Group Project: Identification of NMD sensitive RNAs and isoforms with long-read direct RNA sequencing(DRS)

Gonzalo Cardenal Antolin (GonzaloCardenalAl), Michael Cibien, Tobia Ochsner (ochsneto)

2024-01-12

Introduction

Biological Background – NMD

Nonsense-mediated mRNA decay (NMD) is a eukaryotic pathway that is responsible for degradation of not only aberrant, but also some endogeneous mRNA, that would result in physiologically active proteins. This pathway therefore plays a pivotal role in post-transcriptional gene regulation by eliminating mRNA transcripts that could otherwise lead to the synthesis of truncated or malfunctioning proteins. The underlying mechanisms by which NMD selectively targets mRNAs for degradation are not fully elucidated, however, key proteins implicated in this process have been identified, including UPF1, an RNA helicase; SMG1, a phosphatidylinositol-kinase-related kinase; SMG6, an endonuclease; and SMG5 and SMG7, which function as adaptor proteins. (Karousis et al. 2021) (Karousis and Mühlemann 2022)

Historically, it was posited that the primary role of NMD was to target mRNAs containing premature stop codons. Recent research, however, indicates that the scope of NMD is broader, encompassing additional features of mRNAs. To explore these additional features, particularly in endogenous mRNAs, researchers have employed strategies such as the knockdown of key NMD proteins. This approach increases the intracellular concentration of NMD-targeted mRNA isoforms, facilitating their study through subsequent RNA sequencing (RNA-seq) experiments.

Gene and isoform expression analysis from direct RNA sequencing (DRS)

In conventional differential gene expression analysis short-read sequencing is widely used. However, short-read RNA sequences have limitations in identifying and quantifying gene transcript isoforms due to RNA fragmentation, bias introduction in the reverse transcription into cDNA and the subsequent amplification by PCR. DRS could act as a potential remedy to this problem, since it sequences full-length transcripts, while also characterizing RNA modifications, polyA tails and not needing prior PCR amplification.(Aird et al. 2011) (Steijger et al. 2013)

It has been shown by Josie Gleeson et al (Gleeson et al. 2022) that DRS has the potential to quantify both genes and transcript isoforms in an unbiased manner allowing for a multitude of analyses: gene and isoform quantification; differential gene expression analysis; differential isoform usage analysis (DUI); discovery of novel isoforms. However, the key challenges of long-read RNA sequencing – namely sequencing depth – remains and impacts the sensitivity of isoform identification with a high probability.(Gleeson et al. 2022)

Goal of this Study

We were lucky to receive DRS data from a follow up project on their paper by Dr. Evan Karousis. They assessed endogenous targets of NMD using a combination of long-read and short-read sequencing and found that the juxtaposition of short-read sequencing with long-read sequencing enabled the identification of novel NMD-targeted mRNA isoforms. (Karousis et al. 2021)

By comparing the results of their previous work to the results of this study, where we use DRS, we aim to answer the question of whether DRS alone possesses sufficient resolution to delineate the full spectrum of NMD-targeted transcripts.

Data Background

The data was kindly provided by Dr. Evangelos Karousis, who generated the data while working as post-doctoral researcher in the laboratory of Prof. Dr. Oliver Mühlemann, at the University of Bern in 2019.

Knockdown experiments on HeLa cells were performed by RNA interference (RNAi). For this, plasmids expressing shRNAs against SMG6 and SMG7 were transfected into the "dKD" cells, whereas in the "Scr" (control) cells plasmids targeting nothing were transfected. polyA+ mRNA was then isolated, and direct RNA-seq was performed using the Flowcell SQK-RNA002 from Oxford Nanopore. The MinKNOW instrument software generated fast5 files, on which basecalling was performed using GUPPY (VERSION?) (from Oxford Nanopore Technologies).

The data was divided into 6 folders, NP05_Scr1, NP06_dKD1, NP07_dKD2, NP07B_dKD2, NP08_Scr2, NP08B_Scr2, where "Scr" refers to the controls and "dKD" to the double-knockdown conditions. The two conditions were performed in a biological duplicate, however the sequencing run of dKD2 and Scr2 samples were interrupted, so thats why there are "B" samples. These folders were merged after quality control (before alignment and subsequent analyses) and therefore conditions are referred to as Scr1, Scr2, dKD1 and dKD2.

Preprocessing

Quality Control (QC)

To validate the quality control and the filtering heuristics that were performed by E. Karousis et al with GUPPY, two scripts were written and run. We confirmed that the filtering criterion was a q-score lower than 7. (The scripts available on the github repo under the names "filter.r" and "checkGuppyFilter.r"). We then used NanoPlot, which generated a report of different sequencing quality statistics like mean and median read quality, read lengths etc. These reports are also available in the group project GitHub repository. The results provided by NanoPlot confirmed that our samples were of mean read quality greater than 7 and displayed coherent read length distribution, sequencing error rates, and number of reads.

The sequencing of DRS is more difficult than sequencing cDNA, since mRNA contains dynamic base modifications. The most prevalent base modification is N6-methyladenosine (m6A) (Roundtree et al. 2017). For such modified bases the current produced when passing through the ONT MiniOn pore are atypical and thus are read as different nucleotides. Therefore higher error rates are expected and if we were to investigate mRNA modifications a lower quality score threshold should be contemplated.

Mapping and quantification

To map long reads, the current gold standard is using minimap2. We ran two mappings, one to the genome and another one to the transcriptome. In the genomic mapping, we use [Homo_sapiens.GRCh38.dna.primary_assembly.fa](https://ftp.ensembl.org/pub/release-110/fasta/homo_sapiens/dna/) as the reference genome for each sample. In the transcriptome mapping, we use the cDNA reference from the Ensmbl database.

To quantify the genes we Feature Counts was used, which provides a mode "-L" tailored for long reads. The annotation file was obtained from the Ensembl database. With this mode the raw counts were obtained, which were needed by Deseq2 for the differential gene expression analysis. For the transcript and the different isoform quantification NanoCount was utilized. NanoCount is a recently released tool specifically developed to quantify Oxford Nanopore direct RNA sequencing (DRS) reads. (Gleeson et al. 2022)

The most important quality control metrics were summarized in Figure. Reads had a median quality score of 8.1 - 8.2 across all samples and mean redian lengths were also comparable. The only metric that differed significantly

between the samples was the number of reads, however this is partly due to the aforementioned interruptions of the DRS runs.

	Scr1	Scr2		dKD1	dKD2	
	NP05_Scr1	**NP08_Scr2**	**NP08B_Scr2**	**NP06_dKD1**	**NP07_dKD2**	**NP07B_dKD2**
Number of reads	1'549'683	1'776'660	481'888	1'592'867	915'205	1'340'688
Median read length (nt)	946	950	923	1003	979	912
Median read quality score	8.1	8.2	8.1	8.1	8.2	8.1
Longest alignment length (nt)	288'042	121'433	46'021	166'309	59'725	41'680
Reads aligned to genome (primary) (%)	41.5	41.7		43.5	42.9	
Reads aligned to genome (%)	76.2	76.3		72.3	72.7	

Figure 1: Direct RNA sequencing quality scores and genome alignment metrics.

Statistical Analysis

Transcript Coverage

To assess the coverage of the reads, coverage fractions for the samples were computed using BamSlam scripts. Coverage fraction is defined as the ratio between the transcript length and the theoretical full length of their respective isoforms.

