iCLIP data analysis: A complete pipeline from sequencing reads to RBP binding sites

Anke Busch[a], Mirko Brüggemann[b], Stefanie Ebersberger[a,*], Kathi Zarnack[b,*]

[a]*Institute of Molecular Biology (IMB), Ackermannweg 4, 55128 Mainz, Germany*
[b]*Buchmann Institute for Molecular Life Sciences (BMLS), Goethe University Frankfurt, Max-von-Laue-Str. 15, 60438 Frankfurt, Germany*

Abstract

Precise knowledge on the binding sites of an RNA-binding protein (RBP) is key to understanding the complex post-transcriptional regulation of gene expression. This information can be obtained from individual-nucleotide resolution UV crosslinking and immunoprecipitation (iCLIP) experiments. Here, we present a complete data analysis workflow to reliably detect RBP binding sites from iCLIP data. The workflow covers all steps from the initial quality control of the sequencing reads up to peak calling and quantification of RBP binding. For each tool, we explain the specific requirements for iCLIP data analysis and suggest optimised parameter settings.

Keywords: iCLIP, bioinformatics, data processing, UV crosslink events, binding sites, RNA-binding protein

2010 MSC: 00-01, 99-00

*Corresponding author
Email address: kathi.zarnack@bmls.de (Kathi Zarnack)

Contents

1. Introduction

The precise spatial and temporal regulation of gene expression is essential for cellular function. Post-transcriptional regulation acts on all steps of RNA processing, including splicing, 3' end processing, nuclear export, translation and mRNA decay. Key players are RNA-binding proteins (RBPs) that determine the fate and function of each transcript in the cell. Many RBPs recognise their target mRNAs via specific binding sites, which harbour characteristic RNA sequence motifs and/or structural RNA folds. A comprehensive knowledge of the binding sites of a given RBP throughout the transcriptome allows to discover its RNA binding specificity and to unravel its molecular mode of action.

Starting with UV crosslinking and immunoprecipitation (CLIP) in 2003 [1], a range of high-throughput techniques have been introduced to identify the *in vivo* binding sites of an RBP of interest [2]. Modifications of the protocol include the use of photoreactive ribonucleoside analogues (PAR-CLIP) [3] and affinity purification under denaturing conditions (CRAC) [4]. By increasing on both resolution and sensitivity, 'individual-nucleotide resolution CLIP' (iCLIP) has proven as a powerful tool to precisely map and quantify RBP binding sites throughout the transcriptome [5]. This method achieves nucleotide resolution on the RBP crosslink sites by capturing cDNAs that truncate at the crosslinked peptide during reverse transcription (see 1.1 below). The same truncation principle was adopted in further CLIP variants, such as 'enhanced CLIP' (eCLIP) [6] and 'infrared-CLIP' (irCLIP) [7]. In addition, the iCLIP protocol was evolved to measure N6-methyladenosine (m6A) RNA modifications using 'm6A iCLIP' (miCLIP) [8].

1.1. A brief description of the iCLIP experiment

In order to obtain a snapshot of the *in vivo* RNA binding pattern of an RBP, the iCLIP experiment initiates with the UV irradiation of living cells to covalently crosslink immediate contacts between proteins and nucleic acids (Figure 1A). A partial RNase digestion is then employed to restrict the RNA frag-

ment lengths to a defined range. This has to be carefully optimised, as overdigestion can constrain the detected binding sites [9]. The crosslinked protein-RNA complexes are then immunoprecipitated with a specific antibody against the RBP of interest, and the RBP is subsequently removed from the RNA by proteinase digestion. Importantly, this leaves a small polypeptide which triggers truncation of the reverse transcription, thereby inheriting the positional information of the crosslink site into the resulting cDNAs. The cDNAs are captured and amplified for high-throughput sequencing (see 1.2 below).

In the new iCLIP2 protocol [add citation here], several steps have been optimised to improve the quality and complexity of the iCLIP libraries. Bringing together features of several CLIP variants [6, 7], the iCLIP2 library preparation includes two separate linker ligation reactions, a PCR pre-amplification step to minimise sample loss, and bead-based size selection of the cDNAs. In addition, the barcode sequences have been extended to account for larger library sizes and increased sequencing depths (see below).

For more details on the experimental procedure and the improved library preparation steps in iCLIP2, please refer to [10] and Buchbender *et al.* in this issue [add citation here].

1.2. The iCLIP read structure

Since reverse transcription truncates at the crosslinked polypeptide on the RNA, the produced cDNAs start exactly one position downstream of the crosslinked nucleotide in the RNA. During the iCLIP library preparation, specific linker sequences are ligated to either end of the cDNAs for high-throughput sequencing (adding a total of 155 nucleotides [nt] to the cDNAs). The layout preserves the strand information, such that the sequencing read corresponds to the strand of the bound RNA fragment. The 5' linker additionally harbours a fixed stretch with an experimental barcode and a bipartite unique molecular identifier (UMI) sequence, which are the first nucleotides of each sequencing read (Figure 1B). The format of this stretch is `NNNXXXXNN` in the original iCLIP protocol (where N and X are nucleotides of the UMI and the experimental barcode, respectively),

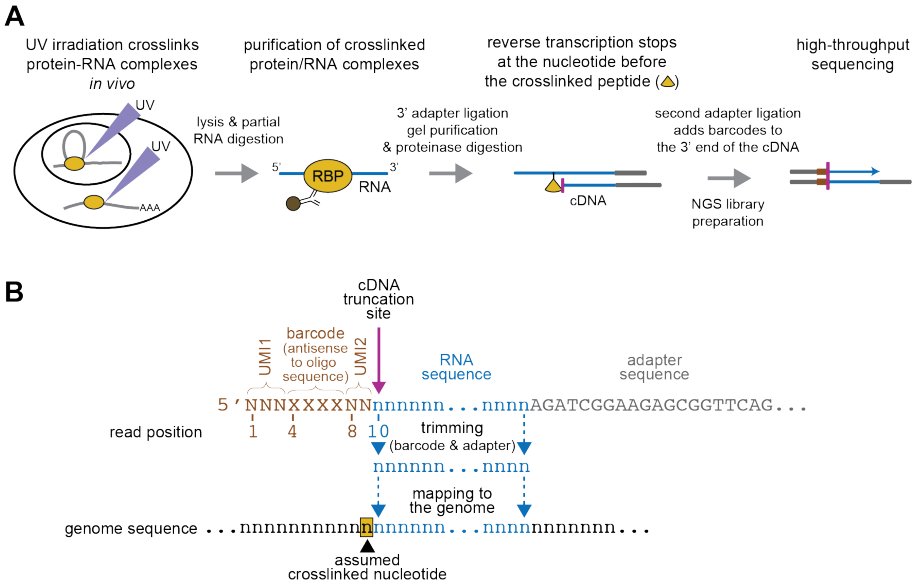


Figure 1: **Overview of the iCLIP experiment (A)** Experimental workflow. UV irradiation of living cells covalently crosslinks proteins and nucleic acids in immediate contact. Upon cell lysis, a partial RNase digestion shortens the RNA molecules, followed by immunoprecipitation of the crosslinked protein-RNA complexes. Proteinase K digestion leaves a small peptide at the crosslink site, which results in truncation of reverse transcription. The resulting cDNAs are captured and subjected to high-throughput sequencing. **(B)** Structure of an iCLIP read. The read starts with a fixed stretch harbouring the bipartite UMI and the experimental barcode (`NNNXXXXNN` layout from the original iCLIP protocol). The 9-nt barcode and UMI stretch at the read starts as well as potential adapter sequences at read ends need to be trimmed before mapping.

and was extended to `NNNNNXXXXXXNNNN` in the iCLIP2 protocol. More details on the oligonucleotide design as well as a list of recommended barcode sequences with the associated IDs can be found in [10] and Buchbender *et al.* in this issue.

For a standard iCLIP experiment, single-end sequencing with a read length of 70-100 nt is generally sufficient for capturing the necessary information.[1] Due to the experimental barcode in the read sequence, several iCLIP libraries can

[1] A possible exception are RBPs that are expected to bind within repetitive regions, for which longer reads or paired-end sequencing may increase the chance for unique mapping.

be multiplexed in one sequencing run. The required read number per sample depends on the expected binding behaviour of the RBP, including the number of binding sites and the extent of background binding, the complexity of the cDNA library and the intended sensitivity for weak binding sites and lowly expressed transcripts, among others. For instance, for human RBPs binding to mature mRNAs, good libraries may start from 1 million reads, whereas at least 10 million reads are advisable for most RBPs binding to pre-mRNA.

2. Computational requirements

2.1. External software

The iCLIP analysis pipeline uses Linux `bash` as well as `R` code for post-processing, downstream analyses and visualisations. In addition, the following external tools and dependencies need to be installed (version numbers in brackets were used for the example described in this manuscript):

- `FastQC` (version 0.11.5) [11] (`http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`)
- `FASTX-Toolkit` (version 0.0.14) [12] (`http://hannonlab.cshl.edu/fastx_toolkit/`)
- `seqtk` (version 1.3) [13] (`https://github.com/lh3/seqtk`)
- `Flexbar` (version 3.4.0) [14] (`https://github.com/seqan/flexbar`)
- `STAR` (version 2.5.4b) [15] (`https://github.com/alexdobin/STAR`)
- `UMI-tools` (version 0.5.5) [16] (`https://github.com/CGATOxford/UMI-tools`)
- `SAMtools` (version 1.5) [17] (`https://github.com/samtools/samtools`)
- `BEDTools` (version 2.27.1) [18] (`https://github.com/arq5x/bedtools2`)
- `kentUtils` (version v365) [19] (`https://github.com/ENCODE-DCC/kentUtils`)
- `iCLIPro` (version 0.1.1) [20] (`http://www.biolab.si/iCLIPro/doc/`)
- `PureCLIP` (version 1.3.0) [21] (`https://github.com/skrakau/PureCLIP`)
- `R` (version 3.5.1) [22] (`https://www.R-project.org/`)
- `R` package `GenomicRanges` (version 1.34.0) [23] (`https://www.bioconductor.org/packages/release/bioc/html/GenomicRanges.html`)
- `R` package `rtracklayer` (version 1.42.1) [24] (`https://www.bioconductor.org/packages/release/bioc/html/rtracklayer.html`)

- `R` package `ggplot2` (version 3.1.0) [25] (`https://ggplot2.tidyverse.org/`)

In addition to the stand-alone versions, most tools have been integrated into the European Galaxy server (`https://usegalaxy.eu/`) [26].

