Supplementary Note

Analysis of single end RNA-seq data

For single-end, bulk RNA-seq (Supplementary Figure 3A), TECtool first generates the required directories, optionally trims the 3' adapters using cutadapt [51] (v1.13)

```
cutadapt \
--adapter {3' adapter} \
--error-rate {error rate} \
--minimum-length {minimum read length} \
--overlap {overlap} \
{input reads} \
| gzip > {output reads}
```

and indexes the genome with the STAR software [52] (v2.5.3a).

```
STAR \
--runMode genomeGenerate \
--sjdbOverhang {read length} \
--genomeDir {genome dir} \
--genomeFastaFiles {genome fasta file} \
--runThreadN {number of threads} \
--sjdbGTFfile {annotation file}
```

The reads are mapped to the genome with the STAR aligner.

```
STAR \
--runMode alignReads \
--twopassMode Basic \
--runThreadN {number of threads} \
--genomeDir {genome dir} \
--sjdbGTFfile {annotation file} \
--readFilesIn {sample fastq file} \
--readFilesCommand zcat \
--outFileNamePrefix {prefix} \
--outSAMtype BAM Unsorted
```

Alignment files (in BAM format) are then sorted

```
samtools sort \
-@ {number of threads} \
{input bam} > {output bam}
```

and indexed

```
samtools index {input bam}
```

using samtools [53] (v1.3).

The sorted alignment file and appropriate input options are provided to TECtool

```
tectool \
--annotation {input annotation file} \
--polyasites {input polya sites} \
--bam {input alignment file} \
--sequencing_direction {sequencing direction option} \
--genome {genome fasta file} \
--minimum_spliced_reads_for_cryptic_exon_start_site 5 \
--output_dir {output directory}
```

to identify novel terminal exons and output an enriched annotation file (in gtf format). The gtf files from different replicates are then merged

```
tectool_add_novel_transcripts_to_gtf_file \
--list_of_gtf_files {list of gtf files} \
--out-dir {output directory}
```

into a single gtf file with a custom TECtool script. A file (fasta format) of transcript sequences is generated based on the annotation file with the gffread script from the cufflinks package [23] (v2.2.1).

```
gffread \
{merged annotation file} \
-g {genome fasta file} \
-w {transcripts fasta file}
```

Finally, the transcriptome is indexed with Salmon [47] (v0.9.1)

```
salmon index \
--transcripts {input transcript sequences} \
--index {output index} \
--kmerLen {kmer length} \
--keepDuplicates \
--threads {number of threads}
```

and the expression levels of transcripts in each replicate are quantified with Salmon.

```
salmon quant \
--index {input index} \
--libType {library type} \
--unmatedReads {input reads} \
--seqBias \
--geneMap {merged annotation file} \
--fldMean {mean fragment length} \
--fldSD {standard deviation of fragment length} \
--useVBOpt \
--threads {number of threads} \
--output {output directory}
```

We used Salmon for isoform quantification (the successor of Sailfish [54]), because we found Sailfish to perform well relative to other methods for isoform quantification [55]. Similar results for Salmon were reported by others [56]. Other methods for transcript quantification can also be used. The output of this final step consists in transcript and gene expression estimates (in TPM).

Analysis of paired-end RNA-seq data

For paired-end bulk RNA-seq (Supplementary Figure 3A) TECtool first selects one of the two mates depending on a user-set parameter and then continues as described above. However, adapter trimming

```
cutadapt \
-a {3' adapter mate 1} \
-A {3' adapter mate 2} \
--error-rate {error rate} \
--minimum-length {minimum read length} \
--overlap {overlap} \
-o {mate 1 output reads} \
-p {mate 2 output reads} \
{mate 1 input reads} {mate 2 input reads}
```

and the transcript expression estimation

```
salmon quant \
--index {input index} \
--libType {library type} \
-1 {mate 1 input reads} -2 {mate 2 input reads} \
--seqBias \
--geneMap {merged annotation file} \
--useVBOpt \
--threads {number of threads} \
--output {output directory}
```

is performed in paired-end and not single-end mode.

