Chapter 20

A Guide to the Chloroplast Transcriptome Analysis Using RNA-Seq

Elena J. S. Michel, Amber M. Hotto, Susan R. Strickler, David B. Stern, and Benoît Castandet

Abstract

Since its first use in plants in 2007, high-throughput RNA sequencing (RNA-Seq) has generated a vast amount of data for both model and nonmodel species. Organellar transcriptomes, however, are virtually always overlooked at the data analysis step. We therefore developed ChloroSeq, a bioinformatic pipeline aimed at facilitating the systematic analysis of chloroplast RNA metabolism, and we provide here a step-by-step user's manual. Following the alignment of quality-controlled data to the genome of interest, ChloroSeq measures genome expression level along with splicing and RNA editing efficiencies. When used in combination with the Tuxedo suite (TopHat and Cufflinks), ChloroSeq allows the simultaneous analysis of organellar and nuclear transcriptomes, opening the way to a better understanding of nucleus–organelle cross talk. We also describe the use of R commands to produce publication-quality figures based on ChloroSeq outputs. The effectiveness of the pipeline is illustrated through analysis of an RNA-Seq dataset covering the transition from growth to maturation to senescence of *Arabidopsis thaliana* leaves.

Key words RNA-Seq, ChloroSeq, Chloroplast, Organelles, Leaf development

1 Introduction

RNA-Seq has become the method of choice for transcriptome studies, largely supplanting microarrays. It generates more precise measurements of transcripts and isoforms, offers greater coverage and lower cost than arrays or real-time RT-PCR and above all, facilitates understanding well beyond traditional differential gene expression (DGE) analyses, since prior knowledge of the genes is not necessary [1–3]. Moreover, a wide array of computational tools and methods has been developed to process and analyze transcriptome data. For example, the Tuxedo suite (Bowtie, Tophat, Cufflinks) [4] democratized DGE analysis with the publication of step-by-step protocols aimed at researchers with little or no bioinformatics experience [5, 6]. The development of the Galaxy web

Eric Maréchal (ed.), *Plastids: Methods and Protocols*, Methods in Molecular Biology, vol. 1829,
https://doi.org/10.1007/978-1-4939-8654-5_20,

interface is another example of how RNA-Seq analyses have become increasingly accessible [7].

Given analytical power along with cost and accessibility factors, it is not surprising that the stockpile of RNA-Seq datasets has increased exponentially over the past decade (see https://www.ncbi.nlm.nih.gov/sra for the number of publicly available experiments). In plants, besides DGE, RNA-Seq has been used, for example, to study RNA secondary structure and its interactions with proteins [8–10], polyadenylation patterns [11], the translational landscape [12–14], to facilitate the discovery of long noncoding RNAs [15], and to discern the relative contributions to the transcriptome of different subgenomes in polyploid species [16].

Although this deluge of discoveries represents an unprecedented opportunity to deepen our understanding of their expression [17], organellar genomes have been to this point virtually invisible. Why is this the case? First, as mRNA represents only a few percent of total cellular RNA [18–20], most RNA-Seq library preparation protocols begin with enrichment of polyadenylated transcripts, to deplete the highly abundant rRNA [21]. Plant organellar (chloroplast and mitochondria) mRNAs, however, are targeted for degradation when polyadenylated [22, 23], and are therefore grossly underrepresented in libraries prepared following this step [24]. Therefore, any analysis of remaining organellar transcripts is likely to be highly biased. An alternative is to selectively deplete rRNAs by hybridization to complementary oligonucleotides, a strategy that is increasingly used because it gives access to the nonpolyadenylated/non-rRNA component of the transcriptome, for example noncoding RNAs [15, 21, 25].

A second major reason for the paucity of organellar RNA-Seq analysis is that most RNA-Seq computational tools are designed around transcript quantification and are therefore not suited to the peculiarities of organellar gene expression. In chloroplasts and mitochondria, post-transcriptional events such as RNA editing, splicing and trimming play a critical role not only in determining the abundance of an mRNA, but also its size and functionality [26–28]. For example, DGE analysis alone for a mutant lacking chloroplast mini-ribonuclease III in *Arabidopsis thaliana* did identify differential expression of the 4.5S rRNA, but failed to reveal other processing defects at the 5′ and 3′ termini of other rRNAs, the accumulation of a noncoding RNA complementary to the 4.5-5S rRNA intergenic region, and a defect in spliced intron degradation [29]. Similarly, mutants for the chloroplast isozyme of the 3′→5′ exoribonuclease polynucleotide phosphorylase, which have been shown to overaccumulate tRNAs, plastid noncoding RNAs, and excised introns, display largely similar mRNA abundance compared to WT, although the transcripts have widespread 3’ terminal extensions [30–33]. These missed opportunities were part of the motivation for

developing ChloroSeq [24, 34], a pipeline that reveals in a concerted way editing, splicing and relative transcript abundance. For an alternative strategy using an in depth statistical approach we refer the reader to the protocol from Malbert et al. (*see* Chapter 19) in this same issue. Given the importance and complexity of retrograde signaling [35], the side-by-side analysis of organellar and nuclear transcripts is also a powerful application of ChloroSeq.

2 Materials

1. Although no coding knowledge is necessary to apply the methods explained here, an understanding of basic Unix commands is helpful.
2. Commands given in the Linux shell are denoted with a $ prefix, and commands given in R are denoted with a > prefix. The software can be run on a 64-bit Linux system, and at least 8 GB of memory is recommended. The software and data will be downloaded to a new directory named "download."
3. Run the command $ mkdir download. All software is freely available online and should be installed in a bin directory in the user's home directory. This bin directory has to be added to the PATH.
4. Run the command $ mkdir $HOME/bin
5. Run the command $ export PATH=$HOME/bin:$PATH. The data and files necessary for the analyses will eventually be placed in a working directory named "development".
6. Run the command $ mkdir development

2.1 Software and Tools

1. ChloroSeq: executable files can be downloaded from https://github.com/BenoitCastandet/chloroseq/tree/master/ChloroSeq_for_bin. Once downloaded to the download directory, the files need to be made executable.
2. Run the command $ chmod u+x *
3. The files are then moved to the bin folder.
4. Run the command $ mv * path-to-USER/bin
5. BEDTools: ChloroSeq relies on BEDTools [36] to extract reads that intersect with the genomic intervals of interest. BEDTools version 2.25 can be downloaded from https://github.com/arq5x/bedtools2/releases/download/v2.25.0/bedtools-2.25.0.tar.gz into the download directory. A step-by-step guide to install BEDTools has been published [36].
6. SAMTools: The SAMtools [37] are necessary for multiple steps in the protocol. The latest version can be downloaded from http://www.htslib.org/download/, and the same web page includes all installation information.

