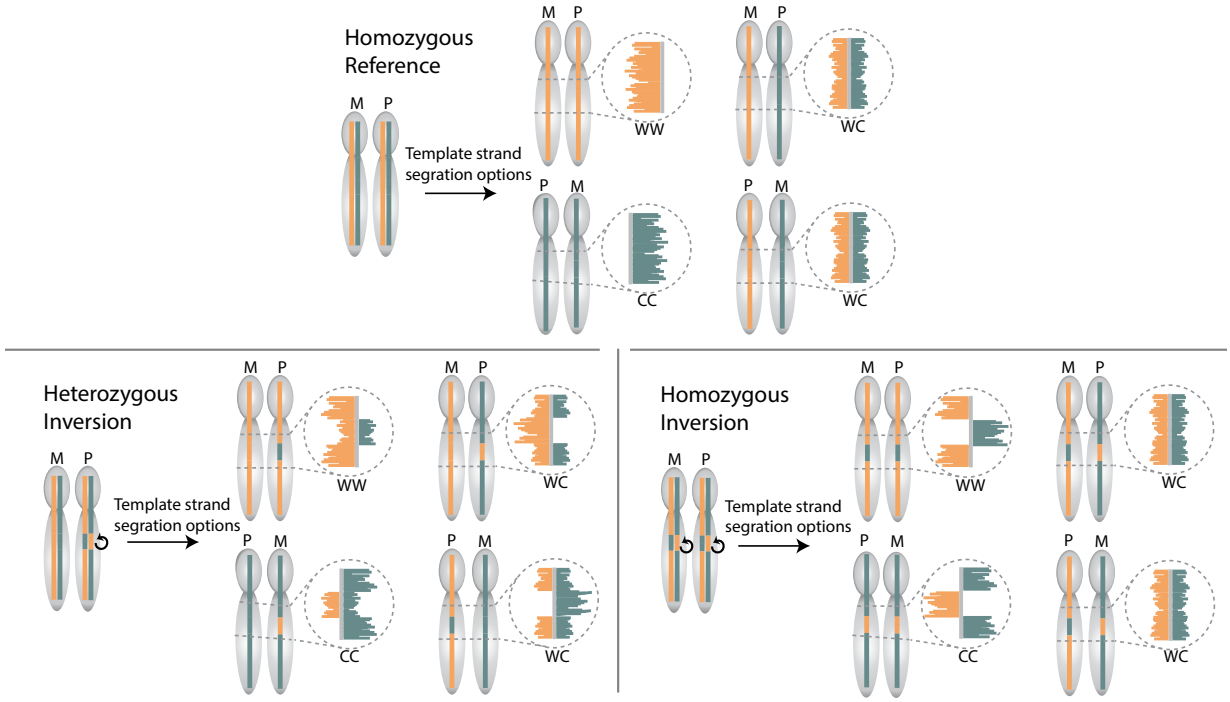
Supplemental Information

Characterizing polymorphic inversions in human genomes by single cell sequencing

Ashley D. Sanders, Mark Hills, David Porubský, Victor Guryev, Ester Falconer, Peter M. Lansdorp

