Supplemental information

breakpointR: an R/Bioconductor package to localize strand state changes in Strand-seq data

David Porubsky[1†*], Ashley D. Sanders[2†], Aaron Taudt[1], Maria Colomé-Tatché[1], Peter M. Lansdorp[1,2,3‡], Victor Guryev[1‡]

1) European Research Institute for the Biology of Ageing, University of Groningen, University Medical Center Groningen, Groningen, The Netherlands.
2) Terry Fox Laboratory, BC Cancer Agency, Vancouver, BC, Canada.
3) Department of Medical Genetics, University of British Columbia, Vancouver, BC, Canada.

Content:

Overview of the breakpointR workflow

BreakpointR detects template-strand-state changes in Strand-seq data.

Strand-seq sequences only template strands for each homologous chromosome in every single cell. In a diploid organism each template strand originates from a maternal and paternal chromosome. Template strands originating from each parental homologue are segregated in daughter cells following random inheritance patterns. Each template strand can be identified based on the directionality it maps to the reference genome, and defined as either Watson template ('W' - maps to negative strand) or Crick template ('C' - maps to positive strand). For a given chromosome, when a daughter cell inherits two Watson templates from both parental homologues we observe reads mapping exclusively in a negative orientation (designated as 'WW' state). Conservely, reads mapping exclusively in the opposite (positive) orientation are observed if both Crick templates were inherited (designated as 'CC' state). Last, in a situation when Watson and Crick template strands are inherited from either parental homologue we observe a mixture of W and C reads aligned to the given chromosome (designated as 'WC' state). The expected frequencies of WW and CC states are 25%, and 50% for the WC state. Because each inherited template strand represents a single parental homologue, localized changes in template-strand-states represent biological features (described below) that can be mapped to one of the homologues.

Template-strand-state changes in Strand-seq data can result from large structural changes in genome organization (*e.g.* such as inversions) or sister chromatid exchange (SCE) events. Inversions change the directionality of a piece of DNA, which is reflected in a localized template-strand-state change on the inverted homologue (Sanders et al. 2016). Similarly, SCEs cause a template-strand-change in single cells as a cause of a double strand break repair occurring during DNA replication (van Wietmarschen and Lansdorp 2016; Claussin et al. 2017). The breakpoints of these rearrangements are seen as transition points between the three strand states described above (*i.e.* the template-strand-state will change from *e.g.* WW to WC). As a result, each breakpoint will exhibit a defined change in the proportion of Watson and Crick reads between segments upstream (3' to) and downstream (5' to) of the transition point. Consequently, by measuring the proportions of Watson and Crick reads along a chromosome breakpoints can be detected.