2.2. Memory requirements

The provided Linux `bash` and `R` code as well as all external tools mentioned in Section 2.1 generally run on any Unix/Linux distribution. For all processing steps until peak calling (Chapters 3-4), our example code was run on a Debian GNU/Linux 9 distribution on a server with 64 cores / 128 threads, dual AMD EPYC 7501 processor and 1024 GB of main memory. Depending on the species/genome used for the experiments, `STAR` might need a substantial amount of RAM to map the iCLIP reads (e.g. $\sim$ 48 GB in the case of human data). Besides, extracting good quality reads might also become RAM-consuming if the data set is large, i.e. 300 million reads or more.

2.3. Code and data availability

The code in this manuscript is available in separate script files in Supplementary Data 1 (`bash` code) and Supplementary Data 2 and 3 (`R` code). As a showcase example to demonstrate the pipeline, we used a previously published iCLIP dataset for the splicing factor U2AF2 (also known as U2AF65)[27]. The demultiplexed `fastq` files of the four replicates are available in GEO under the accession numbers GSM2650195, GSM2650196, GSM2650197 and GSM2650198.

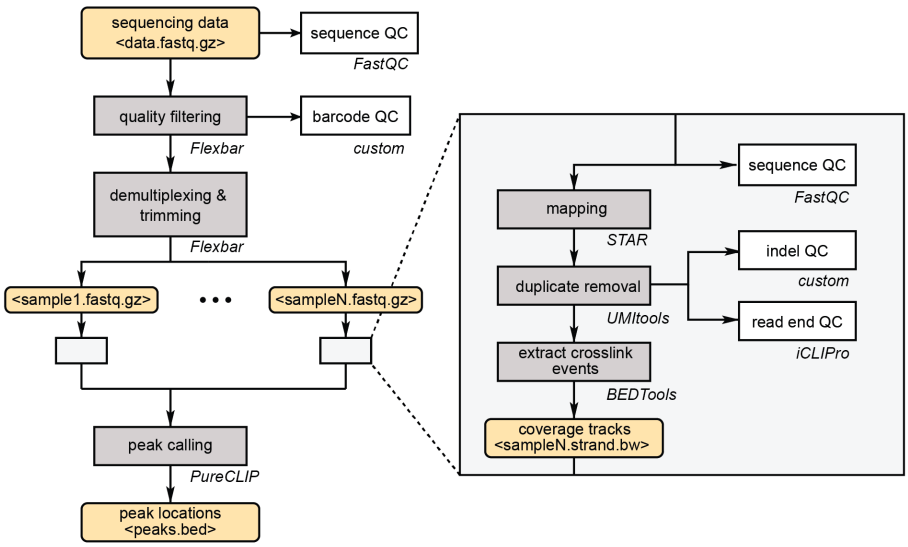


Figure 2: **Analysis overview.** Workflow summarising the processing (grey) and quality control (QC, white) steps together with the major file types produced (orange).

3. Basic read processing

This section describes all steps from the raw sequencing reads to obtaining coverage tracks of crosslink events and for collecting several statistics and quality checks. An overview of the pipeline is shown in Figure 2. The code for all steps in this chapter can be found in **Supplementary Data 1**.

3.1. Settings

Before starting the analysis, a number of parameters need to be defined:

- `barcodes.fasta`: A `fasta` file of barcode sequences that were used in the experiment. Each barcode consists of the two UMIs flanking the experimental barcode, such as:

```
>sample1
NNNGGTTNN
```

- `adapter.seq`: Sequence of the 3' adapter that was used for iCLIP library preparation. Usually `AGATCGGAAGAGCGGTTCAG`. For details, see [10] and Buchbender *et al.* in this issue.

- (`x`,`y`,`z`): Symbols to denote the lengths of the UMI and experimental barcode regions in the following descriptions (`x` = length of the first UMI segment, `y` = length of the experimental barcode, `z` = length of the second UMI segment).

- `readLength`: Read length from high-throughput sequencing.

- `minReadLength`: Minimum length to retain reads after trimming. Usually set to 15 nt.

- `maxReadLength`: Maximum possible read length after trimming. Equals to read length minus UMI and experimental barcode regions (`readLength - (x + y + z)`).

- `minBaseQuality`: Minimum quality (Phred score per base) in the barcode region to ensure high confidence for demultiplexing and deduplication. Often set to 10.

- `genomeMappingIndex`: `STAR` genome index specific to the organism used in the experiment.

3.2. Quality control

3.2.1. General quality check

The starting point of the pipeline is a file of sequencing reads in `fastq` file format (or in a compacted version as `.fastq.gz`), which includes the reads of all samples that were multiplexed in the sequencing run. The quality of the sequencing run can be checked using `FastQC` [11]:

```
fastqc --extract --nogroup --outdir <outdir> <data.fastq.gz>
```

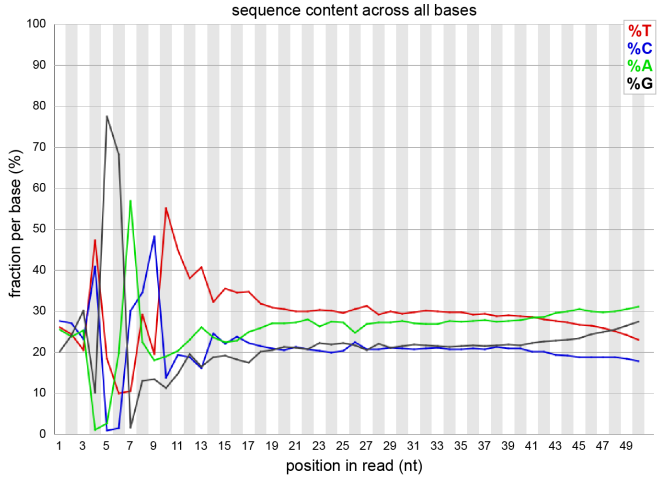


Figure 3: **Quality control of the iCLIP sequencing reads.** Example `FastQC` plot evaluating the Per Base Sequence Content of the full data set.

Among the quality measures reported by `FastQC`, we mainly focus on Per Base Sequence Quality and Per Base Sequence Content:

- Especially for longer reads, the Per Base Sequence Quality (Phred score) can drop towards the end of the reads. In case of extremely low qualities, we recommend trimming the 3' ends of reads.

- The Per Base Sequence Content over all iCLIP reads shows the UMI (here positions 1-3 and 8-9) and the experimental barcode (here positions 4-7; Figure 3). The pattern from position 10 onward reflects the RNA binding preference of the RBP, and consequently varies between the studied proteins. Keep in mind that because this picture looks different from standard RNA-sequencing data, `FastQC` might red-flag some of its checks, but these failed check are not necessarily meaningful in the context of iCLIP data.

3.2.2. Quality filter on the barcode region

If no major problems are reported by `FastQC`, the reads are filtered for a minimum quality in the barcode region (see 1.2). We explicitly filter for the

quality in this region, since sequencing errors in the experimental barcode can result in a misassignment of reads between libraries of a multiplexed sequencing run. Since the quality filter is only applied to a defined segment of the reads, this step has to be customised by either using `bash` commands or a combination of tools as follows:

First, `fastx_trimmer` of the `FASTX-Toolkit` [12] can be used to trim the reads to the barcode region of length `barcodeLength` $= x+y+z$. These barcode regions are only kept if all positions (`-p 100`) have a minimum Phred score of `minBaseQuality` (typically set to 10). Using a short `bash` command, the IDs of the retained sequences are written to a temporary file `tmp/data.qualFilteredIDs.list`.

```
zcat <data.fastq.gz> | fastx_trimmer -l barcodeLength |
fastq_quality_filter -q minBaseQuality -p 100 |
awk 'FNR%4==1 {print $1}' | sed 's/@//' >
<tmp/data.qualFilteredIDs.list>
```

Based on the read IDs, `seqtk subseq` [13] is used to extract the respective reads (in complete length) from the original data file `data.fastq.gz`.

Depending on the sequencing platform used, read IDs may contain whitespaces or other special characters which will results in truncation of the IDs during mapping. In order to circumvent this, these need to be removed. In our example, we replace whitespaces and slashes by `#` using `sed`.

```
seqtk subseq <data.fastz.gz> <tmp/data.qualFilteredIDs.list> |
sed 's/ /#/g; s/\//#/g' | gzip > <data.filtered.fastq.gz>
```

Optionally, the result of the quality filter can be checked by re-running `FastQC`.

3.2.3. Barcode frequencies

In order to assess the relative abundance of all samples and to check for potential contamination, all experimental barcodes are extracted from the full data set. Based on the length x of the first UMI segment and the length y of the experimental barcode, the following `bash` code returns all occurring y-mers in the positions of the experimental barcode, sorted by their frequency.

```
zcat <data.filtered.fastq.gz> | awk -v umi1_len=x
-v exp_bc_len=y '{if (FNR%4==2)
print substr($1,(umi1_len+1),exp_bc_len)}' | sort |
uniq -c | sort -k1,1rn > <exp_barcodes.detected>
```

We recommend to check that all expected barcodes appear among the top hits and to ensure that no additional barcodes are overrepresented. To this end, we visualise the frequency of twice as many barcodes as have been multiplexed in the sequencing run (here: 2 * `#expectedBarcodes` = 8; Figure 4A). Similarly, barplots can be used to show the frequency of all unexpected barcodes and their shortest Hamming distance to any of the expected barcodes (Figure 4B). In an uncontaminated iCLIP library, the expected barcodes commonly outnumber the other possible barcodes by at least 10fold. The most frequently occurring unexpected barcodes usually differ in just one position (Hamming distance = 1), most likely arising from amplification or sequencing errors in the barcode region.

All unexpected barcodes will be removed during demultiplexing (see Section 3.3). Nonetheless, it is important to follow up on the source of contamination and if applicable, to take counteractive measures. An effective intervention to avoid carry-over between experiments is a strict spatial separation, ideally in distinct rooms, of pre- and post-amplification steps during iCLIP library preparation [10]. At later stages, technical problems, such as high error rates in the first cycles of sequencing, can also impair the accuracy of the barcode readout.