Table of Contents

Supplemental Figures

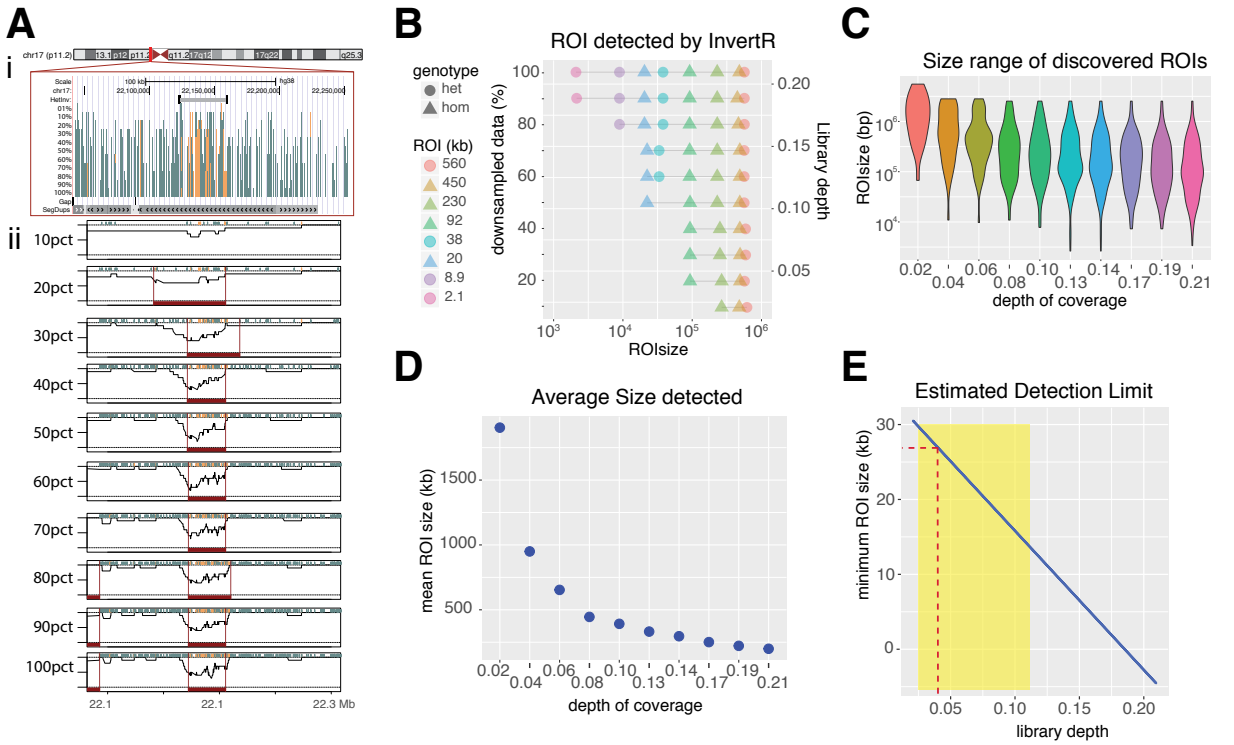


Supplemental Figure S1: Template strand inheritance patterns of inversions, as visualized by Strand-seq

Each parental homologue (M, maternal; P, paternal) is composed of a Watson (W, orange) and Crick (C, blue) strand. During mitosis each strand serves as a template for DNA synthesis, and following cell division the template strands of each homologue segregate into daughter cells as either WW and CC, or WC and CW. After Strand-seq library preparation (Falconer et al. 2012) and sequencing, the orientation of these template strands can be visualized by aligning the sequencing reads to the reference genome using BAIT software (Hills et al. 2013) (zoom inset).

At any given locus, a pair of homologues can contain no inversions with respect to the reference genome (homozygous reference), a single inversion on one homologue (heterozygous), or an inversion on both homologues (homozygous). In Strand-seq libraries, inversions appear as segmental changes in template strand orientation, and the number of homologues harboring an inversion at the locus can be discerned by the magnitude of change seen in the template strands. For instance, heterozygous inversions appear as a 'partial' change in strand orientation, where a WW chromosome switches to WC along the inversion, and a WC chromosome will switch to WW along

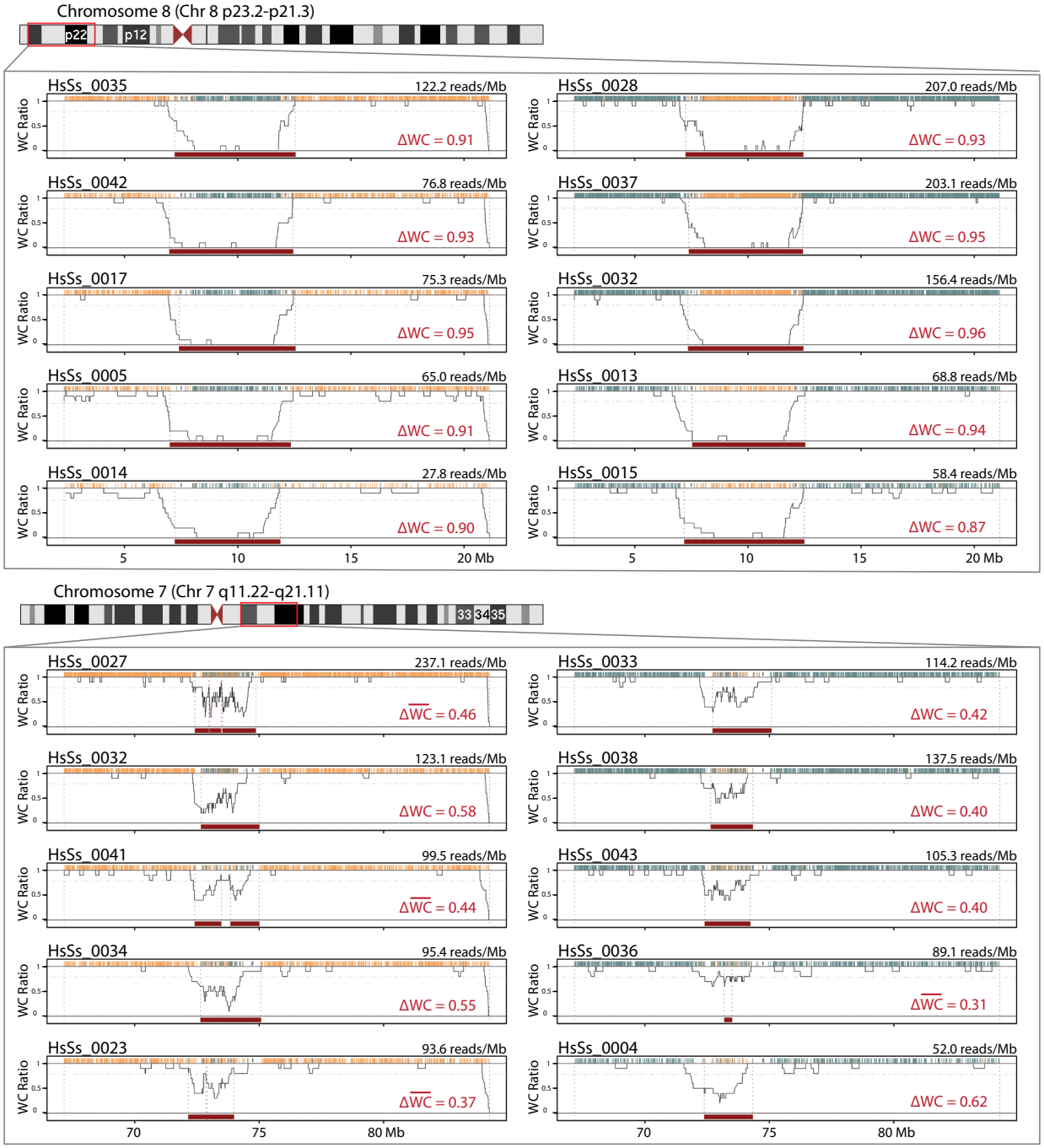
the inversion. Heterozygous inversions will be evident in all chromosomes. On the other hand, homozygous inversions result in a 'complete' change in template strand orientation, where a WW chromosome switches to CC along the inversion and a WC chromosome switches to CW. Consequently, homozygous inversions are masked in WC chromosomes, as they appear to have the same template strand orientation as homozygous reference loci in WC chromosomes. For homozygous inversions, only WW and CC chromosomes are informative.