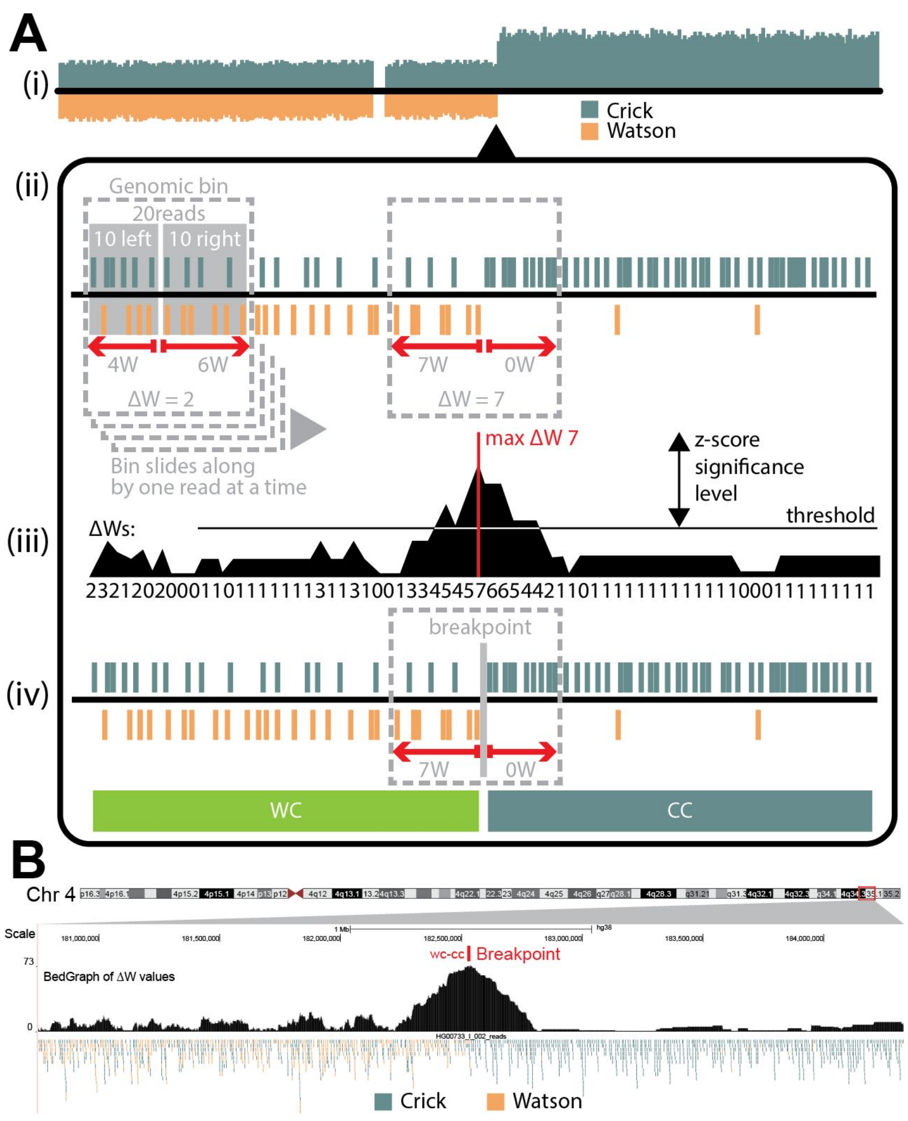


Figure S1: A bi-directional sliding window algorithm implemented in breakpointR

A) To illustrate how the breakpointR algorithm works we will follow breakpoint mapping between a WC to CC template state. (i) Example of binned read counts for a chromosome. Each vertical bar denotes number of 'Crick' (C; teal) and 'Watson' (W; orange) reads in each bin. (ii) breakpointR starts with a user-defined bin of 20 reads in total. Each bin is split in half such that there are 10 reads in the left and 10 in the right portion of the bin (red arrows). The number of Watson (W) reads in the left portion of the bin are subtracted from the number of W reads in the right portion, and the absolute value is reported as the ΔW. The algorithm then slides the bin forward, advancing one read at a time while keeping the total number of reads (i.e.. 20) in each bin. Consequently, the bin is dynamically resized (see lengths of red arrows) to accommodate changes in read density and sequencing coverage seen for the chromosome. (iii) A ΔW value is recorded for each bin (see numeric vector). Peak calling is then applied to search for high-confidence peaks in the ΔW. Peak confidence is determined using z-score statistics to test for significance above a user-defined threshold (default: 1/3 of the highest ΔW). (iv) Significant peaks are considered putative breakpoints that mark the location of template-strand-state changes. Using these breakpoints to define segments for strand state assignment. The strand state is tested between all putative breakpoints by measuring the total number of W and C reads in the segment and assigning the most-probable template-strand-state using the Fisher’s exact test. A breakpoint is retained only if two neighboring segments show different template-strand-state; otherwise the breakpoint is removed and the two segments are merged. B) UCSC Genome Browser view of ΔW distribution (black bars) measured as an example region on chromosome 4. Individual Strand-seq reads are shown as teal (Crick) and orange (Watson) colored ranges along the chromosome.

Read-based binning

breakpointR takes as an input strand-specific sequencing reads aligned to the reference genome in BAM file format. To calculate regional proportions of W and C reads, the chromosome data are first binned. In breakpointR, we have implemented a bi-directional read-based binning approach. We have previously shown that template-strand-state changes can be determined by calculating the ratio of W and C reads within defined genomic regions (bins) using the open-source software BAIT (Hills et al. 2013). However, BAIT fragments the genome into uniform and fixed-sized bins and, therefore, breakpoint resolution depends on selected bin length. Our read-based binning strategy scales each bin dynamically to accommodate a user-defined number of reads and slides the window position one read at a time to preserve the genomic context of each individual sequenced fragment. Importantly, this approach accounts for biases that are caused by genome mappability (Baslan et al. 2012; Navin and Hicks 2011) and variable or sparse sequence coverage typical to single-cell data. This also ensures that every read is independently considered to offer the highest possible breakpoint resolution for each single cell analyzed.

∆W calculation

In addition to a sensitive binning strategy, breakpointR implements a '∆W function' to calculate regional changes in the relative abundance of W and C reads. For a given bi-directional bin, a delta W (∆W) value is calculated as the absolute difference in the total number of W reads found in the first half of the bin to the total number of W reads found in the second half of the bin. This produces a ∆W value, which is assigned to the end position of last read in the left portion of a given bin and to the start position of the first read in the right portion io that bin. Bin then slides forward by one read to repeat the calculation. By sliding across the whole chromosome, a vector of ∆W values is created, which is then interrogated to locate any change in the proportion of W and C reads along the chromosome. In simple terms, the ∆W represents the template-strand-state of the region; a consistent ('unchanged') template-strand-state will produce a low ∆W value, whereas a template-strand-state change will produce a high ∆W value. Thus, by calculating ∆W values for each sliding bin, template-strand-state changes, referred to as 'breakpoints', can be located as peaks in the ∆W values.

