Restrander: rapid orientation and QC of long-read cDNA data

Jakob Schuster, Matthew Ritchie, Quentin Gouil

Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, Victoria 3052, Australia

Principle

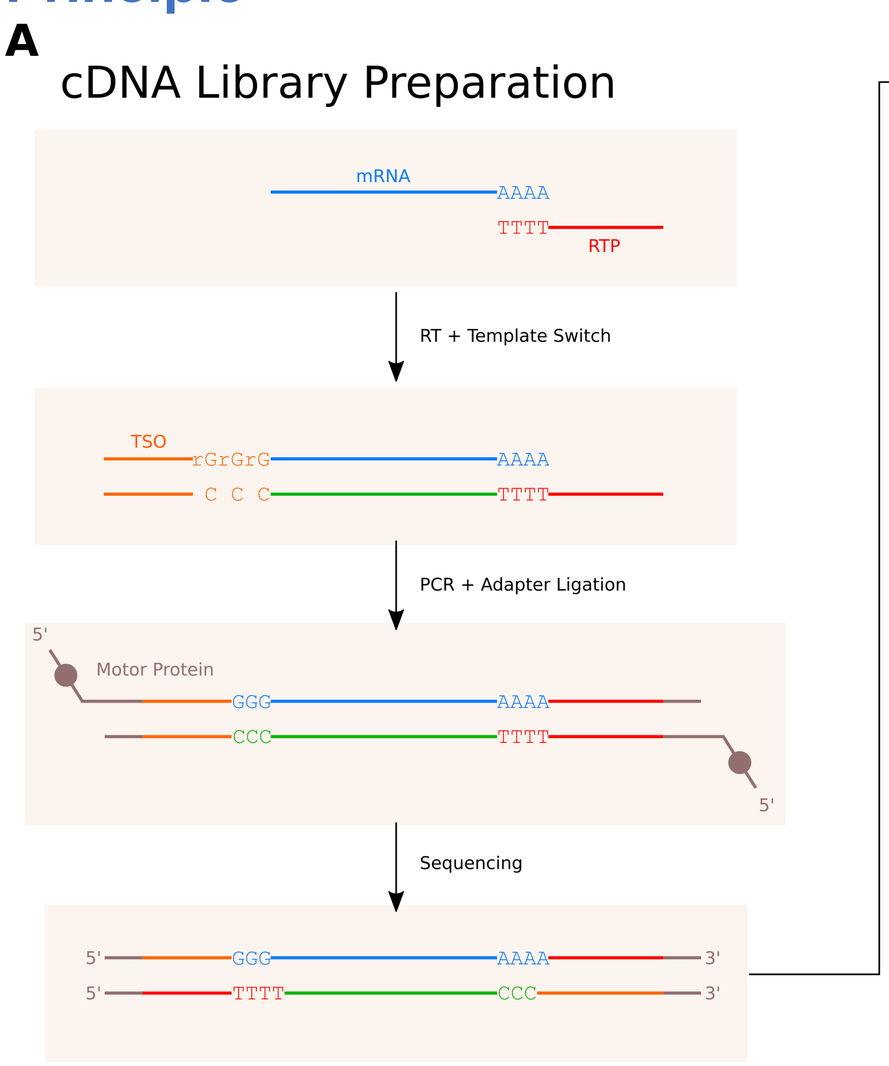

Figure 1: (A) Schematic of PCR-cDNA library preparation and sequencing. Forward reads are characterised by a lead TSO (template switch oligo) sequence, and a polyA-tail followed by the reverse complement of the reverse transcription primer (RTP). Reverse reads lead with RTP, a polyT stretch, and end with the reverse complement of the TSO. Different protocols introduce variations but the principle remains the same. (B) The default processing pipeline used by Restrander to classify the direction of one read. If a read can be restranded using only the easily located polyA/T-tails, no further processing need take place. This optimisation considerably speeds up the process without significantly impacting number of incorrect classifications.

Introduction

When performing gene expression analysis on RNA-sequencing data, preserving information about the direction of the RNA transcript is important, especially in areas of overlapping transcription. In direct-cDNA and PCR-cDNA protocols, either the first or second-strand cDNAs may be sequenced, resulting in both forward and reverse reads that have to be 'restranded' bioinformatically. Here, we introduce Restrander, a lightning-fast, highly accurate and user-friendly tool for restranding cDNA sequencing data.

