins as queries against the prepared transcriptomic database:

```
blastx -query <diplonemids_proteins.fasta> -db <trinity_output_directory>/Trinity.fasta -out <transcripts.tsv> -outfmt 6
```

3. Extract identified transcripts from the assembled transcriptome to a separate file (**<identified_transcripts.fasta>**).
4. Translate found transcripts into protein sequences using "Mold, protozoan" genetic code (NCBI's translation table #4).
5. Identify gene modules by BLASTn searches using identified transcripts as queries against the prepared genomic database:

```
blastn -query <identified_transcripts.fasta> -db <spades_output_directory>/scaffolds.fasta -out <modules.tsv> -outfmt 6
```

6. Extract identified gene modules from the assembled genome to a separate file (**<identified_modules.fasta>**).
7. Map identified modules to a corresponding transcript to corroborate module assignments (*see* **Note 8**).

3.2.4 Chromosome Classification

1. Extend the module-containing contigs by read mapping using BBMap or Geneious built-in mapper (*see* **Note 8**).
2. Compare module-containing contigs among each other using BLASTn:

```
blastn -query <identified_modules.fasta> -db <spades_output_directory>/scaffolds.fasta
```

   This identifies cassettes and constant regions. Cassettes represent unique sequences up- and downstream of modules, which are flanked by constant region that exhibit over 90% identity over 100 bp adjacent to the cassettes. Extract cassette sequences into a separate file (**<identified_cassettes.fasta>**).
3. Assign contigs having the same cassette-flanking regions to the same chromosome class.
4. Order classes from the highest to lowest count of chromosomes and name them A, B, etc.
5. Align contigs of one class using MAFFT:

```
mafft --auto --adjustdirection <one_class_contigs.fasta> > <one_class_contigs_aligned.fasta>
```

6. Place cassette's boundaries at positions where sequence identities drop below 90% over 100 bp.

3.2.5 Identification of DNA Polymorphic and RNA Editing Sites

1. Preliminary RNA editing site identification can be achieved by comparing mature transcripts with aligned genomic modules from **step 7** of Subheading 3.2.3. However, this is only possible when the mitochondrial genome (and consequently transcriptome) does not exhibit a high degree of DNA polymorphism, which is particularly prevalent in hemistasiids [34, 35, 56]. For this reason, we recommend performing variant calling on DNA-Seq and RNA-Seq reads to identify DNA-DNA differences and RNA-DNA differences in a systematic manner as detailed in the following.
2. Map trimmed DNA-Seq reads using Geneious built-in mapper or BBMap (*see* **Note 8**) onto mitochondrial genomic contigs containing annotated cassettes (i.e., the file <**identified_cassettes.fasta**>).

   For Geneious, use the following parameters:

```
Save assembly report; Save list of used reads (include mates);
Save contigs; Do not trim; Map multiple best matches to all;
Only map paired reads that both map; Allow at maximum 2% read
gaps; Maximum gap size 4; Word length 48; Index word length 14;
Minimum overlap identity 95%; Allow at maximum 2% mismatches
per read; Maximum ambiguity 4
```

   For BBMap (e.g., 100 bp reads with library insert sizes <700 bp, "very sensitive" preset):

```
java -Xmx4g -cp <path_to_directory> align2.BBMap pairlen=500
rescuedist=1000 maxindel=2 ambiguous=all k=8 saa=f vslow=t
ref=<identified_cassettes.fasta> out=<DNA_reads_to_identi-
fied_cassettes>.sam nodisk in1=<reads1.fastq> in2=<reads2.
fastq> qin=33
```

3. Generate a sorted BAM from the SAM read-alignment file:

```
samtools view -S -b <DNA_reads_to_identified_cassettes.sam> >
<DNA_reads_to_identified_cassettes.bam>
samtools sort -o <DNA_reads_to_identified_cassettes-sorted.
bam> <DNA_reads_to_identified_cassettes.bam>
samtools index <DNA_reads_to_identified_cassettes-sorted.bam>
```

4. Call DNA-DNA differences using FreeBayes. This can be also done within Geneious, which includes a FreeBayes distribution, or using a stand-alone FreeBayes installation:

```
freebayes -f <identified_cassettes.fasta> --ploidy 100 --min-mapping-quality 30 --min-alternate-count 2 --min-alternate-fraction 0.01 --use-duplicate-reads <DNA_reads_to_identified_cassettes-sorted.bam> > <dna_dna_differences.vcf>
```

   Assign as DNA polymorphic sites those that possess at least two different nucleotides in ≥10% of the DNA-Seq reads (*see* also **Note 9**).