Peak calling

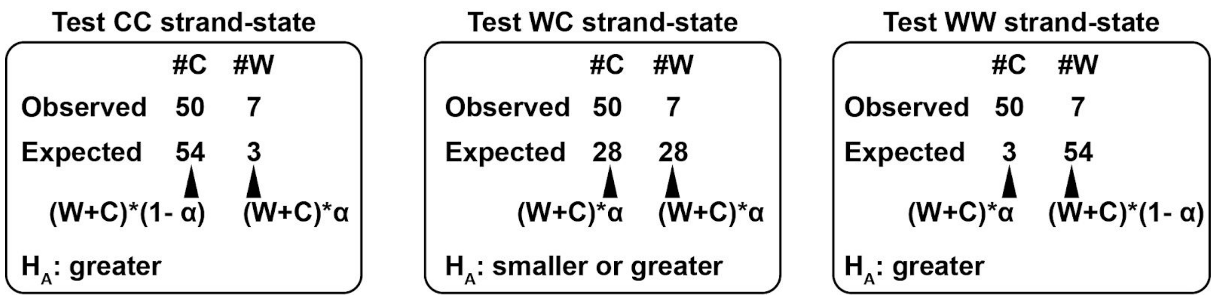
High confidence peaks in the ∆W are determined by z-score statistics based on a user-defined threshold. The threshold is input as a fraction of the highest detected peak (highest ∆W) for the given chromosome. By default this threshold is set as 1/3 of the highest ∆W value. In order to avoid skewing calculation of standard deviation of ∆W we 'trim' user defined percentage of extreme ∆W values (default: 10%). Standard z-score for each ∆W value is calculated as follows.

$z\text{-}score = \Delta W - (\Delta W * threshold) / sd(\Delta W)$

High confidence peaks are considered those with z-score >= 3.29, which correspond to a 99.9% confidence interval that the given ∆W value lies above the set threshold. Such high confidence peaks are considered as putative breakpoints of a change in template-strand-state. breakpointR assigns boundaries of the putative breakpoint to the end position of the first read in the peak and the start position of the last read in the peak. Given these breakpoint boundaries, the chromosome is split into individual segments, each marking a location of a predicted template-strand-state change. To confirm the template-strand-state change breakpointR then performs a strand state assignment for each bin.

Strand-state assignment

For each segment defined during peak calling, the number of W and C reads are counted to assign the most-likely template-strand-state using the Fisher Exact Test or binomial test. In the Fisher Exact Test we consider W and C reads as two categorical variables and test how likely the observed W and C read counts are, given the expected values if the segment was WW, CC or WC. To do this, the expected number of W and C reads is calculated separately for WW, CC and WC strand states based on the total number of reads in the segment and given a user-defined fraction of 'background' reads (*i.e.* proportion of reads expected to map in the opposite direction in WW or CC strand-states). The background accounts for the normal low-level noise seen in a typical Strand-seq library, which is on average ~5% for any given cell. These values are used to fill contingency tables that test the most-probable state given the observed W and C counts for the segment. An example showing the Fisher Exact Tests is shown in Figure S2. Depicted are the contingency tables for a segment containing 57 total reads, where the observed C count is 50 and W count is 7, and where the background level (α) is set to 0.05. The output are three p-values and the most-probable strand-state is CC.



Test CC strand-state

	#C	#W
Observed	50	7
Expected	54	3
	$(W+C)*(1-\alpha)$	$(W+C)*\alpha$

H_A: greater

Test WC strand-state

	#C	#W
Observed	50	7
Expected	28	28
	$(W+C)*\alpha$	$(W+C)*\alpha$

H_A: smaller or greater

Test WW strand-state

	#C	#W
Observed	50	7
Expected	3	54
	$(W+C)*\alpha$	$(W+C)*(1-\alpha)$

H_A: greater

Figure S2: Fisher Exact Test contingency tables.
Fisher Exact Test results in the following p-values for each tested strand state: WC = 0.9999850; CC = 0.9530812; WW = 1.0000000, The smallest p-value is selected as the most likely strand state in a given region.

