Detection of aberrant gene expression events in RNA sequencing data

Vicente A. Yépez[1,2,*], Christian Mertes[1,*], Michaela F. Müller[1], Daniela Klaproth- Andrade[1], Leonhard Wachutka[1], Laure Frésard[3], Mirjana Gusic[4,5], Ines Scheller[1], Patricia F. Goldberg[1], Holger Prokisch[4,5], Julien Gagneur[1,2,*]

[1] Department of Informatics, Technical University of Munich, Munich, Germany
[2] Quantitative Biosciences Munich, Department of Biochemistry, Ludwig-Maximilians Universität, Munich, Germany
[3] Department of Pathology, School of Medicine, Stanford University, Stanford, CA, USA
[4] Institute of Human Genetics, Helmholtz Zentrum München, Neuherberg, Germany
[5] Institute of Human Genetics, Technical University of Munich, Munich, Germany
[*] Corresponding authors: yepez@in.tum.de, mertes@in.tum.de, gagneur@in.tum.de

RNA sequencing (RNA-seq) has emerged as a powerful approach to discover disease-causing gene regulatory defects for individuals affected with a genetically undiagnosed rare disorder. Pioneer studies have shown that RNA-seq could increase diagnostic rates over DNA sequencing alone by 8% to 36 % depending on disease entities and probed tissues. To accelerate the adoption of RNA-seq among human genetics centers, detailed analysis protocols are now needed. Here, we present a step-by-step protocol that instructs how to robustly detect aberrant expression levels, aberrant splicing, and mono-allelic expression of a rare allele in RNA-seq data using dedicated statistical methods. We describe how to generate and assess quality control plots and interpret the analysis results. The protocol is based on DROP (Detection of RNA Outliers Pipeline), a modular computational workflow that integrates all analysis steps, can leverage parallel computing infrastructures, and generates browsable web page reports.

Keywords: RNA-seq, outlier detection, aberrant expression, aberrant splicing, mono-allelic expression, rare disorder, workflow

Introduction

Whole-exome sequencing (WES) and whole-genome sequencing (WGS) are increasingly used to identify disease-causing genetic variants of individuals affected with a genetically undiagnosed rare disorder[1–3]. Depending on the disease entity, analysis of WES or WGS achieves a diagnosis rate of only up to 50%[2–5]. One of the main challenges in these DNA-based molecular diagnostics approaches is the interpretation of the non-coding variants. Many non-coding variants may have a regulatory role in controlling RNA abundance or splicing. However, computational predictions of their regulatory effects from DNA sequence alone remain uncertain[6]. To advance diagnostics, RNA sequencing (RNA-seq) has emerged as a complementary tool because it directly probes gene expression, thereby providing functional data supporting the clinical interpretation of variants. Moreover, RNA-seq can reveal genes with a regulatory defect, even in the absence of candidate regulatory variants, which is particularly important in the situation where WES instead of WGS has been performed. Pilot studies systematically using RNA-seq have obtained increased diagnostic rates over DNA-sequencing alone for a variety of rare disorders ranging between 8% and 36%[7–10].

As the first studies that systematically used RNA-seq for the genetic diagnosis of rare diseases were published only three years ago[7,8], the field is new and lacks established workflows. Here, we provide a step-by-step protocol that instructs how to use RNA-seq data in this context by supporting the detection of aberrant events. Specifically, we provide detailed instructions to identify aberrant expression, aberrant splicing, and mono-allelic expression (MAE) of a rare allele by starting from Binary Sequence Alignment Map (BAM)[11] and Variant Call Format (VCF) files[12] (Table 1). The protocol covers a quality control steps and plots that evaluate the counting and model fitting procedures. In order to detect mixing or swapping of samples, the protocol includes a validation step comparing variants called from RNA and DNA. Lastly, we describe how to analyze individual samples using the results and objects derived from the workflow.

Table 1 | File formats

File format	**Extension**	**Description**
Binary sequence alignment map[11]	.bam	A binary compressed file containing sequencing read (typically from DNA or RNA-seq) alignments.
FASTA	.fa	A text file containing nucleotide sequences.
General Transfer Format	.gtf	A text file containing different gene annotations (e.g. genomic position, type of, transcripts, exons).
Serialized R data	.Rds (.rds, .RDS)	A binary compressed file containing a single R object.
Tab-separated value	.tsv	A text file with columns separated by tabs.
Variant call format[12]	.vcf	A text file containing sequence variants including their genomic positions, quality score, and other annotation.

Application of the protocol

This protocol aims to provide a one-in-all computational workflow starting from VCF and BAM files to call and visualize all forms of aberrant gene expression that have been currently used for rare disease diagnosis purposes. It is based on DROP (Detection of RNA Outliers Pipeline), a computational workflow that automates and standardizes all the steps needed to obtain the results from raw input files. DROP is the result of an improvement and standardization of the concepts and methods described in Kremer et al.[7], and was recently used to successfully diagnose patients with rare disorders[13]. The workflow uses the Snakemake framework[14] (Box 1) to guarantee reproducible results. Overall, our workflow can be easily implemented and provides a comprehensive collection of tools enabling the detection of outliers using RNA-seq.

To showcase the workflow, we have created a dataset based on 100 samples from the Geuvadis dataset[15] (Supplementary Data 1), which is accessible without restriction under https://www.internationalgenome.org/data-portal/data-collection/geuvadis. We refer to this dataset as the test dataset. DROP, including a small demo dataset of 10 samples and chromosome 21 only, is publicly available at https://github.com/gagneurlab/drop. All the plots, results, and analyses of the test dataset can be found here: https://www.cmm.in.tum.de/public/paper/drop_analysis/webDir/html/drop_analysis_index.html.

DROP integrates recent tools leading to increased diagnostic rates. In our pilot study (Kremer et al.[7]), we found the *MCOLN1* gene to be borderline significant for aberrant splicing and expression, and the *CLPP* gene not to be significant for aberrant expression using previously developed approaches (DESeq2[16] to detect aberrant expression and LeafCutter[17] to detect aberrant splicing). After re-analyzing the same dataset with DROP, both of those cases were found significant.

Advantages and limitations

Currently, only one computational workflow for RNA-seq based diagnostics of rare disorders has been published[9]. That workflow includes read alignment, read counting, and quality controls of the counts, as well as functions to filter and annotate variants and prioritize candidates. The main advantage of using DROP is that we automate most of the steps to detect rare aberrant expression events. DROP integrates the specialized statistical methods for detecting expression outliers, OUTRIDER[18], and splicing outliers, FRASER[19], and adds a module to test for MAE. Additionally, it provides quality control through dedicated plots to evaluate the different steps and to verify the proper matching between DNA and RNA samples attributed to the same individual. DROP also provides gene prioritization support by integrating functional annotation and disease phenotypes encoded as Human Phenotype Ontology (HPO) terms[20]. All the results and plots are rendered as HTML pages. DROP is designed with a modular architecture, such that the aberrant expression, aberrant splicing, and MAE modules can be executed without depending on each other. This architecture allows new modules with different functionalities to be added. Another advantage of DROP is that it uses the workflow management framework Snakemake[14] to run and monitor the execution of the workflow (Box 1). By using Snakemake, DROP ensures that the results reflect the latest scripts and input data files, while avoiding unnecessary execution of scripts lying upstream or parallel to the modifications.

Another advantage of DROP is how it manages analysis groups. Users might not want to merge all the available samples into one analysis, but instead, create subsets. These subsets could contain, for example, samples extracted from the same tissue or belonging to the same disease entity. A sample can belong to one or multiple groups, which are specified in the sample annotation. Each sample is processed only once on each module, thus saving space and computational resources. Then, the intermediate outputs are merged in their respective group(s), and a separate analysis is performed for each group.

RNA-seq is not the first tier diagnostic tool. Its power is in the detection of functionally relevant consequences of DNA variants affecting gene expression levels and splicing. In this way, it complements DNA sequencing by supporting the interpretation of variants of uncertain significance and reprioritizing variants overlooked by DNA sequencing. RNA-seq has limited diagnostic potential (the highest reported diagnosis rate increases after WES using RNA-seq are 35%[8] and 36%[10]). This may be because, for instance, the disease-causing variant(s) (e.g. missense) might not have any effect on the transcript. Another issue is that the gene may not be expressed in the available tissue. In order to overcome this, researchers can investigate a priori whether genes are expressed in a certain tissue by using public datasets like the Genotype-Tissue Expression project (GTEx[21]) or the Expression Atlas[22]. Moreover, the web-based tool Panel Analysis of Gene Expression (PAGE[10]) can give insight on which tissues express the largest subset of a given set of genes. Regarding splicing, MAJIQ-CAT (Modeling Alternative Junction Inclusion Quantification – Clinically-Accessible Tissues) allows testing the similarity in splicing events of a group of genes in clinically-accessible tissues with respect to other tissue(s)[23]. Finally, fibroblasts can be transdifferentiated into other cell lines which better reflect the affected tissue of the studied disease, thus allowing detection of aberrant events missed by the original fibroblasts[10].

In order to confidently detect outliers, a sufficiently large sample size is needed. Power analyses showed that a cohort should contain at least 50-60 samples for detecting aberrant expression with OUTRIDER[18], and at least 30 samples for detecting aberrant splicing with FRASER[19]. To support centers to implement RNA-seq based diagnostics with an initial smaller sample size, we are collecting reference count matrices from various tissues and RNA-seq protocols. These count matrices are contributed by colleagues and can be used with no other restriction than citing the original studies (see Box 2 - Online Resources). Another approach is to include BAM files from publicly available cohorts like Geuvadis[15] or GTEx[21] (see 'Dataset Design').

Assuming that local workflows are established for read alignment and variant calling, DROP takes as input BAM and VCF files (Table 1) and does not provide a built-in solution for those steps. However, this protocol provides instructions for those steps as well (Materials). Furthermore, we implement only one method to compute the results of each type of aberrant expression pattern. Nevertheless, the user can modify the workflow to apply any other method to them.

Box 1. Workflow Management Systems used in DROP

Snakemake (https://snakemake.readthedocs.io/en/stable) is a workflow management system to create data analyses guaranteeing reproducibility, automation, and scalability[14]. Snakemake workflows consist of rules that describe how to create output files from the respective input files. These output files are created by defining instructions in shell, Python, or R code. Every time the workflow is executed, Snakemake computes the dependencies among all scripts and data files. Then, it checks if either a data file or a script was modified or added. If a script was added or modified, Snakemake will execute it, together with the downstream steps. If a data file was added or modified, Snakemake will execute any script using it, and all the downstream scripts in cascade. Moreover, Snakemake allows the usage of multiple scheduling systems or parallel backends (multi-core server or clusters under, e.g., SGE (Sun Grid Engine) or SLURM (Simple Linux Utility for Resource Management)) to efficiently use HPC systems and run executions in parallel by automatically determining parallel parts in the workflow.

wBuild (https://wbuild.readthedocs.io) is a framework that automatically creates Snakemake dependencies, workflow rules based on R markdown scripts and compiles the analysis results into a navigable HTML page. All information needed such as input, output, number of threads, and even Python code is specified in a YAML header inside the R script file, thereby keeping code and dependencies together.

