Published in final edited form as:

Methods Mol Biol. 2017 ; 1562: 211–229. doi:10.1007/978-1-4939-6807-7_14.

In Silico Identification of RNA Modifications from High-Throughput Sequencing Data Using HAMR

Pavel P. Kuksa,

Department of Pathology and Laboratory Medicine, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, USA

Yuk-Yee Leung,

Department of Pathology and Laboratory Medicine, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, USA

Lee E. Vandivier,

Department of Biology, University of Pennsylvania, Philadelphia, PA USA

Zachary Anderson,

Department of Biology, University of Pennsylvania, Philadelphia, PA USA

Brian D. Gregory,

Department of Biology, University of Pennsylvania, Philadelphia, PA USA

Li-San Wang

Department of Pathology and Laboratory Medicine, Perelman School of Medicine, University of Pennsylvania, Philadelphia, PA, USA

Abstract

RNA molecules are often altered post-transcriptionally by the covalent modification of their nucleotides. These modifications are known to modulate the structure, function, and activity of RNAs. When reverse transcribed into cDNA during RNA sequencing library preparation, atypical (modified) ribonucleotides that affect Watson-Crick base pairing will interfere with reverse transcriptase (RT), resulting in cDNA products with mis-incorporated bases or prematurely terminated RNA products. These interactions with RT can therefore be inferred from mismatch patterns in the sequencing reads, and are distinguishable from simple base-calling errors, single-nucleotide polymorphisms (SNPs), or RNA editing sites. Here, we describe a computational protocol for the in silico identification of modified ribonucleotides from RT-based RNA-seq read-out using the *High-throughput Analysis of Modified Ribonucleotides* (HAMR) software. HAMR can identify these modifications transcriptome-wide with single nucleotide resolution, and also differentiate between different types of modifications to predict modification identity. Researchers can use HAMR to identify and characterize RNA modifications using RNA-seq data from a variety of common RT-based sequencing protocols such as Poly(A), total RNA-seq, and small RNA-seq.

Keywords

RNA modification; RNA posttranscriptional modification; RNA covalent modification; Small RNA; Small RNA sequencing; Messenger RNA; RNA sequencing; Machine learning; Classification

1 Introduction

Covalent posttranscriptional modifications of specific nucleotide bases in RNA molecules are known to be highly prevalent and physiologically important [1–6]. RNA modifications play a role in maintaining the structure and stability of RNAs [7–10] and affect their maturation [3, 10], translation and cellular abundance [10, 15]. All known classes of RNA molecules harbor various levels of diverse modifications [1, 8, 9, 11–13]. However, the overall abundance and biological function of these modifications are incompletely understood [10, 14, 15].

Experimental methods for low-throughput detection of some types of RNA modifications are well established [16–19]. One such method is primer extension, which relies on the differential ability of reverse transcriptase to produce cDNAs with base-pair substitutions or premature termination at positions occupied by modified nucleotides [17]. However, because many RNA sequencing library preparation protocols require RNA to cDNA conversion by reverse transcriptase (RT), it is possible to use RNA sequencing data to identify sites of modified nucleotides across many classes of RNAs in a high-throughput and transcriptome-wide fashion by identifying positions within RNAs with significant mismatch rates after alignment. The protocol for in silico identification of RNA modifications presented here uses our previously developed HAMR software [20]. The protocol allows for fast and reliable identification of modified nucleotides at single-nucleotide resolution in all RNA classes transcriptome-wide through the analysis of nucleotide substitutions in high-throughput RNA sequencing datasets. This software provides an important tool for detecting and classifying modified RNA ribonucleotides that modulate RT incorporation, e.g., by affecting Watson-Crick base pairing. The examples of HAMR-detectable modifications include but are not limited to m1A, m3C, m5C, and pseudouridine [1, 8, 10, 13, 14, 20–22]. Modifications such as m6A [23], which do not significantly interfere with RT, will not be detected by HAMR.

In this chapter, we describe how to use HAMR [20] for in silico identification of RNA modifications using RNA sequencing data.

HAMR in a nutshell

The starting point of the HAMR analysis is the mapped sequencing reads in the standard BAM file format (<https://samtools.github.io/hts-specs/SAMv1.pdf>). The HAMR software is fully automated and the user can perform genome-wide analysis of RNA modifications in one command (see Subheading 3.5) using BAM files with the corresponding reference genome sequence. The user can also start from raw sequencing data and turn it into mapped

data (BAM) as summarized in Subheading 3.4 for preparing raw sequencing data for HAMR analysis. Subheading 3.1 additionally describes suggestions for RNA library preparation.

Sample study

To illustrate RNA modification analysis using the HAMR software [20], we also provide a sample study of RNA modifications in human brain using RNA-seq data in Subheading 3.8.

Availability

HAMR is available as a stand-alone command-line pipeline (<https://github.com/wanglab-upenn/HAMR/>) and as a Web-based application (accessible at <http://lisanwanglab.org/hamr/>). The HAMR source code is freely available under the MIT license for academic and nonprofit use.

2 Materials

Always use the recommended version of all software dependencies. We also recommend performing a setup as described (*see* Subheading 3.3) before preprocessing the sequencing data from FASTQ to BAM file (*see* Subheading 3.4) and analyzing it using HAMR (*see* Subheadings 3.5 and 3.6).

2.1 Hardware Requirements

Desktop computer or server (multi-core server with 32GB RAM is recommended when starting from raw sequenced reads in human).

2.2 Software Requirements

1. Linux-based operating system (CentOS, Ubuntu, Debian, etc.).
2. C compiler (g++).
3. Standard POSIX programs (awk, grep, bash) (*see* Note 1).
4. Samtools suite [24] <http://www.htslib.org/download/> (version 1.0 or later).
5. Python (version v2.7. x or above) (*see* Note 2).
6. R program <https://www.r-project.org/> (version 3.1.x or above).
7. Additionally, the following software is required for processing raw reads (*see* Subheading 3.4):
 - a. SRA toolkit (<http://www.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?cmd=show&f=software&m=software&s=software>) when analyzing data from the Sequencing Read Archive.
 - b. Cutadapt [25] <https://code.google.com/p/cutadapt/> (version 1.7 and above) to trim adapter sequences for sequenced reads.