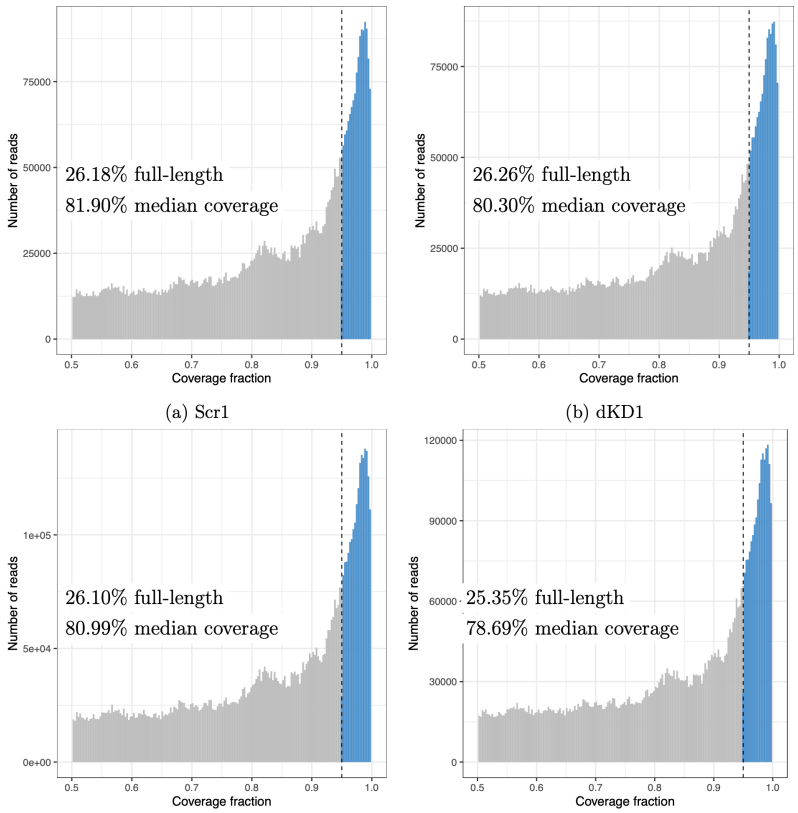
```
Rscript BamSlam.R rna NP05-sorted-transcriptomic-aln.bam NP05_BamSlam_output
Rscript BamSlam.R rna NP06-sorted-transcriptomic-aln.bam NP06_BamSlam_output
Rscript BamSlam.R rna NP07-sorted-transcriptomic-aln.bam NP07_BamSlam_output
Rscript BamSlam.R rna NP08-sorted-transcriptomic-aln.bam NP08_BamSlam_output

NP05_BamSlam_output_stats
```

The median coverage fraction is of special interest to us, as we would expect a 5-10% increase in full-length transcripts upon SMG6 knockdown, since its nuclease activity is suppressed. In our case, the median coverage fraction for Scr1 and Scr2 are 81.90% and 80.99% (average = 81.45%), while in the knockdown samples they are 80.30% and 78.69% for dKD1 and dKD2 respectively (average = 79.50%). This demonstrates that in both conditions most reads covered most of the original RNA transcript lengths. However, we cannot observe a significant change of the median transcript coverage between the two conditions. Additionally, the number of full-length reads does not change significantly either between the two conditions, as can be seen in the Figures.

When assessing the relationship between coverage fraction and known transcript length, it becomes apparent that longer transcripts were less likely to be aligned at full length. This can be observed in all conditions and there does not seem to be a significant difference between them (visually).

Import Data

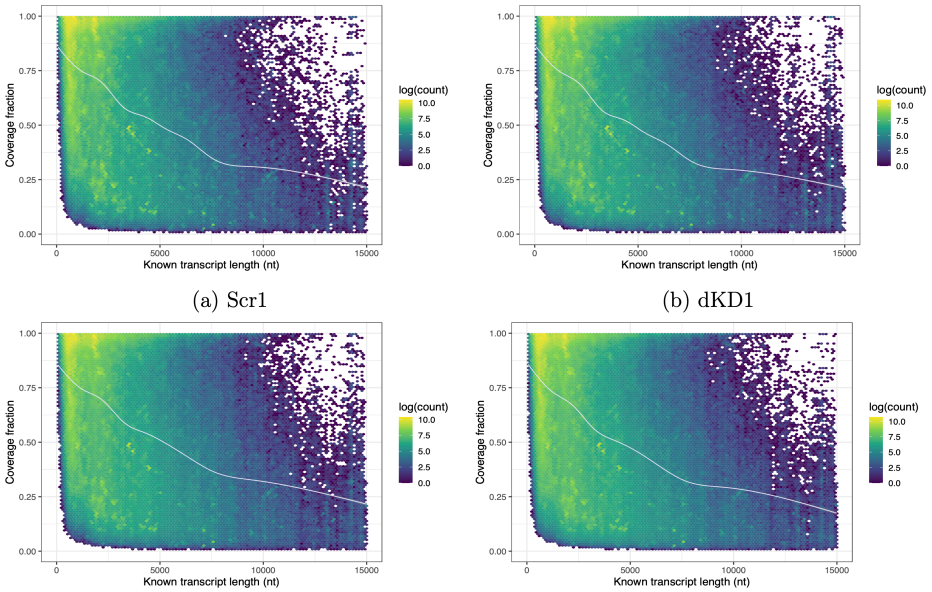
```
samples <- list.files("counts_files")

raw_genomic <- sapply(samples, function(sample) {
  file <- paste0("counts_files/", sample, "/","primary-genomic-quant")
  quant <- read.table(file, header=TRUE, row.names=1, sep="\t")
  gene_id <- rownames(quant)
  raw <- quant[,6]
  raw_counts <- setNames(raw, gene_id)
  return(raw_counts)
```



(a) Scr1 (b) dKD1

(c) Scr2 (d) dKD2

Figure 2: Distribution of transcript coverage fraction. Dotted line represents 95% cutoff for full-length reads. Full-length reads are shaded in blue.



(a) Scr1 (b) dKD1

(c) Scr2 (d) dKD2

Figure 3: Fraction of known transcript length covered by each read (coverage fraction) compared to known transcript length. Trend line was plotted using a generalized additive model, an extension of a generalised linear model where the linear form is replaced by sum of smooth functions.

```
})

raw_genomic <- as.data.frame(raw_genomic)

raw_genomic <- raw_genomic %>%
  rownames_to_column(var = "gene_name") %>%
  arrange(gene_name) %>%
  column_to_rownames(var = "gene_name")

np05_tcrm = read.table(file="counts_files/NP05-Scr1/transcriptomic-quant.tsv", sep = '\t', header = TRUE
```

We are working with direct RNA long-reads. The output from NanoCount are estimated counts (est_count) and transcripts per million (tpm). These metrics are not normalised by transcript length as it is usually done with Illumina data becasue in DRS one read is supposed to represent a single transcript molecule starting from the polyA tail, even if the fragment does not extend to the 5' end.