Workflow

Our workflow is composed of three main steps: i) preparing the input data, ii) fitting the models and extracting the results from aberrant expression, aberrant splicing, and MAE, and iii) analyzing the individual results (Fig. 1). The input data are BAM files, VCF files, a sample annotation table, a configuration file containing the workflow's parameters, a human reference genome (FASTA) file, and a gene annotation (gtf) file (Table 1). DROP then generates intermediate files (e.g., read count matrices), final results, and produces HTML pages for convenient visualization using the framework wBuild (Box 1). The generated files are stored as either tab-separated text (tsv) files or binary R data (Rds) files, depending on their content. See Table 1 for file format overview. Finally, users can access the objects and results in order to plot and analyze the samples and genes of interest.

Aberrant expression

Aberrantly expressed genes, or expression outliers, refer to genes with an expression level aberrantly higher or lower in a sample with respect to other samples. Calling expression outliers in individuals affected with a rare disorder has been successfully used to identify disease-causing genes and to identify or confirm potential disease-causing variants[7,8,10].

To detect aberrant expression, we first compute an RNA-seq read count matrix by counting for each sample the sequencing reads that fully overlap each gene (Supplementary Methods). An external count matrix can also be provided to increase the overall sample size. Then, we apply OUTRIDER[18], a specialized statistical method to detect expression outliers. OUTRIDER computes significance levels for extreme read count values while controlling for hidden confounders. DROP extracts *P* values, z scores, and fold changes for each sample–gene combination from the OUTRIDER returned objects. After correcting for multiple testing, there is typically a handful of outlier genes per sample at a false discovery rate (FDR) of 0.05. For example, there is a median of 1 outlier per sample in the test dataset, consistent with observations from other datasets[7,13,18]. If a sample has more than 0.1% of expressed genes as outliers, we refer to it as an aberrant sample. Pinpointing candidate disease-causing genes among the tens to hundreds of expression outliers of an aberrant sample is difficult. Aberrant samples should be carefully analyzed to find a possible explanation for their high number of outliers, e.g., in case of contamination, or if the sample belongs to a different population (tissue, ethnicity). A biological cause is not to be excluded. For instance, disruption of the function of a transcription factor may cause aberrant expression of many downstream genes.

Other methods to detect aberrant expression consist of applying cutoffs on z scores[9,10] computed after regressing out contributions of hidden confounders estimated by PEER[24] or SVA[9]. However, simulations and enrichment analysis of rare variants among expression outliers have shown that OUTRIDER outperformed these methods[18].

Aberrant splicing

Aberrant splicing is a major cause of Mendelian disorders[25,26]. Despite progress in predicting aberrant splicing from sequence[27,28], the task remains difficult because splicing involves a complex set of cis-regulatory elements that are not yet fully understood. Moreover, some variants affecting splicing cannot be detected by WES owing to being located in intronic sequences not covered by WES kit probes. Moreover, the exact consequences on isoform choice of a variant affecting splicing are not clear. For instance, disruption of a splice site may turn out to be functionally benign thanks to the usage of a nearby in-frame cryptic splice site. Aberrant splicing events and their consequences on isoform choice can instead be detected from RNA-seq data[7–10,29–33].

The aberrant splicing module considers split reads, which are reads aligning to separate locations of the same chromosome and strand, and therefore indicative of an exon-exon junction. Consideration of split reads is done independently of exon annotation, allowing for calling splicing events de novo. In addition, reads spanning the exon-intron boundary (non-split reads) from both donor and acceptor sites are counted to enable intron retention detection. As for the aberrant expression module, external count matrices of split and non-split reads can also be provided to increase the overall sample size. The counts are converted into two intron-centric metrics, percent-spliced-in (PSI, ψ) and splicing efficiency (θ), which are calculated for both the donor *D* (5' splice site) and acceptor *A* (3' splice site) sites of every intron[34] (Supplementary Methods). The θ_5 and θ_3 values are fitted together, hence we jointly refer to them as θ. Junctions expressed at low levels are filtered out (Supplementary Methods). Then, for each of the metrics ψ_5, ψ_3, and θ, we apply the specialized statistical method to detect splicing outliers, FRASER, which uses a denoising autoencoder to control for latent confounders and fits a beta-binomial distribution on every intron or splice site[19]. *P* values, and $\Delta\psi_5$, $\Delta\psi_3$, and $\Delta\theta$ values are then computed from the beta-binomial fits, where the Δ values are defined as the difference between the observed and the expected values. FRASER then reports gene-level *P* values as well as splice site-level *P* values. Similar to expression outliers, FRASER typically returns a handful of aberrant splicing events per sample that are significant at an FDR < 0.1. For example, the median number of outlier events in the test dataset are 2 for ψ_5, 2 for ψ_3, and 5 for θ.

Three other methods have been used to detect aberrant splicing, which are (i) an adaptation of the differential splicing test LeafCutter[17] used in the Kremer et al. study[7], (ii) a cutoff based approach used in the studies of Cummings et al.[8] and Gonorazky et al.[10], and (iii) a z score based method used in the Frésard et al. study[9]. The first one constructs intron clusters and tests for differential usage between one sample and all others, instead of aberrant splicing events. Moreover, it does not control for sample covariation. The second one defines aberrant splicing events as novel introns in genes with enough reads in the affected individual but not (or almost not) appearing in a control cohort, after applying local normalization. The two main caveats are that it depends on arbitrary cut-offs and may fail to recognize aberrant splicing events in weak splice sites[35]. The third method does correct for covariation, but uses a z score approach which does not offer any control for false discovery rate and can be inaccurate in detecting splice sites with low reads. Having in mind these limitations, we have opted for FRASER[19], which not only addresses all those issues but also detects intron retention.

Mono-allelic expression

MAE refers to the expression of a single allele out of the two alleles of a gene. When assuming a recessive mode of inheritance, single heterozygous rare variants are not prioritized after DNA sequencing. However, mono-allelic expression of such a single heterozygous rare variant in an affected individual, which could be due to genetic or epigenetic silencing of the other allele, is consistent with a recessive mode of inheritance. Therefore, detecting MAE of a rare variant has helped to diagnose rare disorders[7,9,10,32,36,37]. Rare disorders can also arise due to *de novo* mutations in haploinsufficient genes[38,39], in which case MAE of either the reference or alternative allele can help highlight those genes.

Detecting mono-allelically expressed genes relies on counting the reads aligned to each allele at genomic positions of heterozygous variants. Several methods have been developed to detect MAE in the context of rare diseases, among which are the one described by Kremer et al.[7], and more recently, ANEVA-DOT[36]. On the one hand, Kremer et al. used a negative binomial test with a fixed dispersion for all genes. On the other hand, ANEVA-DOT implements a binomial-logit-normal test with gene-specific variance, with the caveat that due to insufficient training data, estimates of this variance have been computed so far for only 4,962 genes (in median), depending on the tissue of interest[36]. As using ANEVA-DOT would result in losing more than half of the tested genes, we opted for the training data-independent negative binomial test (Supplementary Methods).

In the test dataset, we found a median of 306.5 heterozygous single nucleotide variants (SNVs) to be mono-allelically expressed per sample. Of them, 239 SNVs (in median) expressed the reference allele, out of which 10 (median) were rare (MAX AF ≤ 0.001). Of the 62.5 SNVs (median) expressing the alternative allele, 2.5 (median) were rare.

VCF-BAM matching

A crucial step when performing multi-omics is to ascertain that all assays performed on samples obtained from the same individual correspond. Therefore, we designed an algorithm to match the variants derived from DNA and RNA sequencing which is based on the ideas proposed by t' Hoen et al.[40] and Lee et al.[41] (Supplementary Methods). We achieved this by comparing the BAM files from RNA-Seq with the VCF files from DNA sequencing at predefined genomic positions of variants that are not in linkage disequilibrium. The proportion of variants derived from the same individual matching in the DNA and RNA has to be significantly higher than the one from different individuals, but will not reach 100% due to MAE[40]. The algorithm is applied not only to the annotated VCF-BAM samples, but to all combinations in order to find other possible unannotated matches.

In the test dataset, the median of the proportion of matching genotypes of samples from the same individual is 0.98 compared to 0.58 in the case of non-matching samples. In the case of family members, we expect a value in-between these. All of the 100 RNA samples from the test dataset match their corresponding DNA, as expected since it is a quality-controlled public dataset. The VCF-BAM matching procedure is part of the MAE module, but can also be run independently. We provide a VCF file containing the set of genomic positions to be tested in our resource folder (see Box 2).

Detection of RNA Outliers Pipeline

The workflow of DROP consists of three independent modules (Fig. 1). It is built using R and Python on top of the workflow manager frameworks Snakemake[14] and wBuild (Box 1). DROP offers a parallelized backend through Snakemake to execute the same processes on different samples at the same time, thus making effective use of computer clusters. It runs on the command line. Detailed instructions for installation, editing the input files, and executing the workflow can be found in the DROP documentation. The functions used throughout the workflow are explained in detail in the Supplementary Methods section. Finally, as the source code is open source and the design of our workflow is flexible, we encourage the community to expand the workflow with new modules, e.g. by adding other multi-omics modules, prediction models, methods for calling variants on RNA-seq data, or gene-fusion detection algorithms.

Dataset design

In general, samples originating from the same tissue and that were prepared and sequenced similarly should be analyzed as separate groups. Power analyses have suggested analyzing groups of at least 50 samples for aberrant expression[18] and at least 30 samples for aberrant splicing[19]. In this section, we discuss whether it is advisable to combine samples from different cohorts if the original sample size is smaller than the recommended values. We recommend that sequencing data are aligned using the same reference genome build, aligner, and parameters, before processing them jointly with DROP.

One strategy is to include samples originating from the same tissue but from another cohort. Simulations detailed below indicate that this setting leads to an increased, yet manageable, list of reported outliers (less than 10 fold larger in these simulations), with no strong loss of sensitivity (less than 30%). We simulated the effect of merging RNA-seq data from one diagnostic lab with RNA-seq data with a public resource (GTEx[21]). The diagnostic lab samples were the Kremer samples, which are derived from skin fibroblast cells[7]. We used the GTEx samples derived from suprapubic skin (not sun exposed) as external samples. The samples from both cohorts were not strand-specifically sequenced and were aligned to the hg19 genome build in the original studies. To investigate the effect on calling expression outliers, we simulated 30 heterogeneous datasets, each with sample size equal to the one of the original Kremer dataset (n=119). Each heterogeneous dataset consisted of 102 randomly picked samples from GTEx and the 17 samples of the Kremer dataset with a confirmed pathogenic protein truncation variant that leads to aberrant expression (probably through nonsense-mediated decay). OUTRIDER could not correct these 30 simulated heterogeneous datasets as effectively as it could correct the original Kremer dataset. Suboptimal OUTRIDER correction is evident from larger correlation values and the 17 Kremer samples clustering together after correction (heatmaps in Supplementary Fig. 1), as well as larger deviations between observed read counts and OUTRIDER estimates (larger post-correction biological coefficient of variation, Supplementary Fig. 2a). Moreover, the number of outliers per sample of the 17 Kremer samples increased by around 8 fold in the heterogeneous setting (median of 23 outliers per sample) compared to the original setting (median of 3 outliers per sample) using the recommended OUTRIDER FDR cutoff of 0.05. Importantly, more than 70% of those 17 pathogenic outliers were detected on median (Supplementary Figure 2b,c). With an FDR cutoff of 0.3, more than 80% of the 17 pathogenic outliers are detected, but at a cost of reporting 67 outliers in median per sample. Heterogeneous datasets were also simulated for investigating aberrant splicing calling, using the 13 Kremer samples with a confirmed pathogenic splicing defect. More than 75% of those 13 splicing pathogenic outliers were detected in the heterogeneous datasets, at a cost of obtaining 34 outliers in median per sample, instead of 14 in the Kremer dataset, using the recommended FRASER FDR cutoff of 0.1 (Supplementary Figure 2d,e). Loosening the FDR cutoff to 0.5 did not recover more true positives (Supplementary Figure 2e). Altogether, this analysis indicates that combining small cohorts with samples from public resources appears to recover pathogenic aberrant expression events at enough sensitivity and specificity to be useful for diagnostics.