7. Bowtie2, TopHat2, Cufflinks, CummeRbund : Detailed steps for installing the Tuxedo suite tools have been published [5, 6]. The binaries from bowtie2 can be downloaded from https://sourceforge.net/projects/bowtie-bio/files/bowtie2/. For TopHat2, a link to the latest version of the binaries can be found at http://ccb.jhu.edu/software/tophat/index.shtml. Cufflinks binaries can be downloaded from http://cole-trapnell-lab.github.io/cufflinks/install/. Users should be sure to download binaries suitable to the operating system, as described here for a 64-bit Linux system. CummeRbund is in R script and documentation for the latest release can be found at http://bioconductor.org/packages/2.11/bioc/html/cummeRbund.html.
8. To install CummeRbund, first start an R session.
9. Run the command $ R
10. Then type the following commands:

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

11. SRA toolkit: Documentation for downloading and installing the SRA toolkit is available at https://trace.ncbi.nlm.nih.gov/Traces/sra/sra.cgi?view=toolkit_doc&f=std.
12. Reference genome: An indexed reference genome is required for Bowtie and TopHat. The *Arabidopsis thaliana* TAIR10 genome is used in this manuscript and can be downloaded from ftp://igenome:G3nom3s4u@ussd-ftp.illumina.com/Arabidopsis_thaliana/NCBI/TAIR10/Arabidopsis_thaliana_NCBI_TAIR10.tar.gz into the download directory. Once in this directory it needs to be unpacked:
13. Run the command $ tar -xvzf Arabidopsis_thaliana_NCBI_TAIR10.tar.gz
14. This creates an Arabidopsis thaliana directory. Although it expands in several subfolders, only the indexed genome is required in the example described here, and is copied into the development working directory (one hierarchical level higher).
15. Run the command $ cp -r Arabidopsis_thaliana/NCBI/TAIR10/Sequence/Bowtie2Index/ ../development/
16. The downloaded data also contains the chloroplast genome annotation file needed to run the analysis, however the *ycf3* gene is not correctly annotated. The corrected annotation file can be downloaded from https://github.com/BenoitCastandet/chloroseq/blob/master/TAIR10_genome_GFF3_genes.gff and then copied to the development directory.
17. Run the command $ cp TAIR10_genome_GFF3_genes.gff ../development/
18. ChloroSeq also calls annotation files containing splicing or editing sites coordinates, that can be downloaded from the

TAIR10_ChrC_files directory on the ChloroSeq web page https://github.com/BenoitCastandet/chloroseq, then copied to the development directory.

19. Run the command $ cp -r TAIR10_ChrC_files/ ../development/

2.2 RNA-Seq Data

The data used in this example is from [38], where the chloroplast, nuclear, and mitochondrial transcriptomes of *Arabidopsis thaliana* leaf tissue were examined from days 4 to 30. The dataset therefore spans leaf development, maturation and senescence. The raw RNA-Seq data are deposited under the series accession number GSE43616 in the Genome Expression Omnibus (GEO). There are two replicates for each time point, but only the second replicate will be used for this demonstration, for the sake of simplicity.

1. The easiest way to access the data is via the NCBI ftp site containing the individual RNA-Seq runs corresponding to this experiment, ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByStudy/sra/SRP/SRP018/SRP018034/. The 14 files corresponding to the runs SRR207985 to SRR2079798 should be copied to the download directory. The data are in the SRA format (.sra files) and should be converted into FASTQ files using the fastq-dump command from the SRA toolkit. As the data are paired ends, the --split-files option has to be used.
2. Run the command $ fastq-dump --split-files SRR*
3. Here, the * flag is used to denote that the command should be carried out for all the files in the current directory that start with SRR. This flag can only be used if all the files whose names start with SRR should be subject to the command. Otherwise, individual file names would need to replace the SRR*, and a separate command would need to be run for each file. Because the data are paired ends, two output files are created for every original .sra file. For example, the file SRR207985.sra produces the files SRR207985_1.fastq and SRR207985_2.fastq as outputs. These are the files to be used for the alignment, and should be moved to the development directory.
4. $ mv *.fastq ../development/
5. Although not done here, a quality control can be performed at this step on the fastq files (*see* **Note 1**).

3 Methods

3.1 Running TopHat

1. From the development working directory, the following commands are entered to map RNA-Seq reads to the *Arabidopsis* genome.
2. Run the command $ tophat2 -p 3 --no-novel-juncs -o day_4_tophat_out/ -G TAIR10_genome_files/TAIR10_genome_

GFF3_genes.gff TAIR10_genome_files/TAIR10_genome_bowtie2_index/TAIR10_genome SRR207985_1.fastq SRR207985_2.fastq

This is an example for the RNA-Seq experiment SRR207985, corresponding to the Day 4 time point.

3. The results are written in the 4days_tophat_out directory that is specified by the –o option. Three processing cores (-p 3) are used for the process, but this can be adapted according to the hardware available. In this case we are not looking for any new possible splicing junctions (--no-novel-juncs), therefore the alignment will only be performed according to the known splicing junctions specified in the file TAIR10_genome_GFF3_genes.gff after the -G option.
4. The output directory contains the accepted_hits.bam file that stores the aligned reads in a binary format, and will be used for the downstream analyses. The same command must be adapted and run for the other data points.
5. The alignment is performed on the full Arabidopsis genome as the final goal is to coanalyze the nuclear and chloroplast transcriptomes. It is possible, however, to align only to the chloroplast genome if desired (*see* **Note 2**).