Software details

Restrander was written with a focus on excellent computational performance, written in optimised C++. The pipeline of read orientation and artefact detection steps can be customised via a JSON configuration file. Most popular protocols are supported out-of-the-box, and support for new primer sequences can easily be configured by users. Restrander is available at *https://github.com/jakob-schuster/restrander*.

Supported Protocols	
PCB109	✓
PCB111	✓
NEBNext Single-Cell/Low Input	✓
10X Genomics Single-Cell 5' / 3'	✓
Additional Protocols	*User-configurable*

Results

Computational performance

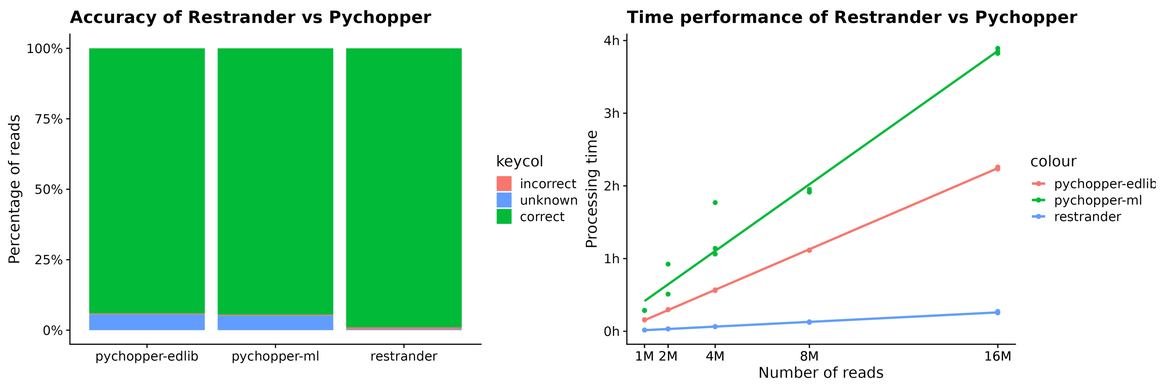

Figure 2: Comparison of time performance of Restrander against Pychopper, using both of Pychopper's computational backends. Due to its implementation in C++ and improved method which searches for polyA/T in addition to TSO/RTP, Restrander outperforms Pychopper in both domains.

In a typical PromethION flow cell sequencing run of 150M reads, it is estimated to take Pychopper's fastest backend roughly 21 hours to fully restrand the data. Restrander would take only 2 hours and 20 minutes, making restranding much less costly.

Isoform Detection

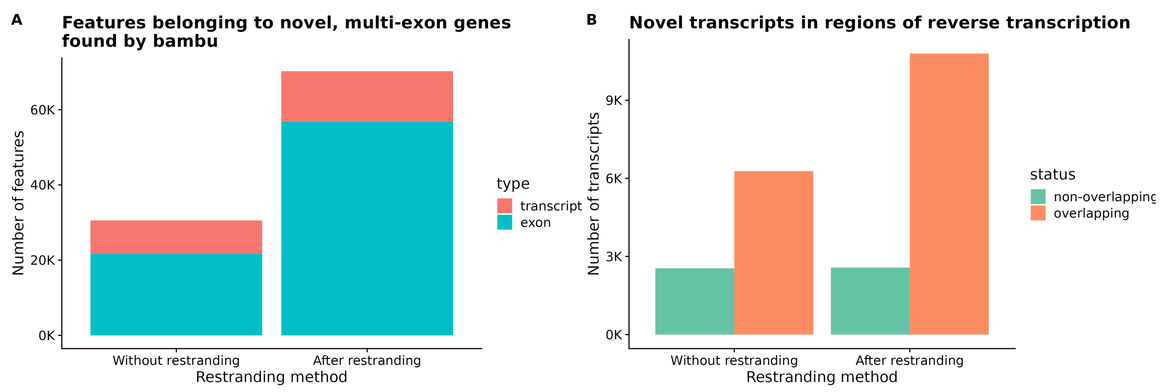

Figure 4: Number of transcripts found by bambu originating from novel, multi-exon genes, comparing non-stranded and restranded data, and quantifying the number of transcripts which overlap with other transcripts on the opposite strand, as an indicator of sense and antisense transcript co-occurrence.