Expression and splicing patterns are known to differ across tissues[21]. In order to test the effect of combining samples from different tissues, we combined 100 GTEx samples from whole blood with 100 samples from either suprapubic skin, skeletal muscle, cerebellum (brain), or liver. Even though the number of expression and splicing outliers did not increase when combining blood with other tissues with respect to blood alone, only ~50% of expression outliers (and 30% of splicing outliers) found in blood were recovered in the combined datasets (Supplementary Fig. 3). Therefore, we conclude that it is better not to merge samples from different tissues.

Sequencing costs can be reduced by higher multiplexing, yielding a lower sequencing depth per sample. To investigate the effect of sequencing depth, we downsampled reads from the 17 expression true positives (and 13 splicing true positives) to sum to a total sequencing depth of ~30 million reads and merged them with the rest of the Kremer dataset which has a median depth of ~86 million reads (Supplementary Fig. 4a). At a lower depth, fewer expression and splicing outliers per sample were detected, retrieving 88% of the 17 pathogenic expression outliers, but only 46% of the 13 pathogenic splicing outliers (Supplementary Fig. 4b-d). Calling splicing outliers relies on split reads which requires a higher sequence depth than calling expression outliers.

We do not recommend merging strand-specific with non-strand-specific samples. Reads that overlap two genes lying on different strands will be assigned to both genes if the sequencing was not strand-specific, while only to the correct gene if it was strand-specific.

Diagnostic labs may encounter further situations we could not investigate (e.g., merging polyA selection with ribo-depleted RNA samples, or FFPE tissues with fresh frozen tissues).

Generally, after running OUTRIDER and FRASER, we recommend investigating the sample correlation heatmaps (Supplementary Fig. 1). If the autoencoder was able to successfully remove the covariation, they should look similar to Figure 2d and Supplementary Fig. 1b and 1f. On the contrary, if the autoencoder fails, the heatmaps will look like Supplementary Fig. 1d and 1h.

Dealing with external count matrices

To overcome the limitation of small sample sizes, DROP is able to integrate them with precomputed count matrices. These pre-computed matrices include gene-level counts for the aberrant expression module, and split counts spanning from one exon to another, and non-split counts covering exon-intron boundaries for the aberrant splicing module. The location of these files is included in the sample annotation table (Procedure). Afterwards, a new analysis is created for samples with matching DROP_GROUP and GENE_ANNOTATION columns. Refer to the documentation for examples and Online Resources (Box 2) for available datasets.

Other uses of RNA-seq in diagnostics of rare diseases

Besides the three strategies already mentioned, RNA-seq data provide other valuable information. First, variants can be called from BAM files derived from RNA-seq. This has limited value when WGS is available, but is useful in the situation where only WES is available as RNA-seq provides better coverage of the UTRs and can reveal intronic variants on aberrantly-created exons. Variant calling in RNA-seq can be done by following the “RNA-seq short variant discovery (SNPs + Indels)” workflow from GATK (Box 2). The inputs are BAM files from RNA-seq and the output are VCF files that can later be annotated and used to find disease-causal variants. Second, variant phasing can be done using RNA-seq data. Due to splicing, variants can be phased over longer distances than WES or WGS. Phasing is useful in the clinical setting by allowing to distinguish between compound heterozygotes and variants on the same allele[42]. Phasing variants can be achieved using phASER[42] (Box 2). The inputs are BAM files from RNA-seq and VCF files from DNA sequencing, and the output is a phased VCF file. Third, gene fusion can be detected from transcriptomic data. Gene fusion happens when, due to chromosomal translocation, inversion, deletion, or duplication, genetic material from different genes are merged and transcribed together. Even though gene fusion has been used more extensively in cancer diagnosis[43,44], it has also been successfully applied to diagnose patients with congenital[45] and rare diseases[46]. CICERO[47] is a recently-developed program to obtain gene fusion calls. The inputs are BAM files from RNA-seq and a reference genome. The output are ranked gene fusion calls which can be visualized interactively using FusionEditor (Box 2). Integration of these and other tools as DROP modules could be considered in the future.

Expertise needed

The installation, creation of a Python environment, and set up of the config file should be done preferably by either a bioinformatician or a system administrator. Afterwards, the protocol can be followed by any scientist.

Materials

Equipment

BAM files from RNA-seq data

This workflow focuses on the analysis of the sequencing data rather than on their generation and preprocessing. In this subsection, we explain how the BAM files (Table 1) should be generated to serve as input for DROP. BAM files are created by aligning FASTA files derived from RNA-seq to a reference genome. We recommend using the aligner STAR[48] with the default parameters and `twopassMode = 'Basic'` to detect novel splice junctions. We strongly recommend performing quality control with tools like MultiQC[49]. The BAM files contain reads that will be used in all of the modules to generate the read count matrices. The location is specified in the sample annotation table. The BAM files must be sorted by position and indexed. This can be achieved using the commands `sort -o` and `index` from SAMtools[11]. Under default settings, DROP does not require the BAM files to contain any particular tag or field.

VCF files from either WES or WGS

VCF files (Table 1) are generated through calling variants on a BAM file. Therefore, the FASTA files derived from DNA sequencing must be aligned to a reference genome first (one possible tool is the Burrows-Wheeler Aligner[50]). BAM files must be sorted by position and indexed. Afterwards, variants can be called by following, for example, the GATK guidelines[51,52]. This will generate either one VCF file per DNA sample, or a single VCF file containing multiple samples.

DROP can handle either option. VCF files must be compressed and indexed. This can be achieved by using the `bgzip -c` and `tabix -p` commands from Tabix[53], respectively. They do not need to be stored in the same folder, as the location is specified in the sample annotation table. We further recommend annotating the VCF file(s) with the Variant Effect Predictor[54] following their tutorial (Box 2). This will add information such as variant consequences (e.g. missense, stop-gained) and allele frequencies. The genome build used to align the files derived from DNA sequencing must be the same as the one used to align the files derived from RNA sequencing.

Human reference genome file

In order to compute the allelic counts, a human reference genome (FASTA) file is needed. Preferably, the file should be the same that was used to align the samples' FASTA files. Otherwise, the user can download it from different sources, as long as the genome build (either GRCh37 (hg19), or GRCh38 (hg38)) matches the BAM files. For example, for build 19 download the FASTA file from https://hgdownload.soe.ucsc.edu/goldenPath/hg19/bigZips/, hg19.fa.gz; and for build 38 https://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/latest/, hg38.fa.gz, from UCSC[55]. Finally, an index (.fai) file must be created in the same directory where the FASTA file is located using the function `faidx` from SAMtools[11].

Gene annotation file

In order to count the reads from the BAM files in the aberrant expression module, a gene annotation file is required. DROP was developed using the 'main annotation' gtf file from the release 29 of the GENCODE annotation[56]. Nevertheless, newer releases already exist and will continue to appear. As gtf files from other sources can have other columns and different names, DROP might not be able to parse them. Therefore, to avoid problems, we recommend users to download the latest release from GENCODE, with the right genome build (matching the one used for read alignment) (https://www.gencodegenes.org/human/).

Hardware

A multi-core computer with at least 16 cores and 64 GB RAM is suggested, but depends heavily on the analyzed dataset (e.g., sample size). For developing and testing DROP on the 100 samples from the Geuvadis dataset, a Scientific Linux 7.7 based server with 32 physical cores and 256 GB memory was used.

Software

- A Linux based operating system
- Conda (https://docs.conda.io/projects/conda/en/latest/user-guide/install/)
- DROP available as a Bioconda package (https://github.com/gagneurlab/drop). The current version is 0.9.2, but look for updates.
- RStudio (https://www.rstudio.com/products/rstudio/download/) or any other integrated development environment.

- Integrative Genomics Viewer (IGV)[57] (https://software.broadinstitute.org/software/igv/)

Procedure

Configuration of the workflow (2-3 h)

1. Define a directory where the analysis code will be stored. Go to that directory and open a terminal. Set up a new DROP-based project by executing `drop init`. This will create a config file, a Snakemake file (Snakefile), and a Scripts directory for analysis that can be modified by the user.
2. Adapt the configuration file automatically generated in the previous step. This file specifies values for parameters following the human-readable YAML format[58] (Box 2). Parameters that always need to be adjusted like file paths are empty. Other parameters come with default values (e.g. FDR cutoff: 0.05). The config file used to run the demo dataset is available in Supplementary Data 2. A detailed description of all the parameters with default and example values is available in the DROP documentation (Box 2).
3. Create the sample annotation table. Each row should correspond to a unique pair of RNA and DNA samples derived from the same individual. An RNA assay can belong to one or more DNA assays, and vice-versa. If so, they must be specified in different rows. The required columns are RNA_ID, RNA_BAM_FILE, and DROP_GROUP, plus other module-specific ones. Save the annotation table in the tsv format (Table 1). The column order does not matter. Also, it does not matter where the table is stored, as the path is specified in the config file. An example annotation table is shown in Supplementary Data 1. Further instructions and examples can be found in the DROP documentation. For a detailed description of the annotation table columns, see Box 3.

?TROUBLESHOOTING

Box 3 – Sample annotation table columns

- RNA_ID: unique identifier from an RNA assay.
- RNA_BAM_FILE: absolute path of the BAM file derived from RNA-seq (e.g. /home/project1/Data/RNA1.bam, and not RNA1.bam). A BAM file can belong to only one RNA_ID, and vice-versa.
- DROP_GROUP: the analysis group(s) that the RNA assay belongs to. Multiple groups must be separated by commas and no spaces (e.g., blood,WES,groupA). We recommend doing a different analysis for each tissue as gene expression and splicing can be tissue-specific.

MAE module- specific columns:

- DNA_ID: unique identifier from a DNA assay.
- DNA_VCF_FILE: absolute path to the corresponding VCF file (e.g., /home/project1/Data/sample1.vcf.gz, and not sample1.vcf). The DNA_ID has to match the ID inside the VCF file. In case a multi-sample VCF is used, write the file name for each sample.

The following columns describe the RNA-seq experimental set-up. They affect the counting procedures of the aberrant expression and splicing modules. For a detailed explanation, refer to the documentation of HTSeq[59].

- PAIRED_END: either TRUE or FALSE, depending on whether the sample comes from paired-end RNA-seq or not.
- STRAND: either yes, no or reverse: “no” means that the sequencing was not strand-specific; “yes” that it was strand-specific, and the first read in the pair is on the same strand as the feature and the second read on the opposite strand; and “reverse” that the sequencing is strand-specific and the first read in the pair is on the opposite strand as the feature and the second read on the same strand.