3.2 Running ChloroSeq

1. By design, ChloroSeq works with single end reads. When working with paired end reads it is necessary to modify the bam file generated by TopHat. From the working directory enter the following commands:

```
$ cd day_4_tophat_out/
$ samtools view -f 147 -F 32 -b accepted_hits.bam > fw.bam
$ samtools view -f 163 -F 16 -b accepted_hits.bam > rv.bam
$ samtools merge single.bam fw.bam rv.bam
```

2. The first step is to move to the tophat_out directory of interest, here 4days_tophat_out. Samtools is then used to extract only the reads from the accepted_hits.bam file that were aligned correctly, only once, and were the first of each pair (flags -f 147 and -f 163), and excluded reads that were the second in a pair (flags -F 32 and -F 16) for both strands.
3. The two files created (fw.bam and rv.bam) are then merged in the final single.bam file that is ready to use in ChloroSeq. The values used for the flags depend on the strand specificity of the library used (*see* **Note 3**).
4. To run ChloroSeq, the following command is entered:

```
$ chloroseq.pl -a all -b single.bam -e ../TAIR10_ChrC_files/TAIR10_ChrC_
exon.gff3 -i
../TAIR10_ChrC_files/TAIR10_ChrC_introns.gff3 -g 128214 -n ChrC -f
../TAIR10_ChrC_files/TAIR10_ChrC_bowtie2_index/TAIR10_ChrC.fa -v
```

```
../TAIR10_ChrC_files/TAIR10_ChrC_editing_sites.gff3 -s
../TAIR10_ChrC_files/TAIR10_ChrC_splice_sites_sort.gff3
```

5. ChloroSeq performs three different analyses that can be run simultaneously or individually (-a option, *see* **Note 4**). The genome size entered (-g 128214) is the size of the full chloroplast genome minus one inverted repeat, because the two inverted repeat copies are 100% identical. The output will consist of six text files: nt_coverage.txt, window_coverage.txt, exon_rpkm.txt, intron_rpkm.txt, splicing_efficiency.txt and editing_efficiency.txt. The same process must be repeated for the 14 different alignments previously performed.

3.3 Running Cuffdiff

1. Cuffdiff is used to compute nuclear gene expression. From the development directory, the following command is entered:

```
$ cuffdiff -o development_diff_out -b
TAIR10_genome_files/TAIR10_genome_bowtie2_index/TAIR10_genome.fa -p 3 -u
TAIR10_genome_GFF3_genes.gff -L
day_4,day_6,day_8,day_10,day_12,day_14,day_16,day_18,day_20,day_22,day_24,
day_26,day_28,day_30 day_4_tophat_out/accepted_hits.bam
day_6_tophat_out/accepted_hits.bam day_8_tophat_out/accepted_hits.bam
day_10_tophat_out/accepted_hits.bam day_12_tophat_out/accepted_hits.bam
day_14_tophat_out/accepted_hits.bam day_16_tophat_out/accepted_hits.bam
day_18_tophat_out/accepted_hits.bam day_20_tophat_out/accepted_hits.bam
day_22_tophat_out/accepted_hits.bam day_24_tophat_out/accepted_hits.bam
day_26_tophat_out/accepted_hits.bam day_28_tophat_out/accepted_hits.bam
day_30_tophat_out/accepted_hits.bam
```

2. The run creates a directory development_diff_out that summarizes information and results. For the remainder of the protocol, focus will be placed on the gene_exp.diff file, which contains the expression values (calculated in FPKM) for all the genes annotated in the TAIR10_genome_GFF3_genes.gff file.

3.4 Results

1. The publication originally linked to the datasets [38] used TopHat to map reads to the genome and applied the Savitzky–Golay method to smooth data and calculate differentially expressed genes. It was reported that the chloroplast transcriptome undergoes major changes during leaf aging that appear to correlate with changes in expression of nuclear genes coding for chloroplast-targeted proteins [38].
2. To add depth and breadth to these observations, the ChloroSeq and Cuffdiff outputs can be used with CummeRbund and ggplot2 (in R) to give an overall view of chloroplast genome expression, and to test for any correlation with nucleus-encoded chloroplast regulatory factors. To this end, a file containing identifiers of the known Arabidopsis nuclear genes

involved in chloroplast RNA metabolism is available (ChrC_RNA_metabolism.txt, accessible on the ChloroSeq webpage, https://github.com/BenoitCastandet/chloroseq).

3. As a control example, the editing efficiency of the ndhD_117166 site (which creates the *ndhD* initiation codon) can be linked to the expression of DYW1 and CRR4, two pentatricopeptide repeat proteins that are required to perform the editing reaction [39].
4. Other useful commands to explore the Cuffdiff outputs have been published elsewhere [5, 6].

3.4.1 Overview of Chloroplast Genome Expression

1. One of the ChloroSeq outputs is the window_coverage.txt file, which yields transcript coverage over tiled 100 nt windows that overlap by 50 nt. This smoothed coverage is ideal to derive an overview of plastome-wide expression, and proves useful for flagging and delimiting areas for more in-depth analyses [24, 29].
2. When comparing more than three conditions, a heat map is generally the best option to represent this output.
3. The first step in generating a heat map is to create a single file containing the window coverage data for each condition (in the present case, time point). The file must have the data organized first by the strand (designated in ChloroSeq as plus and minus), and then by the time point. This can produce a heat map similar to Fig. 1 in [24].
4. In a control example the file is named windowcoverage_hmap.txt, and can be easily created with any spreadsheet software (in **Note 5** UNIX commands are used to create the file).
5. To produce the heat map, an R session is initiated in the development directory, then the libraries for ggplot2, scales, reshape2, grid, and gridExtra packages are loaded into the session:

```
> library(ggplot2)
> library(scales)
> library(reshape2)
> library(gridExtra)
> library(grid)
```

6. Load the text file containing all the data into R as a data frame:

```
> window <- read.table("windowcoverage_hmap.txt")
> colnames(window) <- c("Position", "4d_F", "6d_F",
"8d_F", "10d_F", "12d_F", "14d_F", "16d_F",
"18d_F", "20d_F", "22d_F", "24d_F", "26d_F",
"28d_F", "30d_F", "4d_R", "6d_R", "8d_R", "10d_R",
"12d_R", "14d_R", "16d_R", "18d_R", "20d_R",
"22d_R", "24d_R", "26d_R", "28d_R", "30d_R")
```

```
> window$Position <- as.character(window$Position)
> window$Position <- factor(window$Position, levels
=unique(window$Position))
> longWindow <- melt(window)
> legend_title <- "window coverage"
```

7. The first column is assigned as a character with a unique order to maintain the correct chromosome position when the heat map is plotted, and column names are added to serve as labels for the graph. If not done, data would be reordered alphabetically.
8. The data must then be "melted" to switch the format of the data frame from wide to vertical to allow different formats for graphing. The legend title is purely arbitrary.

