Using RAMPAGE to identify and annotate promoters in insect genomes

R. Taylor Raborn[*1,2] and Volker P. Brendel[1,2]

[1]Department of Biology, Indiana University
[2]School of Informatics and Computing, Indiana University

Department of Biology and School of Informatics and Computing,
Indiana University
212 S. Hawthorne Drive 205 Simon Hall, Bloomington, IN 47401, USA
http://www.brendelgroup.org

Abstract. Application of Transcription Start Site (TSS) profiling technologies, coupled with large-scale next-generation sequencing (NGS) has yielded valuable insights into the location, structure and activity of promoters across diverse metazoan model systems. In insects, TSS profiling has been used to characterize the promoter architecture of *Drosophila melanogaster* [1], and, shortly thereafter, to reveal widespread transposon-driven alternative promoter usage in *D. melanogaster* [2].
In this chapter we highlight the utility of one TSS profiling method, RAMPAGE (RNA annotation and mapping of promoters for analysis of gene expression), for the precise, quantitative identification of promoters in insect genomes. We demonstrate this using our tools GoRAMPAGE [3] and TSRchitect [4], providing details instructions with the aim of taking the user from raw reads to processed results.

Keywords: *cis*-regulatory regions, promoter architecture, transcription initiation, transcription start sites (TSSs)

1 Introduction

1.1 TSS Profiling Identifies Promoters at Genome-Scale

The promoter, defined in eukaryotes as the genomic region bound by RNA Polymerase II immediately prior to transcription initiation [5], is the site where regulatory signals unite to direct gene expression. The identification of promoter regions is a valuable step for understanding the *cis*-regulatory signals that are present in an organism, and is also important for genome annotation. However, despite the rapid accumulation of genome sequences across metazoan and arthropod diversity, accurate annotation of promoter regions remains sparse. This is because—absent empirically-defined information—precisely identifying

* Correspondence: rtraborn@indiana.edu

sequence motifs that demarcate the promoter is unreliable. In contrast with current *in silico* approaches, direct mapping of TSSs identifies the location of the core promoter. Cap Analysis of Gene Expression (CAGE) [6], one of the first methods devised to identify 5′-ends of mRNAs at large-scale, involves selective capture of 5′-capped transcripts, first-strand reverse-transcription and ligation of a short oligonucleotide (CAGE tag).

CAGE was initially utilized by the FANTOM (Functional Annotation of the Mammalian Genome) consortium to identify promoter architecture in human and mouse [7], providing the first glimpse of the global landscape of transcription initiation. At the onset of the NGS era, CAGE was coupled with massively-parallel sequencing to generate 5′-ends of mRNAs at substantially higher scale. This advance provided more extensive coverage of the expressed transcriptome, and provided increased sensitivity for quantitative measurements *i.e.* measurement of promoter activity.

1.2 Promoter Architecture of *Drosophila melanogaster*

Hoskins and colleagues [1] performed CAGE in *D. melanogaster* as part of the modENCODE consortium, identifying promoters at large-scale and characterizing the promoter architecture of an insect genome for the first time. Hoskins [1] indicated that TSS distributions at *Drosophila* promoters exhibit a range of shapes that can be generally grouped into two major classifications: *peaked* and *broad*. Peaked promoters have a single, major TSS position occupying a narrow genomic region, whereas broad promoters lack a single, major TSS and contain TSSs across a wider region [8, 9]. The authors also showed a strong association between promoter class and motif composition (consistent with previous findings [8, 10]). Peaked promoters were associated with positionally-enriched *cis*-regulatory motifs including TATA, Initiator (Inr) and DPE, while broad promoters contained an enrichment of less-well characterized motifs, including *Ohler6* and *Ohler7* [11]. The existence of two promoter classes appears to be conserved among metazoans, and has been reported (using TSS profiling methodolgies) in insects, cladocerans [12], fish [13] and mammals [14, 9].

1.3 Promoter Structure of Insects

Beyond *D. melanogaster*, few investigations have utilized TSS profiling in insect genomes. As a consequence, what is known about promoter architecture in insects is largely restricted to the *Drosophila* genus. As part of the modENCODE effort, CAGE was performed in multiple tissues and developmental stages of the *Drosophila pseudoobscura*. TSSs were found to be highly similar between species: more than 80% of TSSs (81%) of aligned, CAGE-identified TSSs from *D. pseudoobscura* were positioned within 20nt of their counterparts in *D. melanogaster*. An enrichment of the CA dinucleotide was detected at the TSS ([-1, +1]), and the motifs corresponding to TATA, Inr and DPE were positioned at the same locations relative to the TSS in both species.

The only other insect species for which TSS profiling has been applied is the Tsetse fly (*Glossina morsitans morsitans*) [15]. Using TSS-seq (specifically Oligo-capping; for details see [16]), the authors identified 3134 mapping to 1424 genes. The authors found a preference for CA and AA dinucleotides at the TSS, and observe the major core promoter elements observed in *Drosophila*: TATA, Inr, DPE, in addition to MTE (Motif Ten Element). As in *D. melanogaster*, peaked promoters were more likely to contain TATA and Inr than broad promoters. While the taxonomic sampling of species for TSS profiling has been limited, the existing studies are sufficient to provide a general picture of insect promoter architecture. A major demarcation between the promoter architecture of insects and mammals appears to be the large fraction of mammalian promoters found in CpG islands [15]. CpG island promoters (CPIs) form the largest class of promoter in mammals [17]; by contrast, CPIs are not known to exist as a class in invertebrates.

1.4 Paired-end TSS Profiling with RAMPAGE

The most recent major methodological advance in TSS Profiling is RAMPAGE (RNA Annotation and Mapping of Promoters for the Analysis of Gene Expression) [2, 18]. RAMPAGE is a protocol for 5′-cDNA sequencing that combines cap trapping and template-switching with paired-end sequence information. A key advantage of generating paired-end sequence is transcript connectivity, which provides a direct link between a given 5′-end and its associated mRNA molecule [2]. Because short or spurious RNAs are found within the transcriptome, transcript connectivity allows the TSSs (and thus promoters) of full-length mRNAs to be unambiguously identified, which benefits genome annotation and improves interpretation of transcript species.