The following columns describe how the counting should be performed in the aberrant expression module. We recommend setting the count mode to IntersectionStrict to consider only reads that fully align within the gene, and count overlaps to TRUE to detect as many genes as possible. Refer to the documentation of HTSeq[59] for details.

- COUNT_MODE: either “Union”, “IntersectionStrict” or “IntersectionNotEmpty”.
- COUNT_OVERLAPS: either TRUE or FALSE, depending on whether reads overlapping different regions are allowed and counted.

The following columns are optional:

- INDIVIDUAL_ID: unique identifier of an individual. Useful when dealing with replicates.
- HPO_TERMS: comma-separated phenotypes encoded as HPO (Human Phenotype Ontology) terms (e.g., HP:0001479,HP:0005591).

In case external count matrices are used, the following columns are needed:

- GENE_COUNTS_FILE: location of a gene-level count matrix.
- GENE_ANNOTATON: gene annotation used to obtain the count matrix. Must correspond to the key of an entry in the `geneAnnotation` parameter of the config file.
- SPLIT_COUNTS_FILE: location of a matrix containing the split counts.
- NON_SPLIT_COUNTS_FILE: location of a matrix containing the non-split counts.

Other recommended columns are TISSUE, SEX, and DISEASE. Check the documentation for examples.

[PRODUCTION: END OF BOX 3]

4. Execute `snakemake sampleAnnotation`. This command will parse and check the provided sample annotation, and create the annotation overview webpage (Table 2). The annotation overview contains information regarding the sample annotation and different sanity checks. Go through it carefully and, if detected, correct any errors in the sample annotation file. Check if the number of samples in each DROP group is correct. Also, if values were provided in the HPO_TERMS column, this command will create a text file containing all the genes that overlap the HPO terms of each sample (Table 2).
5. Execute `snakemake -n`. It will display all the jobs that must be performed. The full workflow can already be run by executing `snakemake <options>`. Nevertheless, we recommend to run it by modules. For a detailed explanation of the "options", check the DROP documentation. ?TROUBLESHOOTING

The aberrant expression module (3-4 h for 100 samples and 20 cores)

6. To run the aberrant expression module execute `snakemake <options> aberrantExpression`. This step is computationally heavy and can take several hours. This command counts the reads for each sample (Supplementary Methods), performs the OUTRIDER fit, extracts the results, and creates two HTML reports with different plots for counting and results (Supplementary Fig. 5a). This analysis is performed for each annotation and group combination defined in the geneAnnotation parameter (config file) and in the DROP_GROUP column (annotation file), respectively. Different intermediate and final objects are saved locally (Table 2) and can be used for further analyses.

Table 2 | Files created by DROP and their locations. The variables <root>, <htmlOutputPath>, and <geneAnnotation> are defined in the config file. The variable <group> corresponds to the DROP_GROUP column from the sample annotation.

File	**Module**	**Path**
Sample annotation overview	---	<htmlOutputPath>/Scripts_Pipeline_SampleAnnotation.html
Analysis scripts	all	<project>/Scripts/
Merged counts table	Ab. Exp.	<root>/processed_data/aberrant_expression/<geneAnnotation>/outrider/<group>/total_counts.tsv
OUTRIDER full results table	Ab. Exp.	<root>/processed_results/aberrant_expression/<geneAnnotation>/outrider/<group>/OUTRIDER_results_all.Rds
OUTRIDER significant-only results table	Ab. Exp.	<root>/processed_results/aberrant_expression/<geneAnnotation>/outrider/<group>/OUTRIDER_results.tsv
Fitted OUTRIDER dataset (ods) object	Ab. Exp.	<root>/processed_results/aberrant_expression/<geneAnnotation>/outrider/<group>/ods.Rds
Aberrant expression module webpage	Ab. Exp.	<htmlOutputPath>/aberrant-expression-pipeline_index.html
Aberrant expression analysis webpage	Ab. Exp.	<htmlOutputPath>/Scripts_AberrantExpression_Overview.html
Fitted FRASER dataset (fds) object	Ab. Spl.	<root>/<processed_data>/aberrant_splicing/datasets/savedObjects/<group>/fds-object.RDS
FRASER significant-only results table	Ab. Spl.	<root>/<processed_data>/aberrant_splicing/results/{dataset}_results.tsv
Aberrant splicing module webpage	Ab. Spl.	<htmlOutputPath>/aberrant-splicing-pipeline_index.html
Aberrant splicing analysis webpage	Ab. Spl.	<htmlOutputPath>/Scripts_AberrantSplicing_Overview.html
MAE individual results	MAE	<root>/processed_results/mae/samples{DNA_ID}--{RNA_ID}_res.Rds
MAE full results table	MAE	<root>/processed_results/mae/<group>/MAE_results_all_<geneAnnotation>.tsv.gz

MAE significant-only results table	MAE	<root>/processed_results/mae/<group>/MAE_results_<geneAnnotation>.tsv
MAE module webpage	MAE	<htmlOutputPath>/mae-pipeline_index.html
MAE analysis webpage	MAE	<htmlOutputPath>/Scripts_MAE_Overview.html

7. To look at the counting outcome, open the aberrant expression module webpage (Table 2). Go to the Counting tab. It contains one link per analysis group. Click on the analysis group you want to visualize. It contains the following quality control plots:
    - Number of reads counted per sample.
    - Percentage of reads counted per sample (Supplementary Fig. 5b).
    - Size factors per sample (Supplementary Fig. 5c).
    - Size factors vs read count ratio.
    - Raw read count distribution (Fig. 2a).
    - Number of expressed genes (Fig. 2b). A total of 17,259 genes passed the filter in the test dataset. ?TROUBLESHOOTING
8. To investigate the results, open the aberrant expression module webpage. Go to the OUTRIDER tab. It contains one link per analysis group. Click on the analysis group you want to visualize. It contains the following:
    - Fit of the encoding dimension plot (Supplementary Fig. 5d).
    - Number of aberrant genes per sample plot (Supplementary Fig. 5e).
    - Heatmaps showing the sample correlation before (Fig. 2c) and after the autoencoder correction for confounders (Fig. 2d). The clustering should disappear after the correction.
    - Heatmaps showing the log row-centered gene expression of the 50 most variable genes before and after the correction. This is useful to identify the genes that drive the clustering of samples.
    - Gene-wise biological coefficient of variation[60] plot before and after the correction.
    - Table with aberrant samples.
    - Searchable results table with overlapping HPO terms (if specified) and links to download both the full and subset results tables.

The aberrant splicing module (3-4 h for 100 samples and 20 cores)

9. To run the aberrant splicing module, execute `snakemake <options> aberrantSplicing`. This step is also computationally heavy and can take several hours. It creates a FRASER dataset object that contains all the intron-centric metrics and results (Supplementary Methods), an HTML page with overview plots, and the splicing outlier results table (Supplementary Fig. 6a).
10. Open the aberrant splicing module webpage. Go to the Counting tab. Click on the analysis group you want to visualize. It contains the following:
    - Introns and splice sites that passed the filter
    - Filter expression plot (Fig. 3a)
    - Filter variability plot
11. Open the aberrant splicing module overview webpage. Go to the FRASER tab. Click on the analysis group you want to visualize. It contains the following:
    - Plot showing the area under the precision-recall curve using different encoding dimensions (*q*) for the autoencoder for ψ_3 (Supplementary Fig. 6b), ψ_5 (Supplementary Fig. 6c), and θ, in order to find the optimal dimension.
    - Number of aberrant ψ_3, ψ_5 and θ events per sample plot (Fig. 3b).
    - Heatmaps (sample vs. sample) before (Fig. 3c) and after autoencoder correction (Fig. 3d).
    - Searchable results table for all metrics aggregated by gene, with overlapping HPO terms (if specified).

Mono-allelic Expression Module (22-24 h for 100 samples and 20 cores)

12. To run the MAE module execute `snakemake <options> mae`. This creates the allelic counts, aggregated results table and the overview webpage (Supplementary Fig. 7). The results for each sample contain the counts, the nominal and the adjusted *P* values, and the alternative allele ratio of each variant (Supplementary Methods). Moreover, the results are annotated with the minor allele frequencies from gnomAD[61] (for each of the African, American, East Asian and Non-Finnish European populations, and the maximal frequency among them), and a Boolean column "rare" if the allele frequency

was below the corresponding cut-off specified in the config file (default=0.001). This step is computationally heavy and can take several hours.

13. Open the MAE overview webpage (Table 2). Go to the MAE tab. It contains the following information:
    - Cascade plot (Fig. 4a) showing the number of heterozygous SNVs for the four cumulative filter criteria: at least 10 reads, mono-allelically expressed for either the reference or alternative allele, for the alternative only, and rare as defined by the config file allele frequency cutoff (step 12).
    - Searchable results table containing only the significant events with overlapping HPO terms (if specified). To subset for rare variants, simply filter the “rare” column.
    - Links to download full and significant-only results tables.

VCF - BAM matching

14. Download the test VCF file (qc_vcf_1000G.vcf.gz) and its index file from the DROP resource webpage (Box 2). Its location is specified in the config file.
15. Execute `snakemake <options> sampleQC`. This step is computationally heavy and can take several hours.
16. Open the MAE overview webpage (Table 2). Go to the QC tab. It contains:
    - Histogram of the percentage of matching genotypes between DNA and RNA (Fig. 4b).
    - Table containing the samples that were annotated to match, but actually do not (empty if there are no mismatches).
    - Table containing the samples that were not annotated to match, but actually do (empty if there are no such cases).

Exploration of outlier results (30 min per sample)

17. Execute `snakemake <options>`. This will run all the analysis scripts and create a webpage for each of them with case-specific plots.
18. Open the aberrant expression analysis webpage (Table 2). It contains the following:
    - Links to the overview webpages, and the paths of the OUTRIDER dataset objects and results tables.

- Volcano plot ($-\log_{10}$(P value) vs. z score) of the first sample from the results table. Figure 5a shows a typical scenario in which the expression of most of the genes of a sample does not significantly deviate from the rest of the population with the exception of a handful of genes clearly distinguishable from the other ones.
- Expression plot (normalized counts across all samples) of the first gene from the results table (Fig. 5b). This plot not only shows if the gene of interest is aberrantly expressed in a sample, but also the magnitude of the difference. In this case, the expression of gene *MUTYH* lies in a range of 350-500 counts for all samples except for one, where it is around 200 counts, thus implying a ~50% reduction.
- Observed against expected counts plot of the same given gene across all samples (Supplementary Fig. 8a). Large variations of expected values (OUTRIDER predictions, x-axis) are typical and reflect variations in sequencing depth and gene expression co-variation.

19. Open the aberrant expression analysis script (Table 2) in your preferred editor (e.g., RStudio). It contains the code that generated the analysis webpage. Modify the code by specifying a sample and gene of interest and recreate the plots from the previous step by executing the modified lines of code. Alternatively, save the script and execute `snakemake Scripts_AberrantExpressionAnalysis_Overview_R` to generate an updated analysis webpage.
20. Open the aberrant splicing analysis webpage (Table 2). It contains the following:
    - Links to the overview webpages, and the paths of the FRASER dataset objects and results tables.
    - Volcano plot ($-\log_{10}$(P value) vs. $\Delta\psi$) of the first sample from the results table (Supplementary Fig. 8b).
    - Expression plot (junction count versus total junction coverage) for ψ_3 of the first junction from the results table across all samples (Supplementary Fig. 8c). In this example, for most of the samples, the count of the junction was the same as the total count of the corresponding acceptor. Nevertheless, in one sample, the total count for the acceptor is around 3 times higher than the tested junction. This means that most of its spliced reads come from another donor not present in any other sample.