Supplemental Figure S2: Detectable size range of rearrangements visible in Strand-seq libraries

To demonstrate the range of inversions reliably detected by Invert.R, we performed a down-sampling experiment by randomly subsetting reads from a single Strand-seq library (between 10 - 90% of the original library) and testing when specific inversion classes were no longer detected, along with the total range of inversions that were predicted in each subset. A) Example of a ~ 40 kb heterozygous inversion mapped to Chr 17p11.2. i) UCSC Genome Browser view of the down-sampled library, and ii) corresponding Invert.R (bin = 25) histograms of the region. Invert.R did not accurately detect the inversion until 40% of the reads were sampled. B) Plotted results of eight

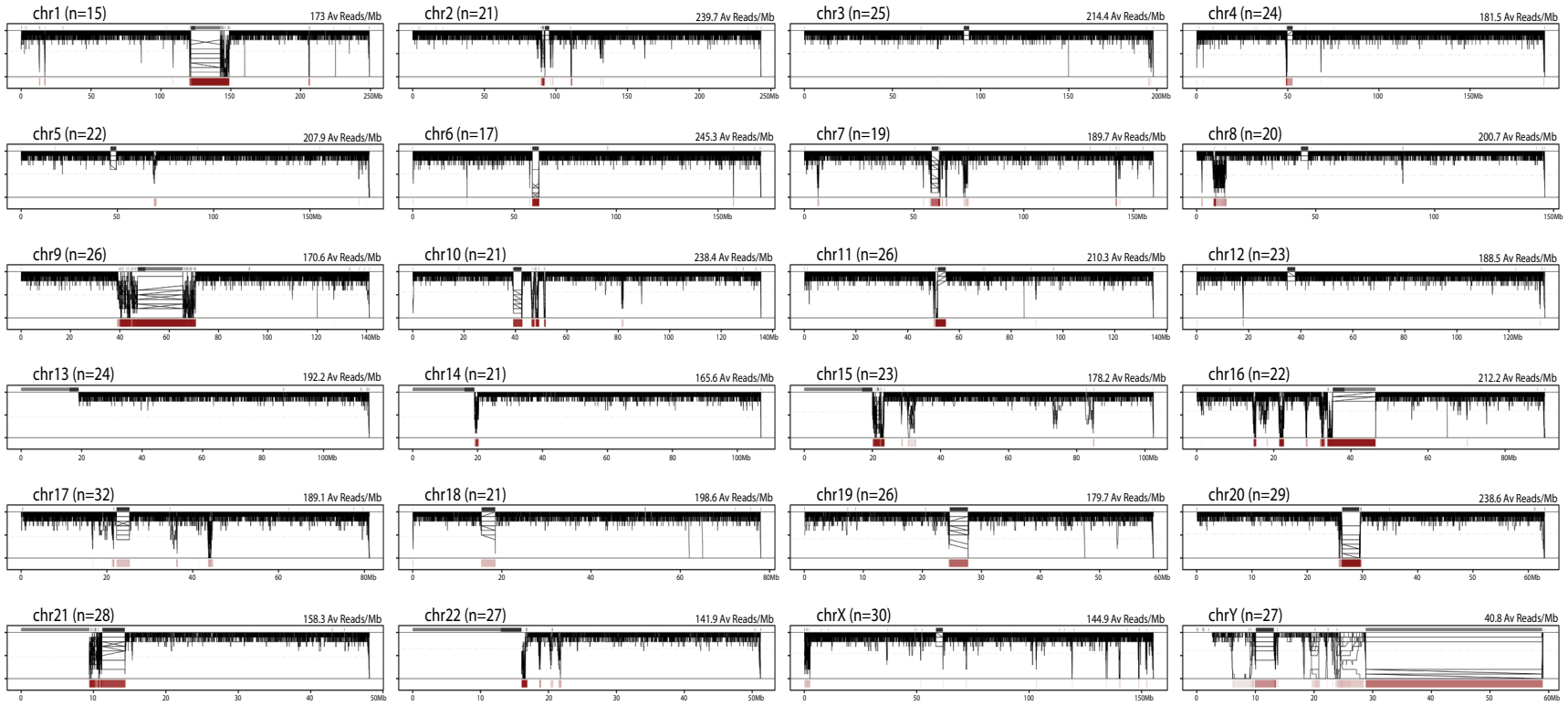
inversion examples representing various lengths and genotypes illustrating the genomic coverage at which an inversion class was no longer detected. For example, at 90% (corresponding to 0.19x coverage) all inversion classes were reliably detected by Invert.R, at 80% (0.16x) the ~ 9 kb heterozygous inversion was not called and at 40% (0.03x) the 20kb homozygous inversion was not called. C, D) For this down-sampled library, the total number and size distribution of inversions predicted by Invert.R was directly correlated to the genomic coverage of the individual library. This allowed us to predict the limits of detection for our technology. E) A predictive model of the minimal depth requirements to accurately locate various inversion classes. The shaded yellow box marks the range of genomic coverage of the single cell libraries used in this study, with the overall average indicated (dotted red line). Abbreviations: Region of interest (ROI); heterozygous inversion (het); homozygous inversion (hom); kilobase (kb); basepair (bp).