¹.These programs are readily available in popular Linux distributions such as Ubuntu, CentOS, or Debian.

².You can check the version of python by typing `python --version` on the command line.

- c. STAR [26] <https://github.com/alexdobin/STAR/releases> (version 2.4.x and above) for mapping reads to the genome.

The required external programs can be installed from Linux repositories (e.g., using package managers such as Apt-get in Ubuntu).

```
sudo apt-get install samtools
sudo apt-get install r-base
```

Alternatively, packages can be manually installed (see Note 3).

3 Methods

3.1 RNA Sequencing Data Generation

The protocol begins with extraction of the target population of RNAs such as mRNA or small RNA using a phenol-based extraction method from tissues or cells. The extracted material is then prepared as a sequencing library and sequenced using any standard RT-based Illumina sequencing protocol for RNA class of interest (e.g., mRNAs or small RNAs).

1. RNA extraction: Use a standard phenol-based RNA extraction to obtain a minimum of 4 µg total RNA using Illumina TruSeq® RNA Library Prep Kit or Illumina TruSeq® Stranded Total RNA Library Prep Kit; alternatively, obtain a minimum of 1 µg total RNA from tissues or cell lines using Illumina TruSeq® Small RNA Library Preparation.
2. Library preparation for RNA sequencing: Use Illumina TruSeq®RNA, Stranded total RNA, or TruSeq Small RNA library preparation kit protocol for standard RT-based sequencing. These include adaptor ligation, cDNA synthesis, and library amplification steps, all adapted from original Illumina library preparation kit protocol. The libraries should be sequenced on an Illumina Genome Analyzer machine (e.g., HiSeq2000, GAIIx, or above) with read length 50–100 bp.
3. Following sequencing is a series of preprocessing steps (not included in the HAMR software, summarized in Fig. 1) for turning the FASTQ file obtained from the Illumina sequencer to mapped (BAM) file format. These preprocessing steps ensure the input BAM file is compatible and ready for use with HAMR. The preprocessing steps include read trimming, read mapping with mismatches, read filtering, and handling of multi-mapped reads (optional for studying highly repetitive RNAs such as tRNAs, or other small RNAs). The HAMR software (see Fig. 2) can then be applied to the prepared BAM file. The HAMR software is generally applicable to any mRNA or small RNA populations [14].

³The simplest way to install programs from source is to build them in your own directory. Download the appropriate package from the source websites (see Subheading 2.2 on software requirements). Extract the files and check the readme files for installation instructions and a list of required dependencies.

3.2 Downloading and Installing HAMR

All HAMR programs are packaged as compiled “binaries” (see Note 4) that can be run from the command line by typing in the program’s file name. In general, HAMR programs will also require other external programs (see Subheading 2.2 on software requirements). To install HAMR:

1. Create a directory for HAMR analyses and enter this directory:

```
mkdir <HAMRdirectory>
cd <HAMRdirectory>
```

2. Download HAMR from <https://github.com/wanglab-upenn/HAMR> to the HAMR directory <HAMRdirectory>. For example, to download HAMR release v1.2 from the command line

```
wget https://github.com/wanglab-upenn/HAMR/archive/v1.2.tar.gz -O
HAMR-v1.2.tar.gz
```

3. Extract HAMR source code from archive:

```
tar xzvf HAMR-v1.2.tar.gz
```

4. Go to the HAMR program directory:

```
cd HAMR-1.2/
```

5. Compile HAMR source code (see Note 4):

```
Make clean
Make
```

3.3 Preparing the Reference Genome

1. Download a reference genomic sequence. Human genome data as well as genomes of other organisms can be obtained from the UCSC Genome Browser (<http://hgdownload.soe.ucsc.edu/downloads.html>). Users can also go to the HAMR website (<http://lisanwanglab.org/hamr/genomes/>) to get the genome FASTA files for human (hg19), mouse (mm9), *D. melanogaster* (dm3), *C. elegans* (ce6), yeast (sacCer3), and *A. thaliana* (TAIR10). Plant genomes, such as Arabidopsis, can be downloaded from the Ensembl Plant project (see Note 5). For instance, to download human reference genome:

⁴.HAMR is packaged as compiled “binaries” that can be run by typing in the program’s filepath. On certain systems, however, the HAMR needs to be recompiled, to do so, please use make clean, then make.

```
cd <HAMRdirectory>
```

```
wget http://lisanwanglab.org/hamr/genomes/hg19\_all\_chr.fas
```

2. Index genome FASTA file (*see* Note 6):

```
samtools faidx hg19_all_chr.fas
```

3.4 Preparing RNA-Seq Data for HAMR

When starting from raw sequenced reads, the user needs to use a Linux-capable computer to execute the following steps (*see* Fig. 1) to prepare an input BAM file for HAMR analysis (*see* Subheading 3.8 for an example of HAMR analysis starting from raw sequenced data). All sorting is done with respect to chromosomal coordinates, not read name.

1. Read trimming

Use Cutadapt software [25] to trim the adaptor sequences from sequencing reads (*see* Subheading 3.8 for an example of how to run cutadapt).

- a. We suggest using the following trimming parameters: 6 % error rate for adaptor sequence with at least 6 nt overlap, and the read length is at least 15 nt after trimming.
- b. For smRNA-seq: The reads are 3' adapter-trimmed, with at least 6 bp of adapter sequence having at most a 6 % mismatch rate is required. All untrimmed reads and trimmed reads shorter than 14 bp should be discarded (*see* Note 7). Only trimmed reads should be used.
- c. For mRNA-seq: Use all reads (both trimmed and untrimmed) that are at least 15 nt.