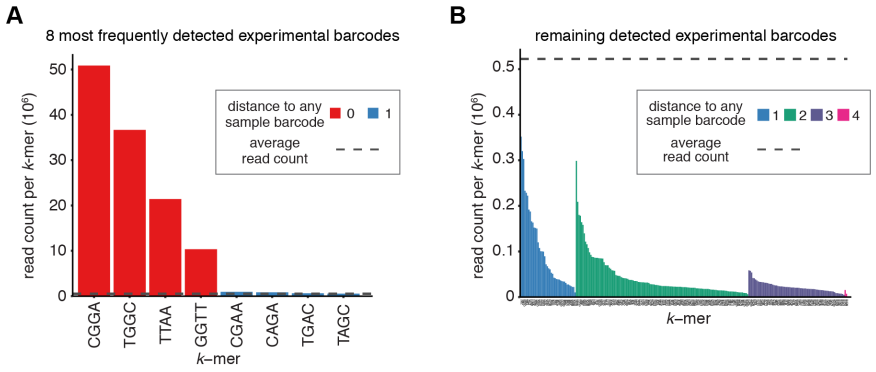


Figure 4: Frequency of y-mers in the position of the experimental barcode. Assuming `#expectedBarcodes` is the number of expected experimental barcodes, **(A)** shows the 2 * `#expectedBarcodes` most frequently found experimental barcodes, while **(B)** shows all remaining experimental barcodes. The dashed line indicates the average frequency of all experimental barcodes, colours indicate the shortest Hamming distance to the expected barcodes.

3.3. Demultiplexing, adapter and barcode trimming

Following initial quality control and quality filtering, demultiplexing and adapter trimming are performed on the quality-filtered data using `Flexbar` [14]. In addition to separating the reads of the different samples, this step extracts the barcode region from the 5' end of the reads (`--barcode-trim-end LTAIL`), and trims adapters from the 3' end if present (`--adapter-trim-end RIGHT`). We recommend not allowing any mismatches for barcode matching (set by `--barcode-error-rate 0`), while an error rate of 0.1 is acceptable for trimming adapter (`--adapter-seq adapter.seq --adapter-error-rate 0.1`). In order to remove all adapter traces, even if only the very first nucleotides of them were sequenced, we commonly require just 1 nt of overlap between the read 3' end and the beginning of the adapter (`--adapter-min-overlap 1`). Since very short sequences are likely to overlap by chance, this stringent setting means that many reads will be trimmed by a few extra bases, but ensures that even the shortest remaining adapter fragments are trimmed off. Only trimmed reads with a remaining length of at least `minReadLength` are kept for further analysis.

Demultiplexing, adapter trimming and barcode removal can either be done

separately or in one step. For the latter, we commonly use `Flexbar` with the following parameters:

```
flexbar -r <data.filtered.fastq.gz> --zip-output GZ

          --barcodes barcodes.fasta  --barcode-unassigned
          --barcode-trim-end LTAIL --barcode-error-rate 0

          --adapter-seq adapter.seq  --adapter-trim-end RIGHT
          --adapter-error-rate 0.1 --adapter-min-overlap 1
          --min-read-length minReadLength

          --umi-tags
```

Using this command, `data.filtered.fastq.gz` is split into separate `fastq` files based on the barcodes and sample names specified in `barcodes.fasta`. The log output will report the number of reads assigned to each sample, the number of unassigned reads as well as the number of reads removed after adapter trimming due to the `minReadLength` cutoff. Reads not assigned to any barcode will be written to a separate file if `--barcode-unassigned` is specified. `--umi-tags` captures the UMI sequence (defined as N positions in `barcodes.fasta`) and adds it to the ID of each read. This allows to preserve the UMI information during genomic mapping, as it is required for the subsequent deduplication step, i.e. the removal of PCR duplicates.

As mentioned in Section 3.2.2, whitespaces and other special characters can lead to truncation of the read IDs during mapping and should therefore be removed before. In the case that read IDs should remain unchanged, the `fastq.gz` files after `Flexbar` demultiplexing can be re-processed to bring the UMIs forward within the read IDs. A `bash` and `awk` code implementing this optional step can be found in the Supplementary Material.

Following demultiplexing and trimming, we recommend running `FastQC`

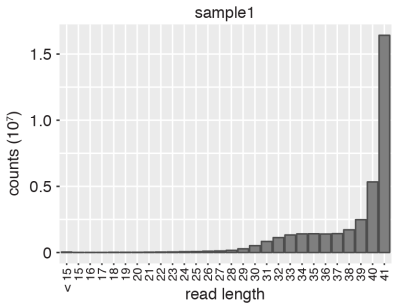


Figure 5: Read length distribution after trimming. In an ideal experiment, the majority of reads are still full-length, i.e. they did not contain adapter sequence and hence were not shortened during trimming.

again to make sure that all samples contain reads of sufficient quality. Furthermore, the length distribution of the reads in each sample can be checked using tools of the `FASTX-Toolkit` as follows:

1. Transform `fastq` files of all samples into `fasta` files:
   `zcat <sampleX.fastq.gz> | fastq_to_fasta -n -r | gzip > <sampleX.fasta.gz>`
2. Create a histogram of the length distribution using `fasta` files:
   `fasta_clipping_histogram.pl <sampleX.fasta.gz> <sampleX.readlength.png>`

Alternatively, running `Flexbar` with parameter `--length-dist` returns a tab-delimited text file of the read length distribution for each sample. This can be used in `R` to produce similar custom-made plots as shown in Figure 5.

3.4. Genomic mapping

After demultiplexing, the individual `fastq` files for each sample are mapped to a reference genome. Here, we used `GRCh38.p7.genome.fa`, downloaded from `ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_25/GRCh38.p7.genome.fa.gz`. We suggest to use the splice-aware alignment software `STAR` [15] for genomic mapping. A gene annotation file `annotation.gtf` can be provided optionally using options `--sjdbGTFfile` and `--sjdbOverhang`.

Here, `gencode.v25.chr_patch_hapl_scaff.annotation.gtf` was used and downloaded from `ftp://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_25/gencode.v25.chr_patch_hapl_scaff.annotation.gtf.gz`. For details on available parameters, see the `STAR` user manual. Most importantly, soft-clipping has to be turned off on the 5' end of reads (`--alignEndsType Extend5pOfRead1`). This is essential to preserve the information on the crosslinked nucleotide in the bound RNA, which corresponds precisely to the position upstream of the start of the iCLIP cDNA.

Following the ENCODE [28] standard option for RNA sequencing (as specified in the `STAR` user manual), we recommend allowing up to 4% mismatched bases (`--outFilterMismatchNoverReadLmax 0.04 --outFilterMismatchNmax 999`). Furthermore, only uniquely mapped reads[2] are kept for further analyses (`--outFilterMultimapNmax 1`). The fraction of uniquely mappable reads depends on the organism as well as the RBP of interest, i.e. the sequence composition of typical binding regions of this protein. For a human RBP binding to non-repetitive regions, unique mapping rates of $80-90\%$ are good to excellent, whereas rates below 70% are considered poor.

An example `STAR` call is as follows:

```
STAR --runMode alignReads
      --genomeDir genomeMappingIndex
      --outFilterMismatchNoverReadLmax 0.04
      --outFilterMismatchNmax 999
      --outFilterMultimapNmax 1
      --alignEndsType Extend5pOfRead1

      --sjdbGTFfile annotation.gtf
```

[2] A common practice for multi-mapping reads is to assign them randomly to one of the possible locations. This impairs accurate duplicate removal based on UMIs. We therefore restrict all analyses to uniquely mapping reads.

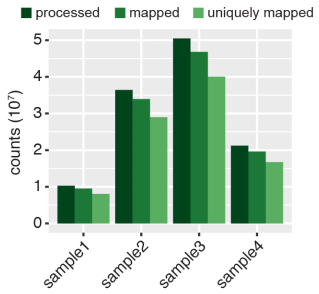


Figure 6: **Genomic mapping.** Read numbers for the four replicates before and after genomic mapping.

```
--sjdbOverhang maxReadLength-1
--outReadsUnmapped Fastx
--outSJfilterReads Unique

--readFilesCommand zcat
--outSAMtype BAM SortedByCoordinate

--readFilesIn <sampleX.fastq.gz>
```

By default, the output will be written to `Aligned.sortedByCoord.out.bam`. For simplicity, we will rename this file to `sampleX.bam`. The mapping statistics for all samples can be visualised as shown in Figure 6.

4. Conversion into crosslink events

4.1. Duplicate removal (deduplication)

Due to the PCR amplification during iCLIP library preparation and the limiting amounts of starting material, technical duplicates can make up a substantial fraction of iCLIP datasets and need to be removed. A sequencing read is considered a technical duplicate of another read if they map to the same coordinates in the genome and harbour identical UMIs. In this case, they most likely originate from the same co-purified RNA fragment and have been multiplied during the PCR reaction. In contrast, two reads are counted as independent crosslink events ('biological duplicates') if they map to the same location, but have different UMIs. Alternative approaches to remove duplicates already at the `fastq` level are inferior, since sequence variations from PCR or sequencing errors will be masked [29].

We suggest removing technical duplicates using `UMI-tools` [16]:

```
umi_tools dedup -I <sampleX.bam> -L <sampleX.duprm.log>
-S <sampleX.duprm.bam> --extract-umi-method read_id
--method unique
```

`UMI-tools` will use the input file `sampleX.bam` and its corresponding index file `sampleX.bam.bai` and write the output without technical duplicates to `sampleX.duprm.bam`. The index file `sampleX.bam.bai` can easily be prepared using `SAMtools` [17]:

```
samtools index <sampleX.bam>
```

By setting the option `--extract-umi-method read_id`, `UMI-tools` will look for the UMI in the read ID. The parameter `--method` defines which reads are considered duplicates. `--method directional` identifies clusters of connected UMIs (based on Hamming distance), while `--method unique` only col-

lapses duplicate reads with identical UMIs. While `--method directional` is
405 `UMI-tools`'s default method to define duplicates, it may lead to wrong assignments and excess removal. For instance, very high-complexity iCLIP libraries may contain hardly any duplicates and, thus, closely connected UMIs occur with low but similar frequencies. In this case, only collapsing reads with identical UMIs will be beneficial. In contrast, highly over-amplified libraries might bene-
410 fit from the use of the `directional` duplicate removal. Generally, if the libraries are not strongly over-amplified, we recommend `--method unique` to avoid incorrect duplicate removal. In the example used throughout the manuscript, `--method directional` was used. See the `UMI-tools` user manual for more details and Section 4.3.1 for the expected number of duplicates. The code for
415 this and all following steps in this chapter can be found in **Supplementary Data 1**.

4.2. Extraction of crosslinked nucleotides

After genomic mapping and deduplication, the mapped reads are transformed into crosslink events. As outlined above, the iCLIP sequencing reads
420 start precisely at the position where the cDNAs truncated during reverse transcription (see 1.2). Unlike for other types of RNA sequencing, we therefore want to retain only the position upstream of the 5' end of the reads (referred to as 'crosslinked nucleotide').