Supplemental Figure S3: Invert.R analyis of a homozygous and heterozygous inversion in single cells

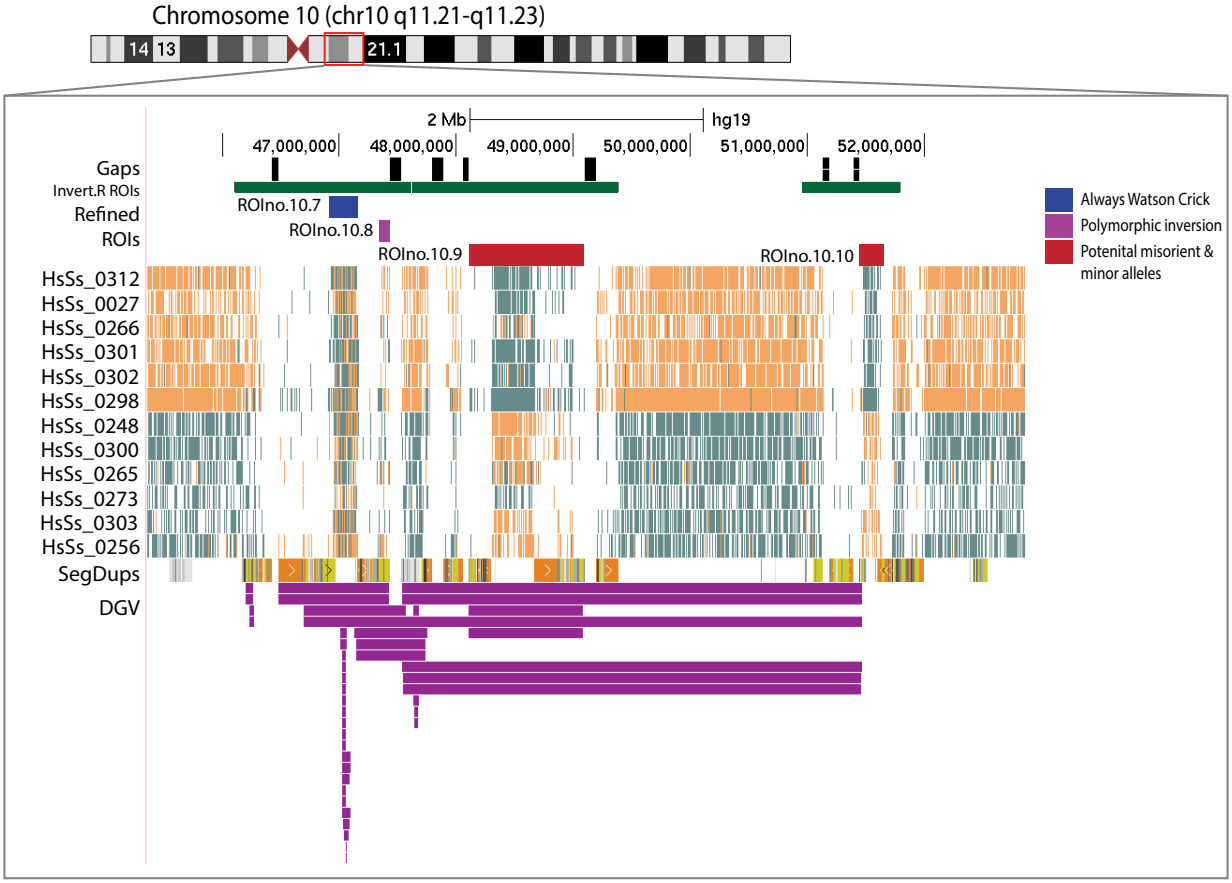
Ten single Strand-seq libraries from a male donor who has a homozygous inversion on Chr 8p23 (upper panel) and a heterozygous inversion on and Chr 7q11 (lower panel). Zoom insets (red box) of the W/C ratio values calculated by Invert.R (bin = 25 reads) are shown as histograms for each cell. Watson (orange) and Crick (blue) Strand-seq reads are shown above each histogram, along with sequence gaps in the reference

genome (grey). A ‘complete’ change in template strand orientation along the homozygous inversion (e.g. WW chromosome of HsSs_0035 changes to CC at the Chr 8 inversion), and a ‘partial’ change in template strand orientation at the heterozygous inversion (e.g. WW chromosome of HsSs_0027 changes to WC at the Chr 7 inversion) is shown by the magnitude of the change in the Invert.R histograms. The putative inversion predictions made by Invert.R are shown below (red bars), and the ∆W/C of the inversion is listed for each cell. In some cases, multiple inversions were predicted along the Chr 7 locus because the W/C ratios crossed the threshold more than once along the inverted segment (e.g. HsSs_0041). The variability seen between the individual cells is likely due to the changes in read densities across the locus, particularly near the inversion breakpoints, which are near large blocks of segmental duplications (see **Fig. 2B**). When multiple inversions were predicted within the region the average ∆W/C is provided. The reads/megabase (Mb) was also calculated for the region and listed for each library. These ten histograms were overlaid (in **Fig. 2B, iii**) and used to refine the inversion breakpoints.