Batut and colleagues [2] generated libraries from total RNA isolated from 36 stages across the life cycle of *D. melanogaster* providing a comprehensive gene expression and promoter atlas for fruit fly and in the process demonstrating the utility of RAMPAGE. RAMPAGE is currently being applied as part of the latest iteration of ENCODE to identify promoters in human, but as of this writing it has not been applied to any non-*Drosophila* insect model system. In anticipation of the future application of TSS profiling into other insect model systems here we provide a documented protocol for the computational processing RAMPAGE data, using selected libraries from Batut *et al.* [2]. This method will consist of two parts: first, we will process, filter and align the sequenced RAMPAGE libraries to the *D. melanogaster* genome. Second, we will identify TSSs and promoters from the aligned sequences and associate them with coding regions. In closing, we will consider further applications of this data and discuss the utility of reproducible workflows in bioinformatic analysis.

2 Materials

The analyses described herein require a workstation capable of doing modern bioinformatics, including a reasonably-appointed laptop. An intermediate understanding of the Linux/Unix command line will be extremely useful, although we make efforts to explain the procedures with clarity. In addition, it will likely be necessary for the participant to have superuser privileges on the machine. If you do not have a machine (or have access to one) that meets these requirements, it is recommended that you consider cloud-based cyberinfrastructure, including Amazon Web Services (AWS; https://aws.amazon.com/) or CyVerse (http://www.cyverse.org/) [19]. The former is a well-known pay-per-use solution, while the latter is an NSF-funded resource that makes compute allocations freely available to the public.

2.1 Hardware

1. x86-64 compatible processors
2. At least 8GB RAM
3. 30GB+ hard disk space

2.2 Operating System

- 64 bit Linux (preferred) or Mac OS X (with Command Line Tools from XCode)

2.3 Software

Below is a list of the software packages required for this demonstration (*see* **Note 1**).

Sequence retrieval

1. SRA Toolkit [20] (https://www.ncbi.nlm.nih.gov/sra/docs/toolkitsoft/)

GoRAMPAGE

1. GoRAMPAGE [3] (https://github.com/brendelGroup/GoRAMPAGE)
2. fastq-multx [21] (https://github.com/brwnj/fastq-multx)
3. FASTX-Toolkit [22] (`http://hannonlab.cshl.edu/fastx_toolkit/Index.html`)
4. TagDust2 [23] (https://sourceforge.net/projects/tagdust/)
5. Samtools [24] (http://www.htslib.org/doc/samtools.html)
6. STAR [25] (https://github.com/alexdobin/STAR)

TSRchitect

1. R (v. 3.4 and up) [26] (https://www.r-project.org/)
2. Bioconductor (v. 3.5 and up) [27] (http://bioconductor.org/)
3. TSRchitect [4] (http://bioconductor.org/packages/release/bioc/html/TSRchitect.html)
4. Various R package dependencies (see **Methods**)

2.4 Online Appendix

We created an online appendix to serve as a companion to this chapter, which contains both scripts and select files to assist you in completing this tutorial. Please find the repository at `https://github.com/rtraborn/MMB_appendix` (*see* **Note 2**).

2.5 Installation of R packages

For installation of the software listed above, please follow the instructions provided by each respective package. Part of our analysis will require the use of R packages found in the Bioconductor suite [27]. To install Bioconductor, please type the following from an R console:

```
source("https://bioconductor.org/biocLite.R")
biocLite()
```

We will use the R package *TSRchitect* to identify promoters from aligned RAMPAGE libraries. Prior to running the analysis, it will be necessary to install a series of prerequisite packages to *TSRchitect* from Bioconductor. Please install these packages as follows (as before, from an R console):

```
source("https://bioconductor.org/biocLite.R")
biocLite(c("AnnotationHub", "BiocGenerics", "BiocParallel",
 "ENCODExplorer",  "GenomicAlignments", "GenomeInfoDb",
 "GenomicRanges", "IRanges", "methods",
 "Rsamtools", "rtracklayer", "S4Vectors",
 "SummarizedExperiment"))
```

To install *TSRchitect*, please type the following from an R console:

```
source("https://bioconductor.org/biocLite.R")
biocLite("TSRchitect")
```

Finally, please confirm that TSRchitect has been installed correctly by loading it from your R console as follows:

```
library(TSRchitect) #installing TSRchitect
```

3 Methods

3.1 Retrieving the RAMPAGE sequence data from NCBI

To begin our analysis, we must download the RAMPAGE data to our workstation. We will utilize tools provided by the SRA Toolkit, which should already be installed on your machine (see **Materials**). The command *fastq-dump* allows one to directly retrieve data from the GEO database using the appropriate identifier(s). While there are 36 RAMPAGE libraries in the Batut *et al.* paper, we will select a subset of these to analyze here. We will compare samples from selected embryonic (E01h-E03h) and larval (L1-L3) tissues, representing the beginning and end of embryonic development. For more information about the experiment and the available RAMPAGE libraries, please see the following link: https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP011193.

First, let's proceed with downloading the libraries from early embryonic tissues (*see* **See Note 3**). We will make a new folder (entitled `"fastq_files/"`) to house these files.

```
mkdir fastq_files
cd fastq_files

fastq-dump --split-files SRR424683
fastq-dump --split-files SRR424684
fastq-dump --split-files SRR424685
```

We continue by downloading the data from late larval tissues.

```
fastq-dump --split-files SRR424707
fastq-dump --split-files SRR424708
fastq-dump --split-files SRR424709
```

Once the download of the aforementioned files are complete, you should see a total of 12 (6 *x* 2) separate fastq files in your current working directory:

```
ls -l *.fastq | wc -l
cd ..
```