In order to perform this transformation with standard tools, we propose
425 the following workflow: First, the `bam` files are converted to `bed` files[3] using `bedtools bamtobed` of the `BEDTools` suit [18]. The `bed` files are then shifted by one base pair into 5' direction, such that afterwards only the 5' position of the new intervals can be extracted and piled up. The shift can be performed using `bedtools shift`, while the subsequent extraction and pile-up of the 5' ends of

[3]When inspecting the files, please keep in mind that `bed` and `bedgraph` files are UCSC file formats which represent genomic coordinates as 0-based half-open intervals. In contrast, `sam` files, the human-readable version of `bam` files, use 1-based closed intervals. For more information on the UCSC file formats, see `https://genome.ucsc.edu/FAQ/FAQformat.html`.

the intervals can be done using `bedtools genomecov`. The latter needs to be run separately for each strand.

The described procedure outputs the crosslink events as coverage tracks, which specify the number of crosslink events on each crosslinked nucleotide along the genome. These are stored e.g. in `bedgraph` file format. If wanted, the `bedgraph` coverage files can be further normalised to the overall library size (e.g. reads per million [RPM]) by providing a scaling factor to `bedtools genomecov`.

For some of the steps, a file of chromosome sizes is needed, with at least two columns specifying the chromosome name and its length. Such a file, named `genome.fasta.fai`, can easily be prepared using `SAMtools` [17] on the genome `fasta` file:

```
samtools faidx <genome.fasta>
```

Alternatively, a file of chromosome sizes is generated automatically when creating the `STAR` index. Its default name is `chrNameLength.txt`. In the following, we will refer to the file of chromosome sizes by `chromo.sizes`.

The following code implements the workflow described above:

1. Convert all read locations to intervals in `bed` file format

   ```
   bedtools bamtobed -i <sampleX.duprm.bam> > <sampleX.bed>
   ```

2. Shift intervals depending on the strand by 1 bp upstream

   ```
   bedtools shift -m 1 -p -1 -i <sampleX.bed>
   -g <chromo.sizes> > <sampleX.shifted.bed>
   ```

3. Extract the 5' end of the shifted intervals and pile up into coverage track in `bedgraph` file format (separately for each strand)

```
bedtools genomecov -bg -strand + -5 -i <sampleX.shifted.bed> -g
<chromo.sizes> > <sampleX.plus.bedgraph>
```

```
bedtools genomecov -bg -strand - -5 -i <sampleX.shifted.bed> -g
<chromo.sizes> > <sampleX.minus.bedgraph>
```

4. For RPM-normalised coverage tracks, use additional parameter `-scale` with $1,000,000/$`#mappedReads`, where `#mappedReads` is the total number of mapped reads in a given sample remaining after duplicate removal.

5. Optionally, the `bedgraph` files can be converted to `bw` files using `bedGraphToBigWig` of the `kentUtils` suite [19]:

   ```
   bedGraphToBigWig <sampleX.strand.bedgraph> <chromo.sizes>
   <sampleX.strand.bw>
   ```

   Depending on the system and the version of `bedGraphToBigWig`, it might be necessary to sort the `bedgraph` files before converting them to `bw` files. This can be done with the following commands:

   ```
   export LC_COLLATE=C
   sort -k1,1 -k2,2n <sampleX.strand.bedgraph> >
   <sampleX.strand.sorted.bedgraph>
   ```

4.3. Diagnostic plots and measures of library complexity

4.3.1. Summary of duplicate removal

The amount of technical PCR duplicates removed during deduplication reflects the quality of an iCLIP library and may inform on potential overamplification.

The following metrics inform on library complexity (visualised in Figure 7):

- Number of uniquely mapped reads, which can be extracted from the `STAR` [15] log files.

- Number of crosslink events, i.e. reads after duplicate removal:

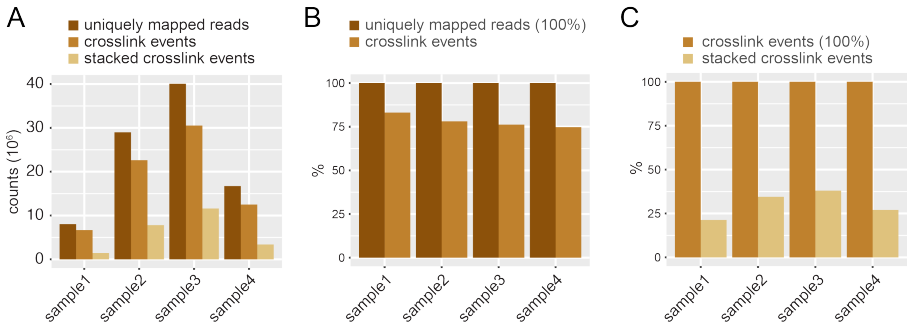


Figure 7: **Visualisation of PCR duplicate removal results** Barplots show the absolute numbers (**A**) and fraction (**B**) of reads that were uniquely mapped and kept after PCR duplicate removal, i.e. crosslink events. (**C**) Depiction of stacked crosslink events, i.e. crosslink events on positions with > 1 crosslink event, as a fraction of all crosslink events.

```
cat <sampleX.plus.bedgraph> <sampleX.minus.bedgraph> |
awk 'BEGIN{totalcount=0}{totalcount+=(($3-$2)*$4)}END{print totalcount}'
```

- Number of crosslinked nucleotides, i.e. positions harbouring crosslinked nucleotides (if both strands are covered, count as 2):

```
cat <sampleX.plus.bedgraph> <sampleX.minus.bedgraph> |
awk 'BEGIN{totalpos=0}{totalpos+=($3-$2)}END{print totalpos}'
```

- Number of stacked crosslink events, i.e. crosslink events on positions with > 1 crosslink event:

```
cat <sampleX.plus.bedgraph> <sampleX.minus.bedgraph> |
awk 'BEGIN{totalstackedcount=0}{if($4>1)
totalstackedcount+=(($3-$2)*$4)}END{print totalstackedcount}'
```

- Number of nucleotides with stacked crosslink events, i.e. positions with > 1 crosslink event:

```
cat <sampleX.plus.bedgraph> <sampleX.minus.bedgraph> |
awk 'BEGIN{totalstackedpos=0}{if($4>1)
totalstackedpos+=($3-$2)}END{print totalstackedpos}'
```

The numbers of uniquely mapped reads and crosslink events can be used

to calculate the removed reads, i.e. the number of technical PCR duplicates. Often, this is given as a percentage of all uniquely mapped reads. In the example shown in Figure 7, the percentage of removed PCR duplicates is between 17% and 25%.

When looking at different iCLIP libraries, we find strong variations in the duplication level. Nonetheless, whether to discard a library usually depends on the studied protein and the intended downstream analyses. Even if the majority of reads consists of PCR duplicates, an iCLIP sample can still capture the binding behaviour of an RBP well if sufficient reads remain. This is exemplified in a comparison of U2AF2 iCLIP signal at 3' splice sites based on three different iCLIP datasets [27, 30] with broad differences in duplication level ($<$ 25% to 70%) and read numbers after duplicate removal (18 million to 72 million reads; see Figure 13C below).

4.3.2. Reads with insertions and deletions

The iCLIP protocol relies on truncation of the reverse transcription reaction which is predominant under standard conditions [31, 2]. In the case of residual read-through events, the resulting reads often contain crosslink-induced mutations (CIMS) at the site of protein-RNA crosslinking, mainly in the form of insertions and deletions [32]. Although the real number of read-through reads cannot be determined, the frequency of insertions/deletions in the uniquely mapped reads can be used as a proxy to compare the incidence of read-through between iCLIP libraries (Figure 8).

In the following code example, insertions or deletions are counted with `bash` commands from the CIGAR strings in the `sam` files. The latter are created from `sampleX.duprm.bam` using `SAMtools`:

```
samtools view <sampleX.duprm.bam> -o <sampleX.duprm.sam>
```

- Number of reads mapped with deletions:

```
cut -f6 <sampleX.duprm.sam> | grep D | wc -l
```

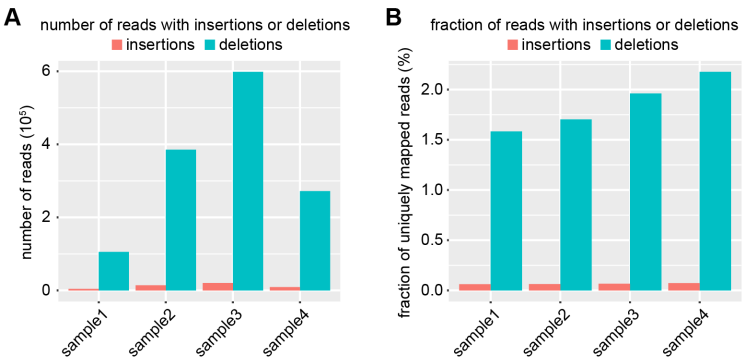


Figure 8: Visualisation of number or percentage of reads mapped with insertions and deletions.

- Number of reads mapped with insertions:

```
cut -f6 <sampleX.duprm.sam> | grep I | wc -l
```

4.3.3. iCLIPro analysis

A typical iCLIP experiment will result in the detection of RNA fragments of different lengths. The general assumption is that the sequencing reads accumulate downstream of the crosslink site [5], irrespective of the individual length of the underlying RNA fragment. However, previous studies found that for some iCLIP libraries, this interpretation may not hold, such as in case of substantial read-through or RNase overdigestion [20, 9]. In the latter situation, sequence constraints of the employed RNases may result in an increased accumulation of read ends in certain positions. As a consequence, if certain read lengths are underrepresented in the library, this results in an incomplete coverage of the associated binding sites. In this case, repeating the experiment to obtain a new iCLIP library is advisable.

The tool `iCLIPro` [20] can be used to detect coinciding read starts, centres or ends with respect to different fragment lengths. For a detailed description of the tool, please see [20]. Assuming a minimal read length of 15 nt and a maximal read length of 20 nt, a possible call of iCLIPro is as follows (keep in mind, while 15 nt is often set as minimal read length, a maximal read length of

20 nt is rather short and just used here for simplicity):

```
iCLIPro -r 50 -b 300 -f 30 -g "L15:15,L16:16,L17:17,L18:18,L19:19,R:20"
-p "L15-R,L16-R,L17-R,L18-R,L19-R" -o <outdir> <sampleX.duprm.bam>
```

An example `iCLIPro` output is shown in Supplementary Figure 1. Moreover, the same U2AF2 iCLIP dataset that we used in our present manuscript is shown in Figure 4C of Hauer *et al.* [20] as an example of a 'good' dataset and compared to other datasets with apparent biases.