Est_count is obtained by multiplying the raw abundance by the number of primary alignments, the latter is the estimated counts obtained by multiplying the raw abundance by 10^6. It was decided to use tpm for the subsequent analyses.

```
np05_tcrm = data.frame(transcript_name = np05_tcrm$transcript_name, tpm_np05 = np05_tcrm$tpm)

np06_tcrm = read.table(file = 'counts_files/NP06-dKD1/transcriptomic-quant.tsv', sep = '\t', header = TR
np06_tcrm = data.frame(transcript_name = np06_tcrm$transcript_name, tpm_np06 = np06_tcrm$tpm)
```

```
np07_tcrm = read.table(file = 'counts_files/NP07-dKD2/transcriptomic-quant.tsv', sep = '\t', header = TR
np07_tcrm = data.frame(transcript_name = np07_tcrm$transcript_name, tpm_np07 = np07_tcrm$tpm)

np08_tcrm = read.table(file = 'counts_files/NP08-Scr2/transcriptomic-quant.tsv', sep = '\t', header = TR
np08_tcrm = data.frame(transcript_name = np08_tcrm$transcript_name, tpm_np08 = np08_tcrm$tpm)

tpm_transcriptome <- Reduce(function(x, y) merge(x, y, by = "transcript_name", all = TRUE), list(np05_tc


# Rename columns to specify the dataset source
tpm_transcriptome <- as.data.frame(tpm_transcriptome)
tpm_transcriptome  <- tpm_transcriptome  %>%
  arrange(transcript_name) %>%
  column_to_rownames(var = "transcript_name")
```

Exploratory Analysis

Once the gene and transcript counts were obtained, an initial exploratory analysis was performed.

```
colSums(raw_genomic)
```

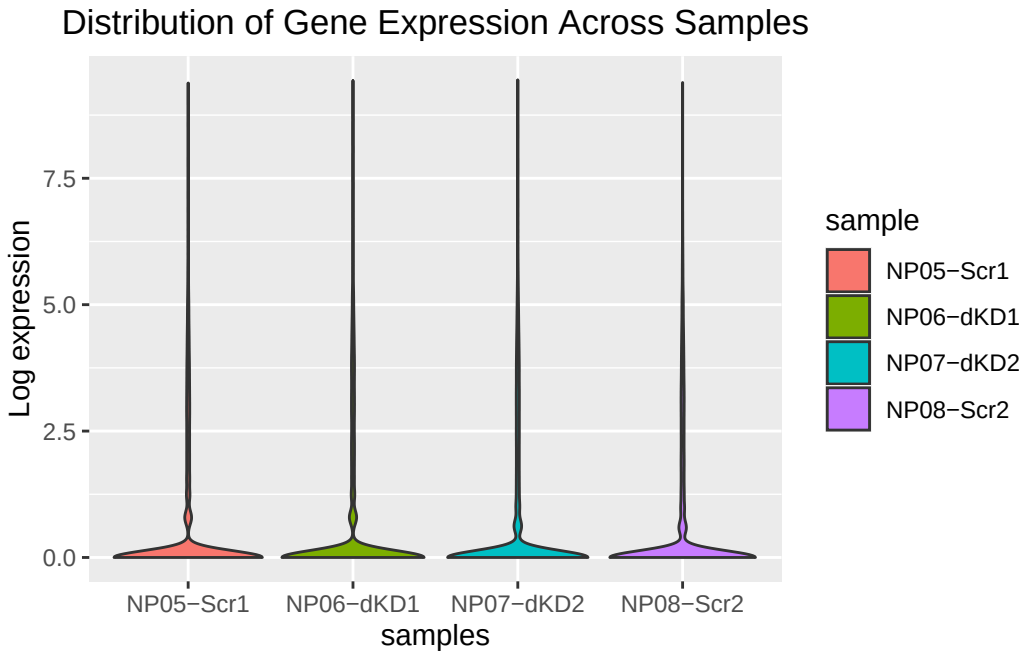
```
NP05-Scr1 NP06-dKD1 NP07-dKD2 NP08-Scr2
   882803    877859   1239173   1323920
```

```
colSums(tpm_transcriptome)
```

```
tpm_np05 tpm_np06 tpm_np07 tpm_np08
   1e+06    1e+06    1e+06    1e+06
```

The samples exhibit the same number of transcript counts and number of total counts, whereas the raw genomic counts differ from the number of total counts. Therefore, a normalization with a scaling factor proportional to the ratio of counts was performed to have the same number of total counts between samples.

```
median_all_counts = median(colSums(raw_genomic))
scaling_factors = median(colSums(raw_genomic))/colSums(raw_genomic)
normalised_genomic=sweep(raw_genomic, 2, scaling_factors, `*`)

log_normalised_genomic=log1p(normalised_genomic)

melted_data <- reshape2::melt(log_normalised_genomic)
colnames(melted_data) <- c("sample","Log_normalised_expression")

ggplot(melted_data, aes(x=sample,
                                  y=Log_normalised_expression, fill=sample)) +
  geom_violin() +
  labs(title='Distribution of Gene Expression Across Samples',
       x='samples',y='Log expression') +
  theme(plot.title = element_text(hjust = 0.5))
```



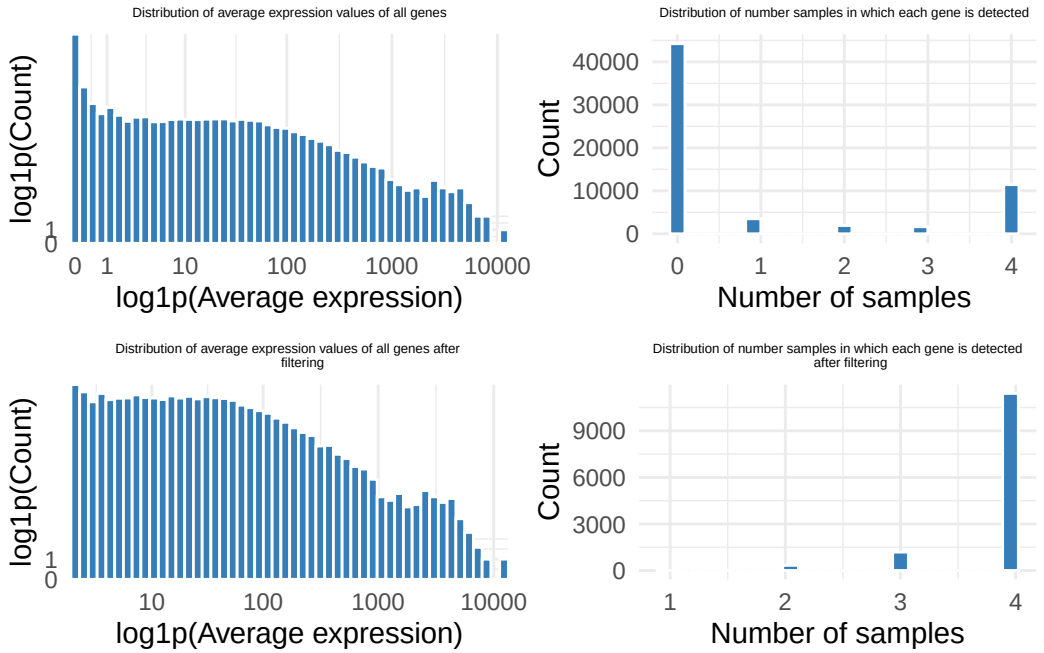
The visualization by the violin plots suggests that all samples have similar gene expression distributions.

```
avg_genomic <- rowMeans(raw_genomic)

par(mfrow = c(1, 2))
layout(matrix(1:2, nrow = 1))
par(mar = c(3, 3, 2, 1))
hist(avg_genomic)
hist(log1p(avg_genomic))

layout <- matrix(c(1, 2), nrow = 1)

plot <- ggplot(data = as.data.frame(avg_genomic), mapping = aes(x = avg_genomic)) +
  geom_histogram(
    color = "white",
    fill = brewer.pal(n = 3, name = "Set1")[2],
    bins = 50
  ) +
  labs(title = "Distribution of Average Expression Values of All Genes",
       x = "Average expression",
       y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))

log_plot <- ggplot(data = as.data.frame(avg_genomic), mapping =
                     aes(x = log(avg_genomic + 1))) +
  geom_histogram(
    color = "white",
    fill = brewer.pal(n = 3, name = "Set1")[2],
```

```
    bins = 50
  ) +
  labs(title = "Log Distribution of average expression values of all genes",
       x = "log10(Average expression + 1)",
       y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5))

# Arrange and display the plots
grid.arrange(plot, log_plot, layout_matrix = layout)
```