2. Read mapping

- a. Use STAR [26] for mapping reads to the genome (*see* Subheading 3.8 on an example of mapping reads using STAR aligner). The reads should be mapped allowing for mismatches between sequencing reads and the reference genomic sequence (*see* Subheading 3.8 on HAMR analysis of RNA modification in the human brain for an example of mapping with STAR).
- b. For both smRNAs and mRNAs, map reads with mismatches. For shorter reads (e.g., smRNA-seq) X = 2 mismatches are suggested. For longer reads (e.g., mRNA-seq), setting the number of mismatches to $\leq 5\%$ of mapped read length is suggested (*see* Note 8).

⁵ Arabidopsis genome FASTA file can be downloaded here: ftp://ftp.ensemblgenomes.org/pub/release-25/plants/fasta/arabidopsis_thaliana/dna/Arabidopsis_thaliana.TAIR10.25.dna.genome.fa.gz

⁶ Make sure to index the FASTA file with genome index before running HAMR for faster execution.

⁷ Untrimmed reads are likely to correspond to larger RNA fragments that are outside the small RNA population.

- c. Allow for soft-clipping (*see* Note 9).
 - d. For mRNA only, filter out or resolve the spliced alignments (*see* Note 10).
 - e. Sort and index the resulting BAM file (*see* Note 11).
3. Handling multimapped reads (optional).

We strongly suggest using only uniquely mapped reads for general purposes and for analyzing longer RNAs such as mRNAs. For example, when using STAR for mapping, the multimapped reads can be filtered out with—`outFilterMultimapNmax 1` option (*see* Subheading 3.8). For analysis of highly repetitive classes of RNA (e.g., tRNA), additional steps need to be performed to assign multimapped reads to the repeat family before running HAMR. Resolving multimapped reads in case of tRNAs is illustrated below:

- a. Take the nuclear tRNA annotations from the “tRNAs” table in the UCSC genome browser (e.g., GRCh37) (*see* Note 12).
- b. Generate the annotations for mitochondrial tRNAs by running tRNAscan-SE (*see* Note 13). Set to organelle mode on the mitochondrial genome (“chrM” in GRCh37/hg19).
- c. Discard reads aligning to repeat regions or annotated RNAs other than tRNAs. This step will keep only the reads mapped exclusively to the target tRNAs. Discard reads mapped to more than one tRNA iso-acceptor family.
- d. Optional: filter out alignments with mismatches overlapping with single nucleotide polymorphisms (SNPs) based on dbSNP for human (*see* Note 14). This will prioritize alignments whose mismatches had no apparent explanation (i.e., likely caused by RNA modification) over mismatches caused by SNPs.
- e. Sort and index the output BAM file (*see* Note 11).

3.5 Running HAMR Pipeline

1. Description:

The HAMR program [20] takes as input a BAM file consisting of the sequencing reads aligned to the reference genome allowing for mismatches between reads

⁸. If no mismatches are allowed, HAMR is not capable of detecting any modifications.

⁹. Soft clipping routinely allows for mapping reads with untrimmed partial adapters at 3' end (e.g., shorter than the minimum overlap length), containing partial exon-exon junctions, etc.

¹⁰. The HAMR program handles continuous read alignments, and spliced reads require additional preprocessing.

¹¹. BAM files need to be sorted by genomic chromosome and start positions, not by the read name.

¹². The tRNAs table in the UCSC genome browser (hg19) can be downloaded here: http://genome.ucsc.edu/cgi-bin/hgTables?hgsql=482380859_VTSVW7eMgPDQJ0DdZlKALAE02bM&clade=mammal&org=Human&db=hg19&hgta_group=genes&hgta_track=tRNAs&hgta_table=0&hgta_regionType=genome&position=chr1%3A1103243--1103332&hgta_outputType=primaryTable&hgta_outFile-Name=.

¹³. tRNAscan-SE can be downloaded and installed from <http://gtrnadb.ucsc.edu/>.

¹⁴. dbSNP can be downloaded from <http://www.ncbi.nlm.nih.gov/projects/SNP/>.

and reference genomic sequences (*see* Subheading 3.4). HAMR first produces a table of observed reference and non-reference (mismatched) read nucleotide frequencies per candidate genomic site. Only high-quality (e.g., Q score of at least 30) read positions are considered when tabulating mismatches. The resulting candidate set of genomic sites are strictly filtered for sequencing errors using a binomial test that assumes a sequencing error-rate tenfold higher than the error-rate corresponding to the minimum quality score. The sites that pass this filter are enriched for mismatches and consist of potential SNPs, RNA editing events, and RNA modification sites. To filter out sites corresponding to homozygous or heterozygous SNPs and RNA editing events, HAMR performs a series of binomial tests using an ensemble of biallelic null hypotheses corresponding to all possible genotypes (AA, AC, AG, AT, CC, etc.). Sites that cannot be explained by any of the homozygous and biallelic genotypes are then inferred to be modifications. These inferred modification sites will be classified as particular modification types based on machine learning models built from the observed mismatch patterns at known tRNA modification sites, as annotated in the MODOMICS database [20, 27, 28] (*see* Note 15).

Here, we describe how to use HAMR for processing mapped BAM files into predictions of RNA modification positions and identity (*see* Fig. 2) (if starting from raw sequencing reads, *see* Subheading 3.4 on read preprocessing steps). The starting point for HAMR analysis is the mapped BAM. For the following steps, we assume that sequenced reads have been aligned, filtered and are stored in BAM format (*see* Subheading 3.4). All HAMR programs are operated from the command line. Using a command-line interface makes it easy to reproduce results and keep track of the analyses performed. Note that variable names within HAMR commands enclosed in <angle brackets> are placeholders for file names and analysis parameters and should be replaced with appropriate names without angle brackets.

HAMR analysis can be performed genome-wide (default) or for specific genomic regions (targeted mode). The only difference between the two modes is that user needs to prepare a BED file (BED6 format, also *see* Note 16) containing genomic regions of interest in BED format for the targeted mode.