Analysis of single cell sequencing data

Single cell sequencing data (Supplementary Figure 6) poses the additional challenge that the coverage of individual genes is generally sparse and non-uniform. Thus, after the necessary directories are created, the genome is indexed with STAR and 3' adapters are trimmed with cutadapt, the read files corresponding to individual cells are concatenated, the reads are mapped to the genome with STAR and the alignment file is sorted and indexed with samtools. PCR duplicates are removed with samtools

```
samtools rmdup \
-s {input alignment file} \
{output alignment file}
```

and the resulting alignment file is provided as input to TECtool to identify novel exons and transcripts. Reads from individual cells are mapped to the genome with STAR and sorted with samtools. PCR duplicates are also removed from the alignment using samtools and then indexed. To quantify expression of transcripts that indeed include specific terminal exons, in spite of the sparse coverage of genes by reads in individual cells, we estimated the expression of novel and annotated transcripts as the number of split reads that fall in the 5' splice junction of the respective exons (novel or annotated terminal exons that do not overlap with annotated internal exons).

Analysis of TECtool running time

Data from a time series of mouse T cell activation (accession GSE52260) were merged and mapped to the genome with STAR. After sorting the alignment file and removal of PCR duplicates with samtools, we kept only primary alignments

```
samtools view \
-F 0x100 \
-bS {input alignment file} \
> {output alignment file}
```

and generated subsets representing 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, and 90% of the total number of reads

```
samtools view \
-s {subset fraction size} \
-b {input alignment file} \
> {output alignment file}
```

with samtools. We applied TECtool to each of the datasets, using the bash command time to obtain the running time. We repeated the procedure 10 times for each data set size to obtain averages and standard deviations of the running time for each data set size. All samples were run in a single core with 64GB of memory. The results are shown in Supplementary Figure 4B.

Analysis of ribosome profiling data

For the analysis of translation, we only considered novel terminal exons containing a stop codon. We mapped ribosome-protected reads to the genome with STAR and the parameters used for bulk RNA-seq data. We counted mapped reads HTSeq [9], constructed profiles of ribosome footprints around the stop codon and estimate the density of ribosomes over the terminal exons. Estimates of transcript abundance (transcripts per million, TPM) from Salmon were used to normalize Ribo-seq read densities (expressed in reads per kilobase per million, RPKM). The profile of ribosome footprints in intronic regions have been normalized to the expression level of the most expressed isoform of that gene (TPM), inferred with Salmon.

Transcript and terminal exon reconstruction with StringTie

StringTie [10] (v1.3.3) has been found most accurate among transcript reconstruction methods in a recent benchmarking study [24]. We mapped RNA-seq reads obtained in a previous study [20] to the genome in paired end mode with the STAR aligner

```
STAR \
--runMode alignReads \
--twopassMode Basic \
--runThreadN {number of threads} \
--genomeDir {genome dir} \
--sjdbGTFfile {annotation file} \
--readFilesIn {sample mate 1 fastq file} {sample mate 2 fastq file} \
--readFilesCommand zcat \
--outFileNamePrefix {prefix} \
--outSAMtype BAM Unsorted \
--outSAMstrandField intronMotif
```

After sorting and indexing the alignments with samtools, we ran StringTie with the following command

```
stringtie \
{input alignment file} \
-G {annotation file} \
-o {output annotation file} \
{library type} \
-p {threads}
```

to generate gtf files with the new annotation. Finally, we extracted novel terminal exons that were located in introns relative to the support level 1-5 transcript annotation of the genome.

Transcript and terminal exon reconstruction with Cufflinks

Cufflinks (v2.2.1) [23] is a second transcript reconstruction method with relatively good performance in a recent benchmarking [24]. We used the alignments of reads to genome obtained as described for StringTie, and after sorting and indexing with samtools we ran Cufflinks with the following options,

```
cufflinks \
--num-threads {number of threads} \
-g {annotation file} \
--library-type {library type} \
-o {output directory} \
{input alignment file}
```

obtaining a gtf file with novel and known transcripts for each sample. From transcripts with at least two exons, we then extracted terminal exons that were located in introns relative to the TSL1-5 annotation, using custom scripts and bedtools [40].