3.2 Creating symlinks to the files

Our workflow expects fastq files that have the format "*.R1/R2.clipped.fq". Rather than rename them, we can simply create brand new symbolic links (symlinks) to the files, as follows:

```
cd ..
mkdir -p output/reads/clipped
cd output/reads/clipped

#embryonic libraries
ln -s ../../../fastq-files/SRR424683_1.fastq E01h.R1.clipped.fq
ln -s ../../../fastq-files/SRR424683_2.fastq E01h.R2.clipped.fq
ln -s ../../../fastq-files/SRR424684_1.fastq E02h.R1.clipped.fq
ln -s ../../../fastq-files/SRR424684_2.fastq E02h.R2.clipped.fq
ln -s ../../../fastq-files/SRR424685_1.fastq E03h.R1.clipped.fq
ln -s ../../../fastq-files/SRR424685_2.fastq E03h.R2.clipped.fq

#larval libraries
```

```
ln -s ../../../fastq-files/SRR424707_1.fastq L1.R1.clipped.fq
ln -s ../../../fastq-files/SRR424707_2.fastq L1.R2.clipped.fq
ln -s ../../../fastq-files/SRR424708_1.fastq L2.R1.clipped.fq
ln -s ../../../fastq-files/SRR424708_2.fastq L2.R2.clipped.fq
ln -s ../../../fastq-files/SRR424709_1.fastq L3.R1.clipped.fq
ln -s ../../../fastq-files/SRR424709_2.fastq L3.R2.clipped.fq

cd ../../.. #returning to the output directory
```

3.3 Downloading genomic data from *D. melanogaster*

Now that we have the fastq files from the RAMPAGE libraries downloaded and named appropriately, we now must retrieve the genome assembly and rRNA sequences from *D. melanogaster*. The genome assembly is required for aligning the RAMPAGE reads, and the rRNA sequences are required to filter out matching reads in the sequenced RAMPAGE libraries, since our sample is intended to contain only capped RNA transcripts. Please download the rRNA sequences from the link we provide below. These sequences were retrieved separately from Genbank at the NCBI database.

To retrieve the genome assembly from the ENSEMBL database, please do the following:

```
mkdir genome
cd genome
wget ftp://ftp.ensembl.org/pub/release-78/fasta/drosophila_melanogaster/dna/Drosophila_m
#uncompressing the file
gzip -d Drosophila_melanogaster.BDGP5.dna.toplevel.fa.gz
cd ..
```

Please navigate to the rRNA file "`Dmel_rRNA.fasta`" found in the Appendix.

```
head -n 3
>ref|NR_133562.1| Drosophila melanogaster 28S ribosomal RNA (28SrRNA:CR45844), rRNA
TTATATACAACCTCAACTCATATGGGACTACCCCCTGAATTTAAGCATATTAATTAGGGGAGGAAAAGAA
ACTAACAAGGATTTTCTTAGTAGCGGCGAGCGAAAAGAAAACAGTTCAGCACTAAGTCACTTTGTCTATA
```

3.4 Filtering and alignment of RAMPAGE reads using GoRAMPAGE

At this stage we are ready to commence with the rRNA filtering and alignment of the RAMPAGE libraries. We will use GoRAMPAGE, a tool we developed, to perform these tasks in a concerted workflow. GoRAMPAGE runs TagDust [23] to remove rRNA and low-complexity reads, and uses STAR [25] to align RAMPAGE (or other paired-end) reads to a given genome assembly.

Setting up the GoRAMPAGE job. Please refer to the script "`GoRAMPAGE_script_MMB.sh`" and (using a text editor) provide the appropriate paths to the genome assembly, output directory (see above) and rRNA sequences (*see* **Note 4**). GoRAMPAGE jobs can optionally be run in parallel (*see* **Note 5**). The script can be executed as follows:

```
#vi GoRAMPAGE_script_MMB.sh #updating with a text editor
./GoRAMPAGE_script_MMB.sh
```

If everything is working correctly you should start to see the results of the job being written to the file "errScript". You can inspect the progress during the run using the *less* command.

```
less -S errScript
```

Should the run fail before completion, any associated error messages will be printed to the errScript file. Once the job is complete, you should see the message "GoRAMPAGE job is complete!" appear on the command-line terminal.

Inspecting the rRNA filtering results. To evaluate the results from Step 3 (rRNA filtering), please navigate to the top level of the "output" directory and open the file "LOGFILES". You'll see the recorded progress of the program Tagdust and a record of the results. We notice that (for the L3h library) 1046448 of reads (78.1%) were "extracted", meaning that slightly more than 20% of reads were removed because of matches with ribosomal sequences. The removed reads from all libraries are found in the "`dusted_discard`" directory, and the extracted reads are found in the current directory. Due to their sheer abundance within cells, ribosomal RNA sequences are an inevitable contaminant within TSS profiling libraries. For analysis purposes, it is important that these sequences be removed, which is what has been completed here.
Since this step was conducted appropriately, we can proceed to the next step.

Evaluating the alignments. The folder "alignments/" in your GoRAMPAGE output folder will now contain 6 .bam files, each representing the distinct RAMPAGE libraries selected for our analysis. Typing "ls -l" from the command line will show that these files are symlinks to the original alignment files found in the "STARoutput/" directory. "STARoutput/", as its name suggests, contains the output from the STAR alignment, and this includes the alignment files "*.sortedByCoord.out.bam", and four additional log files. The files with the suffix "*.STAR.Log.final.out" each contain a summary of the alignment, such as the number of input reads, the percentage of uniquely-mapped reads and the percentage of unmapped reads. An inspection of these log files indicates that the alignments have similar mapping rates (70-80%), a reasonable outcome for our purposes.

Now that our RAMPAGE libraries are filtered and aligned, we can commence with the second half of our analysis.

3.5 Promoter identification from aligned RAMPAGE libraries

We can now use the prepared alignment files to identify TSSs and promoters from the selected RAMPAGE libraries. There are currently several tools available for this purpose. *CAGEr*, developed by Haberle [28], was utilized to perform TSS identification as part of the FANTOM5 efforts. We will use *TSRchitect* in this demonstration, since it was specifically designed to analyze paired-end TSS profiling datasets, and also because it is more flexible with respect to model system (*i.e.* it does not require a corresponding *BSGenome* package). The latter feature will be helpful when analyzing the non-*D. melanagaster* TSS profiling datasets that we expect to be generated in the near future.

Setting up the Analysis. *TSRchitect*, the package we'll use for this analysis, is an R package available in the Bioconductor suite of genomics tools [27]. It makes use of existing packages and data structures within this environment, where available, to identify promoters from sequence alignments. Since you have already installed *TSRchitect* and its dependencies (see section 2.3), we are set to proceed.