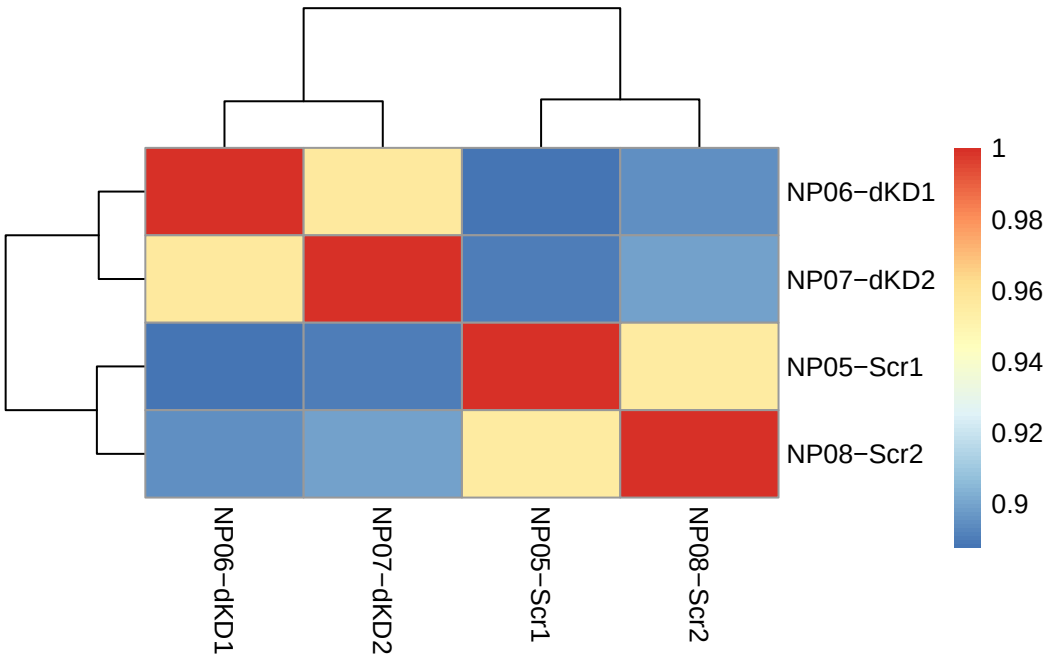
```
avg_genomic <- rowMeans(raw_genomic)
layout <- matrix(c(1, 2, 3, 4), nrow = 2, byrow = TRUE)

plot_1 <- ggplot(data = as.data.frame(avg_genomic), mapping =
                   aes(x = avg_genomic)) +
  geom_histogram(
    color = "white",
    fill = brewer.pal(n = 3, name = "Set1")[2],
    bins = 50
  ) +
  scale_x_continuous(
    breaks = c(0, 1, 10, 100, 1000, 10000, 20000),
    trans = "log1p",
    expand = c(0, 0)
  ) +
  scale_y_continuous(breaks = c(0, 1),
                     expand = c(0, 0),
                     trans = "log1p") +
  labs(title = "Distribution of average expression values of all genes",
       x = "log1p(Average expression)",
       y = "log1p(Count)") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 5))

num_detected_genes <- rowSums(raw_genomic > 0)

plot_2 <- ggplot(data = as.data.frame(num_detected_genes), mapping =
                   aes(x = num_detected_genes)) +
  geom_histogram(
    color = "white",
    fill = brewer.pal(n = 3, name = "Set1")[2],
    bins = 23
  ) +
  labs(title = "Distribution of number samples in which each gene is detected",
       x = "Number of samples",
       y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, size = 5))
```

```r
# filter expressed genes
# threshold: genes must be detected in at least half of the samples
#            or the average counts must be >= 1

expressed <- rowSums(raw_genomic) >= 5
num_filtered_expressed_genes <- rowSums(raw_genomic[expressed,] > 0)

avg_genomic <- data.frame(avg_genomic)
plot_3 <- ggplot(data = as.data.frame(avg_genomic[expressed,]), mapping =
                   aes(x = avg_genomic[expressed,])) +
  geom_histogram(
    color = "white",
    fill = brewer.pal(n = 3, name = "Set1")[2],
    bins = 50
  ) +
  scale_x_continuous(
    breaks = c(0, 1, 10, 100, 1000, 10000, 20000),
    trans = "log1p",
    expand = c(0, 0)
  ) +
  scale_y_continuous(breaks = c(0, 1),
                     expand = c(0, 0),
                     trans = "log1p") +
  labs(title = "Distribution of average expression values of all genes after
       filtering",
       x = "log1p(Average expression)",
       y = "log1p(Count)") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,  size = 5))

plot_4 <- ggplot(data = as.data.frame(num_filtered_expressed_genes), mapping =
                   aes(x = num_filtered_expressed_genes)) +
  geom_histogram(
    color = "white",
    fill = brewer.pal(n = 3, name = "Set1")[2],
    bins = 23
  ) +
  labs(title = "Distribution of number samples in which each gene is detected
       after filtering",
       x = "Number of samples",
       y = "Count") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5,  size = 5))

grid.arrange(plot_1, plot_2, plot_3, plot_4, layout_matrix = layout)
```



The genes were then filtered by removing those with fewer than 5 counts. By looking at the plots on the right side of the figure, we can observe that our filtering criteria filtered not only all the genes that were not expressed in any of the samples, but also most of the genes only expressed in one individual sample.

```
#Plot heatmap
corr_pearson <- cor(log1p(raw_genomic[expressed,]), method = "spearman")
pheatmap(corr_pearson)
```



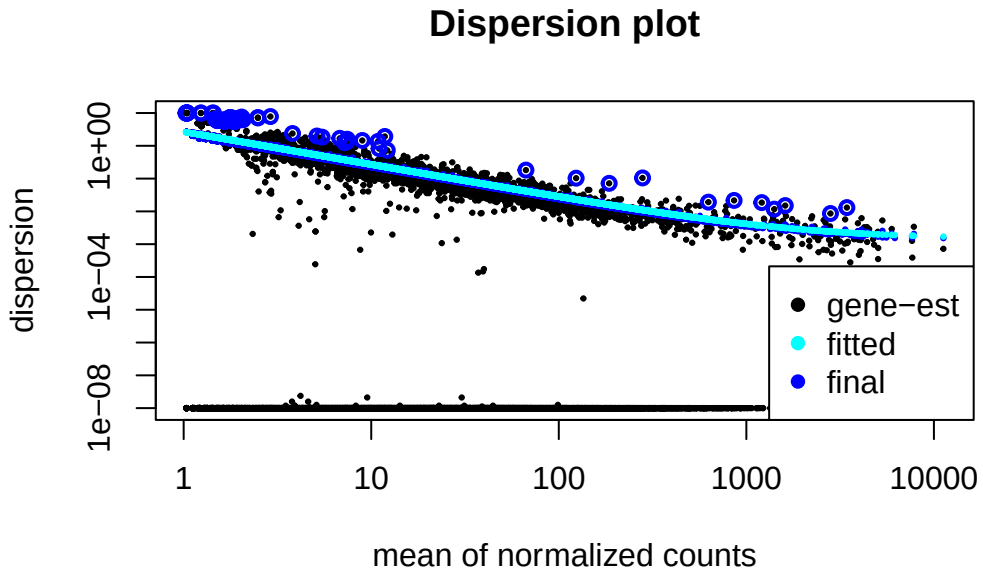
The heatmap clearly shows a correlation between the two control samples, as well as the knockdown samples, which indicates that there is a differential expression between the knockdown and the control. This again indicates that the NMD knockdown is causing a differential gene expression.