5. Identification of RBP binding sites

A key step in iCLIP data analysis is the identification of *bona fide* RBP binding sites from the observed crosslink events. Substantial challenges arise from the strong dependence of the iCLIP signals on the underlying transcript abundance, which can range over several orders of magnitude. Moreover, datasets are prone to variable noise levels, which are influenced by the RNA binding behaviour of the studied RBP, but also by technical aspects, such as the efficiency of the immunoprecipitation. Once the RBP binding sites are defined, it is critical to correct for these biases to allow for a quantification of RBP binding (see Chapter 7). It is important to keep in mind that a clear definition of binding sites may not always be the best way to proceed, for instance for RBPs that show a widespread, promiscuous RNA binding [33]. This may also hold true for particularly noisy datasets, e.g. due to insufficient antibody specificity. In such cases, an analysis at the level of total crosslink events could be considered, such as testing the distribution of crosslink events across different transcript regions or relative to specific functional transcript positions, such as splice sites.

As a first step, we identify sites with significant crosslink signal from CLIP data, also known as 'peak calling'. The most common strategy is to fit a probability distribution to the crosslink event counts (e.g. a Poisson or negative binomial distribution) to identify sites that arise above the background signal with at least a given significance level (for more details on approaches and considerations, see a recent review by Chakrabarti *et al.* [29]). Various tools are available to perform peak calling on different types of CLIP data. Among these, `CLIPper` [34] was developed for the publicly available eCLIP datasets from the ENCODE consortium. In the following section, we use `PureCLIP` [21] which specifically models the characteristics and intrinsic biases of the truncation-based iCLIP and eCLIP data.

5.1. Peak calling with `PureCLIP`

In order to reliably detect positions with significant crosslink signal, `PureCLIP` trains a hidden Markov model based on the diagnostic truncation sites, i.e. the

crosslink sites, and the complete RBP-bound fragments. In order to correct for UV crosslinking biases, the model allows to incorporate crosslink-associated motifs which preferentially respond to UV irradiation and thereby often lead to false-positives [9]. In its latest version 1.3, `PureCLIP` also works with two replicate experiments.

`PureCLIP` takes as input the `bam` file of the deduplicated reads and the associated `bai` index file. The latter can be prepared using `SAMtools` [17] as described in Chapter 4.1. In order to boost the sensitivity of peak detection, we commonly merge replicate experiments for the peak calling step [30, 35] (and subsequently separate them again to assess reproducibility, see Section 6.1). Since the latest `PureCLIP` version 1.3 allows two replicates, the data can be pooled to two larger datasets of roughly equal size. `bam` files can be combined using

```
samtools merge -f <merged.bam> -b <list_of_bam_files>
```

In addition, `PureCLIP` requires a `genome.fasta` file of the reference genome. Option `-ld` allows for higher precision to compute emission probabilities with the drawback of higher memory consumption (and run time).

As output, `PureCLIP` provides individual 'crosslink sites', i.e. positions with enriched crosslink events. Since RBPs rarely bind to isolated nucleotides, adjacent crosslink sites within a certain distance (option `-nt`; default: ≤ 8 nt) are further merged into 'binding regions'. The parameters `-o` (here: `-o PureCLIP.crosslink_sites.bed`) and `-or` (here: `-or PureCLIP.crosslink_regions.bed`) specify the names of the output files for individual 'crosslink sites' and 'binding regions'.

`PureCLIP` can be run with the following command:

```
pureclip -i <merged.bam> -bai <merged.bam.bai> -g <genome.fasta>
-ld -nt 8 -o PureCLIP.crosslink_sites.bed
-or PureCLIP.crosslink_regions.bed
```

5.2. Postprocessing of binding sites

As mentioned above, `PureCLIP` merges adjacent crosslink sites within a certain distance into binding regions. It is anticipated that the width of these binding regions reflects the RNA binding footprint of the studied RBP. However, widths can vary considerably between different binding regions, which can impair their comparability in downstream analyses. For many applications, it is therefore advisable to resize the binding regions to obtain binding sites of a uniform width. The chosen width depends on the expected width of the RBP's footprint, which often relies on prior knowledge, such as the type of RNA binding domains. Estimates can also be deduced from the data, e.g. based on visual inspection of the RBP crosslink events in the genome browser, the spread of crosslink events around the maxima of the binding regions or the local k-mer enrichment (Figure 9A,B). In the case of U2AF2, we decided for 9-nt binding sites (i.e. 4 nt on either side of the summit position), which could accommodate the two adjacent RRM binding events [30].

In order to obtain equal-sized RBP binding sites, we suggest the following postprocessing steps. The code to perform the described procedure is detailed in Supplementary Data 2.

1. Starting from the `PureCLIP` output, the crosslink sites, i.e. positions with enriched crosslink events, are first clustered into regions. The maximum distance between crosslink sites to be clustered together into one region is chosen such that no binding sites will overlap after resizing. Practically, in our example, this means that when crosslink sites will later be extended by 4 nt to either side to obtain 9-nt binding sites, clustering together all crosslink sites with a distance $\leq$ 8 nt ensures that binding sites after resizing can touch but not overlap (`reduce(peaks_sc, min.gapwidth=8)`, see Supplementary Data 2).
2. Next, all regions shorter than 3 nt are removed with the intention to focus on binding sites with substantial signal over a minimum genomic region.
3. In the following step, equal-sized binding sites are assured by (i) extending

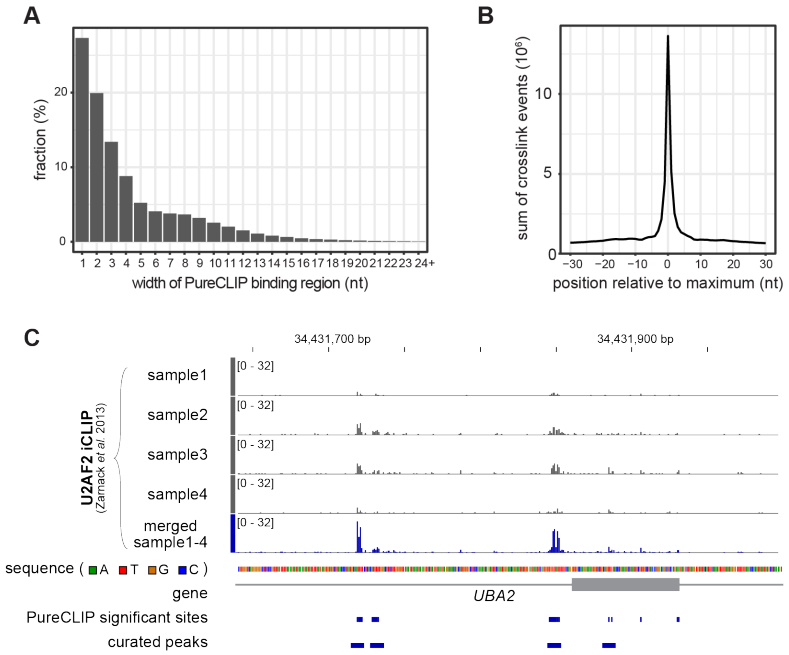


Figure 9: **Peak calling with `PureCLIP`. (A)** Width of initial 'binding regions' that are predicted by `PureCLIP` (`PureCLIP.crosslink_regions.bed`). **(B)** Metaprofile of crosslink events around the maxima of the merged binding regions. In order to estimate the footprint of U2AF2 binding in our iCLIP data, the merged binding regions were centred on the position with the highest number of crosslink events (maximum). **(C)** Genome browser view of the *UBA2* gene locus showing U2AF2 crosslink events from the four individual replicate experiments as well as the merged replicates as used for peak calling, the PureCLIP output ('significant sites') and the curated 9-nt U2AF2 binding sites after postprocessing.

too short binding sites and (ii) decomposing too long binding sites:

- Binding sites which are too short after merging ($\leq$ 8 nt) are centred on the position with the maximum signal and extended by 4 nt to either side of the maximum.
- Binding sites which already have the correct width (9 nt) are centred on the position with the maximum signal.
- Binding sites which are too long after merging ($>$ 9 nt) are split up into disjunctive binding sites of width 9 nt, such that binding sites with the highest summit signal are chosen first. We iteratively place all possible non-overlapping binding sites of width 9 nt, whose centre position is contained in the merged region.

4. Finally, we require at least three positions with crosslink signal within one binding site to assure sufficient support of the binding site.

In our example, peak calling with `PureCLIP` followed by the described post-processing steps yielded a total of 358,747 U2AF2 binding sites. An example region with the U2AF2 crosslink events in the four replicate experiments, the `PureCLIP` output and the derived U2AF2 binding sites after postprocessing is shown in Figure 9C.

6. Downstream analyses

This chapter describes common downstream analyses of the computed RBP binding sites, including the reproducibility between replicate experiments and the assignment to transcript regions. Example code for the major steps is provided in Supplementary Data 3.

6.1. Reproducibility of binding sites

As outlined in Chapter 5, we identify the binding sites based on the merge of all replicates to augment the signal for peak calling and postprocessing. This is followed up by a reproducibility filter to ensure that each binding site is sufficiently supported in the individual replicates. To this end, we compare the number of crosslink events that fall within a given binding site in each replicates in a quantitative manner (see below). As an alternative strategy to test the reproducibility of the identified binding sites, peak calling can also be performed separately on each replicate and the results then intersected in a qualitative comparison [29].

For implementation of the quantitative approach, we recommend the following procedure:

- Peak calling is performed on the merged signal from all replicates to increase the sensitivity, as described in Chapter 5.

- For each binding site, the number of crosslink events per replicate is determined. For a good-quality replicate, the resulting count data typically approximate a negative-binomial distribution (Figure 10). In the case of low-quality replicates, the distribution will be skewed towards lower count values.

- In order to integrate replicates of varying size, we determine the required minimum number of crosslink events individually for each replicate. The replicate-specific threshold is chosen based on a given percentile in the distribution of crosslink counts within the binding sites (Figure 10A). In the

present example, we used the count corresponding to the 10% percentile, meaning that only binding sites with the 90% highest signal enter from each replicate. More stringent thresholds are highlighted for comparison. We additionally applied a lower boundary to ensure that the determined threshold does not drop below a certain value (here: two crosslink events) even in low-read replicates, for which the 10% quantile would overestimate the signal.

- Finally, the minimum level of support that is demanded for a given binding site can be set depending on the number of available replicates and the stringency requirement of the study. In the present example, a binding site was deemed reproducible if the respective thresholds were met in at least three out of four replicates.

In our example, the reproducibility filter removed a total of 15.1% of the initially computed binding sites (54,340 out of 358,747; Figure 10B).