In addition, breakpointR implements strand state assignment using binomial test. Here strand state probability p_t is calculated as the probability of observing a given number of W reads in a genomic region with a strand-state t and is defined as follows:

$$p_t = \begin{cases} 1-\alpha & \text{if } t = \mathrm{WW} \\ 0.5 & \text{if } t = \mathrm{WC} \\ \alpha & \text{if } t = \mathrm{CC} \end{cases}$$

In this definition, the level of background reads are represented as a constant parameter *α*. To account for large counts of W and C reads we operate in a logarithmic space. For both, Fisher Exact Test and binomial test no multiple testing correction is applied as for each genomic segment we merely try to assign the most likely strand state given the observed read counts. Method to assign strand state to a genomic interval is defined by a ‘genoT’ parameter set to either ‘fisher’ or ‘binom’ what refers to Fisher Exact Test or binomial test respectively.
To set the accuracy of strand state assignment for both Fisher Exact Test and binomial test we considered various combinations of W and C reads for a total of 1000 reads (Figure S3). In comparison both Fisher Exact Test and binomial test performs equally well in most cases. Strand states predicted by Fisher Exact Test and binomial test agrees in 91% of all tested combinations of W and C reads (Figure S3). Remaining 9% of cases represent a borderline counts of W and C reads where it is difficult to decide if a given read counts belong to WC state or pure WW and CC state. In such cases Fisher Exact Test leans towards WC state while binomial test leans towards pure WW or CC state.

Breakpoint refining

Based on the results from the strand state assignment test the putative breakpoints that were defined during peak calling are re-evaluated. For any two neighbouring segments that are on either side of a putative breakpoint, the most-probable strand states are compared. The breakpoint is retained only if the two strand states differ (*i.e.* the strand state of the segment upstream (3’) of the breakpoint is *different* to the strand state of the segment downstream (5’) of the breakpoint). The different states suggests the predicted breakpoint accurately located a template-strand-state change in the chromosome. Alternatively the breakpoint is removed if the two neighboring segments do not differ in strand state (*i.e.* the strand state of the segment upstream (3’) of the breakpoint is *the same* as the strand state of the segment downstream (5’) of the breakpoint). In this scenario the predicted breakpoint is considered false as it does not represent a *bonafide* template-strand-state change in the chromosome, and thus the two segments are merged. After testing each neighbouring segment and merging the false breakpoint predictions, breakpointR reiterates the strand state assignment test (for a maximum of 10 times) to confirm all remaining breakpoints accurately predict template-strand-state changes in the chromosome. The results are a list of breakpoints and strand states assigned to all neighbouring segments.

Confidence interval calculation

Once breakpoints are refined the breakpoint region is then expanded to encompass a confidence interval. Confidence intervals are calculated with read-resolution by going outwards from a breakpoint read by read (e.g. to the left), and testing the probability that the reads within the tested interval belong to the other side of the breakpoint (e.g. to the right) (this is equivalent to saying that the breakpoint is falsely positioned). Obviously, in doing so we calculate two confidence intervals, one for the left side and one for the right side of the breakpoint. The following example illustrates the procedure:
We have a breakpoint strand state assigned as ww-wc (watson/watson - watson/crick). Now, for the left side of the breakpoint with ww-strand-strand, the probability of finding watson and crick reads, if the breakpoint was falsely positioned and the reads actually belonged to the

right side with wc-strand-state, is $prob_{watson} = 0.5$ and $prob_{crick} = 0.5$. Conversely, for the right side with wc-strand-state, the probability of finding watson and crick reads, if the breakpoint was falsely positioned and the reads actually belonged to the left side with ww-strand-state, is $prob_{crick} = background$ and $prob_{watson} = 1 - background$. We can now employ a binomial test for finding the probability p telling us how likely it is that the watson- and crick-read counts in the tested interval actually belong to the other side of the breakpoint.
Another way to describe this is to say that for both left and right side of the breakpoint, we test the null hypotheses that the reads within the tested interval are generated by a binomial distribution with parameters from the other side of the breakpoint. A (left-sided) confidence interval with threshold p = 0.01 would mean that the probability is 1% that the reads within the confidence interval belong to the other (right) side of the breakpoint, and thus the confidence is 99% that the true breakpoint lies within that interval.

BreakpointR results

The outputs of a breakpointR analysis include a 'BreakPoint' class object that stores the raw directional reads, the vector of ΔW values, as well as all detected breakpoints for the input file. Additionally, breakpointR prepares bed-formatted files of these data, which enables the user to visualize their results directly in the UCSC Genome Browser. Last, breakpointR outputs genome-wide and chromosome-specific plots of all the template-strand-state changes located per input single cell, as well as a population-scale heatmap representing a summary of all template-strand-states detected in the input dataset. Accordingly, breakpointR provides the user with ample ways to interpret the Strand-seq data for biological discoveries.