Supplemental Figure S4: Putative inversions in a pooled donor cell population, as predicted by Invert.R

Overlaid histograms of W/C ratios, as generated by Invert.R (bin=25), for each chromosome of the pooled donor population cells. Number of cells (n) analyzed and average reads/megabase (Av Reads/Mb) is indicated. Each line in the histogram represents the W/C ratio at a genomic location in a single cell. A change in the W/C ratios along a chromosome represents a change in strand orientation, indicative of a putative inversion. Putative inversions overlapping in at least two cells appear as red heat maps below each histogram, where the intensity of the red hat map reflects the number of cells with a predicted inversion within the region. Reference assembly sequence gaps are shown as grey bars above each histogram.

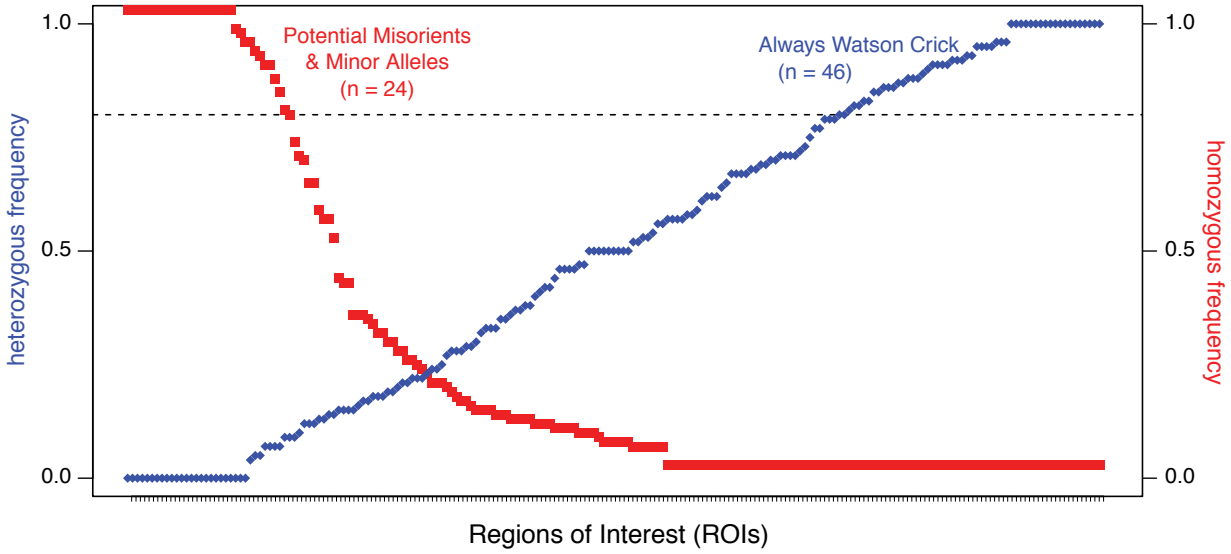


Supplemental Figure S5: Examples of different genomic features evident on Chr 10q11

Zoomed inset (red box) of a UCSC Genome Browser (GrCh37/hg19) view of 12 representative Strand-seq libraries from the pooled donor population. Within this 4 Mb domain, Invert.R predicted three regions of interest (ROIs, upper green bars), which were further refined and categorized based on the genotype frequencies calculated for each ROI. ROIno.10.7 was heterozygous in ≥ 80% of cells, and was categorized as an Always Watson Crick (AWC) region (dark blue bar). Two ROIs (ROIno.10.9 and ROIno.10.10) were homozygous in ≥ 80% of cells, and were categorized as potential misorients or minor alleles (red bars). ROIno.10.8 is a polymorphic inversion found in at least two cells, with different genotypes (dark purple bar). The domain has several reference sequence gaps (uppermost, black bars) and segmental duplications (SegDups) flanking the ROIs, and Database of Genomic Variants (DGV) inversions overlapping with the ROIs.