Differential Gene Expression Analysis

```
#DESeq input
raw_genomic<- as.matrix(raw_genomic[expressed,])
condition <- factor(c("control","knockdown","knockdown","control")) #NP05-Scr1, NP06-dKD1, NP07-dKD2, NP
coldata <- data.frame(row.names=colnames(raw_genomic), condition)

# Make DESeq dataset
dds <- DESeqDataSetFromMatrix(countData=raw_genomic, colData=coldata,
                              design=~condition)

# Run DESeq2 pipeline
dds <- DESeq(dds)
res <- DESeq2::results(dds)

#DESeq2 results
res <- res[order(res$padj), ]

# Merge with normalized count data
resdata <- merge(as.data.frame(res), as.data.frame(counts(dds, normalized=TRUE)),
                 by="row.names", sort=FALSE)
names(resdata)[1] <- "Gene"
resdata$DE <-resdata$padj<0.05
head(resdata)
```

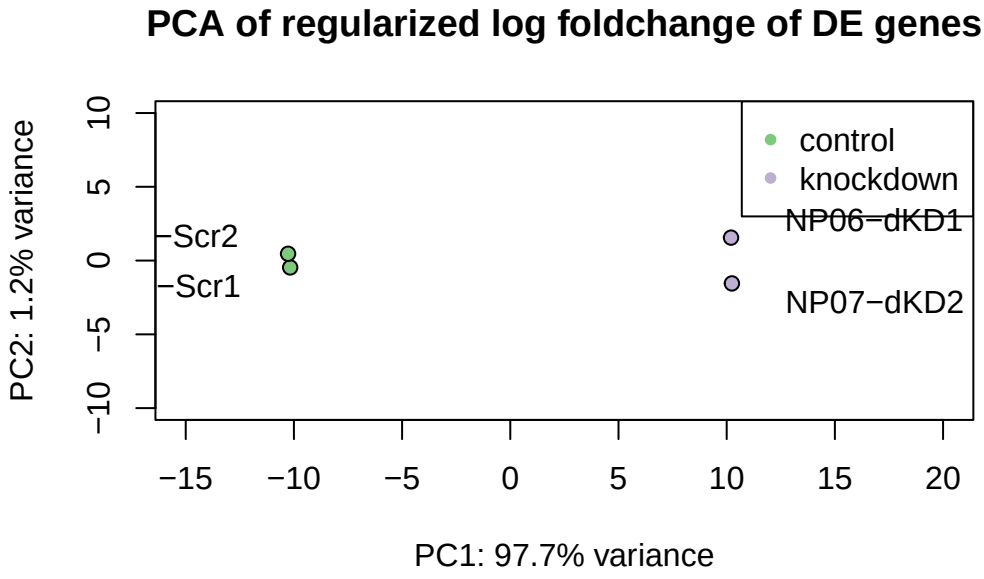
```
            Gene  baseMean log2FoldChange      lfcSE     stat        pvalue
1 ENSG00000224032 1000.8274       4.307767 0.13936391 30.91021 8.709885e-210
2 ENSG00000169242  886.1724       3.720117 0.13127462 28.33843 1.162292e-176
3 ENSG00000107262 1221.1706       2.555759 0.09930271 25.73705 4.501466e-146
4 ENSG00000210082 7819.4672       1.723809 0.06830571 25.23668 1.586237e-140
5 ENSG00000116761  605.5634       3.286235 0.14590196 22.52358 2.438256e-112
6 ENSG00000211459 3961.3526       1.490358 0.06967320 21.39070 1.630984e-101
          padj NP05-Scr1 NP06-dKD1 NP07-dKD2 NP08-Scr2   DE
1 7.773572e-206   89.5916  1837.283  1975.093  101.3427 TRUE
2 5.186729e-173  130.0919  1611.611  1681.869  121.1169 TRUE
3 1.339186e-142  351.0027  2086.889  2088.384  358.4071 TRUE
4 3.539291e-137 3424.1174 11737.182 12278.729 3837.8402 TRUE
5 4.352287e-109  101.8644  1114.679  1084.593  121.1169 TRUE
6  2.426088e-98 2055.6977  6056.650  5632.055 2101.0074 TRUE
```

```
# Plot dispersions
plotDispEsts(dds, main="Dispersion plot", genecol = "black", fitcol = "cyan", finalcol = "blue", legend
```



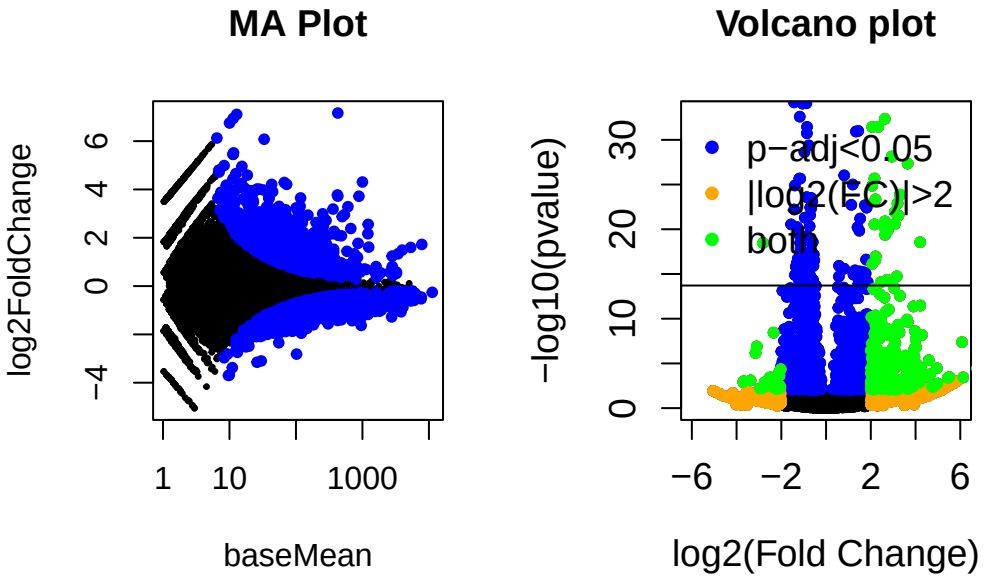
There's a trend where dispersion decreases as the mean of normalized counts increases, which is typical in RNA-seq data due to biological variability being more pronounced in genes with low expression levels. The model (fitted line) captures the overall trend of the dispersion estimates well, as indicated by the fitted points closely following the line.

```
rld <- rlogTransformation(dds) #applies a regularized log transformation to the dds(DESeqDataSet)

#Set colours for plotting
mycols <- brewer.pal(8, "Accent")[1:length(unique(condition))]

# PCA
rld_pca <- function(rld, intgroup = "condition", ntop = 500, colors = NULL, legendpos = "topright", mair

  rv <- rowVars(assay(rld))
  select <- order(rv, decreasing = TRUE)[seq_len(min(ntop, length(rv)))]
  pca <- prcomp(t(assay(rld)[select, ]))
  fac <- factor(apply(as.data.frame(colData(rld)[, intgroup, drop = FALSE]), 1, paste, collapse = " : ")

  pc1var <- round(summary(pca)$importance[2, 1] * 100, digits = 1)
  pc2var <- round(summary(pca)$importance[2, 2] * 100, digits = 1)
  pc1lab <- paste0("PC1: ", as.character(pc1var), "% variance")
  pc2lab <- paste0("PC2: ", as.character(pc2var), "% variance")


  plot(PC2 ~ PC1, data = as.data.frame(pca$x), bg = colors[fac], pch = 21, xlab = pc1lab, ylab = pc2lab,

  with(as.data.frame(pca$x), textxy(PC1, PC2, labs = rownames(pca$x), cex = textcx))
  legend(legendpos, legend = levels(fac), col = colors, pch = 20)
}

rld_pca(rld, colors = mycols, intgroup = "condition", xlim = c(-15, 20), ylim = c(-10, 10))
```