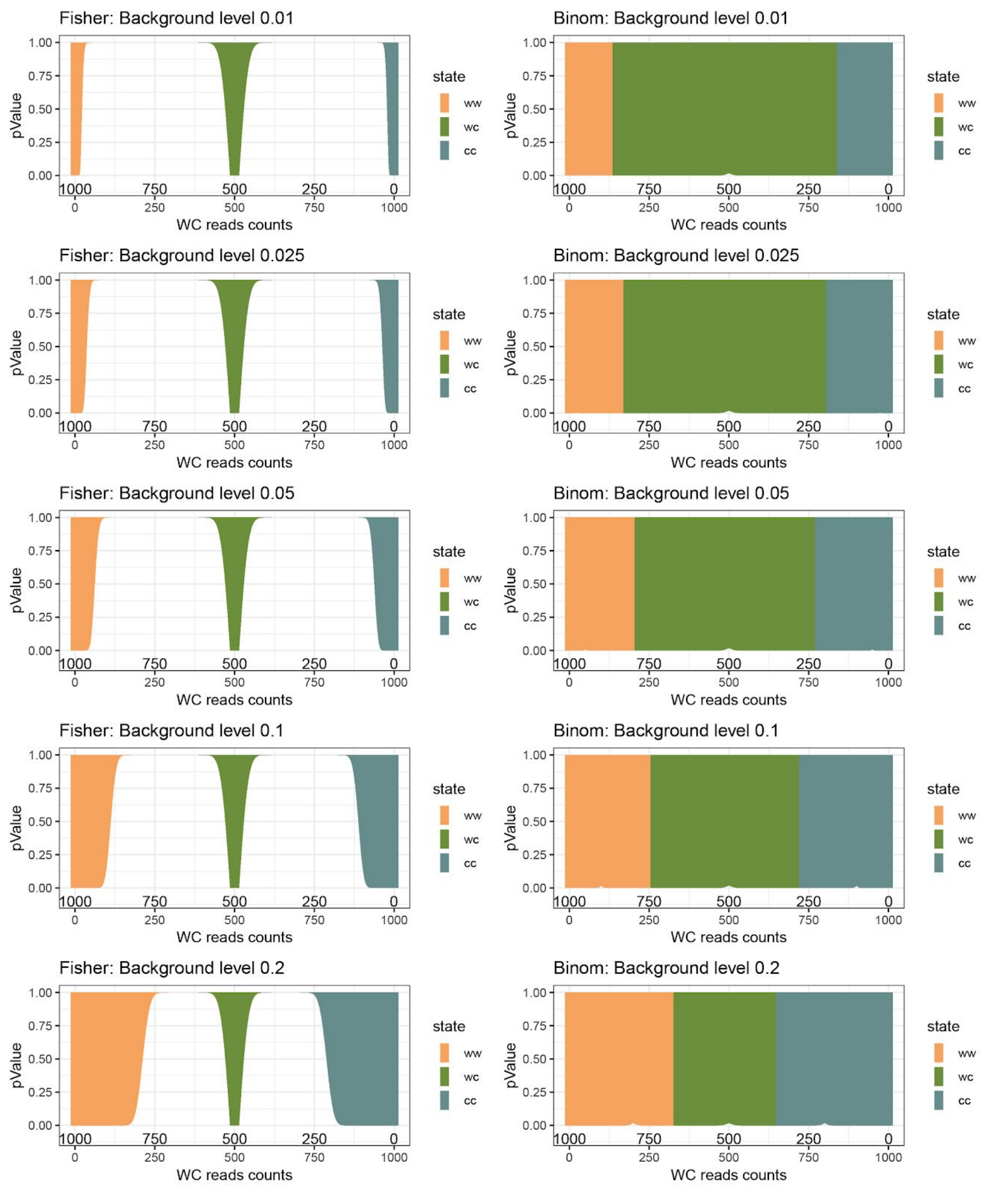


Figure S3: Strand state assignment using Fisher's exact test in comparison to binomial distribution test.

Left column: Distribution of Fisher probabilities calculated for all possible combinations of W and C reads given the total of 1000 reads. X axis shows counts of C reads from 0 to 1000 and counts of W reads from 1000 to 0. Right column: Distribution of binomial probabilities calculated for all possible combinations of W and C reads given the total of 1000 reads. Combinations of directional reads are shown on x-axis. From 1000 to 0 for Watson reads and from 0 to 1000 for Crick reads. For each combination of Watson and Crick reads the most likely strand state (WW, WC or CC) and corresponding probability is reported on y-axis.