Besides preparing a BAM file <prepared_sorted.bam> to run HAMR, the user needs to provide a reference genome file <genome.fa> in FASTA format (*see* Subheading 3.3 on how to download and prepare a reference genome sequence).

2. HAMR usage:

```
python hamr.py [-h] [--target_bed [TARGET_
BED]] [--paired_ends] [--filter_ends]
```

¹⁵tRNA modification information from MODOMICS is chosen as this is by far the best documented list of RNA modification sites. Data from MODOMICS is downloadable at <http://modomics.genesilico.pl/downloads/>.

¹⁶See <https://genome.ucsc.edu/FAQ/FAQformat.html#format1> for a description of the BED file format.


```

<bam> <genome_fas> <prediction_training_set>
<output_folder> <output_prefix> <min_qual>
<min_cov> <seq_err> <hypothesis> <max_p>
<max_fdr> <refpercent>

```

Required parameters

- a. <bam> = prepared sorted BAM file (*see* Subheading 3.4 on preparing BAM file from raw sequenced read data)
- b. <genome_fas> = indexed <genome.fa>(FASTA file)
- c. <prediction_training_set> = data used for training HAMR
- d. <output_folder> = output directory
- e. <output_prefix> = output directory prefix name, contains the following outputs:
 - <output_prefix>.mods.bed= BED file contains detected modification sites
 - <output_prefix>.mods.txt = Text file contains detected modification sites
 - <output_prefix>.raw.txt =full table contains the candidate sites for statistical test
- f. <min_qual> = Minimum base calling quality, suggested 30
- g. <min_cov> = Minimum number of non-reference nucleotide counts, suggested 10 (*see* Note 17)
- h. <seq_err> = expected sequencing error rate / mismatch rate, suggested 0.01–0.05 (*see* Note 18)
- i. <hypothesis> = Statistical test to be performed, H4 is recommended (*see* Note 19)
- j. <max_p> = Maximum p-value (0.01 is recommended)
- k. <max_fdr> = Maximum FDR with replacement
- l. <refpercent>= Minimum percentage of reads with reference nucleotide

Optional parameters

- d. [-h] = prints help screen
- e. [-target_bed [TARGET_BED]] = targeted BED file containing genomic regions of interest (*see* Note 16)

¹⁷Filter out genomic positions with insufficient number (e.g., less than 10) of non-reference nucleotide counts. Such positions do not provide sufficient evidence for HAMR to reliably establish mismatch pattern necessary to detect RNA modifications.

¹⁸Filter out genomic positions with non-reference / reference read count ratio that less than, e.g., 1 %. Such positions are likely to be attributed to sequencing errors, not modification-related mismatches.

¹⁹HAMR tests for homozygous, bi-allelic, and multi-allelic loci, *see* [20] for more information on statistical testing and hypotheses.

- f. [--paired_ends] = set to 1 for paired-end sequencing data
- g. [--filter_ends] = set to 1 to filter out the read ends from read-pileups
3. Example for running HAMR genome-wide mode:

```
python hamr.py trial.human.bam hg19_all_chr.
fas models/euk_trna_mods.Rdata HAMR_out brain
30 10 0.05 H4 0.01 0.05 0.05 --filter_ends
Sample output HAMR_out/brain.mods.txt
```

chr	bp	strand	refuc	A	C	G	T	nonref	ref	h1.p	h1.padj	h4.p	h4.padj	sig	pred.mod
chr1	159111465	-	G	31	43	485	126	200	485	1.70e-93	1.55e-91	7.89e-10	8.22e-08	TRUE	m16
chr1	161510088	+	A	29	1	16	54	71	29	1.21e-68	7.31e-67	9.43e-06	0.001	TRUE	m1a m11 ms216A
chr1	222638403	+	A	78	1	114	418	533	78	0	0	2.56e-14	3.11e-12	TRUE	m1a m11 ms216A

4. Example for running HAMR in targeted mode for specific genomic regions:

```
python hamr.py trial.human.bam hg19_all_chr.
fas models/euk_trna_mods.Rdata HAMR_out brain
30 10 0.05 H4 0.01 0.05 0.05 --filter_ends
```

3.6 Running HAMR Step-by-Step

Besides running HAMR in a genome-wide or target-bed batch mode, the HAMR pipeline (see Fig. 2) can also be run step-by-step.

3.6.1 HAMR-Step-1: RNA Pileup

1. Description: this step produces per-position read pileup for a given BAM file containing mapped reads.
2. Usage:

```
./rnapileup <align.bam> <genome.fa> >
<pileup.raw>
```

<align.bam>: sorted BAM file (see Subheading 3.4) containing uniquely-mapped reads and continuous, un-spliced alignments

<genome.fa>: indexed reference genome (FASTA file)

<pileup.raw>: output file in pileup format

3. Example:

```
./rnapileup trial.human.bam hg19_all_chr.
fas > trial.human.bam.pileup
```

4. Sample RNA pileup output:

```
chr1 161510084 G 100 .....
chr1 161510085 T 109 .....
chr1 161510086 T 109 .....
chr1 161510087 C 109 .....
chr1 161510088 G 109 .....
*chr1 161510089 A 109 .TTTTGT...TT...TTT.TTTT.TTTTGGGTT...TTT.TTGTG.T.TTTTGG.GTTTGGT..GTTTTTTTTGT.TGT..T..TGT.T..TGT.GTG.....C..
chr1 161510090 T 109 .....
chr1 161510091 C 109 .....
chr1 161510092 C 109 .....
chr1 161510093 C 109 .....
chr1 161510094 A 110 .....
chr1 161510095 C 110 .....
```

This output is in a pileup format (see <http://samtools.sourceforge.net/pileup.shtml> for details) and contains chromosome, physical position, reference nucleotide, the number of reads covering the site, read nucleotides, base qualities. In the read nucleotide column:

^~	indicates 5' (start) of the read segment
.	indicates a match between the read nucleotide and reference nucleotide on the forward strand
,	indicates a match between the read nucleotide and reference nucleotide on the reverse strand
[ACGT]	indicates a mismatch (non-reference read nucleotide) on the forward strand
[acgt]	indicates a mismatch (non-reference nucleotide) on the reverse strand