6.2. Annotation of genes and transcript regions

6.2.1. Gene assignment

Once the RBP binding sites are accurately defined, they are typically overlapped with existing gene annotations. It is important to note that depending on the source, annotations can differ in scope and reliability. For instance, ENSEMBL/GENCODE reports the full spectrum of putative isoforms [36], whereas NCBI RefSeq annotation provides a manually curated selection of transcripts [37]. In order to minimise redundancies, it can be useful to filter the annotations prior to any analysis. Here, we use gene annotations from GENCODE on the human genome that were filtered for gene support level ≤ 2 and a transcript support level ≤ 3 (Supplementary Data 3).

Next, the RBP binding sites are overlaid with the filtered gene annotations to assign each binding site to its host gene. Binding sites that do not overlap with any gene are defined as 'intergenic'. Due to the width of the binding sites and adjacent or overlapping annotations, binding sites can overlap with

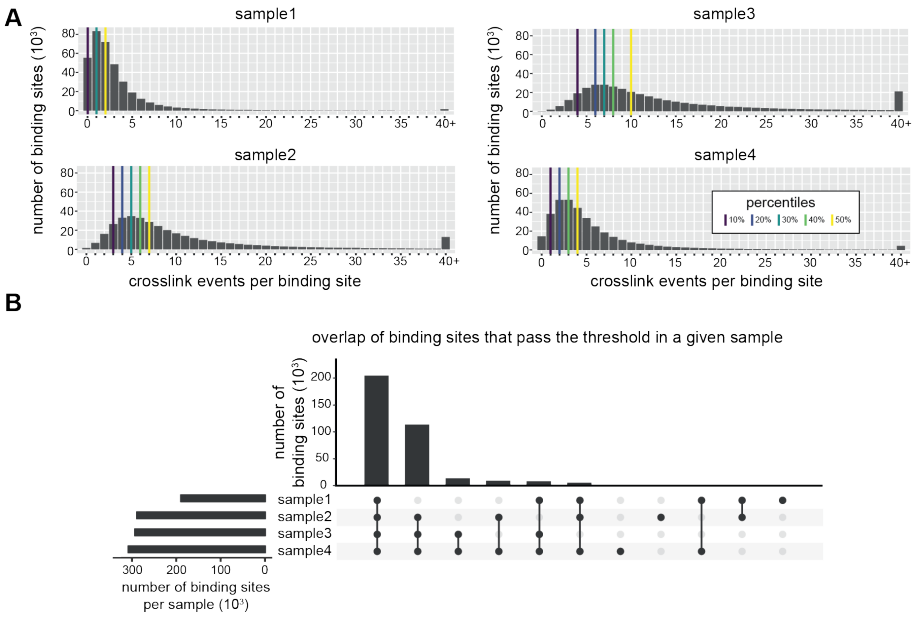


Figure 10: **Reproducibility between replicate experiments.** **(A)** Distribution of crosslink events per binding site in the four U2AF2 iCLIP replicates of our example dataset. 10-50% quantiles are indicated by vertical line. **(B)** Overview of the numbers of binding site that are shared between different replicates.

more than one gene. This can be resolved by several means. In a conservative approach, these binding sites are assigned to an additional category 'ambiguous' or completely removed. Alternatively, a predefined hierarchy can be applied, such that for instance, protein-coding genes are prioritised over non-coding RNA genes. However, care needs to be taken when deciding on these rules e.g. based on the expected behaviour of the studied RBP, as the choice can skew the resulting distribution. In the present example, overlapping annotations were not prevalent, and thus, we simply removed those binding sites.

Following the unique assignment of binding sites to distinct genes, we can visualise the target spectrum of the studied RBP (Figure 11). In our case, we observed that U2AF2 mainly binds to protein-coding genes.

6.2.2. Assignment to transcript regions

The annotations for most transcripts discriminate between introns and exons. In the case of protein-coding genes, the latter are further divided into 5' untranslated region (UTR), coding sequence and 3'UTR. In many cases, the transcript region that is preferentially bound by an RBP hints to its potential function. For instance, a binding preference within introns is compatible with a role in pre-mRNA processing, while many translational regulators bind in the 3'UTRs of their target mRNAs.

In more complex organisms like human, the majority of pre-mRNAs are alternatively spliced into multiple transcript isoforms. In most cases, it is difficult to trace back which transcript isoform was harbouring the RBP binding site. In a simplified approach, the binding sites are commonly assigned to distinct transcript regions. For this task, overlapping annotations need to be resolved. One solution is to choose a single reference isoform for each gene, such as the longest processed transcript or the most highly expressed according to orthogonal RNA-Seq data. If more than one transcript are considered per gene, ambiguous categories or hierarchies can be used, as before.

In our example, we follow a majority vote-based approach, such that a binding site is assigned to the type of transcript region that was most often over-

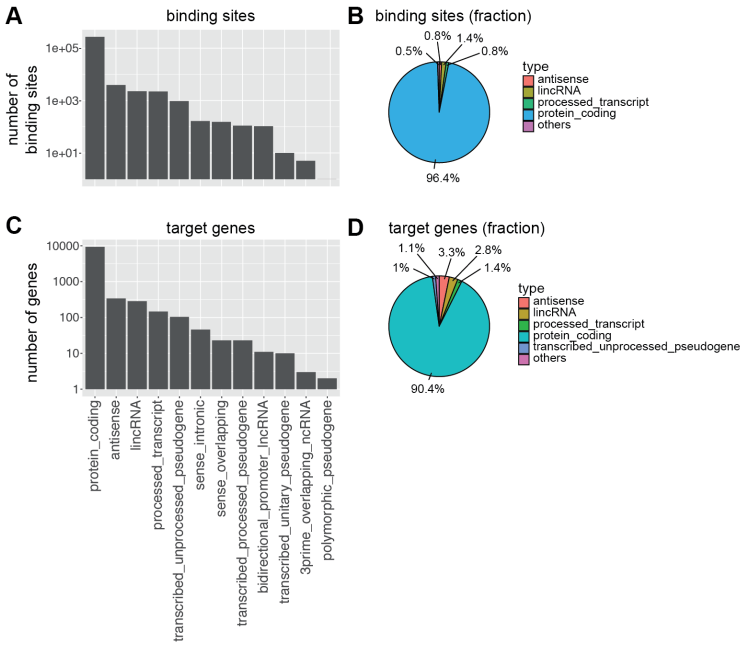


Figure 11: **Gene assignment. (A,B)** Binding site perspective. Shown are absolute number (A) and percent (B) of U2AF2 binding sites that fall within genes of a given type. **(C,D)** Target gene perspective. Shown are absolute number (A) and percent (B) of U2AF2 target genes (i.e. genes harbouring at least one U2AF2 binding site) of a given type. Note that minor categories were merged into 'other' in (B) and (D). Binding sites that overlap with multiple genes were removed from the analyses.

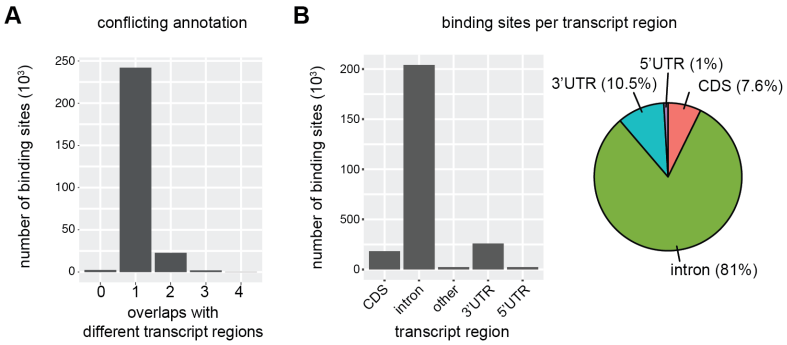


Figure 12: **Distribution of binding sites per transcript region. (A)** Conflict of overlapping annotations. Bar chart shows number of binding sites that overlap with zero, one or more different transcript regions. **(B)** Resolved overlaps after applying majority vote and hierarchy scheme. Total number (left) and percent (right) of binding sites that fall into a given transcript region. The additional category 'other' in the left plot collects binding sites that fall into a protein-coding gene, but overlap only with a non-coding transcript of this gene.

lapping. In the case of ties, we apply a hierarchy based on the most prevalent transcript regions. Using this approach, a total of 266,751 binding sites within protein-coding genes could be assigned to a specific transcript region in our example (Figure 12).

7. Estimation of binding site strength

The iCLIP signal is proportional to the binding site strength, but also to the abundance of the underlying transcript. It is therefore immanent to account for expression differences when comparing between binding sites. One possibility is to use orthogonal information from RNA-Seq data, which are commonly used to measure gene expression. The suitability of these data depends on the RBP of interest. For instance, poly(A)+ RNA-seq usually works well for cytoplasmic RBPs, whereas for a splicing regulator like U2AF2, neither poly(A)+ nor total RNA-Seq accurately informs about intronic regions in pre-mRNAs.

We suggest an alternative approach, in which the background iCLIP signal in the surrounding transcript region is taken as an expression proxy. The underlying assumption is that the dispersed background signal reflects low-affinity binding which should be largely invariant between transcripts and therefore scale with the underlying transcript abundance. Thus, the iCLIP signal in the binding site is normalised to this background signal to obtain the 'signal-to-background ratio' (SBR).

The SBR procedure critically depends on the choice of the region that is used to calculate the background signal. For instance, since introns and exons have a very different half-life in the cell, the background region should not cross exon-intron boundaries. Moreover, on protein-coding transcripts, the background signal tends to be generally lower within the open reading frame, possibly due to clearance of low-affinity binding sites by translating ribosomes.

In the following, we describe how we commonly derive background regions for U2AF2. Keep in mind that other RBPs may require different considerations.

1. Since U2AF2 is expected to bind to introns, each binding site is assigned to its 'hosting intron'. In order to simplify the analysis, we first filter the GENCODE annotation for transcripts with gene support level ≤ 2 and a transcript support level ≤ 3. For binding sites overlapping with more than one intron, we choose the intersection of all possible hosting introns.

2. Within the hosting intron, we calculate the background signal by summing up all crosslink events that do not fall into a binding site. Since we frequently observe that the iCLIP signal may reach beyond the curated 9-nt window, we remove an extended binding site region (± 5 nt).
3. For the normalised background signal, the sum of background crosslink events is divided by the width of the background region. Since a minimum background signal is required to obtain reliable estimates on binding site strength, we only move forward with binding sites in background regions with a normalised signal > 0.2 (i.e. at least 20 background crosslink events per 100 nt on average).
4. Finally, the 'signal-to-background ratio' (SBR) is calculated for each binding site as the number of crosslink events inside the binding site over the normalised background signal.