- Observed vs. predicted counts plot for ψ_3 of the same junction across all samples (Supplementary Fig. 8d).
- Quantile-quantile plot of observed vs. expected *P* values for ψ_3 of the same junction to evaluate the fit (Supplementary Fig. 8e).

21. Open the aberrant splicing analysis script (Table 2) in RStudio. Once again, go through the different lines of code and modify them to specify the sample and junction of interest. Recreate the plots. Run `snakemake Scripts_AberrantSplicingAnalysis_Overview_R` to display the plots in the analysis webpage. These plots indicate if an event was an outlier by visualizing all samples.
22. Sashimi plots provide a better visualization of the nature of the splicing aberration. To create a sashimi plot, load in IGV[57] the BAM file of the sample of interest and also other BAM files that do not have this splicing outlier (File > Load from File, then select the BAM file). Go to the genomic position of the junction of interest obtained from the FRASER results table. Adjust the zoom to cover the entire region of interest and create a sashimi plot by right clicking and selecting 'Sashimi Plot' from the resulting menu. The bars represent the coverage for each alignment track, and the arcs the splice junctions connecting exons with the number of reads split across the junction. Therefore, sashimi plots can be used for both qualitative and quantitative analysis of transcripts. High-quality sashimi plots (Fig. 5c) can be achieved using the MISO package[62]. We provide an example script, config and gff file needed to generate a MISO sashimi plot in the DROP GitHub webpage (Box 2).
23. Open the MAE analysis webpage (Table 2). It contains the following:
    - Links to the overview webpages, and the paths of the results per sample.
    - MA plot (fold change vs. RNA coverage) of the first sample from the results table, colored by significance and rarity (Fig. 5d). As expected, a lower proportion of variants are mono-allelically expressed towards the alternative than towards the reference, and from them only a handful are rare.
    - Alternative vs. reference allele coverage plot of the same sample (Supplementary Fig. 8f), which gives a similar overview as the previous plot.
24. Open the MAE analysis script (Table 2) in RStudio. Once again, go through the different lines of code and modify them to specify the sample of interest. Note that the format of the identifier is {RNA_ID}--{DNA_ID}. Recreate the plots. Run `snakemake Scripts_MAEAnalysis_Overview_R` to display the plots in the analysis webpage.

25. Inspect the mono-allelically expressed variants in IGV. Load the BAM file of interest in IGV. If available, load the corresponding WES or WGS BAM file, and other BAM files as controls. Go to the position of the mono-allelically expressed variant (from the MAE results table) and confirm the detected mono-allelic expression.

Adding or editing input data (highly variable depending on the edits)

26. Every time a new input file is added or edited, execute `snakemake -n`. DROP will automatically identify the steps that need to be run to obtain the new results. Run each of the modules and analysis scripts as done before. Be sure to include any new sample in the sample annotation table. Alternatively, execute `snakemake <options>` to run the full workflow. Changes in the config file and in the sample annotation (except for added rows) are not recognized by DROP. Therefore, executing `snakemake -n` will reflect no new jobs to be performed. A way to force one of the modules to be executed is by running `snakemake --forcerun <module>`. Refer to the documentation for more details.

Troubleshooting

Troubleshooting advice can be found in Table 3.

Table 3. Troubleshooting table.

Step	**Problem**	**Possible reason(s)**	**Solution(s)**
3	The sample annotation does not load properly	There were problems with parsing special values	1. Avoid special characters 2. The DROP_group column has comma separated values, so be sure to save the sample annotation with tab-separated values (tsv). 3. Go through the sample annotation html report and try to find any inconsistency.
7	The gene counts are very low	The read orientation specified in the sample annotation is not the correct one.	Check the RNA-seq protocol of the samples to determine whether they were paired-end, strand-specific and to which strand the first read aligns to. Load the BAM files in IGV and

			color them by first-of-pair strand. Go to a gene that lies on the forward strand. Scroll over the reads and note the “Pair orientation” value. Set the STRAND column from the sample annotation to: - “no” if the reads have multiple colors - “yes” if the “Pair orientation” is F1R2 - “reverse” if the “Pair orientation” is F2R1
7	Very few genes are counted	A gtf file from a different genome build was used.	Be sure to provide a gtf file with the same build (hg 19 or hg38) as used for aligning the BAM files.
		Too low coverage.	Check the coverage of the BAM files. The sequencing depth should be at least 30 million reads.
All	Any other problem during the installation or running of the different modules	Even though we have tested DROP in numerous data sets, there might still be problems that we have not encountered yet.	Go to DROP’s GitHub webpage and click on the ‘Issues’ tab. Scroll through the issues to check if someone else had the same problem already. If not, create a new issue by clicking on ‘New issue’. Copy the error and the step in which it occurred, together with a description of the dataset. We will reply as soon as possible.
All	The results contained either too few or too many outliers	This could be due to many factors such as combining samples sequenced using different protocols, bad choice of cutoffs or other parameters, or a problem with the workflow.	Create an issue in DROP’s GitHub following the steps described in the previous point.

Timing

The timings below are provided for the test dataset (composed of 100 samples) with 20 available CPU cores and 96 GB RAM. However, the performance highly depends on the dataset size, sequencing depth, and the number of available CPU cores and RAM.

Installation and step 1: 3 h max if all dependencies must be installed.

Step 2, Adapting the configuration file: 15 min.

Steps 3 – 5, Creating the sample annotation: ~2 h depending on the experimental design and annotations at hand.

Steps 6 – 8, Aberrant expression module: ~30 min counting/sample/core, ~1 h merging, filtering, fitting and generating the results. The counting needed 4 GB/sample. The step that required the most memory was the OUTRIDER fit with ~55 GB.

Steps 9 – 11, Aberrant splicing module: ~5 min counting split reads per sample using 3 cores, ~3 min counting non-split reads per sample using 3 cores, ~1 h for merging, computing ψ and θ, filtering, fitting and generating the results. The counting of the split reads needed 8 GB/sample. The step that required the most memory was computing z scores and *P* values with ~80 GB.

Steps 12 – 13, MAE module: ~2 h counting/sample/core, 1 h for generation of all results. Running the negative binomial test needed 11 GB/sample, which was also the step that required the most memory.

Steps 14 – 16, VCF - BAM matching: similar to the MAE module.

Steps 17 – 25, Exploration of outlier results: going through each of the three analysis pages can take up to one hour. Then, exploring each sample can take around half an hour per sample.

Step 26, Adding or editing input data: several hours depending on the changes.

Anticipated results

One can expect a handful of expression outliers per sample. In the test dataset the median number of expression outliers is 1 and the 90th percentile of samples had 7 expression outliers (Supplementary Fig. 5e), in line with previous reports[7,18] and other in-house analyses. These expression outliers should be interpreted according to their fold changes. A strong down-regulation (fold-changes < 0.2) probably implies impaired gene function. Fold-changes of weaker amplitudes should be investigated further. In particular a fold-change of 0.5 often reflects loss of expression of one allele. Inspecting the MAE results may further reveal mono-allelic expression of a rare variant harbored by the other allele[7,10].

One can also expect a handful of splicing outliers per sample. These aberrant splicing events should be visualized as sashimi plots, for instance using the genome browser IGV, to detect the nature of the splicing defect, e.g. intron retention, exon truncation, exon elongation, or exon skipping. The next step is to look for direct splice site and near splice site variants that can explain the defect. However, other variants such as synonymous and deep intronic variants, have also been described to activate new splice sites[9] potentially originating from cryptic exons[7,8,10,29].

Analysis of RNA-seq data delivers candidate genes whose aberrant expression or splicing may be contributing to the clinical presentation of the patient. Analysis of the gene fusion results can further provide candidate genes. If HPO-encoded phenotypes are provided in the sample annotation, DROP also outputs whether the outlier genes overlap with them. Further prioritization involves adding specific disease gene lists, or general ones such as OMIM[63]. Promising candidate genes need to be further investigated for potential pathogenic variants. RNA-seq informs about consequences of genetic variants but does not necessarily allow pinpointing the causal variant. Nonetheless, in some instances a splice defect is directly explained by a variant in the splice site. Also, outliers with low expression can be explained by nonsense mediated decay triggered by premature termination codons caused by stop-gain variants or frameshift variants. In this respect the value of RNA-seq is to support prioritization of candidate genes which need further inspection or functional validation of an existing candidate variant. In any case, for molecular diagnostic interpretation we refer users to the ACMG standards and guidelines for the interpretation of sequence variants[64].

Acknowledgements

We thank all the users who helped with their feedback during the revision, especially David Murdock. We also thank Carla Andrade for helping with the DROP logo, as well as the members of the Gagneur Lab for input. V.A.Y., C.M., H.P., and J.G. were supported by the EU Horizon2020 Collaborative Research Project SOUND (633974). The Bavaria California Technology Center supported C.M through a fellowship. The German Bundesministerium für Bildung und Forschung (BMBF) supported the study through the e:Med Networking fonds AbCD-Net (FKZ 01ZX1706A to V.A.Y., C.M., and J.G.), the German Network for Mitochondrial Disorders (mitoNET; 01GM1113C to H.P.), and the E-Rare project GENOMIT (01GM1207 to H.P.). A fellowship through the Graduate School of Quantitative Biosciences Munich supports V.A.Y.. The Genotype-Tissue Expression (GTEx) Project was supported by the Common Fund of the Office of the Director of the National Institutes of Health, and by NCI, NHGRI, NHLBI, NIDA, NIMH, and NINDS.

Author contributions

V.A.Y., C.M., M.F.M., and J.G. participated in the design of the workflow. V.A.Y., C.M., M.F.M., D.K-S.A., I.S., and P.F.G. contributed to the computational workflow. L.F implemented the candidate prioritization workflow. L.W. designed and implemented wBuild. V.A.Y. and J.G. wrote the manuscript with the help of L.F, D.K-S.A., M.G., and I.S.. C.M., H.P., and J.G. supervised the research. All authors revised the manuscript.

Competing interests

The authors declare no competing interests.