3.6.2 HAMR-Step-2: Filter Pileup

- 1. Description: this step removes low-quality reads (i.e., reads with low base calling quality) from pileups and (optionally) filters reads whose 5' or 3' ends coincide with the predicted pileup position.
- 2. Usage:

```
./filter_pileup <pileup.raw> <minQ> <filter_
ends=0|1> > <pileup.filtered>
```

This command reflects the following required parameters:

- <pileup.raw>: raw pileup file from **step 1**
- <minQ>: minimum read quality (30)
- <filter_ends=0|1>: read end filtering option

- 3. Example:

```
./filter_pileup trial.human.bam.pileup 30 1
> trial.human.bam.pileup.filtered.
```

Output of coverage <pileup.filtered> is of the same format as that of <pileup.raw>

- 4. Sample output after filtering RNA pileup:

```

chr1 161510082 T 97 .....
chr1 161510083 G 99 .....
chr1 161510084 G 97 .....
chr1 161510085 T 102 .....
chr1 161510086 T 104 .....
chr1 161510087 C 101 .....
chr1 161510088 G 104 .....
*chr1 161510089 A 100 .....TTGT...TT.TTT.TTTTGGGT...TTT.TGTGG.T.TTTTGG.GTTTTTTTGT.TGT..T..TGT.T..TGTG....C..
chr1 161510090 T 106 .....TTGT...TT.TTT.TTTTGGGT...TTT.TGTGG.T.TTTTGG.GTTTTTTTGT.TGT..T..TGT.T..TGTG....C..
chr1 161510091 C 100 .....
chr1 161510092 C 102 .....
chr1 161510093 C 101 .....
chr1 161510094 A 99 .....
chr1 161510095 C 103 .....

```

3.6.3 HAMR-Step-3: Filter Out Pileups with Below Minimum Coverage

1. Description: this step filters out pileups with below minimum coverage.
2. Usage:

```

awk '$4>=<min_cov>' <pileup.
filtered>><pileup.filtered.min_cov>

```

This command reflects the following required parameters:

<min_cov>: minimum coverage

3. Example:

```

awk '$4>=10' trial.human.bam.pileup.filtered
>trial.human.bam.pileup.filtered.min10

```

4. Sample output of pileups filtered by read coverage:

```

chr1 161510084 G 97 .....
chr1 161510085 T 102 .....
chr1 161510086 T 104 .....
chr1 161510087 C 101 .....
chr1 161510088 G 104 .....
*chr1 161510089 A 100 .....TTGT...TT.TTT.TTTTGGGT...TTT.TGTGG.T.TTTTGG.GTTTTTTTGT.TGT..T..TGT.T..TGTG....C..
chr1 161510090 T 106 .....TTGT...TT.TTT.TTTTGGGT...TTT.TGTGG.T.TTTTGG.GTTTTTTTGT.TGT..T..TGT.T..TGTG....C..
chr1 161510091 C 100 .....
chr1 161510092 C 102 .....
chr1 161510093 C 101 .....

```

3.6.4 HAMR-Step-4: Convert Filtered Pileups to BED Format

1. Description: this step converts filtered pileups to BED format.
2. Usage:

```

./rnapileup2mismatchbed <filtered.min_cov> >
<nuc2nuc.BED>

```

3. Example:

```

./rnapileup2mismatchbed trial.human.bam.
pileup.filtered.min10 > trial.human.bam.mis-
match.bed

```

4. Output is sorted by genomic coordinates.

5. The output BED file contains the number of observed reference nucleotides to read nucleotide transitions per candidate site.
6. Sample read nucleotide mismatch BED output

```
chr1 161510082 161510083 G> 99;2:1,5:7,7:2,11:20,12:33,13:19,14:5,15:7,16:1,17:1,18:1,19:1,20:1 *
chr1 161510083 161510084 G> 97;3:1,6:7,8:1,12:19,13:32,14:19,15:6,16:8,17:1,18:1,19:1,20:1 *
chr1 161510084 161510085 Ts 102;1:1,4:1,7:7,9:2,13:19,14:33,15:21,16:6,17:7,18:1,19:1,20:1,21:1,22:1 *
chr1 161510085 161510086 Ts 104;1:1,2:1,5:1,8:7,10:2,14:20,15:33,16:20,17:6,18:9,19:1,20:1,22:1,23:1 *
chr1 161510086 161510087 Cs 101;2:1,6:1,9:5,11:2,15:20,16:33,17:19,18:6,19:8,20:1,21:1,22:1,23:1,24:2 *
chr1 161510087 161510088 G> 104;3:1,4:1,7:1,10:6,12:2,16:20,17:33,18:20,19:6,20:9,21:1,22:1,23:1,24:1,25:1 *
chr1 161510088 161510089 As 29;4:1,5:1,11:5,17:8,18:4,19:4,20:1,21:4,26:1 *
chr1 161510088 161510089 AsC 1;8:1 *
chr1 161510088 161510089 A>G 16;13:1,17:3,18:7,19:4,22:1 *
chr1 161510088 161510089 AsT 54;11:1,17:9,18:21,19:12,20:4,21:5,23:1,25:1 *
chr1 161510089 161510090 Ts 106;5:1,6:1,9:1,12:7,14:2,18:20,19:33,20:20,21:6,22:9,23:1,24:1,25:1,26:1,27:2 *
chr1 161510090 161510091 Cs 100;6:1,7:1,10:2,13:6,15:2,19:19,20:31,21:19,22:6,23:8,24:1,25:1,26:1,27:1,28:1 *
chr1 161510091 161510092 Cs 102;7:1,8:1,11:2,14:6,16:2,20:20,21:31,22:19,23:6,24:9,25:1,26:1,27:1,28:1,29:1 *
chr1 161510092 161510093 Cs 101;8:1,12:2,15:6,17:2,21:20,22:30,23:19,24:6,25:10,26:1,27:1,28:1,29:1,30:1 *
chr1 161510093 161510094 As 99;9:1,13:2,16:5,18:2,22:19,23:31,24:18,25:6,26:10,27:1,28:1,29:1,30:1,31:1 *
chr1 161510094 161510095 Cs 103;1:1,10:1,11:1,14:2,17:6,19:2,23:19,24:31,25:20,26:6,27:10,28:1,29:1,31:1,32:1 *
chr1 161510095 161510096 Cs 99;2:1,11:1,12:1,15:1,18:6,20:1,24:18,25:31,26:18,27:6,28:10,29:1,30:1,31:1,32:1,33:1 *
chr1 161510096 161510097 Cs 99;3:1,12:1,13:1,16:1,19:6,21:1,25:18,26:31,27:18,28:6,29:10,30:1,31:1,32:1,33:1,34:1 *
```