PCA further confirms there is little variability between biological replicates but high variability between gene expression of control and knockdown, indicating that the DRS long reads are capturing the variability of the NMD knockdown at a gene expression level.

```
# MA Plot
maplot <- function (res, thresh = 0.05, labelsig = FALSE, textcx = 1, ...) {
  with(res, plot(baseMean, log2FoldChange, pch = 20, cex = 0.5, log = "x", ...))
  with(subset(res, padj < thresh), points(baseMean, log2FoldChange, col = "blue", pch = 20, cex = 1))
  if (labelsig) {
    require(calibrate)
    with(subset(res, padj < thresh), textxy(baseMean, log2FoldChange, labs = Gene, cex = textcx, col = 2
  }
}

# Volcano Plot
volcanoplot <- function (res, lfcthresh = 2, sigthresh = 0.05, xlab = "log2(Fold Change)", legendpos = "
  with(res, plot(log2FoldChange, -log10(pvalue), pch = 20, xlab = xlab, cex.axis = 1.2, cex.lab = 1.2, .
  with(subset(res, padj < sigthresh), points(log2FoldChange, -log10(pvalue), pch = 20, col = "blue", ...
  with(subset(res, abs(log2FoldChange) > lfcthresh), points(log2FoldChange, -log10(pvalue), pch = 20, co
  with(subset(res, padj < sigthresh & abs(log2FoldChange) > lfcthresh), points(log2FoldChange, -log10(pv
  legend(legendpos, xjust = 1, yjust = 1, legend = c(paste("p-adj<", sigthresh, sep = ""), paste("|log2(
}

# Set up the layout
par(mfrow = c(1, 2), mar = c(5, 4, 4, 2) + 0.1)

# Plot MA Plot
maplot(resdata, main = "MA Plot")

# Plot Volcano Plot
volcanoplot(resdata, lfcthresh = 2, sigthresh = 0.05, xlim = c(-6, 6), ylim = c(0, 33), legendpos = "top
```




```
# Reset the par settings to default after plotting
par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```

Differential Isoform Usage Analysis

A differential isoform usage analysis between the control and the knockdown with IsoformSwitchAnalyzeR was run. This analysis will output genes whose isoform abundances change from control. Thereby, the identified isoforms should be NMDs sensitive isoforms.

```
sampleID = c("tpm_np05","tpm_np06","tpm_np07","tpm_np08")
myDesign = data.frame(sampleID= sampleID, condition = condition)
tpm_transcriptome$isoform_id <- rownames(tpm_transcriptome)

aSwitchList <-  importRdata(
  isoformCountMatrix   = tpm_transcriptome,
  isoformRepExpression = tpm_transcriptome,
  designMatrix         = myDesign,
  isoformExonAnnoation = "files/Homo_sapiens.GRCh38.110.chr_patch_hapl_scaff.gtf",
  isoformNtFasta       = "files/transcriptome_reference.fa",
  showProgress = FALSE
)

SwitchListFiltered <- preFilter(
  switchAnalyzeRlist = aSwitchList,
  geneExpressionCutoff = 5,
  isoformExpressionCutoff = 5,
  removeSingleIsoformGenes = TRUE)

SwitchListAnalyzed <- isoformSwitchTestDEXSeq(
  switchAnalyzeRlist = SwitchListFiltered,
```

```
  reduceToSwitchingGenes=TRUE,
  reduceFurtherToGenesWithConsequencePotential = FALSE,
  alpha = 0.05,
  dIFcutoff = 0.1,
  onlySigIsoforms = FALSE
)

extractSwitchSummary(SwitchListAnalyzed)
summary_isoforms <- SwitchListAnalyzed$isoformFeatures
write.csv(summary_isoforms[(summary_isoforms$iso_biotype == "nonsense_mediated_decay"),],"NMD_isoformfea
```

From the output of the IsoformSwitchAnalyzeR, we get a dataset where one of the columns classifies the isoforms according to the biological function ($iso_biotype). This classification is based on the gencode annotation. Although, all the isoforms switch obtained should be NMD sensitive, the number classified by the gencode annotation is only . The criteria used by genconde is the following: -If the coding sequence (following the appropriate reference) of a transcript finishes >50bp from a downstream splice site then it is tagged as NMD. If the variant does not cover the full reference coding sequence then it is annotated as NMD if NMD is unavoidable i.e. no matter what the exon structure of the missing portion is the transcript will be subject to NMD.

However, as discussed with our lab, classifying an isoform as NMD sensitive is not an easy task and the gencode criteria does not properly achieve this.

For that reason also, we compare this to the file with high confidence NMDs which were identified in (Karousis et al. 2021) (the lab who provided the data).

```
new_nmd = read.csv(file = 'NMD_exons_to_long_reads_transcriptome_mapping.csv',
                   sep = ',', header = TRUE)
head(new_nmd)
```

```
  chrom_ref start_ref   end_ref strand_ref     gene_id   transcript_id
1        20  50096483  50096534          - MSTRG.21153  MSTRG.21153.13
2        20  50096483  50096534          - MSTRG.21153 ENST00000483534
3        20  50096483  50096534          - MSTRG.21153 ENST00000470565
4         4 157145210 157145273          + MSTRG.25835   MSTRG.25835.4
5         4 157145210 157145273          + MSTRG.25835   MSTRG.25835.5
6        11  67432356  67432492          +  MSTRG.5855    MSTRG.5855.9
  gene_name     ref_gene_id
1
2    UBE2V1 ENSG00000244687
3    UBE2V1 ENSG00000244687
4
5
6
```

This file contains more novel identified isoforms not contained in the annotation file from ensembl, these novel identified transcripts are named with MSTRG. We filter them as we only have ensmbl annotated isoforms.

```
# Assuming new_nmd is a data frame
filtered_new_nmd <- new_nmd[grepl("^ENST", new_nmd$transcript_id), ]
```

```r
# Assuming summary_isoforms is a data frame
summary_isoforms$isoform_id <- sub("\\..*", "", summary_isoforms$isoform_id)

# View the updated data
head(summary_isoforms)

# Count the number of matching isoform_ids

count_same_ids <- summary_isoforms$isoform_id %in% filtered_new_nmd$transcript_id
transcript_id_equal <- summary_isoforms$isoform_id[count_same_ids]
genes_id_equal <- summary_isoforms$gene_id[count_same_ids]

count_same_ids_sum <- sum(count_same_ids)

# Print the count of exactly matching ids
print(count_same_ids)
```

This file contains more novel identified isoforms not contained in the annotation file from ensembl. To be able to , we would need the reference transcriptome they assembled and used for the alignment. Unluckly, this file was not avaible so we can only compare to the

On the other side, we check if our results contain BAG1. BAG1 is well-characterised as been NMD sensitive, so if it's present this will work as a sanity check.