Box 2: Online resources

DROP documentation: https://drop-rna.readthedocs.io/en/latest/
DROP in github: https://github.com/gagneurlab/DROP
DROP resource folder: https://www.cmm.in.tum.de/public/paper/drop_analysis/resource/
DROP test dataset analysis:
https://www.cmm.in.tum.de/public/paper/drop_analysis/webDir/html/drop_analysis_index.html
DROP Kremer dataset analysis:
https://www.cmm.in.tum.de/public/paper/drop_analysis/kremer/kremer/kremer_index.html
FRASER in Bioconductor:
http://bioconductor.org/packages/release/bioc/html/FRASER.htmlGeuvadis dataset:
https://www.internationalgenome.org/data-portal/data-collection/geuvadis
FusionEditor: https://proteinpaint.stjude.org/FusionEditor/
Human Phenotype Ontology: https://hpo.jax.org/
HTSeq documentation: https://htseq.readthedocs.io/en/release_0.11.1/count.html
OUTRIDER in Bioconductor:
https://bioconductor.org/packages/release/bioc/html/OUTRIDER.html
phASER: https://github.com/secastel/phaser
Resource of count matrices: https://github.com/gagneurlab/drop#datasets
Snakemake documentation: https://snakemake.readthedocs.io/en/stable/
Variant effect predictor tutorial:
https://uswest.ensembl.org/info/docs/tools/vep/script/vep_tutorial.html
wBuild documentation: https://wbuild.readthedocs.io/en/latest/
YAML documentation:
https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html

Data availability

A subset of the Geuvadis dataset[15] comprising 100 samples (Supplementary Data 1), was used to test and demonstrate the workflow.. We refer to this dataset as the test dataset. It is accessible without restriction under https://www.internationalgenome.org/data-portal/data-collection/geuvadis. The analyses performed in the 'Dataset design' section used the GTEx and Kremer et al.[9] datasets. The GTEx dataset was downloaded from the GTEx Portal on June 12, 2017, under accession number dbGaP: phs00424.v6.p1. The count matrices from the Kremer et al. dataset were downloaded from Zenodo (doi: 10.5281/zenodo.3887451).

Code availability

DROP, including a small demo dataset of 10 samples and chromosome 21 only, is publicly available at https://github.com/gagneurlab/drop under MIT license. The current version is 0.9.2 which is fixed with doi: 10.5281/zenodo.4106177. All the plots, results, and analyses of the test dataset can be found here: https://www.cmm.in.tum.de/public/paper/drop_analysis/webDir/html/drop_analysis_index.html.

References

1. Bamshad, M. J. *et al.* Exome sequencing as a tool for Mendelian disease gene discovery. *Nat. Rev. Genet.* **12**, 745–755 (2011).
2. Yang, Y. *et al.* Clinical Whole-Exome Sequencing for the Diagnosis of Mendelian Disorders. *N. Engl. J. Med.* **369**, 1502–1511 (2013).
3. Taylor, J. C. *et al.* Factors influencing success of clinical genome sequencing across a broad spectrum of disorders. *Nat. Genet.* **47**, 717–726 (2015).
4. Lionel, A. C. *et al.* Improved diagnostic yield compared with targeted gene sequencing panels suggests a role for whole-genome sequencing as a first-tier genetic test. *Genet. Med.* **20**, 435–443 (2018).
5. Chong, J. X. *et al.* The Genetic Basis of Mendelian Phenotypes: Discoveries, Challenges, and Opportunities. *Am. J. Hum. Genet.* **97**, 199–215 (2015).
6. Cooper, G. M. Parlez-vous VUS? *Genome Res.* 1423–1426 (2015) doi:10.1101/gr.190116.115.
7. Kremer, L. S. *et al.* Genetic diagnosis of Mendelian disorders via RNA sequencing. *Nat.*

Commun. **8**, 15824 (2017).

8. Cummings, B. B. *et al.* Improving genetic diagnosis in Mendelian disease with transcriptome sequencing. *Sci. Transl. Med.* 12 (2017).
9. Frésard, L. *et al.* Identification of rare-disease genes using blood transcriptome sequencing and large control cohorts. *Nat. Med.* **25**, 911–919 (2019).
10. Gonorazky, H. D. *et al.* Expanding the Boundaries of RNA Sequencing as a Diagnostic Tool for Rare Mendelian Disease. *Am. J. Hum. Genet.* **104**, 466–483 (2019).
11. Li, H. *et al.* The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**, 2078–2079 (2009).
12. Danecek, P. *et al.* The variant call format and VCFtools. *Bioinformatics* **27**, 2156–2158 (2011).
13. Murdock, D. R. *et al.* Transcriptome-directed analysis for Mendelian disease diagnosis overcomes limitations of conventional genomic testing. *J. Clin. Invest.* (2020).
14. Koster, J. & Rahmann, S. Snakemake--a scalable bioinformatics workflow engine. *Bioinformatics* **28**, 2520–2522 (2012).
15. Lappalainen, T. *et al.* Transcriptome and genome sequencing uncovers functional variation in humans. *Nature* **501**, 506–511 (2013).
16. Love, M. I., Huber, W. & Anders, S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol.* **15**, 550 (2014).
17. Li, Y. I. *et al.* Annotation-free quantification of RNA splicing using LeafCutter. *Nat. Genet.* **50**, 151–158 (2018).
18. Brechtmann, F. *et al.* OUTRIDER: A Statistical Method for Detecting Aberrantly Expressed Genes in RNA Sequencing Data. *Am. J. Hum. Genet.* **103**, 907–917 (2018).
19. Mertes, C. *et al. Detection of aberrant splicing events in RNA-Seq data with FRASER*. https://www.biorxiv.org/content/10.1101/2019.12.18.866830v1 (2019) doi:10.1101/2019.12.18.866830.

20. Köhler, S. *et al.* Expansion of the Human Phenotype Ontology (HPO) knowledge base and resources. *Nucleic Acids Res.* **47**, D1018–D1027 (2019).

21. GTEx Consortium. Genetic effects on gene expression across human tissues. *Nature* **550**, 204–213 (2017).

22. Papatheodorou, I. *et al.* Expression Atlas: gene and protein expression across multiple studies and organisms. *Nucleic Acids Res.* **46**, D246–D251 (2018).

23. Aicher, J. K., Jewell, P., Vaquero-Garcia, J., Barash, Y. & Bhoj, E. J. Mapping RNA splicing variations in clinically accessible and nonaccessible tissues to facilitate Mendelian disease diagnosis using RNA-seq. *Genet. Med.* (2020) doi:10.1038/s41436-020-0780-y.

24. Stegle, O., Parts, L., Piipari, M., Winn, J. & Durbin, R. Using probabilistic estimation of expression residuals (PEER) to obtain increased power and interpretability of gene expression analyses. *Nat. Protoc.* **7**, 500–507 (2012).

25. Scotti, M. M. & Swanson, M. S. RNA mis-splicing in disease. *Nat. Rev. Genet.* **17**, 19–32 (2016).

26. Singh, R. K. & Cooper, T. A. Pre-mRNA splicing in disease and therapeutics. *Trends Mol. Med.* **18**, 472–482 (2012).

27. Cheng, J. *et al.* MMSplice: modular modeling improves the predictions of genetic variant effects on splicing. *Genome Biol.* **20**, 48 (2019).

28. Jaganathan, K. *et al.* Predicting Splicing from Primary Sequence with Deep Learning. *Cell* **176**, 535-548.e24 (2019).

29. Lee, H. *et al.* Diagnostic utility of transcriptome sequencing for rare Mendelian diseases. *Genet. Med.* (2019) doi:10.1038/s41436-019-0672-1.

30. Gonorazky, H. *et al.* RNAseq analysis for the diagnosis of muscular dystrophy. *Ann. Clin. Transl. Neurol.* **3**, 55–60 (2016).

31. Kernohan, K. D. *et al.* Whole-transcriptome sequencing in blood provides a diagnosis of spinal muscular atrophy with progressive myoclonic epilepsy. *Hum. Mutat.* **38**, 611–614

(2017).

32. Hamanaka, K. *et al.* RNA sequencing solved the most common but unrecognized NEB pathogenic variant in Japanese nemaline myopathy. *Genet. Med.* **21**, 1629–1638 (2019).
33. Wang, K. *et al.* Whole-genome DNA/RNA sequencing identifies truncating mutations in RBCK1 in a novel Mendelian disease with neuromuscular and cardiac involvement. *Genome Med.* **5**, 67 (2013).
34. Pervouchine, D. D., Knowles, D. G. & Guigo, R. Intron-centric estimation of alternative splicing from RNA-seq data. *Bioinformatics* **29**, 273–274 (2013).
35. Kapustin, Y. *et al.* Cryptic splice sites and split genes. *Nucleic Acids Res.* **39**, 5837–5844 (2011).
36. Mohammadi, P. *et al.* Genetic regulatory variation in populations informs transcriptome analysis in rare disease. *Science* **366**, 351–356 (2019).
37. Albers, C. A. *et al.* Compound inheritance of a low-frequency regulatory SNP and a rare null mutation in exon-junction complex subunit RBM8A causes TAR syndrome. *Nat. Genet.* **44**, 435–439 (2012).
38. van Haelst, M. M. *et al.* Further confirmation of the MED13L haploinsufficiency syndrome. *Eur. J. Hum. Genet.* **23**, 135–138 (2015).
39. Lindstrand, A. *et al.* Different mutations in *PDE4D* associated with developmental disorders with mirror phenotypes. *J. Med. Genet.* **51**, 45–54 (2014).
40. ’t Hoen, P. A. C. *et al.* Reproducibility of high-throughput mRNA and small RNA sequencing across laboratories. *Nat. Biotechnol.* **31**, 1015–1022 (2013).
41. Lee, S. *et al.* NGSCheckMate: software for validating sample identity in next-generation sequencing studies within and across data types. *Nucleic Acids Res.* **45**, e103–e103 (2017).
42. Castel, S. E., Mohammadi, P., Chung, W. K., Shen, Y. & Lappalainen, T. Rare variant phasing and haplotypic expression from RNA sequencing with phASER. *Nat. Commun.* **7**, 12817 (2016).

43. Mitelman, F., Johansson, B. & Mertens, F. The impact of translocations and gene fusions on cancer causation. *Nat. Rev. Cancer* **7**, 233–245 (2007).

44. Dai, X., Theobard, R., Cheng, H., Xing, M. & Zhang, J. Fusion genes: A promising tool combating against cancer. *Biochim. Biophys. Acta BBA - Rev. Cancer* **1869**, 149–160 (2018).

45. van Heesch, S. *et al.* Genomic and Functional Overlap between Somatic and Germline Chromosomal Rearrangements. *Cell Rep.* **9**, 2001–2010 (2014).

46. Oliver, G. R. *et al.* A tailored approach to fusion transcript identification increases diagnosis of rare inherited disease. *PLOS ONE* **14**, e0223337 (2019).

47. Tian, L. *et al.* CICERO: a versatile method for detecting complex and diverse driver fusions using cancer RNA sequencing data. *Genome Biol.* **21**, 126 (2020).

48. Dobin, A. *et al.* STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29**, 15–21 (2013).

49. Ewels, P., Magnusson, M., Lundin, S. & Käller, M. MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics* **32**, 3047–3048 (2016).

50. Li, H. & Durbin, R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* **25**, 1754–1760 (2009).

51. McKenna, A. *et al.* The Genome Analysis Toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.* **20**, 1297–1303 (2010).

52. Van der Auwera, G. A. *et al.* From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline. in *Current Protocols in Bioinformatics* 11.10.1-11.10.33 (John Wiley & Sons, Inc., 2013). doi:10.1002/0471250953.bi1110s43.

53. Li, H. Tabix: fast retrieval of sequence features from generic TAB-delimited files. *Bioinformatics* **27**, 718–719 (2011).

54. McLaren, W. *et al.* The Ensembl Variant Effect Predictor. *Genome Biol.* **17**, 122 (2016).

55. Haeussler, M. *et al.* The UCSC Genome Browser database: 2019 update. *Nucleic Acids Res.* **47**, D853–D858 (2019).

56. Frankish, A. *et al.* GENCODE reference annotation for the human and mouse genomes. *Nucleic Acids Res.* **47**, D766–D773 (2019).