This output contains the following information:

Fourth column = type of mismatch (A>T, A>G, A>C, etc.) or match (A>., T>., C>., G>.) between the reference and read nucleotide

Fifth column = first number is the total number of reads with match or mismatch event, followed by the detailed list of positions within the reads where the event is observed

3.6.5 HAMR-Step-5: Convert BED to HAMR-FREQ- TABLE

1. Description: this step converts BED to HAMR-FREQ-TABLE while keeping only positions with >0 non-reference nucleotides, and filters the nucleotide frequency table by removing SITES (rows) with insufficient sequencing-based information.
2. Usage:

```
mismatchbed2table <nuc2nuc.BED> > <HAMR-FREQ-TABLE>
```

3. Example:

```
bash mismatchbed2table.sh trial.human.bam. mismatch.bed >
trial.human.bam.freq.table
```

4. Sample output of read nucleotide frequencies:

chr	bp	strand	refnuc	A	C	G	T	nonref	
chr1	161450359	+	A	36	1	0	0	0	1
chr1	161450360	+	G	0	0	180	17	17	
chr1	161493634	-	T	0	3	0	7	3	
chr1	161493635	-	C	0	9	0	1	1	
chr1	161493641	-	A	14	1	0	0	1	
chr1	161493651	-	A	5	0	0	7	7	
chr1	161500129	-	T	0	9	0	2	9	
chr1	161500130	-	A	2	8	0	0	8	
chr1	161500146	-	A	15	1	0	0	1	
chr1	161510080	+	G	1	0	98	0	1	
*chr1	161510088	+	A	29	1	16	54	71	
chr1	169076221	+	T	0	17	0	1	17	
chr1	169101940	+	A	26	1	0	0	1	
chr1	171570078	-	T	0	5	0	39	5	
chr1	171966723	-	C	0	0	268	0	268	

- Additional filtering of sites based on observed read nucleotide counts is imposed, including:

minimum number of reads with non-reference nucleotide [=10]

minimum number of reference reads [=10] minimum ratio of non-reference/reference[=1 %]

- Usage:

```
awk '{cov=$5+$6+$7+$8;nonref=$9; ref.=cov-
nonref; if (ref/cov>=<refpercent>+)
print;}' '<HAMR-FREQ-TABLE>' <HAMR-FREQ-
TABLE-FINAL>
```

This command reflects the following required parameters: <refpercent> = Minimum percentage of reads with reference nucleotide

- Example:

```
awk '{cov=$5+$6+$7+$8;nonref=$9; ref=cov-
nonref; if (ref/cov>=0.05) print;}' trial.
human.bam.freq.
table > trial.human.bam.freq. final.table
```

- Output table contains the HAMR frequency information on the FINALIZED set of candidate sites with enough sequencing support.

3.6.6 HAMR-Step-6: Finding Modification Sites Based on Statistical Testing

- Description: this step finds modification sites based on significant deviations of the observed nucleotide frequencies from the expected homozygous or biallelic distributions.
- Usage:

```
Rscript detect_mods.R <HAMR-NUC-TABLE>
<EXPECTED-SEQ-ERROR-RATE> <HYPOTHESIS-TYPE>
<P-VALUE-THRESHOLD> <Q-VALUE--THRESHOLD> >
<RAW-MOD-TABLE>
```

This command reflects the following required parameters:

<HAMR-NUC-TABLE> = HAMR frequency table from **step 5**

<EXPECTED-SEQ-ERROR-RATE> = expected sequencing error rate / mismatch rate, suggested 0.01–0.05 (see Note 18)

<HYPOTHESIS-TYPE> = Statistical test to be performed, H4 is recommended (see Note 19)

<P-VALUE-THRESHOLD> = Maximum p-value (0.01 is recommended)

<Q-VALUE-THRESHOLD> = Maximum FDR with replacement

3. Example:

```
Rscript detect_mods.R trial.human.bam.freq.final.
table 0.05 H4 0.01 0.05 > trial.human.bam.raw.txt
```

4. Sample output of HAMR predicted modification sites:

chr	bp	strand	refnuc	A	C	G	T	nonref	ref	h1.p	h1.padj	h4.p	h4.padj	sig	1	1	FALSE
chr1	161500129	-	T	0	9	0	2	9	2	9.79e-11	2.38e-09	1	1	FALSE			
chr1	161500130	-	A	2	8	0	0	8	2	1.60e-09	3.77e-08	1	1	FALSE			
chr1	161500146	-	A	15	1	0	0	1	15	0.56	1	1	1	FALSE			
chr1	161510000	+	G	1	0	98	0	1	98	0.99	1	1	1	FALSE			
*chr1	161510008	+	A	29	1	16	54	71	29	1.21e-68	7.37e-67	9.43e-06	1	0.001	TRUE		
chr1	169076221	+	T	0	17	0	1	17	1	1.31e-21	5.30e-20	1	1	FALSE			
chr1	169101940	+	A	26	1	0	0	1	26	0.75	1	1	1	FALSE			
chr1	171570078	-	T	0	5	0	39	5	39	0.07	1	1	1	FALSE			
chr1	172100004	+	A	22	0	1	0	1	22	0.69	1	1	1	FALSE			
chr1	172100013	+	A	23	0	1	0	1	23	0.71	1	1	1	FALSE			