5. Map trimmed RNA-Seq reads using the Geneious built-in mapper or BBMap (*see* **Note 8**) onto mitochondrial identified transcripts (*see* **Note 10**).

   When using Geneious:

```
Save assembly report; Save list of used reads (include mates); Save contigs; Do not trim; Map multiple best matches to all; Allow at maximum 10% read gaps; Maximum gap size 4; Word length 20; Index word length 14; Allow at maximum 10% mismatches per read; Maximum ambiguity 8
```

   When using BBMap (e.g., 100 bp reads with library insert sizes <400 bp, "very sensitive" preset):

```
java -Xmx4g -cp /path/to/directory align2.BBMap pairlen=200 rescuedist=400 maxindel=2 ambiguous=all k=8 saa=f vslow=t ref=<identified_transcripts.fasta> out=<RNA_reads_to_identified_transcripts.sam> nodisk in1=<reads1.fastq> in2=<reads2.fastq> qin=33
```

6. Generate a sorted BAM from the SAM read-alignment file:

```
samtools view -S -b <RNA_reads_to_identified_transcripts.sam> > <RNA_reads_to_identified_transcripts.bam>
samtools sort -o <RNA_reads_to_identified_transcripts-sorted.bam> <RNA_reads_to_identified_transcripts.bam>
samtools index <RNA_reads_to_identified_transcripts-sorted.bam>
```

7. Call RNA-DNA differences using FreeBayes:

```
freebayes -f <identified_transcripts.fasta> --ploidy 100 --min-mapping-quality 30 --min-alternate-count 2 --min-alternate-fraction 0.01 --use-duplicate-reads <RNA_reads_to_identified_transcripts-sorted.bam> > <rna_dna_differences.vcf>
```

8. To identify all genuine editing sites, i.e., variant positions present among the RNA-DNA differences, but absent from among the genomic polymorphisms, intersect the VCF files of RNA-DNA and DNA-DNA differences with the tool vcf-isec of the VCFtools suite:

```
vcf-isec -c <rna_dna_differences.vcf> <dna_dna_differences.
vcf> > <rna_editing_sites.vcf>
```

   Assign as the default RNA editing sites those are present in ≥50% of the RNA-Seq reads.

4 Notes

1. For historical reasons, the kinetoplastid mtDNA genes still use slightly different naming than in other eukaryotes although representing homologous sequences. Genes encoding NAD$^+$ subunits of NADH dehydrogenase (complex I) are abbreviated as *ND*, cytochrome *b* (complex III) as *CYb* instead of *cob*, *cox* subunits of cytochrome *c* oxidase (complex IV) as *CO*, and the ATP6 subunit of the ATP synthase (complex V) as *A6*. Moreover, several genes were reported as having unknown function and were named MURF (mitochondrial unassigned reading frame) and CR (C-rich region) [57, 58]. With more advanced HMM analysis and structural modeling, MURF1 was assigned to *nad2* (or ND2 in the kinetoplastid terminology), MURF5 to *rps3*, CR3 to *nad4L* (or G3), CR4 to *nad6* (or G4), and CR5 to *nad3* (or ND3) [59–61].
2. When assembling the genome with `flye`, the tool requires an estimate of genome size, i.e., the expected total assembly size. If you use mitochondrial fraction enrichment, set the `--genome-size` parameter to the expected maxicircle size (e.g., "***35k***" for *Leishmania*). If you use a total-cell DNA library, set this parameter to "***32m***", which is a typical nuclear genome size of trypanosomatid. For diplonemids, the typical values are "***500k***" when using enriched mitochondrial DNA and "***250m***" when starting from total cell DNA.
3. The assembly of large minicircles from species like *Vickermania ingenoplastis* is usually possible only with long sequencing reads. Flye assembler can be used with default settings, though genome length should be set to the value around "***50m***".
4. The key option of the `find_orfs` tool is **--orf_tracing_mode**, which determines the preferrable way of reads overlap. It can take the value "extension" (prefers best extension of ORF, default, usually gives best results), "overlap" (more confident, best read overlap preferred), "editing" (prefers extension with