57. Robinson, J. T. *et al.* Integrative genomics viewer. *Nat. Biotechnol.* **29**, 3 (2011).

58. Ben-Kiki, O. & Evans, C. YAML Ain't Markup Language (YAML^{TM}) Version 1.2. 80.

59. Anders, S., Pyl, P. T. & Huber, W. HTSeq--a Python framework to work with high-throughput sequencing data. *Bioinformatics* **31**, 166–169 (2015).

60. McCarthy, D. J., Chen, Y. & Smyth, G. K. Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. *Nucleic Acids Res.* **40**, 4288–4297 (2012).

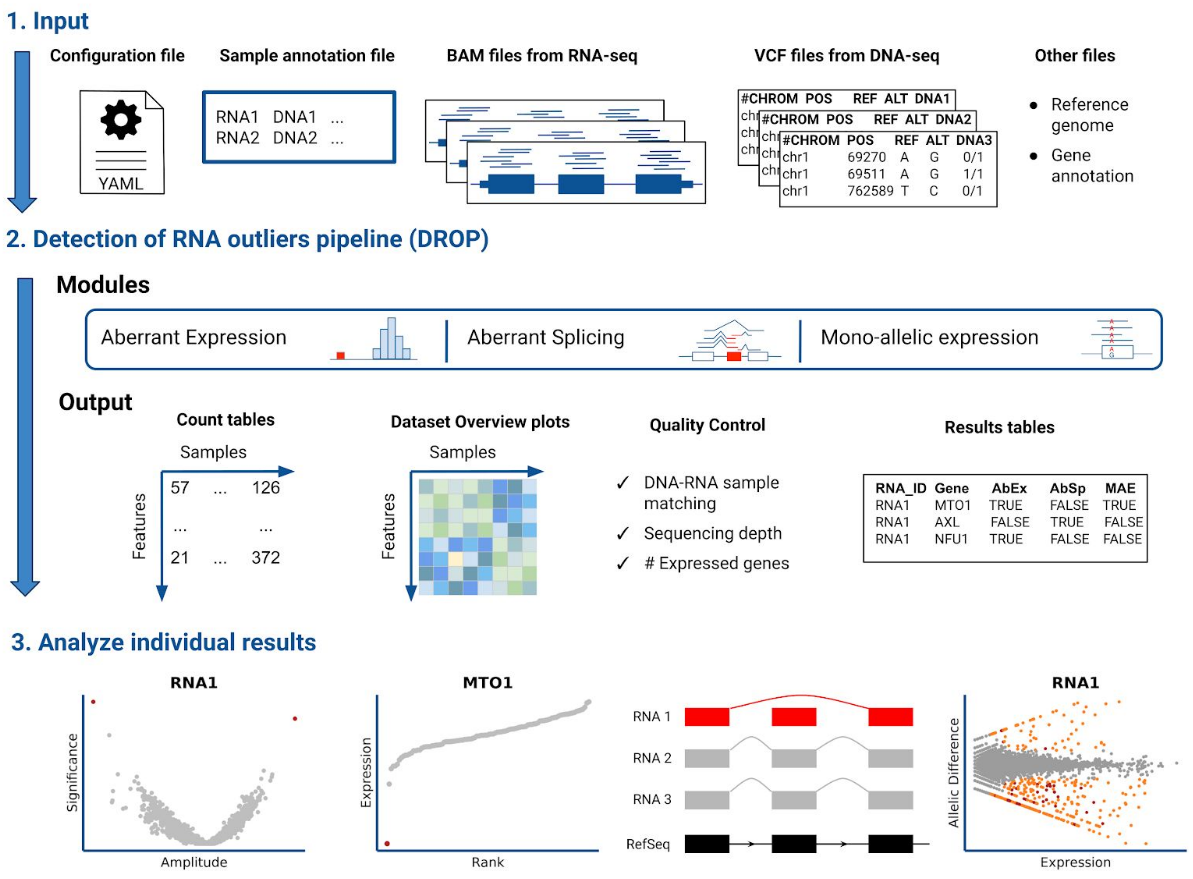
61. Karczewski, K. J. *et al.* The mutational constraint spectrum quantified from variation in 141,456 humans. *Nature* **581**, 434–443 (2020).

62. Katz, Y. *et al.* Quantitative visualization of alternative exon expression from RNA-seq data. *Bioinformatics* **31**, 2400–2402 (2015).

63. Amberger, J. S., Bocchini, C. A., Scott, A. F. & Hamosh, A. OMIM.org: leveraging knowledge across phenotype–gene relationships. *Nucleic Acids Res.* **47**, D1038–D1043 (2019).

64. Richards, S. *et al.* Standards and guidelines for the interpretation of sequence variants: a joint consensus recommendation of the American College of Medical Genetics and Genomics and the Association for Molecular Pathology. *Genet. Med.* **17**, 405–423 (2015).

Figure legends



RNA1	DNA1	...
RNA2	DNA2	...

#CHROM	POS	REF	ALT	DNA3
chr1	69270	A	G	0/1
chr1	69511	A	G	1/1
chr1	762589	T	C	0/1

57	...	126
...		...
21	...	372

RNA_ID	Gene	AbEx	AbSp	MAE
RNA1	MTO1	TRUE	FALSE	TRUE
RNA1	AXL	FALSE	TRUE	FALSE
RNA1	NFU1	TRUE	FALSE	FALSE

Fig. 1 | Workflow overview. As input, DROP requires a configuration file, a sample annotation file, BAM files from RNA-seq, and VCF files. DROP processes for each module the input data and generates count tables, overview plots (e.g. sample covariation heatmap), quality control plots, and result tables. Lastly, users can perform case-by-case analyses with the help of different visualizations.

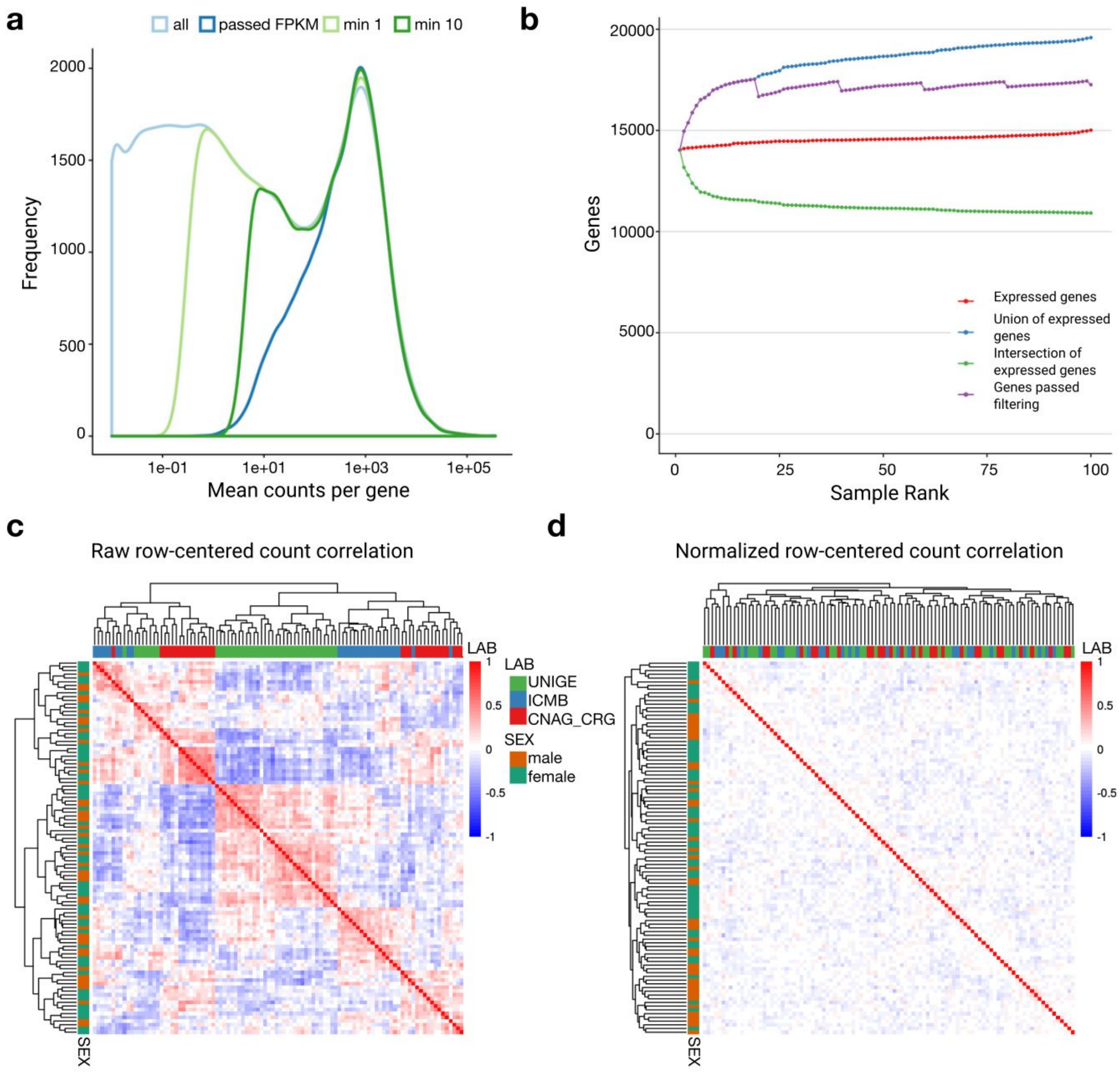


Fig. 2 | The Aberrant Expression Module. a, Histogram of raw read count distribution colored by different filtering strategies: all, minimum 1 count in 5% of the samples, minimum 10 counts in 5% of the samples, and minimum 1 (or user-defined) FPKM in 5% of the samples. **b,** Number of expressed genes cumulative across all samples. Colors represent the union of all detected genes (blue), genes that passed the FPKM filter as a group (violet), genes that are expressed on each sample (red), and the intersection of expressed genes (green). **c,** Heatmap of the correlation of row-centered log-transformed read counts between samples before and after (**d**) the autoencoder correction including the samples' sexes and the laboratories where they were sequenced. Before correction, samples cluster by laboratory.