Comparison of SCE detection between BAIT and breakpointR

We establish accuracy of strand-state-change detection using BAIT and breakpointR using simulated set of Strand-seq libraries sampling, we simulated libraries containing various fractions (1, 2.5, 5 and 7.5% of genome coverage, using GRCh38 reference) in order to mimic range of genome coverage of real Strand-seq libraries (Chaisson et al. 2019). We used our custom PERL script to simulate single-end Strand-seq libraries with 76bp long reads by cutting the reference genome (GCA_000001405.15_GRCh38_no_alt_analysis_set.fna) into 150bp (+/- 10SD around the mean fragment size) long fragments (https://github.com/daewoooo/breakpointR_paper). The simulated reads have then been mapped to the reference genome using BWA (version 0.17.17-r1188). Aligned reads have been sorted by SAMtools (version 1.4.1) and duplicated reads have been marked by sambamba (version 0.6.6). In total 10 single-end Strand-seq libraries have been simulated for each above mentioned fraction of the genome coverage (Table S1). For each library we have simulated 5-7 randomly distributed strand-state-changes around the genome (reflecting typical rate of sister chromatid exchange in our studies). All simulated BAM files are freely available at zenodo site (https://doi.org/10.5281/zenodo.3359715).
To compare BAIT and breakpointR we have downloaded the latest version of BAIT (v1.4) available at Sourceforge (https://sourceforge.net/p/bait/wiki/Home/). We note that this version of BAIT doesn’t work properly with the latest version of SAMtools (v1.9) and therefore older version of SAMtools (version 1.4.1) have been used instead. All simulated Strand-seq libraries were analyzed by both BAIT and breakpointR using 200kb and 1Mb long genomic window. We kept constant BAIT parameters (BAIT -a -r -6 200000 and 1000000). Likewise, we analyzed all simulated data were analyzed using breakpointR function called ‘breakpointr’ with the following parameters: windowsize = 200000 and 1000000, binMethod = 'size', pairedEndReads = FALSE, min.mapq = 10, filtAlt = TRUE, background = 0.05, minReads = 100, conf=95. Our results of simulated strand-state-change detection are freely available for both BAIT and breakpointR on github (https://github.com/daewoooo/breakpointR_paper).
To compare BAIT and breakpointR we have assessed the fraction of simulated strand-state-changes (n=247, by simulation of 5-7 events per 10 cells for 4 different genome coverage levels) that overlaps with predicted strand-state-change by both tools either directly or within a specified range of 10kb, 100kb or 500kb distance from the simulated breakpoint (Figure S4, Table S2). Overall we found breakpointR to be able to detect more simulated strand-state-changes than BAIT using both 200kb and 1Mb size genomic window. Another important advantage of breakpointR over BAIT is that apart from calling breakpoint as range between two consecutive reads assigned to different strand states, it also calculates user defined confidence intervals (CI) for each detected breakpoint and thus accounts for various levels of noise in single-cell Strand-seq data (see Calculation of confidence intervals). In summary, 46.56% and 49.8% breakpoints predicted by breakpointR overlap directly with simulated breakpoints using 200kb and 1MB window, respectively.

Notably, more than 92% of simulated breakpoints falls within predicted confidence intervals (CI = 0.95) of each breakpoint. In contrast, BAIT was able to directly detect only 27.13% and 21,86% simulated breakpoints using 200kb and 1MB window, respectively. The number of detected breakpoints increases when we allow for overlap within 10kb, 100kb and 500kb away from the simulated breakpoint. This true for both, breakpointR and BAIT while breakpointR consistently provides more detected breakpoints than BAIT. Notably, using larger window size (1Mb) BAIT was able to detect less than 68% of simulated breakpoints (Figure S4, Table S2). While BAIT provides higher breakpoint resolution than breakpointR, it does so at costs of missing the real breakpoint position. This is shown by consistently larger distances of BAIT breakpoints from the simulated ones (Figure S4A.B, ii and iii). In this tool comparison only 4 out 247 simulated breakpoints were missed by breakpointR. All of these missed breakpoints lied within 1Mb from the start of the chromosome and therefore are difficult to detect using window size used in this comparison. Overall, breakpointR achieved a better precision and recall of simulated breakpoints, at both tested window sizes (200kb and 1Mb), than BAIT (Figure S4C).

Table S1: Summary read mapping statistics of simulated Strand-seq libraries

	Total # of reads	Depth of coverage	% of genome covered
COV01	368492	0.0096	0.96
COV025	921404	0.024	2.38
COV05	1842927	0.048	4.69
COV075	2763904	0.0721	6.95