There are two general ways one can choose to run *TSRchitect*. The first is interactively *i.e.* typing the instructions directly into an R console. While this is a perfectly acceptable way to run analyses using package, for larger jobs it will likely be more efficient (and likely more reproducible) to run a dedicated R script. We have provided a sample script `"MMB_chapter_TSRchitect.R"` to make it easier for you to set up an R script. In the section to follow, we will go through the output of the analysis. For further details on how to use *TSRchitect*, please see its documentation at its Bioconductor page found here: https://www.bioconductor.org/packages/release/bioc/html/TSRchitect.html.

Running the Analysis. To run TSRchitect using the batch script, provide full paths for the variables "BAMDIR" and "DmAnnot" in the script provided (*see* **Note 6**). *BAMDIR* should be a path to the subdirectory "alignments/" in RAMPAGE output directory you specified earlier, and *DmAnnot* should be a full path to the *D. melanogaster* gene annotation listed above.

Once this is complete, we can run the batch script from the Linux command-line as follows:

```
R CMD BATCH MMB_chapter_TSRchitect.R
#assumes variables BAMDIR and DmAnnot have already been set
bg #puts this job in the background
```

Once the job is underway, you can monitor its progress by looking at the contents of the .Rout file (in this case, `"MMB_chapter_TSRchitect.Rout"`). The job should complete within an hour on most systems.

Reviewing the *TSRchitect* script. Before we evaluate the results (which will have been written to your working directory after running the batch script), there are some important aspects of the analysis to review. We discuss these for informational purposes only; it will not necessary to perform these commands separate from the batch script provided. First, we must initialize the *tssObject* (which stores the information about the experiment) appropriately (*see* **Note 7**).

The inputs in this case are BAM files (*inputType*="bam"); *TSRchitect* also accepts input in BED format.

```
DmRAMPAGE <- loadTSSobj(experimentTitle = "RAMPAGE Tutorial", \
 inputDir=BAMDIR, inputType="bam", isPairedEnd=TRUE, \
 sampleNames=c("E1h","E2h", "E3h", "L1", "L2", "L3"), \
 replicateIDs=c(1,1,1,2,2,2))
```

A critical step in our analysis is identifying TSRs from the aligned TSS data; to do this we use the function *determineTSR*. We have selected the job to run on 4 cores in this example (*n.cores*=4). Please enter the number of cores appropriate for your system. Because we want to identify TSRs from every one of the selected RAMPAGE libraries, we specify *tssSet*="all". The parameter *tagCountThreshold* was set to 25, meaning that only TSSs supported by 25 or more $5'$ RAMPAGE reads will be included within a TSR. Setting *writeTable* to "TRUE" means that the identified TSRs from each set will be written to the working directory.

```
DmRAMPAGE <- determineTSR(experimentName=DmRAMPAGE, n.cores=4, \
 tsrSetType="replicates", tssSet="all", tagCountThreshold=25, \
 clustDist=20, writeTable=TRUE)
```

TSRchitect can incorporate the tag abundances from each of the samples and append them to the list of identified TSRs. This is useful for downstream analysis of differential expression.

```
DmRAMPAGE <- addTagCountsToTSR(experimentName=DmRAMPAGE, \
tsrSetType="replicates",  tsrSet=1, tagCountThreshold=10, \
 writeTable=TRUE)
```

We can use *TSRchitect* to import an annotation file (or, alternatively, use an existing one from *AnnotationHub*) and use it to associate our set of identified TSRs with coding genes. We can specify the maximum distances (both up- and downstream) between the TSR and the annotation using the arguments *upstreamDist* and *downstreamDist*.

```
DmRAMPAGE <- importAnnotationExternal(experimentName=DmRAMPAGE, \
 fileType="gff3",  annotFile=DmAnnot)

DmRAMPAGE <- addAnnotationToTSR(experimentName=DmRAMPAGE, \
 tsrSetType="replicates", tsrSet=1, \
upstreamDist=1000, downstreamDist=200, feature="gene", \
 featureColumnID="ID", writeTable=TRUE)
```

Now we have generated a set of identified TSSs, TSRs from all 6 RAMPAGE libraries, and have associated the identified TSRs with annotated genes. Next, we will merge the libraries into two samples according to condition: early embryonic (E1h, E2h, E3h) and late larval (L1, L2, L3) using the information we provided when we initialized the *tssObject* at the start of this section. After merging, we identify promoters i) within the merged samples and ii) within the entire dataset combined, and associate with the *D. melanogaster* gene annotation as described previously (not shown).

```
#merging the sample data into two groups
DmRAMPAGE <- mergeSampleData(DmRAMPAGE)

# ... identifying TSRs from the merged samples:
DmRAMPAGE <- determineTSR(experimentName=DmRAMPAGE, \
n.cores=4, tsrSetType="merged", \
 tssSet="all", tagCountThreshold=40, \
 clustDist=20, writeTable=TRUE)
```

Evaluating the results Our analysis using *TSRchitect* is now complete. Your working directory should now contain the following:

- TSSs from each sample *e.g.* TSSset-1.txt: (6)
- TSRs from each sample (in both .txt and .tab formats): (12)
- TSRs from each merged group (in both .txt and .tab formats): *e.g.* TSRsetMerged-1.txt: (4)
- TSRs from the combined set of TSSs: TSRsetCombined.tab: (1)

Let's briefly review the files (*see* **Note 8**). We can quickly obtain the counts on the command line, as follows:

```
wc -l *.tab
8377 TSRset-1.tab
6159 TSRset-2.tab
4814 TSRset-3.tab
17924 TSRset-4.tab
11851 TSRset-5.tab
3242 TSRset-6.tab
13986 TSRsetCombined.tab
7344 TSRsetMerged-1.tab
12126 TSRsetMerged-2.tab
85823 total
```

We will see that we have identified between roughly 3,200 and 18,000 TSRs within the individual RAMPAGE samples, which is attributable to the differences in library sizes. We detect 7,344 TSRs within the early embryonic samples ("TSRsetMerged-1.tab") and 12,126 TSRs in the late larval samples ("TSRsetMerged-2.tab"). Within the combined samples ("TSRsetCombined.tab") we find 13,986 TSRs, which is similar to the number reported by Hoskins *et. al.* [1].