Note that the AWC (ROIno.10.7) is flanked by large blocks of segmental dupli-

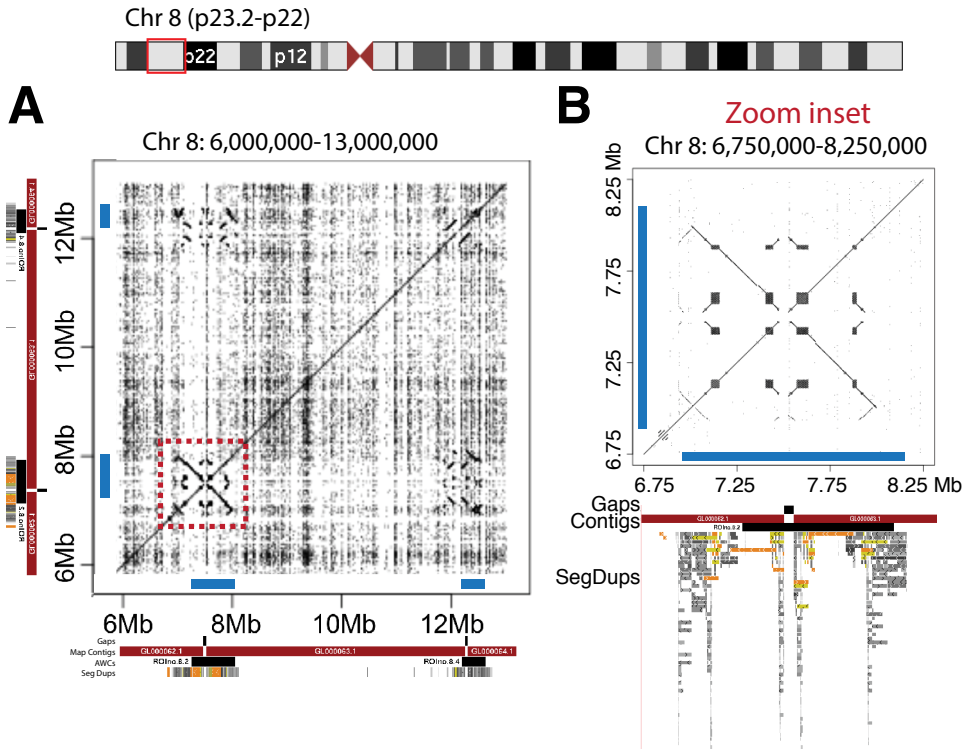
cations and is in a region where few high-quality reads align, while the read density at this AWC itself is quite high. We predict that ROIs classified as AWCs identify repetitive sequences in the human genome that are currently underrepresented in the reference assembly. If these sequences are present on multiple chromosomes, they are expected to have template strand inheritance patterns that match the chromosomes they reside on, and will frequently appear WC since each chromosome that harbors these sequences will have independent segregation patterns. Also, note that at ROIno.10.9 two cells (HsSs_0266 and HsSs_0273) harbor a heterozygous inversion whereas the remaining cells appear to have a homozygous inversion at the locus, making this ROI a probable minor allele present in the reference assembly. This is distinguished from ROIno.10.10, where every cell has a homozygous inversion, making this ROI a probable sequence fragment that is misoriented in the reference assembly. The ROIs classified as misorients or minor alleles point to regions in the human reference genome where the assembled sequence is not representative of the vast majority of individuals seen in our population.



Supplemental Figure S6: Genotype frequency of regions of interest in the mixed population

All cells were genotyped at all regions of interest (ROIs) to assess the frequency of heterozygosity (blue diamonds) and homozygosity (red squares) for each ROI, as plotted. The genotype was determined by counting the number of Watson and Crick reads in the region, where at least ten reads were required for inclusion, and then per-

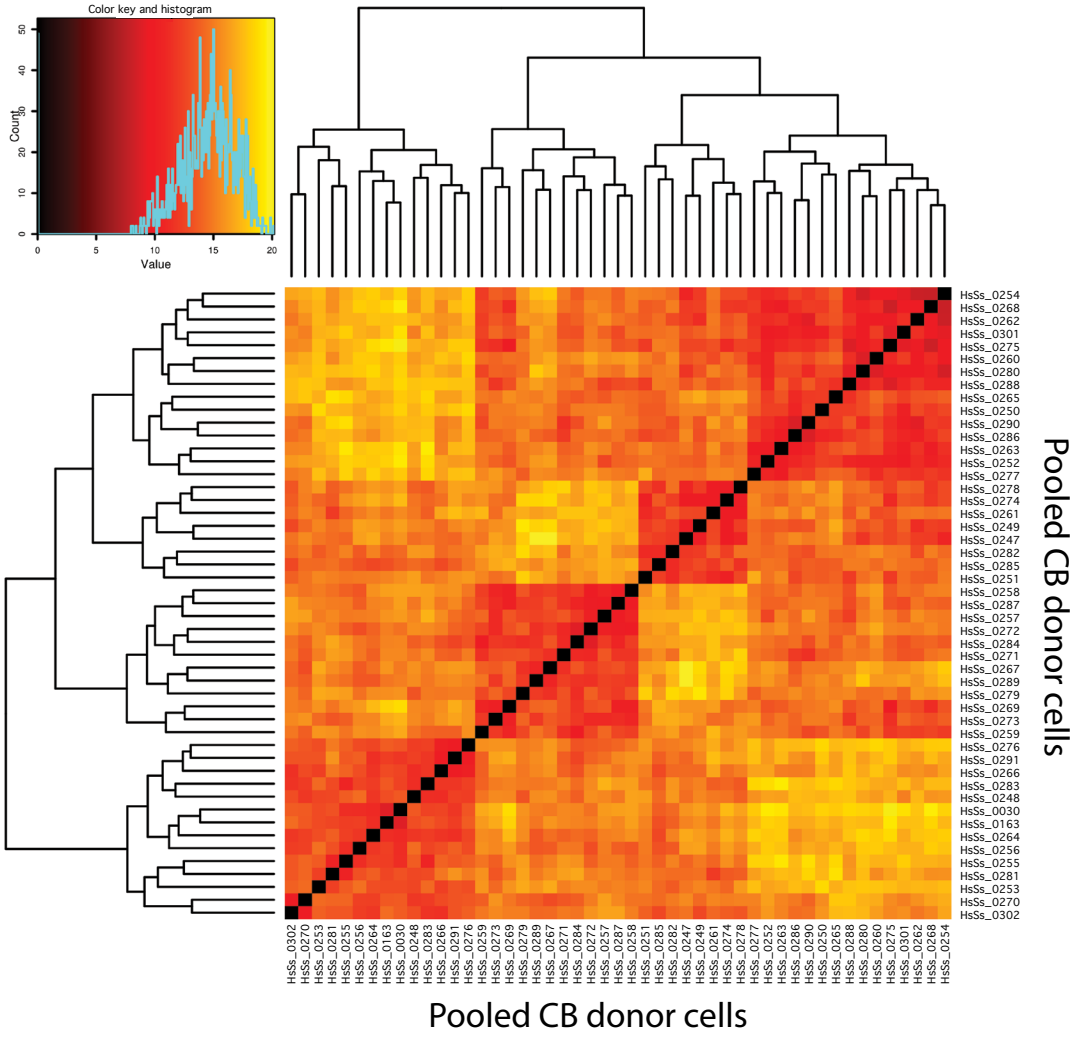
forming three Fisher's exact tests (for a homozygous reference, heterozygous inversion, and homozygous inversion) to determine the best fit genotype based on p-values (see Supplemental Methods for details). Frequencies were calculated as the proportion of genotyped cells showing either a heterozygous or homozygous state at each ROI. 46 ROIs had a minimum of ten cells with a heterozygous frequency of at least 0.8 (dotted line) and are defined as Always Watson Crick (AWC) regions. 24 ROIs had a minimum of ten cells with homozygous frequency of at least 0.8, and are misorients or minor alleles present in the human genome reference assembly. See Supplemental Tables S2 and S3 for the genomic coordinates of these ROIs, as well as Hardy-Weinberg statistical tests