read sequence that has maximal number of edited sites out of all possible, can work best for pan-edited cryptogenes), or "coverage" (prefers extension passing through most supported editing states). The option --**orf_filter_esd** filters out unedited reads (reads with 10+ edits will be used by example command above), for partially edited cryptogenes (with 5′-short edited domains, such as in the case of *Leishmania* CYb), this option should not be used! Other useful options are --**orf_min_overlap** and --**orf_min_extension**, which control minimal overlap and extension and are set to 10 by default. It is recommended to increase these values to 15 or 20 for longer reads (MiSeq platform) or for very deeply covered genes. Option --**orf_output_orf_alignments true** will output alignment of assembled transcript against cryptogene sequence and generate a separate. fasta file for each assembled transcript.

5. The output of the `find_orfs` tool usually includes various partially edited and alternatively edited mRNA sequences assembled from sequencing data. The priority task is to identify the canonically edited mRNA. The most straightforward approach is to use the predicted translated proteins output file of `find_orfs` tool as a BLAST database (use `makeblastdb` to create it) and a set of known homologous sequences for model species (*L. tarentolae*, *T. brucei*, and others) as queries.
6. Note that `find_grna` models mRNA:gRNA alignments only for a given strand of assembled minicircles. That means that if you expect that both strands of minicircles can encode gRNAs, you should also run the tool with reverse-complemented minicircles assembly file as input.
7. Alignment gRNA:mRNA modelling is controlled by command line options of `find_grna` tool. Allow less mismatches and G:U pairs, and set longer seed with higher seed score to produce more strict alignments, which will reflect longer gRNAs (suits well for *Leishmania*). In contrast, for species with short and divergent gRNAs use short seed and decrease the alignment length and maximal score, which will allow to model alignments with higher mismatch/G:U per nucleotide density.
8. Our best experience has been with the built-in aligner and mapper of the Geneious software. However, since this is not an open-source software, other software (e.g., BBMap [37] or Bowtie2 [40]) may be used to essentially the same effect.
9. Some apparent polymorphisms can arise from mitochondrial sequences that were transferred to the nuclear genome (a.k.a. NUMTs). These can even be expressed from the nuclear genome because many localize to 3′ UTRs, as documented extensively for *P. papillatum* [62]. This confounding effect is

especially important for low-copy-number mitochondrial chromosomes.

10. The identified mitochondrial transcript sequences (**identified_transcripts.fasta**) represent the edited transcriptome reference. While the Geneious built-in mapper is usually able to properly map essentially all RNA-Seq reads whether they originate from fully edited, partially edited, or pre-edited transcripts, other tools (e.g., BBMap, Bowtie2, minimap2) require a virtual sequence representing the pre-edited transcriptome reference, especially when using a short-read technology. This virtual reference is created by concatenating genomic module sequences of individual genes in the same order as in the identified transcripts. Thus, when using Bowtie2 or similar tools, map RNA-Seq reads to both references and then merge the resulting BAM files using the edited transcripts as the default reference. When interested in examining substitution RNA editing intermediates, the most practical approach is to map reads onto a virtual sequence, in which known sites of, e.g., A-to-I and C-to-U editing have been replaced by R and Y, respectively; most tools accept such "degenerate" references, even if read mapping generally will not be as efficient as when using the virtual pre-edited reference or the edited transcriptome reference, especially for editing clusters that are longer than half the read size.