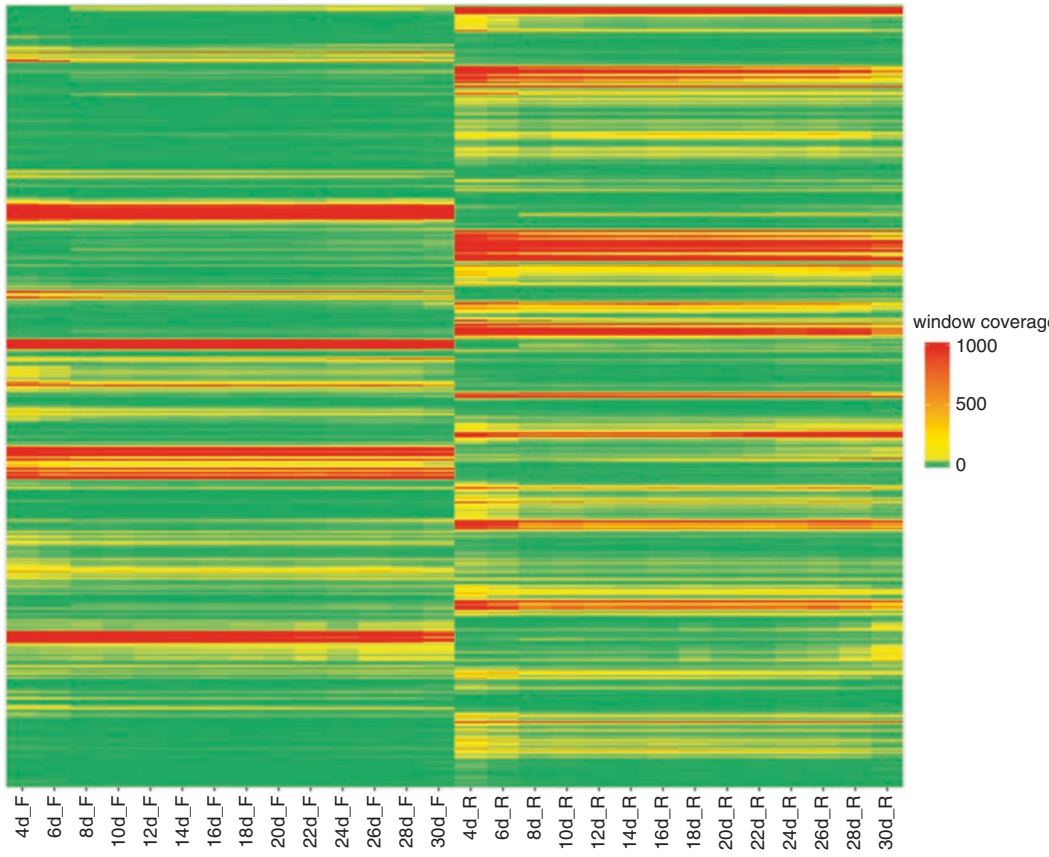

Fig. 1 Overview of the chloroplast genome expression during leaf aging using the window coverage output from ChloroSeq. Low to high expression is represented by a green to red transition. Results are split between the forward (left) and reverse (right) strands. Each line represents a 100 nt interval overlapping by 50 nt with the adjacent ones. Changes in expression values can be identified by changes in color between the data points. The early time points, Days 4 and 6, stand apart, and a subsequent decline in expression levels can be seen

The heat map can now be constructed:

```
> windowcoverage.hmap <- ggplot(data=longWindow, aes(x = variable, y =
rev(Position))) + scale_fill_gradientn(legend_title,limits=c(0,1000),
breaks=c(0,500,1000), values = rescale(c(0,50,1000)), labels=c(0, 500,
1000), colours
=c("springgreen3", "yellow", "red1"), oob = squish) + geom_
tile(data=longWindow,
aes(fill = value)) + theme(panel.grid = element_blank(), panel.
border=element_blank(), axis.title.y=element_blank(), axis.text.y=element_
blank(), axis.ticks.y=element_blank(),axis.title.x=element_blank(),axis.
text.x=element_text(angle = 90, vjust = 0.5, hjust=1))
```

9. Most of the code here is purely cosmetic and can be adapted to fit any personal preferences. The order of the chromosome position column is reversed, designating the Y-axis as rev(Position). This keeps the coordinates of the chloroplast chromosome in the correct orientation for ease of viewing.
10. While the coverage for some regions of the chromosome is much greater than 1000, setting a scale of 0–1000 is used to avoid an irrelevant scale bar. Any coverage greater than 1000 is considered out of bounds and appears with the same color as a coverage of 1000, using the code oob=squish.

3.4.2 Expression of Nucleus-Encoded Genes Involved in Chloroplast RNA Metabolism

1. For the present purposes, the nuclear gene expression data generated by Cuffdiff will focus on genes involved in chloroplast RNA maturation, which are primarily ribonucleases, and ribosomal and RNA-binding proteins.
2. A list of accession numbers, gene names, and functions for all known genes in this category has been created in a text file named Chrc_RNA_metabolism.txt, that will be used to limit the Cuffdiff output to these genes.
3. The first step is to copy the gene_exp.diff into the development directory. In this example it is renamed total_gene_exp.diff.
4. From the development directory, type:

```
$ cp /development_diff_out/gene_exp.diff total_gene_exp.diff
```

5. The total_gene_exp.diff file containing the expression values is then reduced to the set of genes of interest. It first needs to be sorted according to the first column, keeping the headers, and then joined to the ChrC_RNA_metabolism.txt file (that is already sorted).
6. Run the command $ awk 'NR == 1; NR > {print $0 | "sort -k 1,1n"}' total_gene_exp.diff > file1.txt
7. Run the command $ join --header -j 1 -t $'\t' -i file1.txt ChrC_RNA_metabolism.txt > gene_exp.diff
8. Run the command $ rm file1.txt

9. Run the command $ mv gene_exp.diff /development_diff_out/
10. The first column (containing the accession numbers) is used to join the files. To use the CummeRbund package it is necessary to have the file containing the genes of interest named gene_exp.diff, in the directory containing the Cuffdiff output.
11. From the development directory, start an R session and load cummeRbund, ggplot2, scales, reshape2, grid, and gridExtra packages:

```
> library(cummeRbund)
> library(ggplot2)
> library(scales)
> library(reshape2)
> library(grid)
> library(gridExtra)
```

12. Then use the cummeRbund package to load the gene_exp.diff data:

```
> cuff <- readCufflinks("development_diff_out/")
```

13. Next, only the accession numbers from ChrC_RNA_metabolism.txt will be loaded into R. These accession numbers are used to extract the corresponding data from gene_exp.diff.

```
> diffGeneIDs <- read.table("ChrC_RNA_metabolism.txt")
```