In addition to identifying the position of a given TSRs, *TSRchitect* records other useful information about its properties. The *width* of a TSR refers the span of the genomic region it occupies (in bp), and the *Shape Index* (SI) is measure of the relative peakedness of the TSR. We can see an example of this in the file "TSRsetMerged-1.txt".

```
seq      start    end      strand  nTSSs   tsrWidth        shapeIndex      featureID
2L.67043.67044.+           2L      67043   67044   +       270     2       1       NA
2L.74089.74115.+           2L      74089   74115   +       341     27      0.13    NA
2L.94739.94752.+           2L      94739   94752   +       1650    14      0.55    FBgn0031
2L.102386.102386.+         2L      102386  102386  +       284     1       2       FBgn0031
```

3.6 Summary

The workflow provided here is intended to serve as a useful entry point for the analysis of TSS profiling data in insects. On the computational side, we have provided an open source set of tools so that the uninitiated genome scientist can begin to analyze RAMPAGE (or other forms of TSS profiling data) quickly. While the analysis centered on *D. melanogaster* via the use of public datasets, it is anticipated that this will assist groups who may be interested in performing TSS profiling in their preferred insect model system.The application of TSS profiling technology across a more representative sample of insect diversity will improve our understanding of the positions and general structure *cis*-regulatory regions in this phylum.

3.7 Figures

4 Notes

1. Please consult the GoRAMPAGE documentation found here:
   https://github.com/BrendelGroup/GoRAMPAGE.
   Installation instructions for the prerequisites of GoRAMPAGE (which includes some of the items listed) are found at the following link:
   https://github.com/BrendelGroup/GoRAMPAGE/tree/master/src.
2. You can clone this appendix to your workspace on the command line using git, as follows:

   ```
   git clone https://github.com/rtraborn/MMB_appendix.git
   ```

   The "scripts/" folder in the Appendix contains code for you to run the two major workflows described in this chapter. The `"additional_files/"` folder contains the following files which are necessary for the analysis: i) a fasta file containing ribosomal RNA sequences for *D. melanogaster* (`Dmel_rRNA.fasta`) and ii) a gene annotation for *D. melanogaster* (`Drosophila_melanogaster.BDGP5.78.gff`).

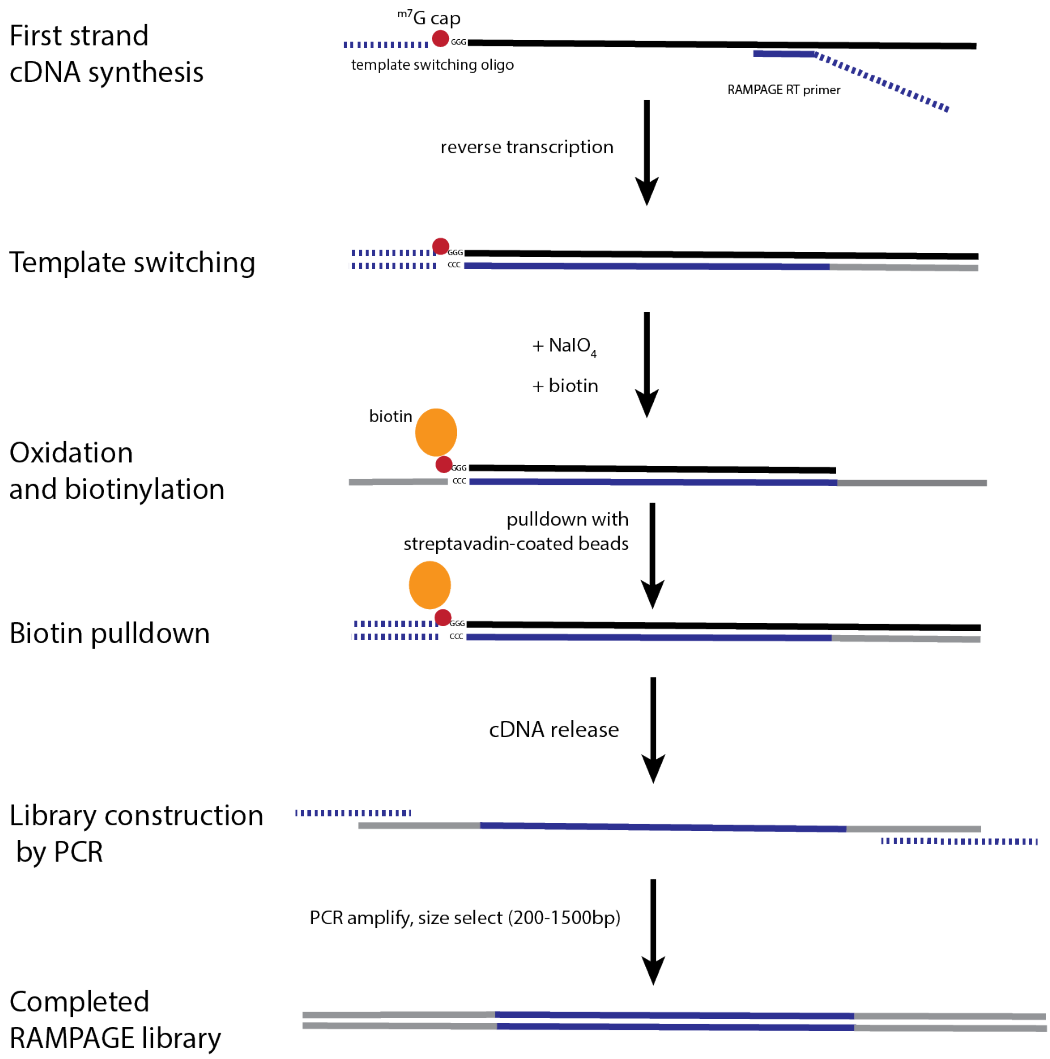

Fig. 1. A brief summary of the RAMPAGE protocol. Starting with high-quality total RNA, first-strand cDNA synthesis is initiated using a cap-bound oligonucleotide and a custom RAMPAGE RT primer, creating a double-stranded DNA-RNA hybrid molecule. Next, the 5′-m7G cap is oxidized, bound with biotin and pulled down with streptavadin-coated beads. The single-stranded cDNA molecules is released and the final RAMPAGE library construction is completed with PCR using custom oligonucleotides, followed by size-selection. This illustration was adapted from [18].