3.6.7 HAMR-Step-7—Predicting modification identity

1. Description: predict the identity of modifications for each of the modification sites (see Note 20).
2. Usage:

```
Rscript classify_mods.R <TABLE-significant.
sites.only> <training.set.Rdata> > <TABLE-
with-predicted-mod-types>
```

This command reflects the following required parameters: <TABLE-significant.sites.only> = Table contains detected modification sites that are statistically significant <training.set.Rdata> = prediction training set (e.g., the training set models/euk_trna_mods.Rdata provided in HAMR that consists of mismatch patterns observed in confirmed modification site in tRNAs)

²⁰For a list of HAMR-detectable modification types, refer to Supplementary table 1 in [20].

3. Example:

```
Rscript classify_mods.R trial.human.bam.raw.
txt models/euk_trna_mods.Rdata > trial.human.
bam.mods.txt and CHANGE font to match other
command lines
```

4. Sample output of HAMR modification sites and predicted modification types:

chr	bp	strand	refnuc	A	C	G	T	nonref	ref	n1.p	n1.padj	n4.p	n4.padj	sig	pred.mod
chr1	159111465	-	G	31	43	485	126	200	485	1.70e-93	1.55e-01	7.89e-10	8.22e-08	TRUE	m1G
*chr1	161510088	+	A	29	1	16	54	71	29	1.21e-68	7.31e-67	9.43e-06	0.001	TRUE	m1A[m1I][ms216A
chr1	222638483	+	A	78	1	114	418	533	78	0	0	2.56e-14	3.11e-12	TRUE	m1A[m1I][ms216A

3.7 Using HAMR Webserver

HAMR is also available as a web-based application, which can be accessed at <http://lisanwanglab.org/hamr/>.

The web interface allows specification of a remote, indexed BAM file and BED file with targeted intervals for querying. The user may specify parameters for the preprocessing steps, such as minimum base call quality score, minimum read coverage at a site, assumed sequencing error rate, and significance level. Additionally, the user may use the webserver to predict the modification type based on mismatch patterns.

3.8 Application of HAMR to Existing Human Small RNA-Seq Data

1. Here, we present the application of HAMR to a Illumina small RNA-seq data generated from human brain tissue (GSE48552, <http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE48552>). With a few preprocessing steps prior to HAMR, users can run a single command and all information about modification sites and types will be summarized in a table.
2. Download the SRA file from SRA database (<http://www.ncbi.nlm.nih.gov/geo/>):

```
wget ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/
ByStudy/sra/SRP/SRP026/SRP026562/SRR1103937/SRR1103937.sra-O
brainpfc.sra
```

3. Build genomic index:

```
STAR--runMode genomeGenerate --runThreadN 8
--genomeDir genomes/star/hg19 --genomeFas-
taFiles hgl9_all_chr.fas
```

4. Convert SRA to FASTQ file using SRAtoolkit:

```
fastq-dump --split-3 brainpfc.sra
```


5. Trim adapter sequences from reads (Illumina v1.5 small RNA 3' adapter in this case):

```
cutadapt brainpfc.fastq -a ATCTCGTATGCCGTCTTCTGCTTG -e 0.06 -O 6 -
m 14
--too-short-output=brainpfc.tooshort.fastq
--untrimmed-output=brainpfc.untrimmed.fastq
--length --tag="length=" -o brainpfc.trimmed.
fastq
```

6. Map trimmed reads to the human reference genome hg19 using STAR:

```
mkdir -p STAR_brainpfc
STAR --genomeDir genomes/star/hg19 --genom-
eLoad LoadAndKeep --readFilesIn brainpfc.
trimmed.fastq --runThreadN 4 --alignIntron-
Max 1 --outSAMattributes NH HI NM MD --out-
FilterMultimapNmax 1 --outReadsUnmapped
Fastx --outFilterMismatchNmax 2 --outFil-
terMatchNmin 15 --outFileNamePrefix STAR_
brainpfc/
```

7. Sort and index output BAM:

```
samtools sort -@ 8 -O bam -T Aligned.out.
sort STAR_brainpfc/Aligned.out.sam > STAR_
brainpfc/Aligned.out.sorted.bam
samtools index STAR_brainpfc/Aligned.out.
sorted.bam
```

8. Perform modification analysis with HAMR:

```
python hamr.py STAR_brainpfc/Aligned.out.sorted.
bam genomes/hg19.fa models/euk_trna_mods.Rdata
HAMR_results brainpfc 30 10 0.05 H4 0.01 0.05
0.05 --filter_ends
```

HAMR_results/brainpfc.mods.txt will contain a list of RNA modifications found by HAMR.

Acknowledgments

This work is supported by the National Institute of General Medical Sciences [R01-GM099962 to P.P.K., Y.Y.L., B.D.G., and L.S.W.], National Institute on Aging [U24-AG041689 to L.S.W.], National Science Foundation

[CAREER Award MCB-1053846, MCB-1243947, and IOS-1444490 to B.D.G.]. We thank Alexandre Amlie-Wolf and other members of the Wang and Gregory labs for their comments and help with this work.