14. Next, RNA metabolism genes can be called out using the getGenes function. This function will fail if the cases of the accession numbers in the text file do not match the accession numbers within gene_exp.diff.
15. Run the command > diffGenes <- getGenes(cuff, diffGeneIDs[,1])
16. The [,1] flag is used to specify only to use the first column, because diffGeneIDs is not a vector and contains three columns. The list of RNA metabolism genes and their respective FPKMs is now designated as "diffGenes." The cummeRbund package can be utilized from this point to create a variety of graphs and figures. Here, cummeRbund was used to generate a leaf developmental heat map of all genes involved in chloroplast RNA metabolism using csHeatmap (Fig. 2).
17. Run the command > cshmap <- csHeatmap(diffGenes, fullnames=F) + theme(axis.text.x=element_text(size=14), axis.text.y=element_blank())

```
> cshmap
```

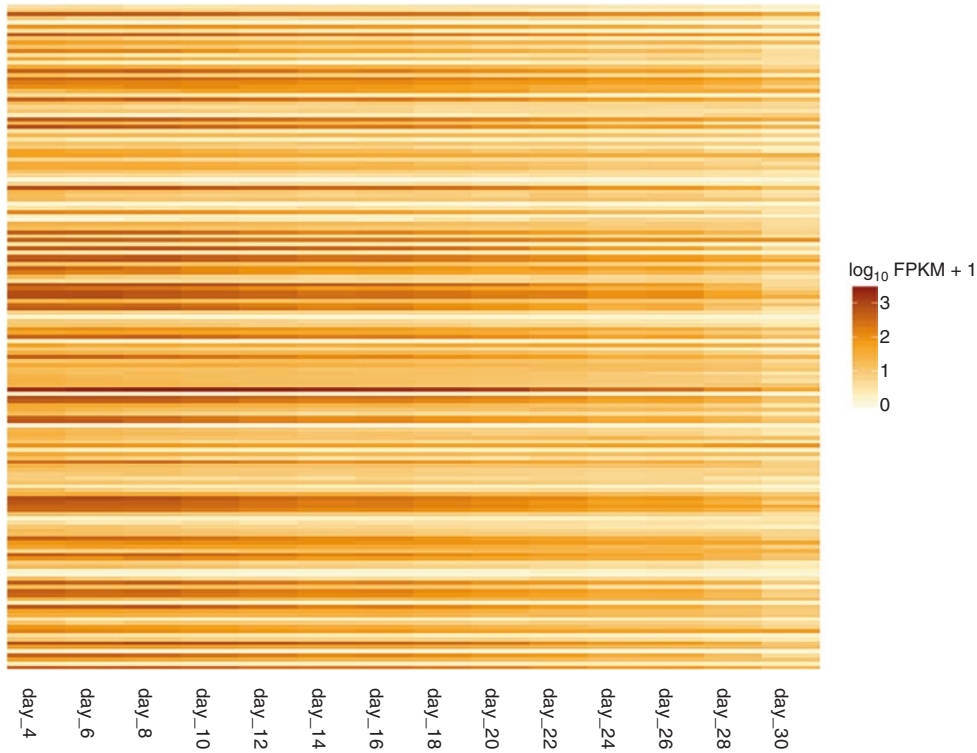

Fig. 2 Expression of nuclear encoded genes involved in chloroplast RNA metabolism during leaf aging. The heat map was produced using the CummeRbund package based on the expression values calculated by Cuffdiff. Each line represents a nuclear gene (here not labeled) involved in chloroplast RNA metabolism, sorted by genome location. They display a clear decrease in their expression during leaf aging, a result in agreement with the analysis from [38]

18. Not only does the csHeatmap generated by cummeRbund provide a useful overview of data, but it also acts as a simple way to extract FPKM data for genes of interest, as shown in the next section.

3.4.3 Comparing RNA Editing Efficiency to the Expression of Nuclear Genes Encoding Cognate Editing Factors

1. In a final example, a figure comparing the editing efficiency of a specific site (obtained from ChloroSeq) to the transcript abundance for the known editing factors of that site (obtained from Cuffdiff), will be generated. Specifically, the editing efficiency during leaf development of *ndhD* at position 117166 will be compared to the transcript abundance for the DYW1 and CRR4 proteins. There are many ways to represent this comparison, here representing editing efficiency as a line graph and FPKM as a heat map, as shown in Fig. 3.
2. The first step is to create two text files within the development working directory, containing the accession numbers and data to be represented. To create a file containing the accession numbers of the two editing factors, use grep to

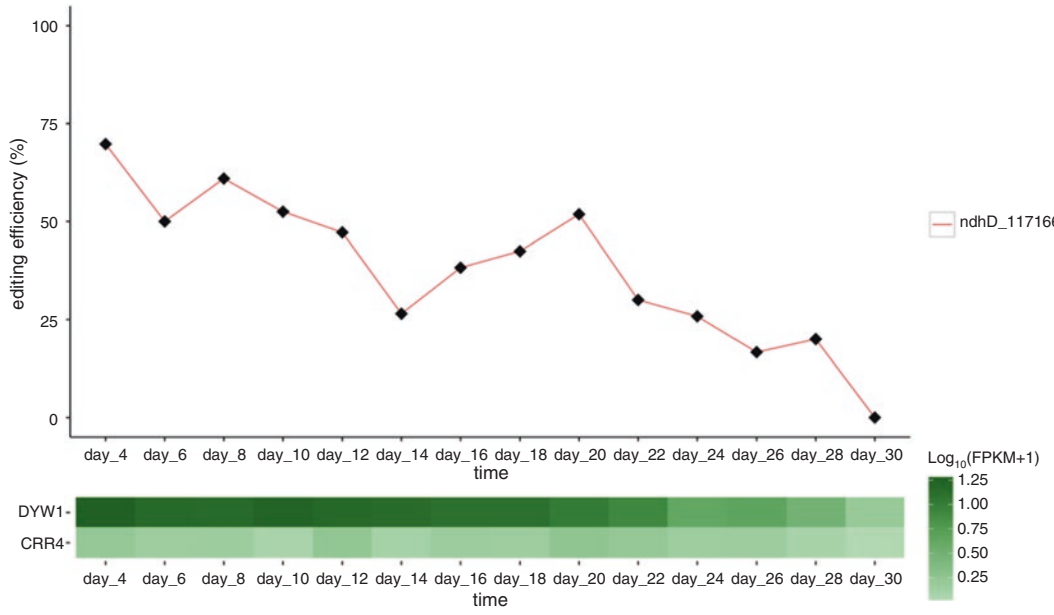

Fig. 3 Comparing editing efficiency of the *ndhD*_117166 editing site to expression of the corresponding *trans* factors, PPR proteins DYW1 and CRR4. Editing efficiency calculated by ChloroSeq analysis 3 is represented by a line graph, and the expression values for the editing proteins are represented by heat maps. Editing efficiency of *ndhD*_117166 decreases with leaf aging, along with expression of the DYW1 expression factor gene. Expression of the CRR4 gene remains low during the leaf life span and is unlikely to have an impact on editing efficiency

search for the specific accession numbers within ChrC_RNA_metabolism.txt, then print them to a new file. This file will be used to call out the FPKM values using cummeRbund in R.