In our example, we tested the performance of the SBR normalisation by comparing raw iCLIP signal and SBR values of binding sites in genes with increasing expression levels (Figure 13). As expected, the mean number of crosslink events per binding site continuously rises with the underlying transcript abundance. The SBR normalisation compensates for this bias, resulting in an even distribution of the estimated binding site strengths across transcripts with different abundances. The SBR metric thereby facilitates comparisons between binding sites on different transcripts. We previously applied variations of this procedure to estimate the binding site strengths and to rank binding sites for motif analyses [30, 35].

Variant:

In a related approach, the local background signal can be be used to integrate the iCLIP signal from multiple regions in a metaprofile. Here, we normalise the signal on all nucleotide positions in the respective region, irrespective of whether they fall within or outside of a binding site. The normalisation accounts for differences in library size, balances out expression level differences of the underlying transcripts and ensures that all regions enter the metaprofile on a

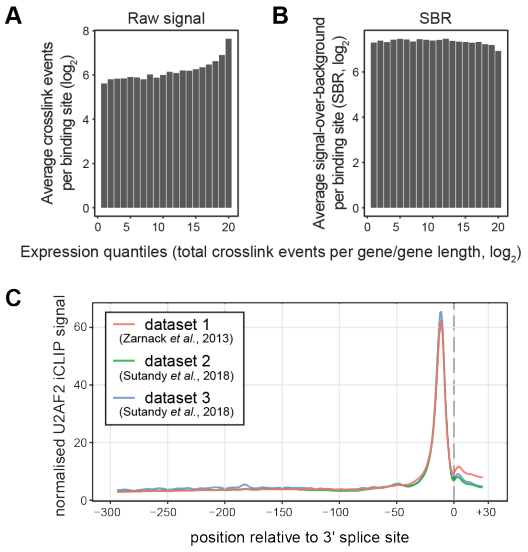


Figure 13: **Signal-to-background ratios allow to correct for transcript level biases in the iCLIP signal. (A)** The raw iCLIP signal follows the abundance of the underlying transcript. U2AF2-bound genes were stratified by the total number of crosslink events therein (normalised to gene length) into 20 bins, taken as a proxy for increasing expression. Shown is the average number of crosslink events per binding site for all binding sites in each bin. **(B)** SBR values are independent of the expression level of the underlying transcript. Average SBR values for all binding sites in each expression bin are shown as in (A). **(C)** Metaprofile of U2AF2 iCLIP signal at 3' splice sites ('RNAmap') from three independent datasets with different duplication level and size. All three datasets yield an almost identical U2AF2 binding profile, even though dataset 3 is considerably smaller. Comparison of summed normalised U2AF crosslink events from dataset 1 (taken from [27], used throughout this manuscript; merged replicates, <25% PCR duplicates, 72 million reads after duplicate removal), dataset 2 (library JKRS17 from [30], 30% PCR duplicates, 58 million reads), and dataset 3 (library JKRS13 from [30], 70% PCR duplicates, 18 million reads). Only introns with sufficient signal were taken into account (n = 21,813, 17,893 and 3,001 introns, respectively). All datasets were generated with the same amount of HeLa cells and anti-U2AF2 antibody.

comparable scale.

In order to illustrate this approach, we generated a metaprofile of summed crosslink events of U2AF2 binding in a 300-nt window upstream of exons (also known as 'RNAmap'; Figure 13C). We compared the dataset that we used in the present manuscript with two previously published U2AF2 iCLIP datasets [30]. To facilitate a direct comparison, we normalised the iCLIP signal separately within each dataset as follows: First, all regions were assigned to their hosting intron based on protein-coding transcripts (GENCODE annotation, gene support level$\leq$ 2, transcript support level $\leq$ 3). Introns were required to be $\geq$ 300 nt long and with $\geq$ 100 crosslink events in the 300 nt upstream of the 3' splice site to assure decent coverage (n = 21,813, 17,893 and 3,001 introns for dataset 1, 2 and 3, respectively). If multiple introns fulfilled these criteria and overlapped, the one with most signal in the 300 nt upstream of the 3' splice site was used. Within each intron, the crosslink events on all positions were then normalised by the total intron signal over intron length and multiplied by a scaling constant to obtain values in a reasonable range. Finally, the regions were aligned at the 3' splice site, and the normalised crosslink events were summed up on each positions and scaled for the number of contributing introns.

Importantly, after normalisation, the three datasets generate almost identical peaks of U2AF2 enrichment upstream of 3 splice sites (Figure 13C). This is particularly notable for dataset 3, in which just 3,001 introns harboured sufficient U2AF2 iCLIP signal to enter the analysis, compared to 21,813 and 17,893 introns in the other two datasets. The example analysis highlights how normalisation to the background signal allows for comparisons between datasets, even if major differences in library size and duplication level are present.

8. Acknowledgements

We would like to thank Dr. Julian König for fruitful discussions. This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) to Z.K. (FOR2333 and SFB902).

References

[1] J. Ule, K. B. Jensen, M. Ruggiu, A. Mele, A. Ule, R. B. Darnell, CLIP identifies Nova-regulated RNA networks in the brain, Science 302 (5648) (2003) 1212–1215.

[2] F. C. Y. Lee, J. Ule, Advances in CLIP Technologies for Studies of Protein-RNA Interactions, Mol. Cell 69 (3) (2018) 354–369.

[3] M. Hafner, M. Landthaler, L. Burger, M. Khorshid, J. Hausser, P. Berninger, A. Rothballer, M. Ascano, A. C. Jungkamp, M. Munschauer, A. Ulrich, G. S. Wardle, S. Dewell, M. Zavolan, T. Tuschl, Transcriptome-wide identification of RNA-binding protein and microRNA target sites by PAR-CLIP, Cell 141 (1) (2010) 129–141.

[4] S. Granneman, G. Kudla, E. Petfalski, D. Tollervey, Identification of protein binding sites on U3 snoRNA and pre-rRNA by UV cross-linking and high-throughput analysis of cDNAs, Proc. Natl. Acad. Sci. U.S.A. 106 (24) (2009) 9613–9618.

[5] J. König, K. Zarnack, G. Rot, T. Curk, M. Kayikci, B. Zupan, D. J. Turner, N. M. Luscombe, J. Ule, iCLIP reveals the function of hnRNP particles in splicing at individual nucleotide resolution, Nat. Struct. Mol. Biol. 17 (7) (2010) 909–915.

[6] E. L. Van Nostrand, G. A. Pratt, A. A. Shishkin, C. Gelboin-Burkhart, M. Y. Fang, B. Sundararaman, S. M. Blue, T. B. Nguyen, C. Surka, K. Elkins, R. Stanton, F. Rigo, M. Guttman, G. W. Yeo, Robust transcriptome-wide discovery of RNA-binding protein binding sites with enhanced CLIP (eCLIP), Nat. Methods 13 (6) (2016) 508–514.

[7] B. J. Zarnegar, R. A. Flynn, Y. Shen, B. T. Do, H. Y. Chang, P. A. Khavari, irCLIP platform for efficient characterization of protein-RNA interactions, Nat. Methods 13 (6) (2016) 489–492.

[8] B. Linder, A. V. Grozhik, A. O. Olarerin-George, C. Meydan, C. E. Mason, S. R. Jaffrey, Single-nucleotide-resolution mapping of m6A and m6Am throughout the transcriptome, Nat. Methods 12 (8) (2015) 767–772.

[9] N. Haberman, I. Huppertz, J. Attig, J. König, Z. Wang, C. Hauer, M. W. Hentze, A. E. Kulozik, H. Le Hir, T. Curk, C. R. Sibley, K. Zarnack, J. Ule, Insights into the design and interpretation of iCLIP experiments, Genome Biol. 18 (1) (2017) 7.

[10] I. Huppertz, J. Attig, A. D'Ambrogio, L. E. Easton, C. R. Sibley, Y. Sugimoto, M. Tajnik, J. König, J. Ule, iCLIP: protein-RNA interactions at nucleotide resolution, Methods 65 (3) (2014) 274–287.

[11] S. Andrews, `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`.

[12] `http://hannonlab.cshl.edu/fastx_toolkit/`.

[13] `https://github.com/lh3/seqtk`.

[14] J. T. Roehr, C. Dieterich, K. Reinert, Flexbar 3.0 - SIMD and multicore parallelization., Bioinformatics 33 (18) (2017) 2941–2942.

[15] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, T. R. Gingeras, STAR: ultrafast universal RNA-seq aligner, Bioinformatics 29 (1) (2013) 15–21.

[16] T. S. Smith, A. Heger, I. Sudbery, UMI-tools: Modelling sequencing errors in Unique Molecular Identifiers to improve quantification accuracy., Genome Res 27 (3) (2017) 491–499.

[17] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, The Sequence alignment/map (SAM) format and SAMtools., Bioinformatics 25 (16) (2009) 2078–9.

[18] A. R. Quinlan, I. M. Hall, BEDTools: a flexible suite of utilities for comparing genomic features., Bioinformatics 26 (6) (2010) 841–842.

[19] https://github.com/ENCODE-DCC/kentUtils.

[20] C. Hauer, T. Curk, S. Anders, T. Schwarzl, A.-M. Alleaume, J. Sieber, I. Hollerer, M. Bhuvanagiri, W. Huber, M. W. Hentze, A. E. Kulozik, Improved binding site assignment by high-resolution mapping of RNA-protein interactions using iCLIP., Nat Commun 6 (2015) 7921.

[21] S. Krakau, H. Richard, A. Marsico, PureCLIP: capturing target-specific protein-RNA interaction footprints from single-nucleotide CLIP-seq data, Genome Biol. 18 (1) (2017) 240.

[22] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria (2018).
URL https://www.R-project.org/

[23] M. Lawrence, W. Huber, H. Pages, P. Aboyoun, M. Carlson, R. Gentleman, M. T. Morgan, V. J. Carey, Software for computing and annotating genomic ranges, PLoS Comput. Biol. 9 (8) (2013) e1003118.

[24] M. Lawrence, R. Gentleman, V. Carey, rtracklayer: an R package for interfacing with genome browsers, Bioinformatics 25 (14) (2009) 1841–1842.

[25] H. Wickham, ggplot2: Elegant Graphics for Data Analysis, Springer-Verlag New York, 2016.

[26] E. Afgan, D. Baker, B. Batut, M. van den Beek, D. Bouvier, M. Cech, J. Chilton, D. Clements, N. Coraor, B. A. Gruning, A. Guerler, J. Hillman-Jackson, S. Hiltemann, V. Jalili, H. Rasche, N. Soranzo, J. Goecks, J. Taylor, A. Nekrutenko, D. Blankenberg, The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update, Nucleic Acids Res. 46 (W1) (2018) W537–W544.