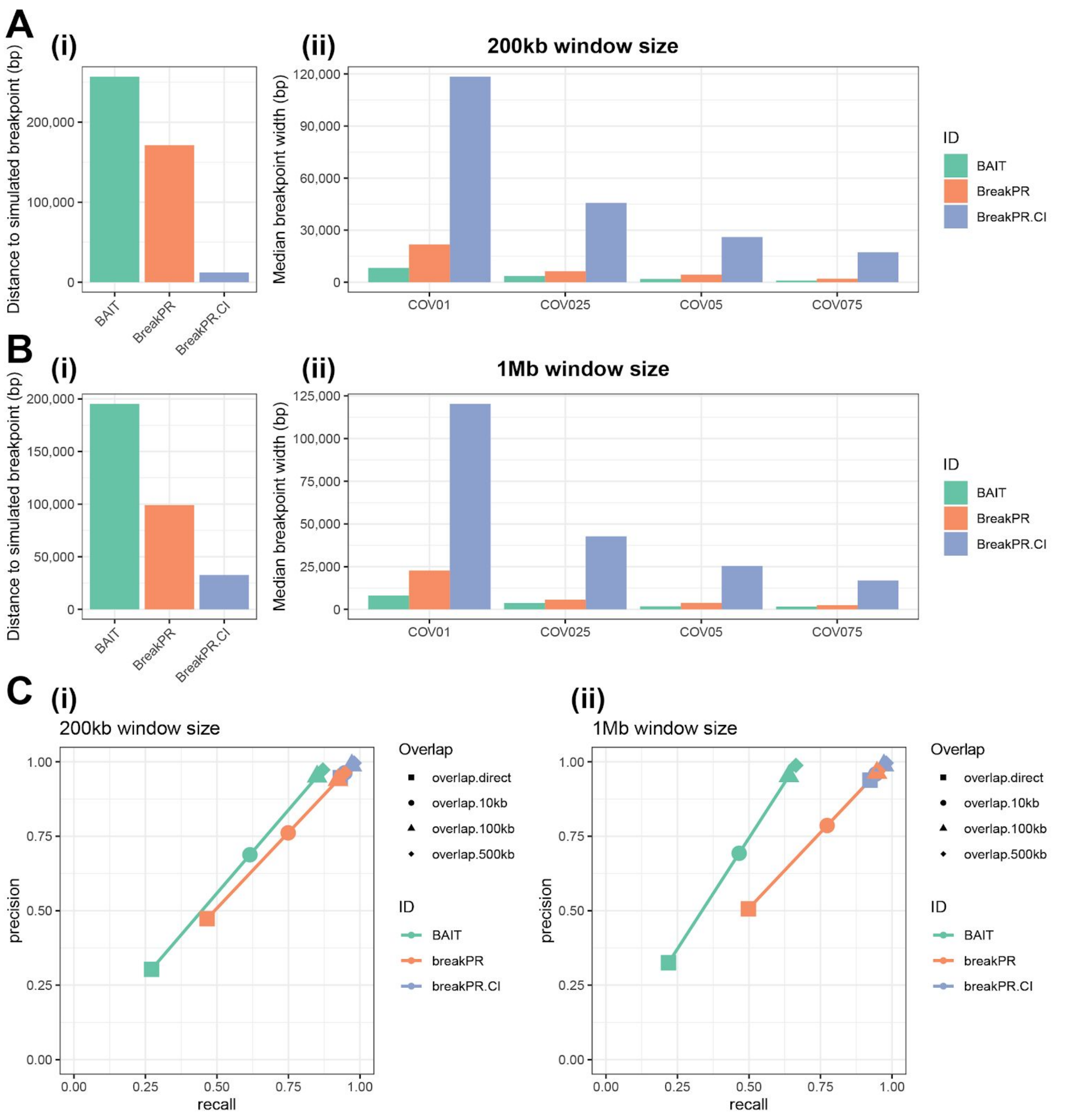


Figure S4: Comparison of breakpoint detection between breakpointR and BAIT on simulated data

A) Comparison of BAIT and breakpointR for 200kb genomic window. (i) Distance of the detected breakpoint boundaries either by BAIT or breakpointR (BreakPR and BreakPR.CI - 95% confidence intervals) to the simulated breakpoint. (ii) Shows a median width (bp) of breakpoints predicted by BAIT and breakpointR.

B) Comparison of BAIT and breakpointR for 1Mb genomic window.

C) Precision-recall curve comparing performance of BAIT and breakpointR for 200kb (i) and 1Mb (ii) long genomic window. This plot evaluates how many simulated SCEs (‘simul’) overlap directly (‘overlap.direct’, that is when simulated breakpoint falls within predicted breakpoint range), with breakpoint calls made by BAIT, breakpointR (‘breakPR’) and breakpointR confidence intervals (‘breakPR.CI’). In addition predicted breakpoints that are within 10kb (‘overlap.10kb’), 100kb (‘overlap.100kb’) and 500kb (‘overlap.500kb’) away from the simulated SCE breakpoint are counted. The total number of SCE breakpoints detected by both tools (‘total.sce’) is shown as well.

Table S2: Summary of simulated breakpoint detection

Column 'Overlap' distinguishes 'overlap.direct' meaning that a simulated breakpoint falls directly within predicted breakpoint range either by BAIT or breakpointR. 'overlap.10kb' allows predicted breakpoint to be within 10kb distance from the simulated breakpoint. Similarly 'overlap.100kb' and 'overlap.500kb' allow for even larger distance.