Acknowledgments

The work on kinetoplastids was primarily supported by the European Union's Operational Program "Just Transition" (LERCO CZ.10.03.01/00/22_003/0000003 to V.Y.) and the Czech Ministry of Education, Youth and Sports (INTER-EXCELLENCE-LUASK22033 to V.Y.). The work on diplonemids was supported by the Fonds de Recherche du Québec—Nature et Technologies (FRQNT; grant 2018-PR-206806 to G.B.) and the Natural Sciences and Engineering Research Council of Canada (NSERC; grants RGPIN-2014–05286 and RGPIN-2019–04024 to G.B.). Computational resources were provided by the e-INFRA CZ project (ID:90254) supported by the Czech Ministry of Education, Youth and Sports.

References

1. Záhonová K, Lax G, Leonard G et al (2021) Single-cell genomics unveils a canonical origin of the diverse mitochondrial genomes of euglenozoan. BMC Biol 19:103
2. Vickerman K (1976) Comparative cell biology of the kinetoplastid flagellates. In: Vickerman K, Preston TM (eds) Biology of Kinetoplastida, vol 1. Academic, London, pp 35–130

3. Kostygov AY, Karnkowska A, Votýpka J et al (2021) Euglenozoa: taxonomy, diversity and ecology, symbioses and viruses. Open Biol 11: 200407
4. Lukeš J, Guilbride DL, Votýpka J et al (2002) Kinetoplast DNA network: evolution of an improbable structure. Eukaryot Cell 1(4): 495–502
5. d'Avila-Levy CM, Boucinha C, Kostygov A et al (2015) Exploring the environmental diversity of kinetoplastid flagellates in the high-throughput DNA sequencing era. Mem Inst Oswaldo Cruz 110(8):956–965
6. Moreira D, López-García P, Vickerman K (2004) An updated view of kinetoplastid phylogeny using environmental sequences and a closer outgroup: proposal for a new classification of the class Kinetoplastea. Int J Syst Evol Microbiol 54:1861–1875
7. Stuart K, Brun R, Croft S et al (2008) Kinetoplastids: related protozoan pathogens, different diseases. J Clin Invest 118(4):1301–1310
8. Jensen RE, Englund PT (2012) Network news: the replication of kinetoplast DNA. Ann Rev Microbiol 66:473–491
9. Yurchenko V, Hobza R, Benada O et al (1999) *Trypanosoma avium*: large minicircles in the kinetoplast DNA. Exp Parasitol 92(3): 215–218
10. Gerasimov ES, Afonin DA, Korzhavina OA et al (2022) Mitochondrial RNA editing in *Trypanoplasma borreli*: new tools, new revelations. Comput Struct Biotechnol J 20:6388–6402
11. Camacho E, Rastrojo A, Sanchiz A et al (2019) *Leishmania* mitochondrial genomes: maxicircle structure and heterogeneity of minicircles. Genes 10(10):758
12. Afonin DA, Gerasimov ES, Škodová-Sveráková I et al (2024) *Blastocrithidia nonstop* mitochondrial genome and its expression are remarkably insulated from nuclear codon reassignment. Nucleic Acids Res 52:3870–3885