3. Run the command $ grep -i -e AT2G45350 -e AT1G47580 ChrC_RNA_metabolism.txt > edit_fpkm.txt
4. Then, extract the editing efficiency from the ChloroSeq output file editing.txt from each tophat_out directory. This is similar to what has been done in **3.4.1** to create the windowcoverage_hmap.txt file. This time, the grep command will also be used to extract only the row containing data on editing of *ndhD* at position 117166. From the development directory type:
5. Run the command $ grep –e 117166 /day_4_tophat_out/editing.txt | cut –f 9 > 4d_edit.txt
6. This command must be modified for all the other time points, until 14 different edit.txt files are in the development directory. These files are then pasted together to create the file that will be used in R:

```
$ paste *_edit.txt > ndhd_edit_line.txt
```

7. Run the command $ rm *_edit.txt
8. From the development directory, start an R session and load the Cuffdiff output along with the necessary libraries:

```
> cuff <- readCufflinks("development_diff_out/")
> library(cummeRbund)
> library(ggplot2)
> library(scales)
> library(reshape2)
> library(grid)
> library(gridExtra)
```

9. Then, load edit_fpkm.txt into R and use the file to call out the FPKM values from the two genes. This is similar to the example developed in Subheading 3.4.2.
10. Run the following commands:

```
> editfpkmIDs <- read.table("edit_fpkm.txt")
> editfpkmgenes <- getGenes(cuff, editfpkmIDs[,1])
```

11. Then use cummeRbund to extract the FPKM values from the matrix. Run the following command:

```
> edit_FPKM_cummeRbund_extract <- fpkmMatrix(editfpkmgenes)
```

12. This is a very convenient way to get a properly oriented matrix of FPKM values, but the function only removes the raw data, and not the data that is adjusted by taking the $\log_{10}$(FPKM +1). Therefore, the data should be scaled using the log10 function in R:

```
> FPKM_log <- log10(edit_FPKM_cummeRbund_extract+1)
```

13. At this point, the rows are renamed by protein rather than by accession number:

```
> rownames(FPKM_log) <- c("DYW1", "CRR4")
```

14. The row names are specified along with the legend title:

```
> id <- rownames(FPKM_log)
> FPKM_log <- cbind(id = id, FPKM_log)
> longFPKMlog <- melt(FPKM_log)
> legend_title <- expression("Log"[10]* "(FPKM+1)")
```

15. Then the heat map is ready to be generated:

```
> FPKM.heatmap <- ggplot(data=longFPKMlog, aes(x = variable, y = id)) +
scale_fill_gradient2(legend_title,low = "white", mid = "darkseagreen1",
high = "darkgreen") + geom_tile(data=longFPKMlog, aes(fill = value)) + theme_
bw() + theme(panel.grid = element_blank(), panel.border=element_blank()) +
coord_fixed(ratio=1) + scale_x_discrete(name = "time") + scale_y_discrete(name
= element_blank())
```

16. The code and parameters are similar to the heat map created in Fig. 3, thus the reader may refer to Subheading

3.4.1 for additional details. These parameters are only meant as a guideline, and are fully customizable. Next, the line graph representing the editing efficiency of *ndhD* is created. Keeping the same working directory, upload the file containing the specific editing efficiency data created earlier.

17. Run the command > ndhd_edit <- read.table("ndhd_edit_line.txt")
18. As the row name of this dataframe is only the position number and not the gene name, it is possible to change the row name in a similar way as before if desired. Here, the row was relabeled as ndhD_117166:

```
> rownames(ndhd_edit)[1] <- " ndhD_117166"
> id <- rownames(ndhd_edit)
> ndhd_edit <- cbind(id = id, ndhd_edit)
```

19. It is also important to rename each column to match the X-axis of the corresponding heat map for the editing factors, so that the two graphs align correctly:

```
> colnames(ndhd_edit) <- c("id", "day_4", "day_6", "day_8", "day_10", "day_12",
"day_14", "day_16", "day_18", "day_20", "day_22", "day_24", "day_26", "day_28",
"day_30")
```

20. Then the data is ready to be melted and made into a line graph using ggplot2:

```
> ndhdlong <- melt(ndhd_edit)
> ndhd.line <- ggplot(ndhdlong, aes(x = variable, y = value, group =
id)) + geom_line(aes(colour = id)) + geom_point(pch = 18, size = 4) +
scale_x_discrete(name = "time") + scale_y_continuous(name = "editing ef-
ficiency (%)", limits = c(0,100)) + theme_bw() + theme(panel.grid.major =
element_blank(), panel.grid.minor = element_blank(), panel.border = ele-
ment_blank(), axis.line.x = element_line(colour = "black"), axis.line.y =
element_line(colour = "black"), legend.title=element_blank())
```

21. This graph is less customized compared to the previous heat map, and the parameters chosen are almost entirely for aesthetic reasons. They are given as an example and can be changed to suit the needs of the data being represented.
22. Finally, the two graphs to be compared are aligned along their X-axes using grid and gridExtra in R:

```
> grid.draw(rbind(ggplotGrob(ndhd.line), ggplotGrob(FPKM.heatmap), size =
"last"))
```

23. This is one example of the many different analyses /visualizations that can be done using the data produced by ChloroSeq and the Tuxedo tools.

24. Using the same code with minor modifications, we repeated the analysis for the entire set of editing sites and their known editing factors.
25. From the results, it is obvious that editing of some sites varies dramatically during leaf aging, and that there is little to no correlation with the expression of the corresponding editing factor. This supplementary analysis can be accessed online at https://github.com/BenoitCastandet/editing_development.

4 Notes

1. It is highly encouraged to perform a quality control on the sequencing files before performing the alignment, because the quality of deposited data is variable. Such a step ensures removal of any leftover adapter sequences, and establishes thresholds for quality and length. We routinely use fastq-mcf software (https://github.com/ExpressionAnalysis/ea-utils/blob/wiki/FastqMcf.md) for this purpose. For example, for the day 4 time point, the following command might be entered:

```
$ fastq-mcf -o SRR985_1.fq -o SRR985_2.fq -q 30 -l 60 adapters.fa
SRR207985_1.fastq SRR207985_2.fastq
```

   This will trim both SRR207985.fastq files for any adapter sequences (provided in a fasta file, adapters.fa), and will retain only sequences longer than 60 nt and bases with a quality higher than 30.