Abbreviations

3'	3-prime (3')
5'	5-prime (5')
bp	Base pair
cDNA	Complementary DNA
HAMR	High-throughput annotation of modified ribonucleotides
mRNA	Messenger RNA
mRNA-seq	messenger RNA sequencing
nt	Nucleotide
RNA-seq	RNA sequencing
RT	Reverse transcriptase
smRNA	Small RNA
smRNA-seq	Small RNA sequencing
SNP	single nucleotide polymorphism
tRNA	Transfer RNA

References

1. Dominissini D, Nachtergaele S, Moshitch-Moshkovitz S et al. (2016) The dynamic N(1)-methyladenosine methylome in eukaryotic messenger RNA. *Nature* 530:441–446 [PubMed: 26863196]
2. Eigenbrod T, Keller P, Kaiser S et al. (2015) Recognition of specified RNA modifications by the Innate immune system. *Methods Enzymol* 560:73–89 [PubMed: 26253966]
3. Li S, Mason CE (2014) The pivotal regulatory landscape of RNA modifications. *Annu Rev Genomics Hum Genet* 15:127–150 [PubMed: 24898039]
4. Lee M, Kim B, Kim VN (2014) Emerging roles of RNA modification: m(6)A and U-tail. *Cell* 158:980–987 [PubMed: 25171402]
5. Satterlee JS, Basanta-Sanchez M, Blanco S et al. (2014) Novel RNA modifications in the nervous system: form and function. *J Neurosci* 34:15170–15177 [PubMed: 25392485]
6. Delatte B, Wang F, Ngoc LV et al. (2016) Transcriptome-wide distribution and function of RNA hydroxymethylcytosine. *Science* 351:282–285 [PubMed: 26816380]
7. Sundaram M, Durant PC, Davis DR (2000) Hypermodified nucleosides in the anticodon of tRNA^{Lys} stabilize a canonical U-turn structure. *Biochemistry* 39:12575–12584 [PubMed: 11027137]
8. Kierzek E, Mąłowska M, Lisowiec J, Turner DH, Gdaniec Z, Kierzek R (2014) The contribution of pseudouridine to stabilities and structure of RNAs. *Nucleic Acids Res* 42:3492–3501 [PubMed: 24369424]

9. Schwartz S, Mumbach MR, Jovanovic M et al. (2014) Perturbation of m6A writers reveals two distinct classes of mRNA methylation at internal and 5' sites. *Cell Rep* 8:284–296 [PubMed: 24981863]
10. Meyer KD, Jaffrey SR (2014) The dynamic epitranscriptome: N6-methyladenosine and gene expression control. *Nat Rev Mol Cell Biol* 15:313–326 [PubMed: 24713629]
11. Karijolich J, Yu YT (2015) The new era of RNA modification. *RNA* 21:659–660 [PubMed: 25780180]
12. Sun WJ, Li JH, Liu S et al. (2016) RMBase: a resource for decoding the landscape of RNA modifications from high-throughput sequencing data. *Nucleic Acids Res* 44:D259–D265 [PubMed: 26464443]
13. Schwartz S, Bernstein DA, Mumbach MR, Jovanovic M, Herbst RH, León-Ricardo BX, Engreitz JM, Guttman M, Satija R, Lander ES, Fink G, Regev A (2014b) Transcriptome-wide mapping reveals widespread dynamic-regulated pseudouridylation of ncRNA and mRNA. *Cell* 159:148–162 [PubMed: 25219674]
14. Vandivier LE, Campos R, Kuksa PP et al. (2015) Chemical modifications mark alternatively spliced and uncapped messenger RNAs in arabidopsis. *Plant Cell* 27:3024–3037 [PubMed: 26561561]
15. Meyer KD, Saletore Y, Zumbo P (2012) Comprehensive analysis of mRNA methylation reveals enrichment in 3' UTRs and near stop codons. *Cell* 149:1635–1646 [PubMed: 22608085]
16. Gupta RC, Randerath K (1977) Use of specific endonuclease cleavage in RNA sequencing. *Nucleic Acids Res* 4:1957–1978 [PubMed: 331267]
17. Woodson SA, Muller JG, Burrows CJ et al. (1993) A primer extension assay for modification of guanine by Ni(II) complexes. *Nucleic Acids Res* 21:5524–5525 [PubMed: 8265373]
18. Motorin Y, Muller S, Behm-Ansmant I, Branlant C (2007) Identification of modified residues in RNAs by reverse transcription-based methods. *Methods Enzymol* 425:21–53 [PubMed: 17673078]
19. Behm-Ansmant I, Helm M, Motorin Y (2011) Use of specific chemical reagents for detection of modified nucleotides in RNA. *JNucleic Acids* 2011:408053 [PubMed: 21716696]
20. Ryvkin P, Leung YY, Silverman IM et al. (2013) HAMR: high-throughput annotation of modified ribonucleotides. *RNA* 19: 1684–1692 [PubMed: 24149843]
21. Dominissini D, Moshitch-Moshkovitz S, Schwartz S et al. (2012) Topology of the human and mouse m6A RNA methylomes revealed by m6A-seq. *Nature* 485:201–206 [PubMed: 22575960]
22. Squires JE, Patel HR, Nousch M et al. (2012) Widespread occurrence of 5-methylcytosine in human coding and non-coding RNA. *Nucleic Acids Res* 40:5023–5033 [PubMed: 22344696]
23. Horowitz S, Horowitz A, Nilsen TW et al. (1984) Mapping of N6-methyladenosine residues in bovine prolactin mRNA. *Proc Natl Acad Sci U S A* 81:5667–5671 [PubMed: 6592581]
24. Li H, Handsaker B, Wysoker A et al. (2009) The sequence alignment/map format and SAMtools. *Bioinformatics* 25:2078–2079 [PubMed: 19505943]
25. Martin M (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnetJ* 17(1):10–12
26. Dobin A, Davis CA, Schlesinger F et al. (2013) STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* 29:15–21 [PubMed: 23104886]
27. Machnicka MA, Milanowska K, Osman OO et al. (2013) MODOMICS: a database of RNA modification pathways: 2012 update. *Nucleic Acids Res* 41:D262–D267 [PubMed: 23118484]
28. Leung YY, Ryvkin P, Ungar LH et al. (2013) CoRAL: predicting non-coding RNAs from small RNA-sequencing data. *Nucleic Acids Res* 41:e137 [PubMed: 23700308]

Fig. 1.
Preparing the raw sequencing data (from FASTQ to BAM) for running HAMR software [20]

Fig. 2. Steps used by HAMR software [20] for in silico detection and classification of RNA modifications