13. Berná L, Greif G, Pita S et al (2021) Maxicircle architecture and evolutionary insights into *Trypanosoma cruzi* complex. PLoS Negl Trop Dis 15(8):e0009719
14. Gerasimov ES, Zamyatnina KA, Matveeva NS et al (2020) Common structural patterns in the maxicircle divergent region of Trypanosomatidae. Pathogens 9(2):100
15. Benne R, Van den Burg J, Brakenhoff JP et al (1986) Major transcript of the frameshifted *coxII* gene from trypanosome mitochondria contains four nucleotides that are not encoded in the DNA. Cell 46(6):819–826
16. Read LK, Lukeš J, Hashimi H (2016) Trypanosome RNA editing: the complexity of getting U in and taking U out. Wiley Interdiscip Rev RNA 7(1):33–51
17. Maslov DA, Opperdoes FR, Kostygov AY et al (2019) Recent advances in trypanosomatid research: genome organization, expression, metabolism, taxonomy and evolution. Parasitology 146(1):1–27
18. Koslowsky D, Sun Y, Hindenach J et al (2014) The insect-phase gRNA transcriptome in *Trypanosoma brucei*. Nucleic Acids Res 42(3): 1873–1886
19. Gerasimov ES, Kostygov AY, Yan S et al (2012) From cryptogene to gene? *ND8* editing domain reduction in insect trypanosomatids. Eur J Protistol 48(3):185–193
20. Aphasizheva I, Alfonzo J, Carnes J et al (2020) Lexis and grammar of mitochondrial RNA processing in trypanosomes. Trends Parasitol 36(4):337–355
21. Gerasimov ES, Gasparyan AA, Afonin DA et al (2021) Complete minicircle genome of *Leptomonas pyrrhocoris* reveals sources of its non-canonical mitochondrial RNA editing events. Nucleic Acids Res 49(6):3354–3370
22. Seiwert SD, Stuart K (1994) RNA editing: transfer of genetic information from gRNA to precursor mRNA *in vitro*. Science 266(5182): 114–117
23. Gerasimov ES, Afonin DA, Škodová-Sveráková I et al (2025) Evolutionary divergent kinetoplast genome structure and RNA editing patterns in the trypanosomatid *Vickermania*. Proc Natl Acad Sci USA 122(15):e2426887122
24. Hong M, Simpson L (2003) Genomic organization of *Trypanosoma brucei* kinetoplast DNA minicircles. Protist 154(2):265–279
25. Yurchenko V, Kolesnikov AA (2001) Minicircular kinetoplast DNA of Trypanosomatidae. Mol Biol (Mosk) 35(1):1–10
26. Thomas S, Martinez LL, Westenberger SJ et al (2007) A population study of the minicircles in *Trypanosoma cruzi*: predicting guide RNAs in the absence of empirical RNA editing. BMC Genomics 8:133
27. Cooper S, Wadsworth ES, Ochsenreiter T et al (2019) Assembly and annotation of the mitochondrial minicircle genome of a differentiation-competent strain of *Trypanosoma brucei*. Nucleic Acids Res 47(21): 11304–11325
28. Cooper S, Wadsworth ES, Schnaufer A et al (2022) Organization of minicircle cassettes and guide RNA genes in *Trypanosoma brucei*. RNA 28(7):972–992