```r
"BAG1" %in% summary_isoforms$gene_id
```

Splicing Analysis

Furthermore, we use SplAdder (Kahles et al. 2016) to perform a differential splicing analysis similar to (Karousis et al. 2021). SplAdder first generates a splicing graph based on the RNA sequencing data and extracts alternative splicing events (compared to the reference genome). The detected splicing events correspond to single or multiple skipped exons, intron retentions, alternative 3' or 5' splicing, and mutually exclusive exons. In a second step, it runs a differential test in order to find events that occur with a significantly different frequency in the knockdown samples compared to wildtype.

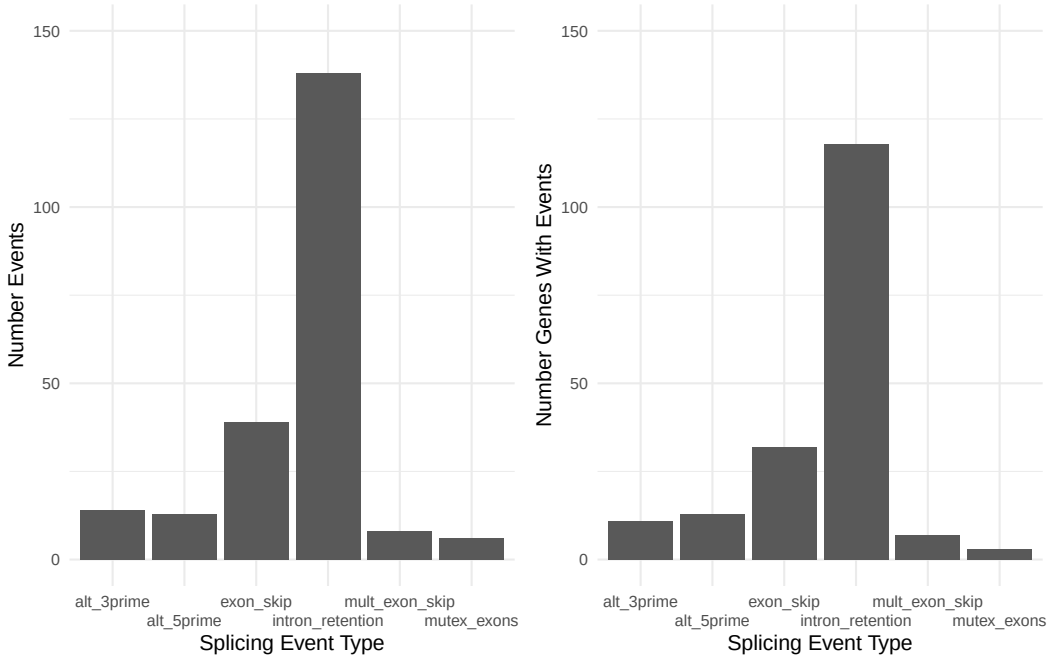
```
python -m spladder.spladder build --annotation $annotationFile \
                                  --bams $wt1,$wt2,$kd1,$kd2 \
                                  --outdir $outFolder

python -m spladder.spladder test \
              --conditionA $wt1,$wt2 \
              --conditionB $kd1,$kd2 \
              --parallel 24 \
              --outdir $outFolder
```

```r
# we load all by SplAdder detected events and merge them into one table

alt_3prime <- as.data.frame(read.table("splicing/test_results_C3_alt_3prime.tsv", sep = "\t", header = T
alt_5prime <- as.data.frame(read.table("splicing/test_results_C3_alt_5prime.tsv", sep = "\t", header = T
exon_skip <- as.data.frame(read.table("splicing/test_results_C3_exon_skip.tsv", sep = "\t", header = TRU
```

```
intron_retention <- as.data.frame(read.table("splicing/test_results_C3_intron_retention.tsv", sep = "\t'
mult_exon_skip <- as.data.frame(read.table("splicing/test_results_C3_mult_exon_skip.tsv", sep = "\t", he
mutex_exons <- as.data.frame(read.table("splicing/test_results_C3_mutex_exons.tsv", sep = "\t", header =

splicing_events <- Reduce(function(x, y) merge(x, y, all=TRUE), list(
  transform(alt_3prime, Type="alt_3prime"),
  transform(alt_5prime, Type="alt_5prime"),
  transform(exon_skip, Type="exon_skip"),
  transform(intron_retention, Type="intron_retention"),
  transform(mult_exon_skip, Type="mult_exon_skip"),
  transform(mutex_exons, Type="mutex_exons")
))

# filter only significant events
splicing_events <- splicing_events[splicing_events$p_val_adj < 0.05,]

# filter events with positive fold change
splicing_events <- splicing_events[splicing_events$log2FC_event_count < 0,]
```

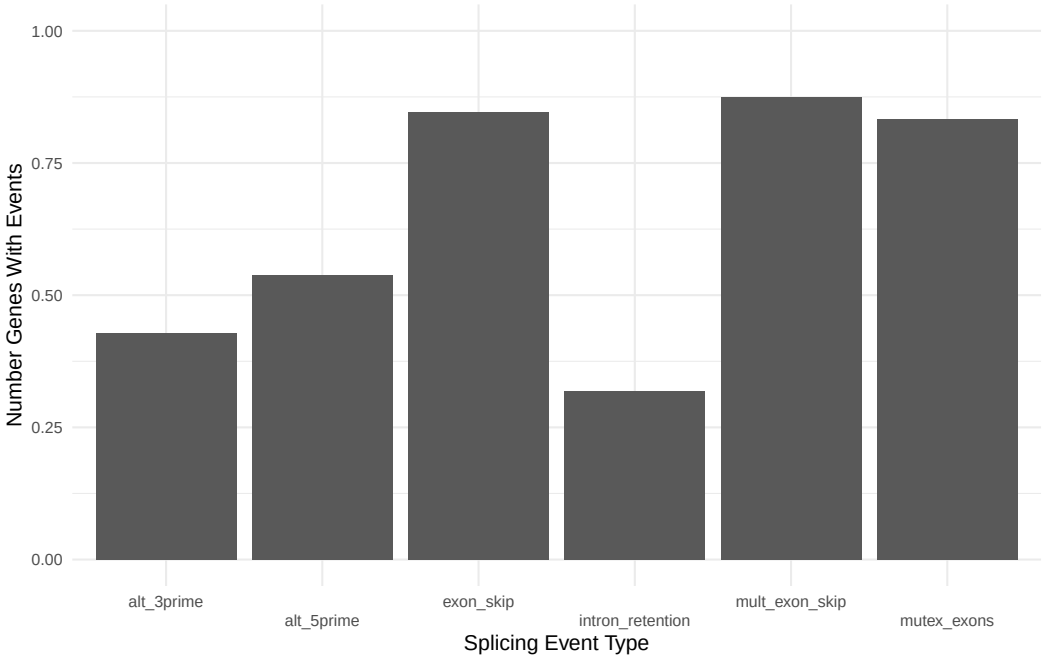
We first look the number of events per event type and the number of unique genes per event type. Note that we are only looking at events that occur significantly more often in the knockdown samples compared to wildtype, as we are interested in NMD sensitive isoforms.