[27] K. Zarnack, J. König, M. Tajnik, I. Martincorena, S. Eustermann, I. Stevant, A. Reyes, S. Anders, N. M. Luscombe, J. Ule, Direct competition between hnRNP C and U2AF65 protects the transcriptome from the exonization of *Alu* elements, Cell 152 (3) (2013) 453–466.

[28] C. A. Davis, B. C. Hitz, C. A. Sloan, E. T. Chan, J. M. Davidson, I. Gabdank, J. A. Hilton, K. Jain, U. K. Baymuradov, A. K. Narayanan, K. C. Onate, K. Graham, S. R. Miyasato, T. R. Dreszer, J. S. Strattan, O. Jolanki, F. Y. Tanaka, J. M. Cherry, The Encyclopedia of DNA elements (ENCODE): data portal update., Nucleic Acids Res 46 (Database issue) (2018) D794–D801.

[29] A. Chakrabarti, N. Haberman, A. Praznik, N. Luscombe, J. Ule, Data Science Issues in Studying Protein-RNA Interactions with CLIP Technologies, Annu. Rev. Biomed. Data Sci. 1 (2018) 235–261.

[30] F. X. R. Sutandy, S. Ebersberger, L. Huang, A. Busch, M. Bach, H. S. Kang, J. Fallmann, D. Maticzka, R. Backofen, P. F. Stadler, K. Zarnack, M. Sattler, S. Legewie, J. König, In vitro iCLIP-based modeling uncovers how the splicing factor U2AF2 relies on regulation by cofactors, Genome Res. 28 (5) (2018) 699–713.

[31] E. L. Van Nostrand, A. A. Shishkin, G. A. Pratt, T. B. Nguyen, G. W. Yeo, Variation in single-nucleotide sensitivity of eCLIP derived from reverse transcription conditions, Methods 126 (2017) 29–37.

[32] C. Zhang, R. B. Darnell, Mapping in vivo protein-RNA interactions at single-nucleotide resolution from HITS-CLIP data, Nat. Biotechnol. 29 (7) (2011) 607–614.

[33] B. Rogelj, L. E. Easton, G. K. Bogu, L. W. Stanton, G. Rot, T. Curk, B. Zupan, Y. Sugimoto, M. Modic, N. Haberman, J. Tollervey, R. Fujii, T. Takumi, C. E. Shaw, J. Ule, Widespread binding of FUS along nascent RNA regulates alternative splicing in the brain, Sci Rep 2 (2012) 603.

[34] M. T. Lovci, D. Ghanem, H. Marr, J. Arnold, S. Gee, M. Parra, T. Y. Liang, T. J. Stark, L. T. Gehman, S. Hoon, K. B. Massirer, G. A. Pratt, D. L. Black, J. W. Gray, J. G. Conboy, G. W. Yeo, Rbfox proteins regulate alternative mRNA splicing through evolutionarily conserved RNA bridges, Nat. Struct. Mol. Biol. 20 (12) (2013) 1434–1442.

[35] A. Hildebrandt, M. Brüggemann, C. Rücklé, S. Boerner, J. B. Heidelberger, A. Busch, H. Hänel, A. Voigt, M. M. Möckel, S. Ebersberger, A. Scholz, A. Dold, T. Schmid, I. Ebersberger, J. Y. Roignant, K. Zarnack, J. König, P. Beli, The RNA-binding ubiquitin ligase MKRN1 functions in ribosome-associated quality control of poly(A) translation, Genome Biol. 20 (1) (2019) 216.

[36] A. Frankish, M. Diekhans, A. M. Ferreira, R. Johnson, I. Jungreis, J. Loveland, J. M. Mudge, C. Sisu, J. Wright, J. Armstrong, I. Barnes, A. Berry, A. Bignell, S. Carbonell Sala, J. Chrast, F. Cunningham, T. Di Domenico, S. Donaldson, I. T. Fiddes, C. Garcia Giron, J. M. Gonzalez, T. Grego, M. Hardy, T. Hourlier, T. Hunt, O. G. Izuogu, J. Lagarde, F. J. Martin, L. Martinez, S. Mohanan, P. Muir, F. C. P. Navarro, A. Parker, B. Pei, F. Pozo, M. Ruffier, B. M. Schmitt, E. Stapleton, M. M. Suner, I. Sycheva, B. Uszczynska-Ratajczak, J. Xu, A. Yates, D. Zerbino, Y. Zhang, B. Aken, J. S. Choudhary, M. Gerstein, R. Guigo, T. J. P. Hubbard, M. Kellis, B. Paten, A. Reymond, M. L. Tress, P. Flicek, GENCODE reference annotation for the human and mouse genomes, Nucleic Acids Res. 47 (D1) (2019) D766–D773.

[37] N. A. O'Leary, M. W. Wright, J. R. Brister, S. Ciufo, D. Haddad, R. McVeigh, B. Rajput, B. Robbertse, B. Smith-White, D. Ako-Adjei, A. Astashyn, A. Badretdin, Y. Bao, O. Blinkova, V. Brover, V. Chetvernin, J. Choi, E. Cox, O. Ermolaeva, C. M. Farrell, T. Goldfarb, T. Gupta, D. Haft, E. Hatcher, W. Hlavina, V. S. Joardar, V. K. Kodali, W. Li, D. Maglott, P. Masterson, K. M. McGarvey, M. R. Murphy, K. O'Neill, S. Pujar, S. H. Rangwala, D. Rausch, L. D. Riddick, C. Schoch, A. Shkeda, S. S. Storz, H. Sun, F. Thibaud-Nissen, I. Tolstoy, R. E. Tully, A. R. Vatsan, C. Wallin, D. Webb, W. Wu, M. J. Landrum, A. Kimchi, T. Tatusova, M. DiCuccio, P. Kitts, T. D. Murphy, K. D. Pruitt, Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation, Nucleic Acids Res. 44 (D1) (2016) D733–745.

9. Supplementary Material

9.1. Optional reformatting of read names after `flexbar`

```
zcat <sampleX.fasta.gz> |

  awk '{if(FNR%4==1) {
            if (NF==1) {
                print;
            } else {
                n=split($NF,v,"_");
                printf "%s_%s",$1,v[n];
                for (i=2;i<NF;i++) {
                    printf " %s",$i;
                }
                printf " ";
                for (i=1;i<(n-1);i++) {
                    printf "%s_",v[i];
                }
                print v[n-1];
            }
        } else {
            print
        }
  }' | gzip > <sampleX.fasta.gz>
```

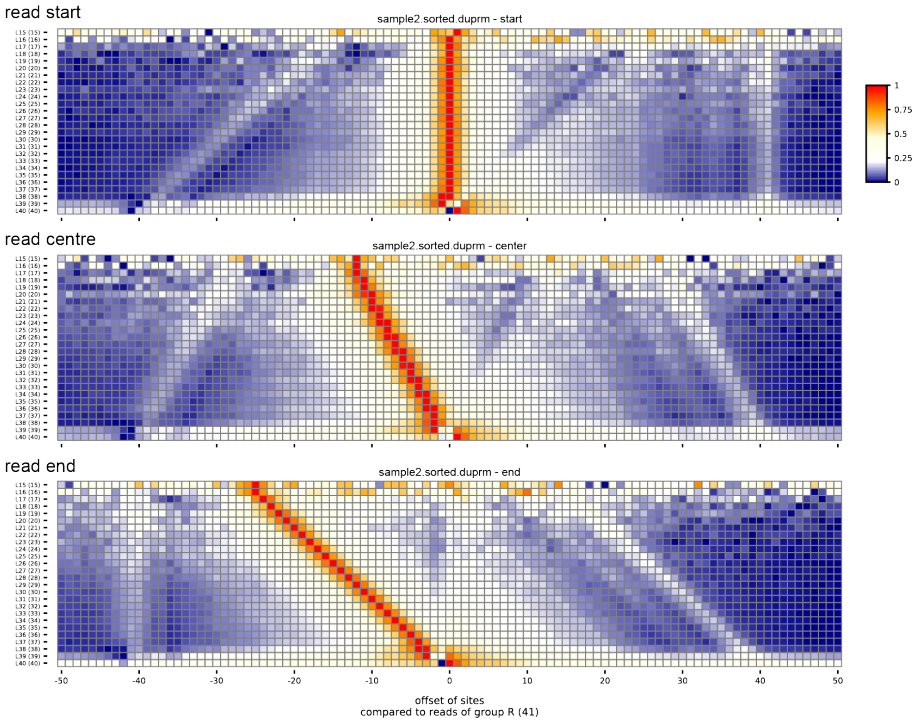
9.2. Supplementary Data Files

Supplementary Data 1: `Bash` code for all steps from raw reads until peak calling. This file provides the `bash` code for basic read processing, conversion into crosslink events and peak calling as described in Chapters 3-5.1. Details on the input files, preset variables and external tools required to run this code are listed in Chapters 3.1 and 2.1.

Supplementary Data 2: `R` code for postprocessing of `PureCLIP` output. This file provides the `R` code to postprocess the output of `PureCLIP` peak calling as described in Chapter 5.2. Running this code requires a bed file with 'crosslink sites' output by `PureCLIP` (`PureCLIP.crosslink_sites_short.bed`) and `bw` files with crosslink events (`sampleX.strand.bw`). The code to obtain these files is described in Supplementary Data 1 and Chapter 4.2.

Supplementary Data 3: `R` code for reproducibility and downstream analysis. This file provides the `R` code to reproducibility analyses and assignment of genes and transcript regions as described in Chapters 6.1 and 6.2. Running this code requires a `bed` file with binding sites (e.g. curated `PureCLIP` output, see Chapter 5), a `gtf` file with gene/transcript annotations, and `bw` files with crosslink events in the individual replicates (`sampleX.strand.bw`, see Chapter 4.2).

9.3. Supplementary Figure



Supplementary Figure 1: High-resolution read overlap heatmaps for sample 2 in our dataset, generated with `iCLIPro`. x-axis shows distance of start, centre and end positions of iCLIP reads that were trimmed to a given length (y-axis) relative to the start positions of reads that were not trimmed as reference positions (41-nt length, see Figure 5). In absence of read-end constraints, most reads starts align at the reference positions, irrespective of their length [9], whereas read centres and ends form a diagonal reflecting the read length after trimming. For more details on the underlying analysis steps and the visualisation, see [20] and `http://www.biolab.si/iCLIPro/doc/`.