Supplemental Figure S7: AWC regions mark repetitive elements with complex architecture

Self-alignment lastz dot plots of Chr 8 (coordinates listed above) illustrate the architecture of Always Watson-Crick regions (AWCs; blue bars). **A**) The two AWC regions flanking Chr 8p23 inversion had heterozygous frequencies of 85% (ROIno.8.2) and 95% (ROIno.8.4) and coincided with known segmental duplications (Seg Dups; depicted adjacent to axes). We can see the degree and orientation of sequence similarity between

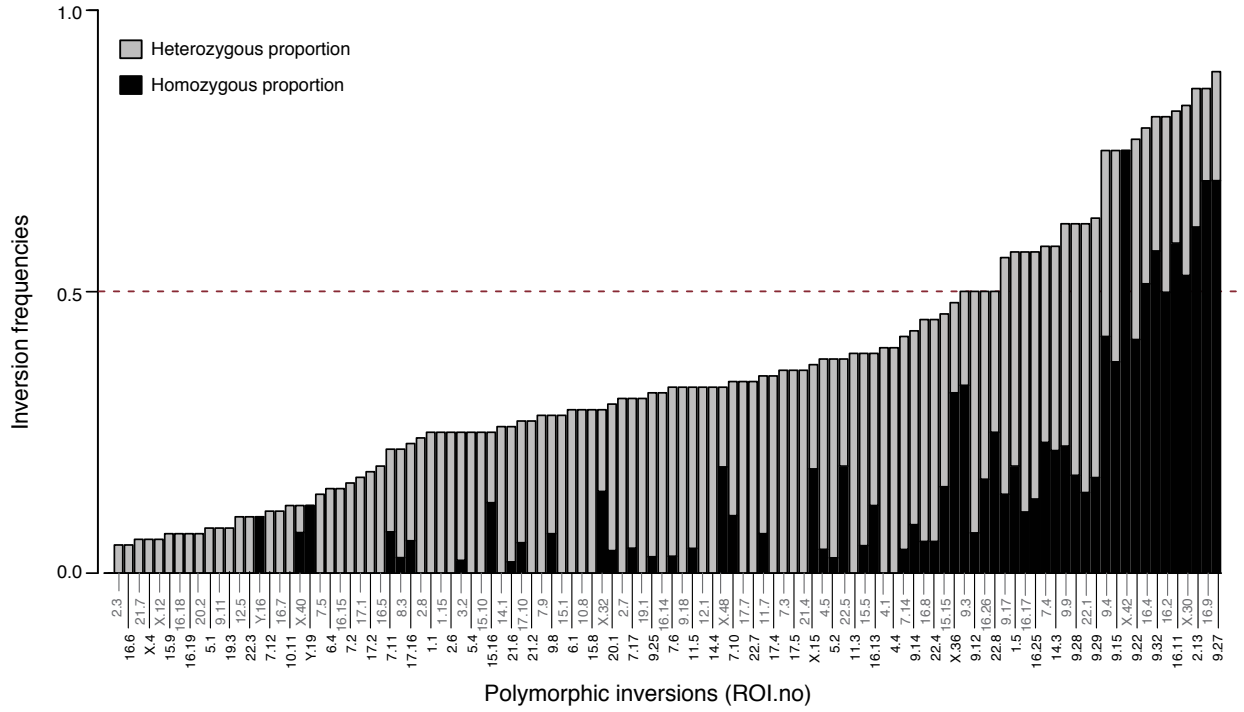
the AWCs, where ROIno.8.2 was a repeated palindrome of about 500 kb that was partially duplicated in ROIno.8.4. **B**) Zoomed inset (dotted red box in i) of ROIno.8.2 illustrating the palindromic duplication contains 4 copies of a minisatellite of variable sizes, highlighting the complex architecture at the locus.