2. If one is only interested in analyzing the chloroplast genome, there is no need to perform the alignment on the full genome. In this case, for the day 4 time point, the command would be:

```
$ tophat2 -p 3 --no-novel-juncs -o day_4_tophat_out/ -G TAIR10_ChrC_files/
TAIR10_ChrC_ genes.gff3 TAIR10_ChrC_files/TAIR10_ChrC_bowtie2_index/TAIR10_
ChrC SRR207985_1.fastq SRR207985_2.fastq
```

   All the necessary files are in the TAIR10_ChrC_files directory, downloaded to the development directory. It is possible to add the -g 2 option to allow up to 2 alignments to the genome (taking into account the inverted repeat), but in our experience this does not impact the final result.

3. This step depends on the library preparation protocol that has been used. If the strand specificity is opposite to the one used in this example, the commands would be:

```
$ samtools view -f 99 -F 16 -b accepted_hits.bam > fw.bam
$ samtools view -f 83 -F 32 -b accepted_hits.bam > rv.bam
$ samtools merge single.bam fw.bam rv.bam
```

When analyzing a chloroplast transcriptome, an easy way to verify that the strategy used is the correct one is to open the window coverage output from ChloroSeq. It should show a clear coverage bias favoring the reverse strand at the beginning of the file for Arabidopsis corresponding to transcription of *psbA* on this strand.

4. The general command for ChloroSeq is:

```
chloroseq.pl [-h] -a <analysis> -b <accepted_hits.bam> -e <exon.gff3> -i
<intron.gff3> -g <genome_size>
-n <name> -v <editing_sites.gff3> -f <fasta> -s <splice_sites.gff3> -k <keep_files>
```

The available options are detailed in the documentation that can be accessed through this command:

```
$ perldoc chloroseq.pl
```

In particular, if one is only interested in a single analysis they can be run individually using the options -a 1 for coverage, -a 2 for splicing, and -a 3 for editing.

5. Although the windowcoverage_hmap.txt can be created with any spreadsheet software, UNIX commands are faster and use less memory when handling large files. From the development directory, first extract the coverage values from the window_coverage.txt file for the Day 4 time point:

```
$ cut -f 2,4 /day_4_tophat_out/window_coverage.txt > 4d_F.txt
$ cut -f 5 /day_4_tophat_out/window_coverage.txt > 4d_R.txt
```

In the first command, the chromosome position (second column of the window_coverage.txt, -f 2 option) is also extracted, to be used later as the Y-axis label on the heat map. This will only be done once. F and R (for forward and reverse) represent the two strands of the chloroplast chromosome. The same process needs to be performed for every other time point, this time only extracting the coverage. For the Day 6 time point the command would be:

```
$ cut -f 4 /day_6_tophat_out/window_coverage.txt > 6d_F.txt
$ cut -f 5 /day_6_tophat_out/window_coverage.txt > 6d_R.txt
```

Once completed, the development directory should contain 14 F.txt files and 14 R.txt files. These files are then joined to create the windowcoverage_hmap.txt file necessary to generate the heat map.

```
$ paste *F.txt *R.txt > windowcoverage_hmap.txt
```

Once can then clean up the development directory by erasing the intermediate F.txt and R.txt files. Run the following commands:

```
$ rm *F.txt
$ rm *R.txt
```

Acknowledgments

This work was supported by Grant DE-FG02-10ER20015 from the Division of Chemical Sciences, Geosciences, and Biosciences, Office of Basic Energy Sciences, of the US Department of Energy. The authors would also like to thank the Boyce Thompson Institute Bioinformatics Help Desk for assistance with experimental methods. This is a consulting service for early-stage ideas or issues in bioinformatics. The IPS2 benefits from the support of the LabEx Saclay Plant Sciences-SPS (ANR-10-LABX-0040-SPS).

References

1. Ozsolak F, Milos PM (2011) RNA sequencing: advances, challenges and opportunities. Nat Rev Genet 12(2):87–98. https://doi.org/10.1038/nrg2934
2. van Dijk EL, Auger H, Jaszczyszyn Y et al (2014) Ten years of next-generation sequencing technology. Trends Genet 30(9):418–426. https://doi.org/10.1016/j.tig.2014.07.001
3. Wang Z, Gerstein M, Snyder M (2009) RNA-Seq: a revolutionary tool for transcriptomics. Nat Rev Genet 10(1):57–63. https://doi.org/10.1038/nrg2484
4. Trapnell C, Williams BA, Pertea G et al (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat Biotechnol 28(5):511–515. https://doi.org/10.1038/nbt.1621
5. Ghosh S, Chan CK (2016) Analysis of RNA-Seq data using TopHat and cufflinks. Methods Mol Biol 1374:339–361. https://doi.org/10.1007/978-1-4939-3167-5_18
6. Trapnell C, Roberts A, Goff L et al (2012) Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and cufflinks. Nature Protoc 7(3):562–578. https://doi.org/10.1038/nprot.2012.016
7. Goecks J, Nekrutenko A, Taylor J (2010) Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 11(8):R86. https://doi.org/10.1186/gb-2010-11-8-r86
8. Ding Y, Tang Y, Kwok CK et al (2014) In vivo genome-wide profiling of RNA secondary structure reveals novel regulatory features. Nature 505(7485):696–700. https://doi.org/10.1038/nature12756
9. Gosai SJ, Foley SW, Wang D et al (2015) Global analysis of the RNA-protein interaction and RNA secondary structure landscapes of the Arabidopsis nucleus. Mol Cell 57(2):376–388. https://doi.org/10.1016/j.molcel.2014.12.004
10. Zheng Q, Ryvkin P, Li F et al (2010) Genome-wide double-stranded RNA sequencing reveals the functional significance of base-paired RNAs in Arabidopsis. PLoS Genet 6(9):e1001141. https://doi.org/10.1371/journal.pgen.1001141
11. Wu X, Liu M, Downie B et al (2011) Genome-wide landscape of polyadenylation in Arabidopsis provides evidence for extensive alternative polyadenylation. Proc Natl Acad Sci U S A 108(30):12533–12538. https://doi.org/10.1073/pnas.1019732108
12. Chotewutmontri P, Barkan A (2016) Dynamics of chloroplast translation during chloroplast differentiation in maize. PLoS Genet 12(7):e1006106. https://doi.org/10.1371/journal.pgen.1006106
13. Juntawong P, Girke T, Bazin J et al (2014) Translational dynamics revealed by genome-wide profiling of ribosome footprints in Arabidopsis. Proc Natl Acad Sci U S A 111(1):E203–E212. https://doi.org/10.1073/pnas.1317811111
14. Liu MJ, Wu SH, Wu JF et al (2013) Translational landscape of photomorphogenic Arabidopsis. Plant Cell 25(10):3699–3710. https://doi.org/10.1105/tpc.113.114769
15. Di C, Yuan J, Wu Y et al (2014) Characterization of stress-responsive lncRNAs in Arabidopsis thaliana by integrating expression, epigenetic and structural features. Plant J 80(5):848–861. https://doi.org/10.1111/tpj.12679
16. Bombarely A, Edwards KD, Sanchez-Tamburrino J et al (2012) Deciphering the complex leaf transcriptome of the allotetraploid species Nicotiana tabacum: a phylogenomic perspective. BMC Genomics 13:406. https://doi.org/10.1186/1471-2164-13-406