```
event_counts_per_event_type <- splicing_events %>%
  group_by(Type) %>%
  summarise(UniqueCount = n_distinct(event_id))

gene_counts_per_event_type <- splicing_events %>%
  group_by(Type) %>%
  summarise(UniqueCount = n_distinct(gene_id))

events <- ggplot(event_counts_per_event_type, aes(x = Type, y = UniqueCount)) +
  geom_bar(stat = "identity") +
  ylim(0, 150) +
  labs(x = "Splicing Event Type",
       y = "Number Events") +
  theme_minimal(base_size = 8) +
  scale_x_discrete(guide = guide_axis(n.dodge = 2))

genes <- ggplot(gene_counts_per_event_type, aes(x = Type, y = UniqueCount)) +
  ylim(0, 150) +
  geom_bar(stat = "identity") +
  labs(x = "Splicing Event Type",
       y = "Number Genes With Events") +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  theme_minimal(base_size = 8)

ggarrange(events, genes, ncol = 2, nrow = 1)
```



SplAdder only detected 218 of significant events. This leads to a low sensitivity of 0.0121429. However, for isoforms with skipped or mutually exclusive exons, the specificity is above 80%:

```
splicing_events$nmd_sensitive <- splicing_events$gene_id %in% filtered_new_nmd$ref_gene_id

specificity <- splicing_events %>%
  group_by(Type) %>%
  summarise(average_value = mean(nmd_sensitive, na.rm = TRUE))

ggplot(specificity, aes(x = Type, y = average_value)) +
  ylim(0, 1) +
  geom_bar(stat = "identity") +
  labs(x = "Splicing Event Type",
       y = "Number Genes With Events") +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +
  theme_minimal(base_size = 8)
```



SplAdder identified 116 isoform candidates of genes that are not present in the precompiled gene set.

Furthermore, we can use SplAdder to confirm different splicing behavior previously reported in literature. One example is the apoptosis-modulating Bcl-2-associated athanogene-1 (BAG-1). (Karousis et al. 2021) reported that a BAG1 isoform with an included alternative exon is stabilized upon NMD inhibition. This effect can also be observed in our data set:

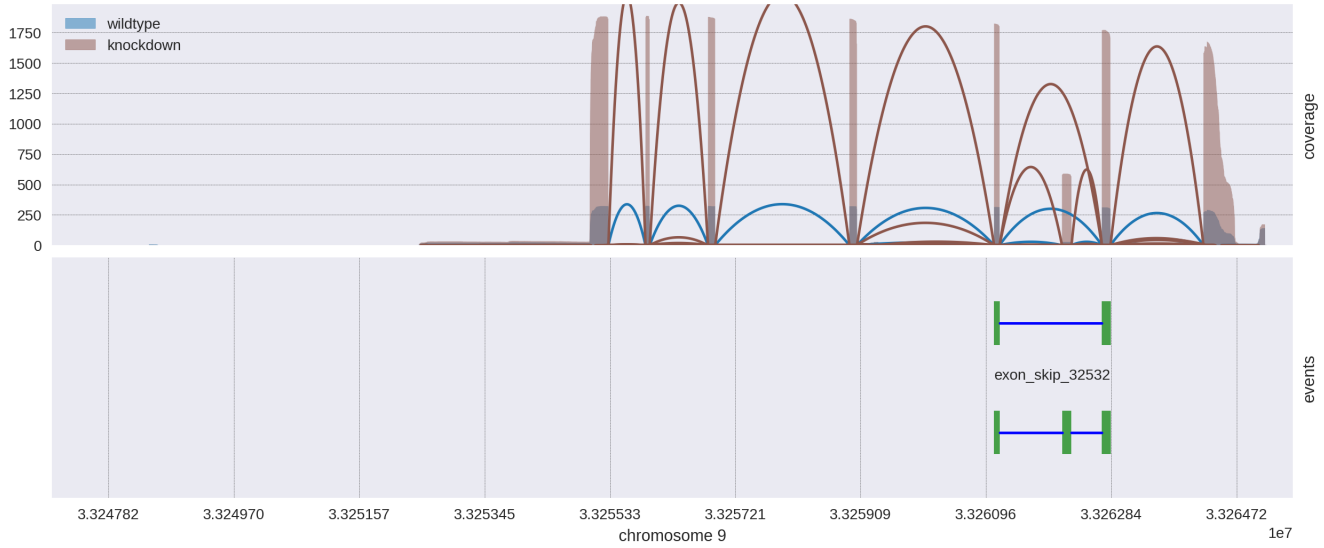


Figure 4: SplAdder Analysis of the BAG-1 Gene. One can observe that an alternative isoform is predominant in the knockdown sample, an observation that is in line with previous literature.

```
python -m spladder.spladder viz \
              --range gene ENSG00000107262 \
              --track coverage wildtype:$wt1,$wt2 knockdown:$kd1,$kd2 \
              --track event exon_skip \
              -O plot --format png \
              --outdir $outFolder
```

Discovery of New Isoforms

We use FLAIR (Tang et al. 2020) to detect new isoforms in the sequences of the knockdown samples when compared to the reference transcriptome.

```
python -m flair.flair correct \
--gtf $annotationFile \
--genome $referenceFile \
--query $experimentFolder/alignments/primary-genomic-aln.bed12 \
--output $experimentFolder/flair/correct

python -m flair.flair collapse \
--gtf $annotationFile \
--reads $rawReads \
--genome $referenceFile \
--query $experimentFolder/flair/correct_all_corrected.bed \
--trust_ends \
--stringent \
--threads 16 \
--output $experimentFolder/flair/collapse

/cluster/home/ochsneto/gffcompare-0.12.6.Linux_x86_64/gffcompare -r $annotationFile -o $experimentFolder
```

The analysis resulted in 241 novel exons as compared to the reference transcriptome:

Table 1: Results of the FLAIR analysis comparing the isoforms in the knockdown samples with the reference transcriptome.

Type	Result
Missed exons	477423/666967 (71.6%)
Novel exons	241/81798 (0.3%)
Missed introns	282313/402495 (70.1%)
Novel introns	0/73365 (0.0%)
Missed loci	49706/57652 (86.2%)
Novel loci	228/8579 (2.7%)

References

Aird, Daniel, Michael G Ross, Wei-Sheng Chen, Maxwell Danielsson, Timothy Fennell, Carsten Russ, David B Jaffe, Chad Nusbaum, and Andreas Gnirke. 2011. "Analyzing and Minimizing PCR Amplification Bias in Illumina Sequencing Libraries." *Genome Biology* 12 (2): 1–14.

Gleeson, Josie, Adrien Leger, Yair DJ Prawer, Tracy A Lane, Paul J Harrison, Wilfried Haerty, and Michael B Clark. 2022. "Accurate Expression Quantification from Nanopore Direct RNA Sequencing with NanoCount."

Nucleic Acids Research 50 (4): e19–19.

Kahles, Andre, Cheng Soon Ong, Yi Zhong, and Gunnar Rätsch. 2016. "SplAdder: Identification, Quantification and Testing of Alternative Splicing Events from RNA-Seq Data." *Bioinformatics* 32 (12): 1840–47.

Karousis, Evangelos D, Foivos Gypas, Mihaela Zavolan, and Oliver Mühlemann. 2021. "Nanopore Sequencing Reveals Endogenous NMD-Targeted Isoforms in Human Cells." *Genome Biology* 22 (1): 1–23.

Karousis, Evangelos D, and Oliver Mühlemann. 2022. "The Broader Sense of Nonsense." *Trends in Biochemical Sciences*.

Roundtree, Ian A, Molly E Evans, Tao Pan, and Chuan He. 2017. "Dynamic RNA Modifications in Gene Expression Regulation." *Cell* 169 (7): 1187–1200.

Steijger, Tamara, Josep F Abril, Pär G Engström, Felix Kokocinski, Tim J Hubbard, Roderic Guigó, Jennifer Harrow, and Paul Bertone. 2013. "Assessment of Transcript Reconstruction Methods for RNA-Seq." *Nature Methods* 10 (12): 1177–84.

Tang, Alison D., Cameron M. Soulette, Marijke J. van Baren, Kevyn Hart, Eva Hrabeta-Robinson, Catherine J. Wu, and Angela N. Brooks. 2020. "Full-Length Transcript Characterization of SF3B1 Mutation in Chronic Lymphocytic Leukemia Reveals Downregulation of Retained Introns." *Nature Communications* 11 (1): 1438. https://doi.org/10.1038/s41467-020-15171-6.