29. Zimmer SL, Simpson RM, Read LK (2018) High throughput sequencing revolution reveals conserved fundamentals of U-indel editing. Wiley Interdiscip Rev RNA 9(5): e1487
30. Gerasimov ES, Gasparyan AA, Kaurov I et al (2018) Trypanosomatid mitochondrial RNA editing: dramatically complex transcript repertoires revealed with a dedicated mapping tool. Nucleic Acids Res 46(2):765–781
31. Simpson RM, Bruno AE, Bard JE et al (2016) High-throughput sequencing of partially edited trypanosome mRNAs reveals barriers to editing progression and evidence for alternative editing. RNA 22(5):677–695
32. Tashyreva D, Simpson AGB, Prokopchuk G et al (2022) Diplonemids – a review on "new" flagellates on the oceanic block. Protist 173(2): 125868
33. Vlcek C, Marande W, Teijeiro S et al (2011) Systematically fragmented genes in a multipartite mitochondrial genome. Nucleic Acids Res 39(3):979–988
34. Valach M, Moreira S, Hoffmann S et al (2017) Keeping it complicated: mitochondrial genome plasticity across diplonemids. Sci Rep 7(1): 14166
35. Kaur B, Záhonová K, Valach M et al (2020) Gene fragmentation and RNA editing without borders: eccentric mitochondrial genomes of diplonemids. Nucleic Acids Res 48(5): 2694–2708
36. Moreira S, Valach M, Aoulad-Aissa M et al (2016) Novel modes of RNA editing in mitochondria. Nucleic Acids Res 44(10): 4907–4919
37. Bushnell B, Rood J, Singer E (2017) BBMerge – accurate paired shotgun read merging *via* overlap. PLoS One 12(10):e0185056
38. Quinlan AR (2014) BEDTools: the swiss-army tool for genome feature analysis. Curr Protoc Bioinformatics 47:11.12.11–11.12.34
39. Camacho C, Coulouris G, Avagyan V et al (2009) BLAST+: architecture and applications. BMC Bioinformatics 10:421
40. Langmead B, Salzberg SL (2012) Fast gapped-read alignment with Bowtie 2. Nat Methods 9(4):357–359
41. Li H, Durbin R (2010) Fast and accurate long-read alignment with Burrows-Wheeler transform. Bioinformatics 26(5):589–595
42. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 25(14):1754–1760
43. Slater GS, Birney E (2005) Automated generation of heuristics for biological sequence comparison. BMC Bioinf 6:31
44. Andrews S (2019) FastQC: a quality control tool for high throughput sequence data. http://www.bioinformatics.babraham.ac.uk/projects/fastqc. Accessed 27 Apr 2025
45. Chen S (2023) Ultrafast one-pass FASTQ data preprocessing, quality control, and deduplication using fastp. iMeta 2(2):e107
46. Kolmogorov M, Yuan J, Lin Y, Pevzner PA (2019) Assembly of long, error-prone reads using repeat graphs. Nat Biotechnol 37(5): 540–546
47. Garrison E, Marth G (2012) Haplotype-based variant detection from short-read sequencing. arXiv 1207:3907
48. Kearse M, Moir R, Wilson A et al (2012) Geneious basic: an integrated and extendable desktop software platform for the organization and analysis of sequence data. Bioinformatics 28(12):1647–1649
49. Katoh K, Standley DM (2013) MAFFT multiple sequence alignment software version 7: improvements in performance and usability. Mol Biol Evol 30(4):772–780
50. Danecek P, Bonfield JK, Liddle J et al (2021) Twelve years of SAMtools and BCFtools. Gigascience 10(2):1–4
51. Prjibelski A, Antipov D, Meleshko D et al (2020) Using SPAdes *de novo* assembler. Curr Protoc Bioinformatics 70(1):e102
52. Grabherr MG, Haas BJ, Yassour M et al (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nat Biotechnol 29(7):644–652
53. Danecek P, Auton A, Abecasis G et al (2011) The variant call format and VCFtools. Bioinformatics 27(15):2156–2158
54. Van den Broeck F, Savill NJ, Imamura H et al (2020) Ecological divergence and hybridization of Neotropical *Leishmania* parasites. Proc Natl Acad Sci USA 117(40):25159–25168
55. Geerts M, Schnaufer A, Van den Broeck F (2021) rKOMICS: an R package for processing mitochondrial minicircle assemblies in population-scale genome projects. BMC Bioinf 22(1):468
56. Yabuki A, Tanifuji G, Kusaka C et al (2016) Hyper-eccentric structural genes in the mitochondrial genome of the algal parasite *Hemistasia phaeocysticola*. Genome Biol Evol 8(9): 2870–2878
57. Duarte M, Tomás AM (2014) The mitochondrial complex I of trypanosomatids–an overview of current knowledge. J Bioenerg Biomembr 46(4):299–311
58. Opperdoes FR, Michels PA (2008) Complex I of Trypanosomatidae: does it exist? Trends Parasitol 24(7):310–317

59. Kannan S, Burger G (2008) Unassigned *MURF1* of kinetoplastids codes for NADH dehydrogenase subunit 2. BMC Genomics 9: 455
60. Valach M, Leveille-Kunst A, Gray MW, Burger G (2018) Respiratory chain Complex I of unparalleled divergence in diplonemids. J Biol Chem 293(41):16043–16056
61. Ramrath DJF, Niemann M, Leibundgut M et al (2018) Evolutionary shift toward protein-based architecture in trypanosomal mitochondrial ribosomes. Science 362(6413): eaau7735
62. Valach M, Moreira S, Petitjean C et al (2023) Recent expansion of metabolic versatility in *Diplonema papillatum*, the model species of a highly speciose group of marine eukaryotes. BMC Biol 21(1):99