17. Smith DR (2013) RNA-Seq data: a goldmine for organelle research. Brief Funct Genomics 12(5):454–456. https://doi.org/10.1093/bfgp/els066
18. Gros F, Hiatt H, Gilbert W et al (1961) Unstable ribonucleic acid revealed by pulse labelling of Escherichia coli. Nature 190:581–585
19. Jacob F, Monod J (1961) Genetic regulatory mechanisms in the synthesis of proteins. J Mol Biol 3:318–356
20. Ycas M, Vincent WS (1960) A ribonucleic acid fraction from yeast related in composition to desoxyribonucleic acid. Proc Natl Acad Sci U S A 46(6):804–811
21. Cui P, Lin Q, Ding F et al (2010) A comparison between ribo-minus RNA-sequencing and polyA-selected RNA-sequencing. Genomics 96(5):259–265. https://doi.org/10.1016/j.ygeno.2010.07.010
22. Lange H, Sement FM, Canaday J et al (2009) Polyadenylation-assisted RNA degradation processes in plants. Trends Plant Sci 14(9):497–504. https://doi.org/10.1016/j.tplants.2009.06.007
23. Rorbach J, Bobrowicz A, Pearce S et al (2014) Polyadenylation in bacteria and organelles. Meth Mol Biol 1125:211–227. https://doi.org/10.1007/978-1-62703-971-0_18
24. Castandet B, Hotto AM, Strickler SR et al (2016) ChloroSeq, an optimized chloroplast RNA-Seq bioinformatic pipeline, reveals remodeling of the organellar transcriptome under heat stress. G3 (Bethesda). https://doi.org/10.1534/g3.116.030783
25. Zhao W, He X, Hoadley KA et al (2014) Comparison of RNA-Seq by poly (A) capture, ribosomal RNA depletion, and DNA microarray for expression profiling. BMC Genomics 15:419. https://doi.org/10.1186/1471-2164-15-419
26. Barkan A (2011) Expression of plastid genes: organelle-specific elaborations on a prokaryotic scaffold. Plant Physiol 155(4):1520–1532. https://doi.org/10.1104/pp.110.171231
27. Germain A, Hotto AM, Barkan A et al (2013) RNA processing and decay in plastids. Wiley Interdiscip Rev RNA 4(3):295–316. https://doi.org/10.1002/wrna.1161
28. Stern DB, Goldschmidt-Clermont M, Hanson MR (2010) Chloroplast RNA metabolism. Annu Rev Plant Biol 61:125–155. https://doi.org/10.1146/annurev-arplant-042809-112242
29. Hotto AM, Castandet B, Gilet L et al (2015) Arabidopsis chloroplast mini-ribonuclease III participates in rRNA maturation and intron recycling. Plant Cell 27(3):724–740. https://doi.org/10.1105/tpc.114.134452
30. Castandet B, Hotto AM, Fei Z et al (2013) Strand-specific RNA sequencing uncovers chloroplast ribonuclease functions. FEBS Lett 587(18):3096–3101. https://doi.org/10.1016/j.febslet.2013.08.004
31. Germain A, Herlich S, Larom S et al (2011) Mutational analysis of Arabidopsis chloroplast polynucleotide phosphorylase reveals roles for both RNase PH core domains in polyadenylation, RNA 3'-end maturation and intron degradation. Plant J 67(3):381–394. https://doi.org/10.1111/j.1365-313X.2011.04601.x
32. Hotto AM, Schmitz RJ, Fei Z et al (2011) Unexpected diversity of chloroplast noncoding RNAs as revealed by deep sequencing of the arabidopsis transcriptome. G3 (Bethesda) 1(7):559–570. https://doi.org/10.1534/g3.111.000752
33. Walter M, Kilian J, Kudla J (2002) PNPase activity determines the efficiency of mRNA 3'-end processing, the degradation of tRNA and the extent of polyadenylation in chloroplasts. EMBO J 21(24):6905–6914
34. Smith DR, Sanita Lima M (2016) Unraveling chloroplast transcriptomes with ChloroSeq, an organelle RNA-Seq bioinformatics pipeline. Brief Bioinform. https://doi.org/10.1093/bib/bbw088
35. Chan KX, Phua SY, Crisp P et al (2016) Learning the languages of the chloroplast: retrograde signaling and beyond. Annu Rev Plant Biol 67:25–53. https://doi.org/10.1146/annurev-arplant-043015-111854
36. Quinlan AR (2014) BEDTools: the Swiss-army tool for genome feature analysis. Curr Protoc Bioinformatics 47:11.12.1–34. doi:https://doi.org/10.1002/0471250953.bi1112s47
37. Li H, Handsaker B, Wysoker A et al (2009) The sequence alignment/map format and SAMtools. Bioinformatics 25(16):2078–2079. https://doi.org/10.1093/bioinformatics/btp352
38. Woo HR, Koo HJ, Kim J et al (2016) Programming of plant leaf senescence with temporal and inter-organellar coordination of transcriptome in Arabidopsis. Plant Physiol 171(1):452–467. https://doi.org/10.1104/pp.15.01929
39. Boussardon C, Salone V, Avon A et al (2012) Two interacting proteins are necessary for the editing of the NdhD-1 site in Arabidopsis plastids. Plant Cell 24(9):3684–3694. https://doi.org/10.1105/tpc.112.099507