Artefact Detection

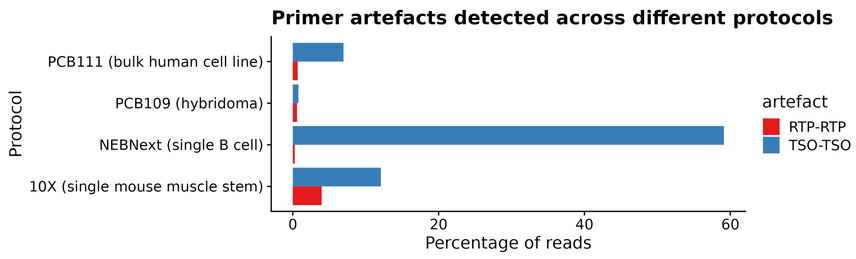

Figure 3: Since we are searching for TSO and RTP sequences, we can also detect and quantify TSO and RTP artefacts, making Restrander a useful tool for library preparation QC.

Improving visualisations

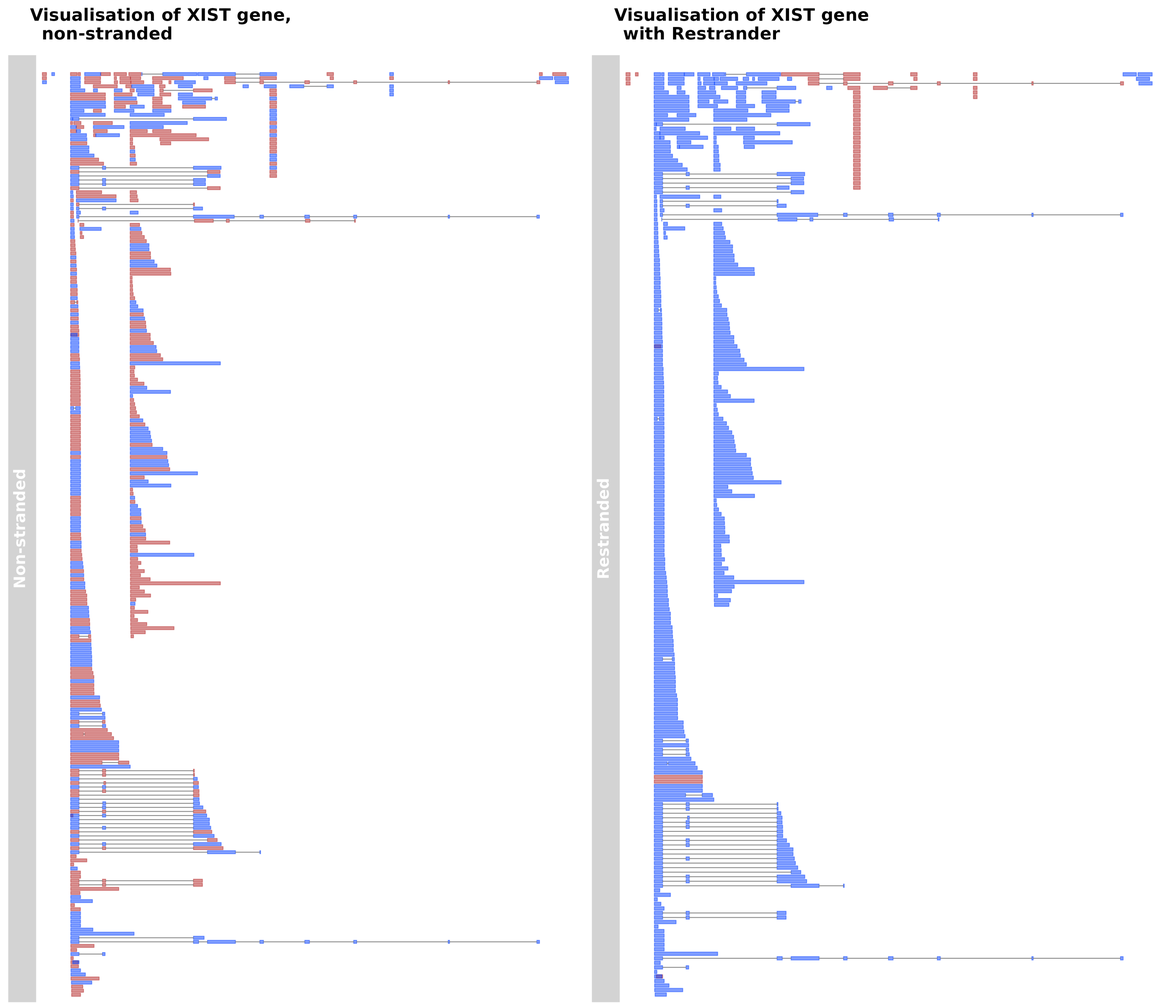

Figure 5: Visualisation of a subsection of the XIST gene, with read directionality made clearer by the restranding process.