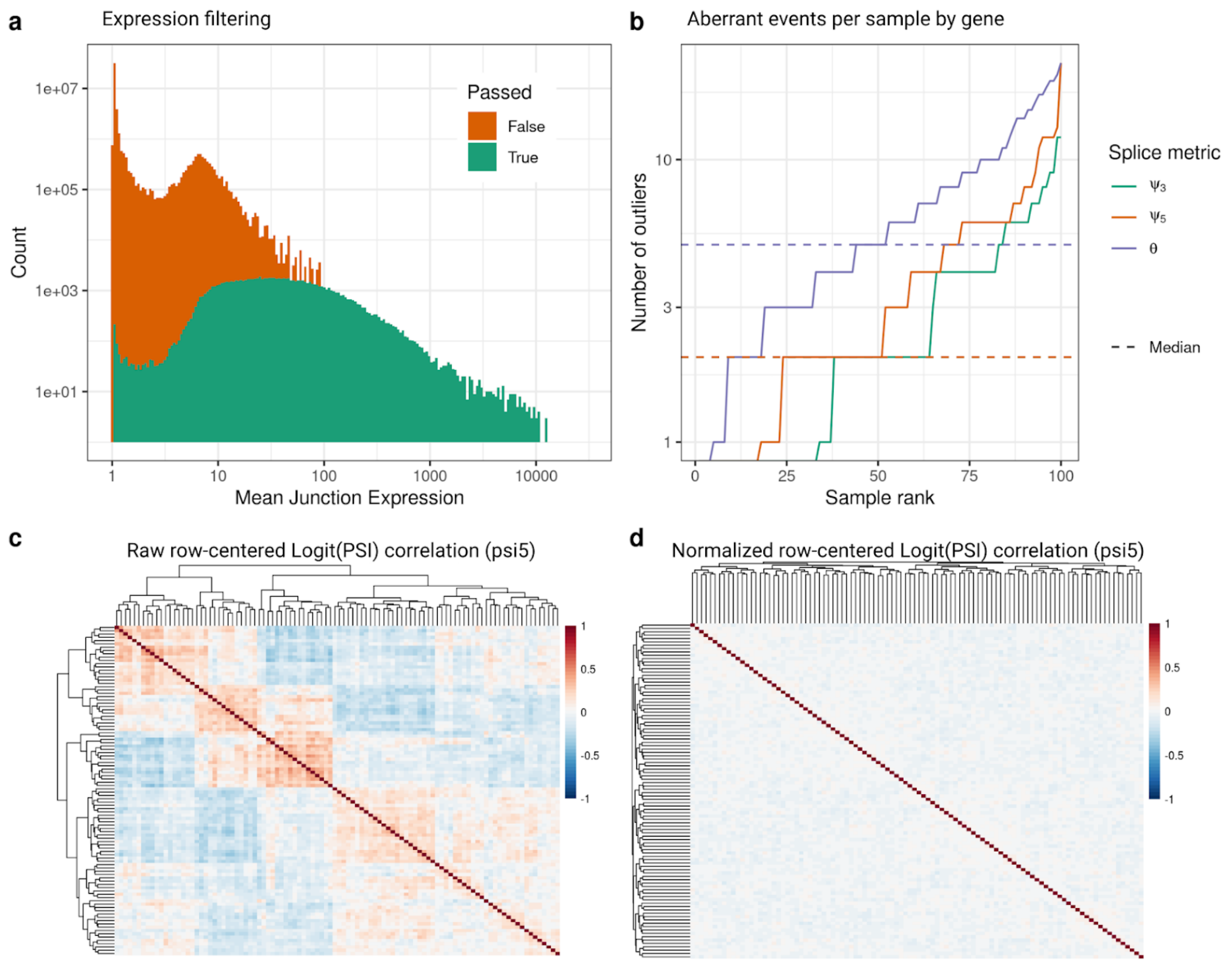
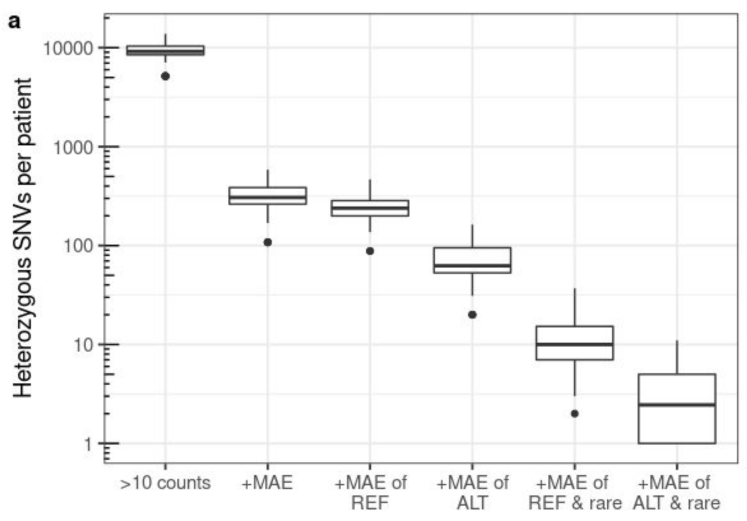


Fig. 3 | The aberrant splicing module. a, Histogram showing the junctions that passed the filter. **b,** Number of aberrant splicing events per sample by gene colored by metric. **c,** Heatmap of the correlation of row-centered logit-transformed read count ratios between samples before and after (**d**) the correction.



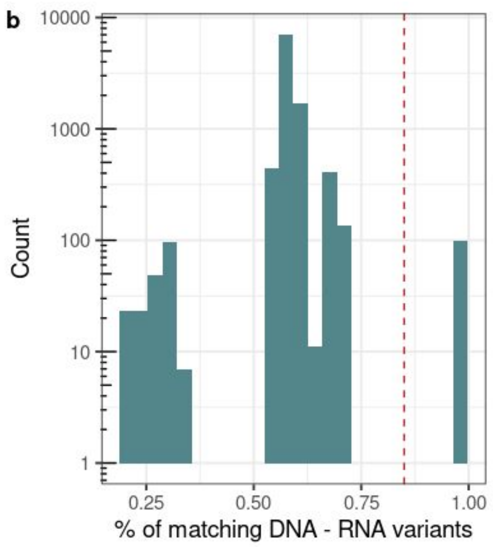


Fig. 4 | Mono-allelic Expression Module. a, Cascade plot showing the number of heterozygous SNVs with at least 10 reads (median=9,171), that are mono-allelically expressed (median=306.5), toward the reference allele (median=239), toward the alternative allele (median=62.5), rare toward the reference (median=10), and rare toward the alternative (median=2.5). The lower and upper limits of the boxes correspond to the first and third quartiles, respectively. Whiskers extend to the most extreme value, no further than 1.5x the interquartile range. **b,** Histogram (y-axis in log scale) of the proportion of matching genotypes from DNA and RNA when comparing all sample combinations. The median of matching samples is 0.98 compared to 0.58 of non-matching samples. One sample had a very low rate of matching with the rest of samples (median=0.26). The red dashed line represents the cutoff provided in the config file to distinguish between the samples that match and the ones that do not (default: 0.85).

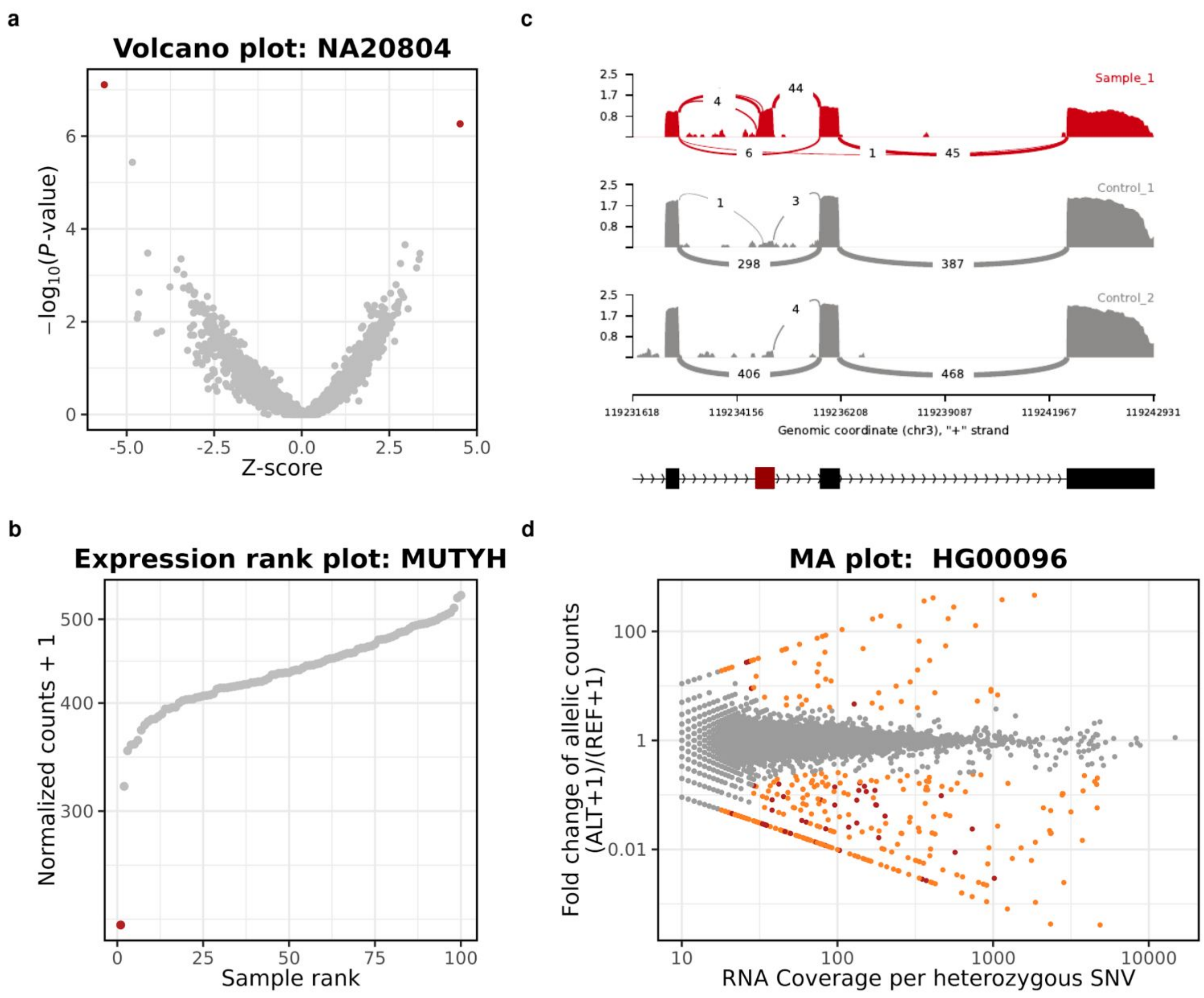


Fig. 5 | Downstream analysis of outlier results. a, Negative log-transformed nominal *P* values (y-axis) versus z scores (x-axis) derived from the expression of all genes of sample NA20804 showing one over- and one under-expression outlier (red). **b,** Normalized counts (y-axis) of gene *MUTYH* of all samples (x-axis) reflecting one under-expression outlier (red). **c,** Representative sashimi plot showing the creation of a new exon for Sample 1 which is skipped by the other two samples. The height represents the coverage in $\log_{10}$ RPKM and the number of the arcs the junction's coverage. On the bottom, the corresponding gene model is depicted (the cryptic exon is in red). **d,** Fold change between the counts of the alternative and reference alleles (y-axis) compared against the total coverage of each heterozygous SNV (with at least 10 reads) of sample HG00096, highlighted by significance (FDR < 0.05, orange), and significance and rarity (MAF < 0.001, red).

Supplementary Data:

Supplementary Data 1: Sample annotation

Supplementary Data 2: Config file

Supplementary Figures:

- Supplementary Figure 1: Raw and OUTRIDER-normalized count correlation heatmaps of different datasets
- Supplementary Figure 2: Analysis of a combination of patient samples with controls
- Supplementary Figure 3: Analysis of a combination of different GTEx tissues
- Supplementary Figure 4: Analysis of a combination of different sequencing depths
- Supplementary Figure 5: Aberrant expression module
- Supplementary Figure 6: Aberrant splicing module
- Supplementary Figure 7: Mono-allelic expression workflow
- Supplementary Figure 8: Sample-centric analysis plots

Supplementary Methods