		200kb window		1Mb window	
Overlap	**Tool**	**SCEs found**	**% SCEs found**	**SCEs found**	**% SCEs found**
overlap.direct	BAIT	67	27.13	54	21.86
overlap.direct	breakPR	115	46.56	123	49.80
overlap.direct	breakPR.CI	230	93.12	228	92.31
overlap.10kb	BAIT	152	61.54	115	46.56
overlap.10kb	breakPR	185	74.90	191	77.33
overlap.10kb	breakPR.CI	234	94.74	233	94.33
overlap.100kb	BAIT	210	85.02	158	63.97
overlap.100kb	breakPR	228	92.31	234	94.74
overlap.100kb	breakPR.CI	240	97.17	240	97.17
overlap.500kb	BAIT	215	87.04	164	66.40
overlap.500kb	breakPR	234	94.74	239	96.76
overlap.500kb	breakPR.CI	242	97.98	242	97.98
total.sce	BAIT	221	89.47	166	67.21
total.sce	breakPR	243	98.38	243	98.38
total.sce	breakPR.CI	243	98.38	243	98.38

Library selection criteria

Low quality single cell libraries should be discarded prior to breakpointR analysis to ensure specificity of breakpoint detection. This is because variations in BrdU labeling, BrdU removal, cell sorting efficiency and library preparation can all result in variability in the number and distribution of reads seen for a single Strand-seq cell. Such poor quality Strand-seq data can result in false-positive breakpoint calls, and therefore we recommend a strict library selection step prior to breakpointR analysis. Library selection criteria have been described in detail in (Porubský et al. 2016; Sanders et al. 2017). Briefly, this involves discarding cells that exhibit low sequence coverage (*i.e.* libraries containing less than 50,000 reads), cells that contain 'noisy' Strand-seq data (*i.e.* libraries containing more than 10% background reads on WW or CC chromosomes), or cells that show non-uniform read distribution (*i.e.* libraries with patchy or puntate read coverage). By evaluating the aligned Strand-seq data, selection of input cells should be performed prior to analysis to ensure only high quality libraries are considered during breakpoint detection.

Filtering repetitive and complex genomic regions

It is worth noting that imperfections such as repeated sequences that are collapsed to a single locus in reference genome might cause false positive calls. For instance, centromeric regions or regions with spurious mapping of short-reads in regions such as segmental duplications (as judged by higher coverage and/or regional WC patterns invariant across all single cells) can result in measurable template-strand-state changes that do not reflect biological features. To account for this breakpointR contains a parameter called 'maskRegions' where the user can specify genomic regions that should be excluded in the analysis, such as (peri-) centromeric regions, regions with unexpectedly high read coverage, or regions that show WC state in all libraries. This feature should eliminate potential artefacts that might arise in such structurally complex regions of the human or other genomes.

References

Chaisson, Mark J. P., Ashley D. Sanders, Xuefang Zhao, Ankit Malhotra, David Porubsky, Tobias Rausch, Eugene J. Gardner, et al. 2019. "Multi-Platform Discovery of Haplotype-Resolved Structural Variation in Human Genomes." *Nature Communications* 10 (1): 1784.

Claussin, Clémence, David Porubský, Diana Cj Spierings, Nancy Halsema, Stefan Rentas, Victor Guryev, Peter M. Lansdorp, and Michael Chang. 2017. "Genome-Wide Mapping of Sister Chromatid Exchange Events in Single Yeast Cells Using Strand-Seq." *eLife* 6 (December). https://doi.org/10.7554/eLife.30560.

Porubský, David, Ashley D. Sanders, Niek van Wietmarschen, Ester Falconer, Mark Hills, Diana C. J. Spierings, Marianna R. Bevova, Victor Guryev, and Peter M. Lansdorp. 2016. "Direct Chromosome-Length Haplotyping by Single-Cell Sequencing." *Genome Research* 26 (11): 1565–74.

Sanders, Ashley D., Ester Falconer, Mark Hills, Diana C. J. Spierings, and Peter M. Lansdorp. 2017. "Single-Cell Template Strand Sequencing by Strand-Seq Enables the Characterization of Individual Homologs." *Nature Protocols* 12 (6): 1151–76.

Sanders, Ashley D., Mark Hills, David Porubský, Victor Guryev, Ester Falconer, and Peter M. Lansdorp. 2016. "Characterizing Polymorphic Inversions in Human Genomes by Single-Cell Sequencing." *Genome Research* 26 (11): 1575–87.

Wietmarschen, Niek van, and Peter M. Lansdorp. 2016. "Bromodeoxyuridine Does Not Contribute to Sister Chromatid Exchange Events in Normal or Bloom Syndrome Cells." *Nucleic Acids Research* 44 (14): 6787–93.