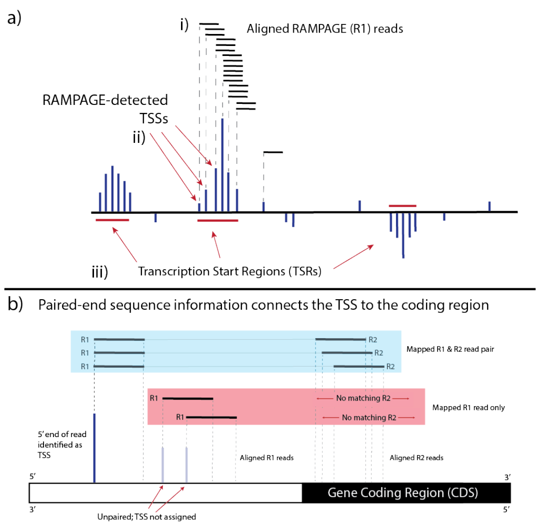

Fig. 2. An overview of promoter identification using RAMPAGE. a) RAMPAGE reads are aligned to the genome. The 5′-most genomic coordinate from each properly-paired R1 read is estimated as a TSS. The ambundance of mapped 5′-ends at a given TSS is a measure of its abundance. TSSs above a minimum threshold will be clustered into TSRs. b) RAMPAGE-derived Paired-end sequence information provides a connection between a 5′-mRNA end and a gene coding region. Only properly-paired R1 reads (*i.e.* with an aligned R2 read) are identified as TSSs and then included in the downstream clustering procedure described in part *a*.

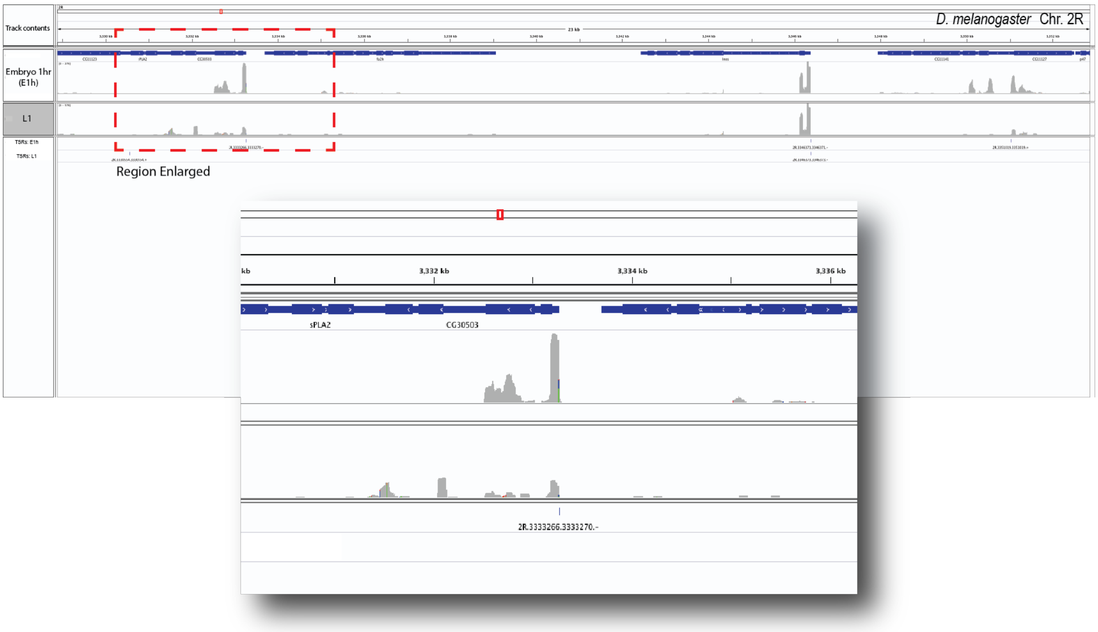

Fig. 3. Test caption for Figure 3

3. Since these fastq files are paired-end, we use the argument *–split-files* to generate separate files for each read pair.

4. If you are running this on a cluster with a job scheduler you'll need to add the necessary headers to the top of the script and submit the job in the appropriate manner.

5. For parallel execution, GoRAMPAGE uses the Linux package *GNU parallel* [29]. Please see the GoRAMPAGE documentation for more information.

6. To do this, please edit the batch script `TSRchitect_script_MMB.R` with a text editor of your choice.

7. Because the samples provided derive from related developmental stages, we will merge them for annotation purposes using the argument *replicateIDs*, (though it must be emphasized that they are not replicates).

8. All of *TSRchitect's* output files are labeled according to the order that they are loaded onto the *tssObject*. For example, *TSSset-1.txt* corresponds to the first RAMPAGE dataset (in our case E1h), and *TSSset-2.txt* corresponds to the second RAMAPGE dataset (for this example E2h), and so on. You can check which datasets are loaded on the *tssObject* by simply entering it on an R console. Please see the *TSRchitect* documentation for more information.

Acknowledgments

The authors would like to thank Philippe Batut for generous technical assistance with the RAMPAGE protocol, and to Nathan Keith for his help establishing the protocol in our laboratory.

Disclosure Declaration

The authors declare that they have no competing interests.