Parallel analysis of long and short read data

We used cutadapt [51] to trim 5' and 3' adapters, polyA and polyT stretches from PACbio reads, with the following commands:

```
cutadapt \
--front {5' adapter} \
--error-rate {error rate} \
--minimum-length {minimum read length} \
--overlap {overlap} \
{input reads} \
| gzip > {output reads}
```

and

```
cutadapt \
--adapter {3' adapter} \
--error-rate {error rate} \
--minimum-length {minimum read length} \
--overlap {overlap} \
{input reads} | gzip > {output reads}
```

We indexed the genome with the STAR software (v2.5.3a),

```
STARlong \
--runMode genomeGenerate \
--sjdbOverhang {read length} \
--genomeDir {genome dir} \
--genomeFastaFiles {genome fasta file} \
--runThreadN {number of threads} \
--sjdbGTFfile {annotation file}
```

and mapped the long, PACbio reads to the genome with the STARlong version of the STAR aligner.

```
STARlong \
--runMode alignReads \
--outFilterMultimapScoreRange 20 \
--outFilterScoreMinOverLread 0 \
--outFilterMatchNminOverLread 0.5 \
--outFilterMismatchNmax 1000 \
--winAnchorMultimapNmax 200 \
--seedSearchStartLmax 50 \
--seedPerReadNmax 100000 \
--seedPerWindowNmax 100 \
--alignTranscriptsPerReadNmax 100000 \
--alignTranscriptsPerWindowNmax 10000 \
--genomeSAsparseD 4 \
--outSAMunmapped Within \
--runThreadN {threads} \
--genomeDir {genome dir} \
--sjdbGTFfile {annotation file} \
--readFilesIn {input reads} \
--readFilesCommand zcat \
--outFileNamePrefix {prefix} \
--outSAMtype BAM Unsorted
```

We sorted

```
samtools sort \
-@ {number of threads} \
{input bam} \
> {output bam}
```

and indexed

```
samtools index {input bam}
```

the alignments with samtools [53]. After trimming the soft clipped parts of the mapped sequences,

```
samtools view -h {input alignment file in bam format} \
| awk 'BEGIN {OFS="\t"} { \
  split($6,C,/[0-9]*/); \
  split($6,L,/[SMDIN]/); \
  if (C[2]=="S"){ \
            $10=substr($10,L[1]+1); \
            $11=substr($11,L[1]+1); \
  }; \
  if (C[length(C)]=="S") { \
            L1=length($10)-L[length(L)-1]; \
            $10=substr($10,1,L1); \
            $11=substr($11,1,L1); \
  }; \
  gsub(/[0-9]*S/,"",$6); \
  print \
  }' \
  | samtools view -bS - > {output alignment file in bam format}
```

we extracted transcript coordinates from the alignment file with bedtools (v2.26.0)

```
bedtools bamtobed \
-i {alignment file} \
-bed12 \
-splitD \
> {bed12 coordinates file}
```

to generate a bed12 file, and the extracted transcript sequences from the genome

```
bedtools getfasta \
-split \
-s \
-name \
-fi {genome sequence} \
-bed {transcript coordinates} \
-fo {transcript sequences}
```

In parallel we generated enriched annotation files by applying TECtool to the short read data also generated for the respective samples. We then mapped the PACbio-sequenced transcripts to annotated and novel transcripts with blast [57], first building the index of annotated and novel transcripts

```
makeblastdb \
-in {transcript sequences} \
-dbtype nucl \
-out {Blast database name}
```

and then running BLAST [58] 2.6.0+

```
blastn \
-num_threads {threads} \
-db {db_prefix} \
-query {PACbio extracted transcripts} \
-outfmt \
"6 qseqid sseqid pident qlen length slen mismatch gapopen evalue bitscore" \
-out {blast result}
```

From the blast output files we kept only transcripts that did not contain any internal gaps (gapopen=0), for which the alignment length (length) did not differ by more than 40 nts from either query (qlen) and target (slen) sequence lengths. This allows only small differences in the initiation/termination sites, but not incorrect assignment of splice variants.

References

51. Martin, M. Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnet.journal 17, 10–12 (2011).

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

53. Li, H. et al. The Sequence Alignment/Map format and SAMtools. Bioinformatics 25, 2078–2079 (2009).

54. Patro, R., Mount, S. M. & Kingsford, C. Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. Nat. Biotechnol. 32, 462–464 (2014).

55. Kanitz, A. et al. Comparative assessment of methods for the computational inference of transcript isoform abundance from RNA-seq data. Genome Biol. 16, 150 (2015).

56. Teng, M. et al. A benchmark for RNA-seq quantification pipelines. Genome Biol. 17, 74 (2016).

57. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. J. Mol. Biol. 215, 403–410 (1990).

58. Camacho, C. et al. BLAST+: architecture and applications. BMC Bioinformatics 10, 421 (2009).