5 References

References

1. R. A. Hoskins, R. A. Hoskins, J. M. Landolin, J. M. Landolin, J. B. Brown, J. B. Brown, J. E. Sandler, J. E. Sandler, H. Takahashi, H. Takahashi, T. Lassmann, T. Lassmann, C. Yu, C. Yu, B. W. Booth, B. W. Booth, D. Zhang, D. Zhang, K. H. Wan, K. H. Wan, L. Yang, L. Yang, N. Boley, N. Boley, J. Andrews, J. Andrews, T. C. Kaufman, T. C. Kaufman, B. R. Graveley, B. R. Graveley, P. J. Bickel, P. J. Bickel, P. Carninci, J. W. Carlson, J. W. Carlson, S. E. Celniker, and S. E. Celniker, "Genome-wide analysis of promoter architecture in Drosophila melanogaster." *Genome Research*, vol. 21, no. 2, pp. 182–192, Feb. 2011.
2. P. J. Batut, A. Dobin, C. Plessy, P. Carninci, and T. R. Gingeras, "High-fidelity promoter profiling reveals widespread alternative promoter usage and transposon-driven developmental gene expression." *Genome Research*, Aug. 2012.
3. V. P. Brendel and R. T. Raborn, "Gorampage- a workflow for promoter detection by 5'-read mapping," https://github.com/brendelGroup/GoRAMPAGE, 2016.
4. R. T. Raborn and V. Brendel, *TSRchitect: Promoter identification from large-scale TSS profiling data*, 2017, r Bioconductor package version 1.0.0. [Online]. Available: http://bioconductor.org/packages/release/bioc/html/TSRchitect.html
5. J. T. Kadonaga, "Perspectives on the RNA polymerase II core promoter." *Wiley Interdisciplinary Reviews: Developmental Biology*, vol. 1, no. 1, pp. 40–51, Jan. 2012.
6. R. Kodzius, M. Kojima, H. Nishiyori, M. Nakamura, S. Fukuda, M. Tagami, D. Sasaki, K. Imamura, C. Kai, M. Harbers, Y. Hayashizaki, and P. Carninci, "CAGE: cap analysis of gene expression." *Nature Methods*, vol. 3, no. 3, pp. 211–222, Mar. 2006.
7. P. Carninci, T. Kasukawa, S. Katayama, J. Gough, M. C. Frith, N. Maeda, R. Oyama, T. Ravasi, B. Lenhard, C. Wells, R. Kodzius, K. Shimokawa, V. B. Bajic, S. E. Brenner, S. Batalov, A. R. R. Forrest, M. Zavolan, M. J. Davis, L. G. Wilming, V. Aidinis, J. E. Allen, A. Ambesi-Impiombato, R. Apweiler, R. N. Aturaliya, T. L. Bailey, M. Bansal, L. Baxter, K. W. Beisel, T. Bersano, H. Bono, A. M. Chalk, K. P. Chiu, V. Choudhary, A. Christoffels, D. R. Clutterbuck, M. L. Crowe, E. Dalla, B. P. Dalrymple, B. de Bono, G. Della Gatta, D. di Bernardo, T. Down, P. Engstrom, M. Fagiolini, G. Faulkner, C. F. Fletcher, T. Fukushima, M. Furuno, S. Futaki, M. Gariboldi, P. Georgii-Hemming, T. R. Gingeras, T. Gojobori, R. E. Green, S. Gustincich, M. Harbers, Y. Hayashi, T. K. Hensch, N. Hirokawa, D. Hill, L. Huminiecki, M. Iacono, K. Ikeo, A. Iwama, T. Ishikawa, M. Jakt, A. Kanapin,

M. Katoh, Y. Kawasawa, J. Kelso, H. Kitamura, H. Kitano, G. Kollias, S. P. T. Krishnan, A. Kruger, S. K. Kummerfeld, I. V. Kurochkin, L. F. Lareau, D. Lazarevic, L. Lipovich, J. Liu, S. Liuni, S. McWilliam, M. Madan Babu, M. Madera, L. Marchionni, H. Matsuda, S. Matsuzawa, H. Miki, F. Mignone, S. Miyake, K. Morris, S. Mottagui-Tabar, N. Mulder, N. Nakano, H. Nakauchi, P. Ng, R. Nilsson, S. Nishiguchi, S. Nishikawa, F. Nori, O. Ohara, Y. Okazaki, V. Orlando, K. C. Pang, W. J. Pavan, G. Pavesi, G. Pesole, N. Petrovsky, S. Piazza, J. Reed, J. F. Reid, B. Z. Ring, M. Ringwald, B. Rost, Y. Ruan, S. L. Salzberg, A. Sandelin, C. Schneider, C. Schönbach, K. Sekiguchi, C. A. M. Semple, S. Seno, L. Sessa, Y. Sheng, Y. Shibata, H. Shimada, K. Shimada, D. Silva, B. Sinclair, S. Sperling, E. Stupka, K. Sugiura, R. Sultana, Y. Takenaka, K. Taki, K. Tammoja, S. L. Tan, S. Tang, M. S. Taylor, J. Tegner, S. A. Teichmann, H. R. Ueda, E. van Nimwegen, R. Verardo, C. L. Wei, K. Yagi, H. Yamanishi, E. Zabarovsky, S. Zhu, A. Zimmer, W. Hide, C. Bult, S. M. Grimmond, R. D. Teasdale, E. T. Liu, V. Brusic, J. Quackenbush, C. Wahlestedt, J. S. Mattick, D. A. Hume, C. Kai, D. Sasaki, Y. Tomaru, S. Fukuda, M. Kanamori-Katayama, M. Suzuki, J. Aoki, T. Arakawa, J. Iida, K. Imamura, M. Itoh, T. Kato, H. Kawaji, N. Kawagashira, T. Kawashima, M. Kojima, S. Kondo, H. Konno, K. Nakano, N. Ninomiya, T. Nishio, M. Okada, C. Plessy, K. Shibata, T. Shiraki, S. Suzuki, M. Tagami, K. Waki, A. Watahiki, Y. Okamura-Oho, H. Suzuki, J. Kawai, Y. Hayashizaki, F. Consortium, R. G. E. R. Group, and G. S. G. G. N. P. C. Group, "The transcriptional landscape of the mammalian genome," *Science (New York, NY)*, vol. 309, no. 5740, pp. 1559–1563, Sep. 2005.

8. E. A. Rach, H.-Y. Yuan, W. H. Majoros, P. Tomancak, and U. Ohler, "Motif composition, conservation and condition-specificity of single and alternative transcription start sites in the Drosophila genome." *Genome Biology*, vol. 10, no. 7, p. R73, 2009.
9. B. Lenhard, A. Sandelin, and P. Carninci, "Metazoan promoters: emerging characteristics and insights into transcriptional regulation." *Nature Reviews Genetics*, vol. 13, no. 4, pp. 233–245, Apr. 2012.
10. T. Ni, D. L. Corcoran, E. A. Rach, S. Song, E. P. Spana, Y. Gao, U. Ohler, and J. Zhu, "A paired-end sequencing strategy to map the complex landscape of transcription initiation." *Nature Methods*, vol. 7, no. 7, pp. 521–527, Jul. 2010.
11. U. Ohler, G.-c. Liao, H. Niemann, and G. M. Rubin, "Computational analysis of core promoters in the Drosophila genome." *Genome Biology*, vol. 3, no. 12, pp. research0087.1–0087.12, 2002.
12. R. T. Raborn, K. Spitze, V. P. Brendel, and M. Lynch, "Promoter Architecture and Sex-Specific Gene Expression in Daphnia pulex." *Genetics*, vol. 204, no. 2, pp. 593–612, Aug. 2016.
13. C. Nepal, Y. Hadzhiev, C. Previti, V. Haberle, N. Li, H. Takahashi, A. M. M. Suzuki, Y. Sheng, R. F. Abdelhamid, S. Anand, J. Gehrig, A. Akalin, C. E. M. Kockx, A. A. J. van der Sloot, W. F. J. van IJcken, O. Armant, S. Rastegar, C. Watson, U. Strahle, E. Stupka, P. Carninci, B. Lenhard, and F. Muller, "Dynamic regulation of the transcription initiation landscape at single nucleotide resolution during vertebrate embryogenesis," *Genome Research*, vol. 23, no. 11, pp. 1938–1950, Nov. 2013.
14. P. Carninci, A. Sandelin, B. Lenhard, S. Katayama, K. Shimokawa, J. Ponjavic, C. A. M. Semple, M. S. Taylor, P. G. Engström, M. C. Frith, A. R. R. Forrest, W. B. Alkema, S. L. Tan, C. Plessy, R. Kodzius, T. Ravasi, T. Kasukawa, S. Fukuda, M. Kanamori-Katayama, Y. Kitazume, H. Kawaji, C. Kai, M. Naka-

mura, H. Konno, K. Nakano, S. Mottagui-Tabar, P. Arner, A. Chesi, S. Gustincich, F. Persichetti, H. Suzuki, S. M. Grimmond, C. A. Wells, V. Orlando, C. Wahlestedt, E. T. Liu, M. Harbers, J. Kawai, V. B. Bajic, D. A. Hume, and Y. Hayashizaki, "Genome-wide analysis of mammalian promoter architecture and evolution," *Nature Genetics*, vol. 38, no. 6, pp. 626–635, Apr. 2006.

15. S. Mwangi, G. Attardo, Y. Suzuki, S. Aksoy, and A. Christoffels, "TSS seq based core promoter architecture in blood feeding Tsetse fly (Glossina morsitans morsitans) vector of Trypanosomiasis," *BMC Genomics*, vol. 16, no. 1, p. 722, Sep. 2015.
16. K. Tsuchihara, Y. Suzuki, H. Wakaguri, T. Irie, K. Tanimoto, S.-i. Hashimoto, K. Matsushima, J. Mizushima-Sugano, R. Yamashita, K. Nakai, D. Bentley, H. Esumi, and S. Sugano, "Massive transcriptional start site analysis of human genes in hypoxia cells," *Nucleic Acids Research*, vol. 37, no. 7, pp. 2249–2263, Apr. 2009.
17. N. Cvetesic and B. Lenhard, "Core promoters across the genome," *Nature Biotechnology*, vol. 35, no. 2, pp. 123–124, Feb. 2017.
18. P. J. Batut and T. R. Gingeras, "RAMPAGE: Promoter Activity Profiling by Paired-End Sequencing of 5'-Complete cDNAs." in *Current Protocols in Molecular Biology.* Current protocols in molecular biology / edited by Frederick M Ausubel [et al], 2013, pp. 25B.11.1–25B.11.16.
19. N. Merchant, E. Lyons, S. Goff, M. Vaughn, D. Ware, D. Micklos, and P. Antin, "The iPlant Collaborative: Cyberinfrastructure for Enabling Data to Discovery for the Life Sciences." *PLoS Biology*, vol. 14, no. 1, p. e1002342, Jan. 2016.
20. R. Leinonen, H. Sugawara, M. Shumway, and International Nucleotide Sequence Database Collaboration, "The sequence read archive." *Nucleic Acids Research*, vol. 39, no. Database issue, pp. D19–21, Jan. 2011.
21. E. Aronesty, "Comparison of Sequencing Utility Programs," *The Open Bioinformatics Journal*, vol. 7, no. 1, pp. 1–8, Jan. 2013.
22. H. Lab, "FASTX Toolkit." [Online]. Available: `http://hannonlab.cshl.edu/fastx_toolkit/`
23. T. Lassmann, "TagDust2: a generic method to extract reads from sequencing data," *BMC Bioinformatics*, vol. 16, no. 1, p. 1, Jan. 2015.
24. H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. R. Abecasis, R. Durbin, and 1000 Genome Project Data Processing Subgroup, "The Sequence Alignment/Map format and SAMtools," *Bioinformatics (Oxford, England)*, vol. 25, no. 16, pp. 2078–2079, Aug. 2009.
25. A. Dobin and T. R. Gingeras, "Optimizing RNA-Seq Mapping with STAR," in *Transcription Factor Regulatory Networks.* New York, NY: Springer New York, Apr. 2016, pp. 245–262.
26. R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2017. [Online]. Available: https://www.R-project.org
27. M. Lawrence and M. Morgan, "Scalable Genomics with R and Bioconductor," *Statistical Science*, vol. 29, no. 2, pp. 214–226, May 2014.
28. V. Haberle, A. R. R. Forrest, Y. Hayashizaki, P. Carninci, and B. Lenhard, "CAGEr: precise TSS data retrieval and high-resolution promoterome mining for integrative analyses." *Nucleic Acids Research*, vol. 43, no. 8, pp. gkv054–e51, Feb. 2015.
29. O. Tange, "Gnu parallel - the command-line power tool," *;login: The USENIX Magazine*, vol. 36, no. 1, pp. 42–47, Feb 2011. [Online]. Available: http://www.gnu.org/s/parallel

6 Checklist of Items to be Sent to Volume Editors

Here is a checklist of everything the volume editor requires from you:

- [ ] The final LaTeX source files
- [ ] A final PDF file
- [ ] A copyright form, signed by one author on behalf of all of the authors of the paper.
- [ ] A readme giving the name and email address of the corresponding author.