Supplemental Figure S8: Cluster analysis of the inversion profiles for each pooled donor cell.

A clustered heat map of all pooled cord blood (CB) cells based on their inversion profiles. To characterize each cell's inversion profile we interrogated the 111 polymorphic inversions and (providing sufficient reads were present) genotyped the cell based on the number of W and C reads at each inversion, as determined by Fisher's exact test. We then compared the inversion profiles of all the cells to each other using a hierarchical clustering model based on these genotypes, from a pairwise dissimilarity matrix

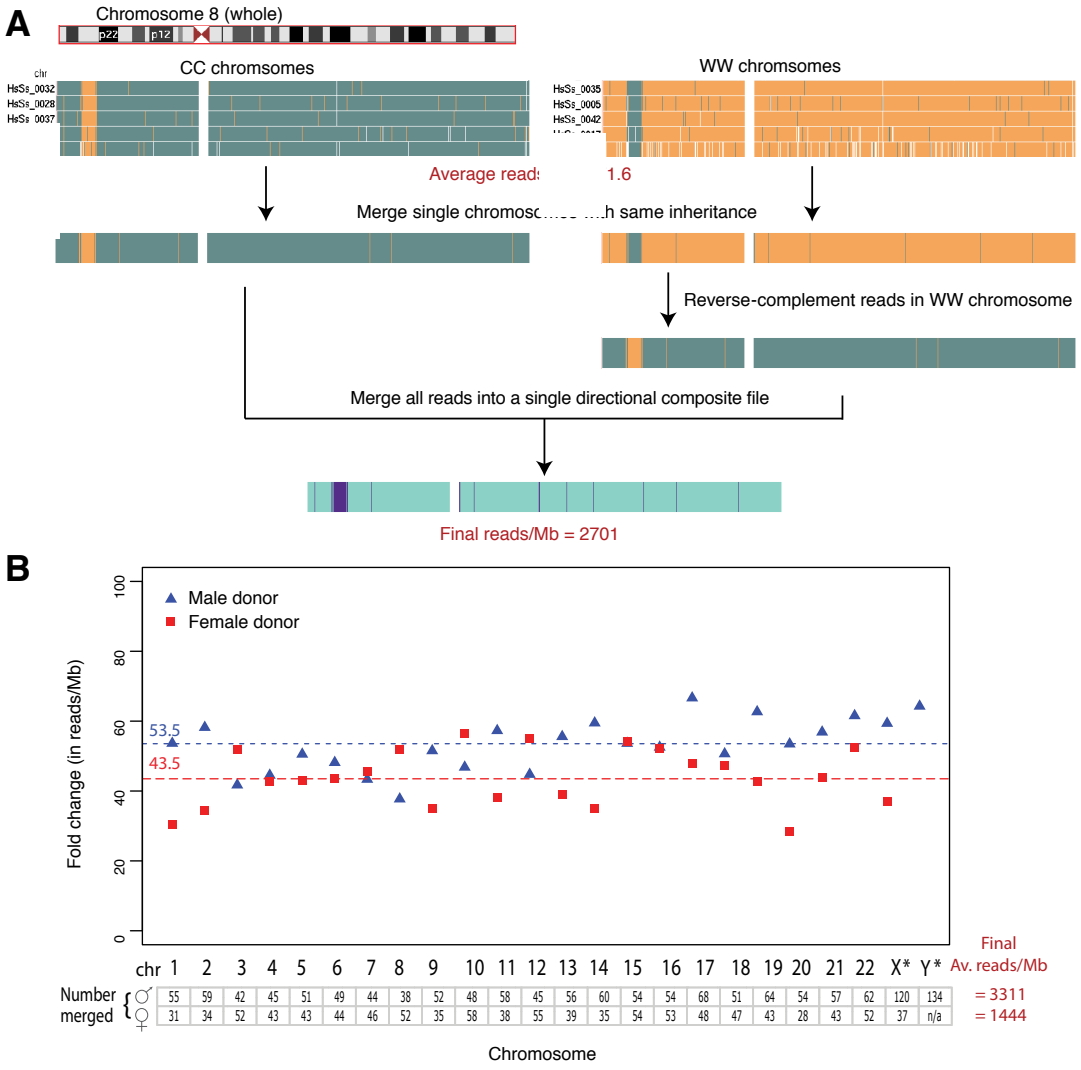
(see Methods). The heat map shows how each cell is identical to itself (diagonal line of black pixels). Cells that show similar profiles cluster together in deep red, whereas cells that are highly dissimilar are in yellow clusters. Related cells are grouped together based on their inversion profiles (e.g. upper left-hand and right-hand corners), however no two cells had an identical set of inversions, suggesting they each represent a unique individual from the pooled donor sample. Note that since only WW or CC chromosomes were included, the inversions analyzed in each cell may not represent the entire inversion load for each cell. Nevertheless, relationships between individual cells in a heterogeneous sample can be visualized by the set of inversions present in a single Strand-seq library.



Supplemental Figure S9: Allelic frequencies of polymorphic inversions in the mixed population.

The bar graph depicts the allelic frequencies found for all 111 polymorphic inversions identified in the pooled donor population. The frequency of alleles in a reference versus inverted state was calculated based on the genotypes found for each cell (see Supplemental Methods). The height of each bar represents the frequency of the inverted allele, where the proportion contributed by cells in a heterozygous state is shown in grey, and the proportion contributed by homozygous cells in black. The minimal inver-

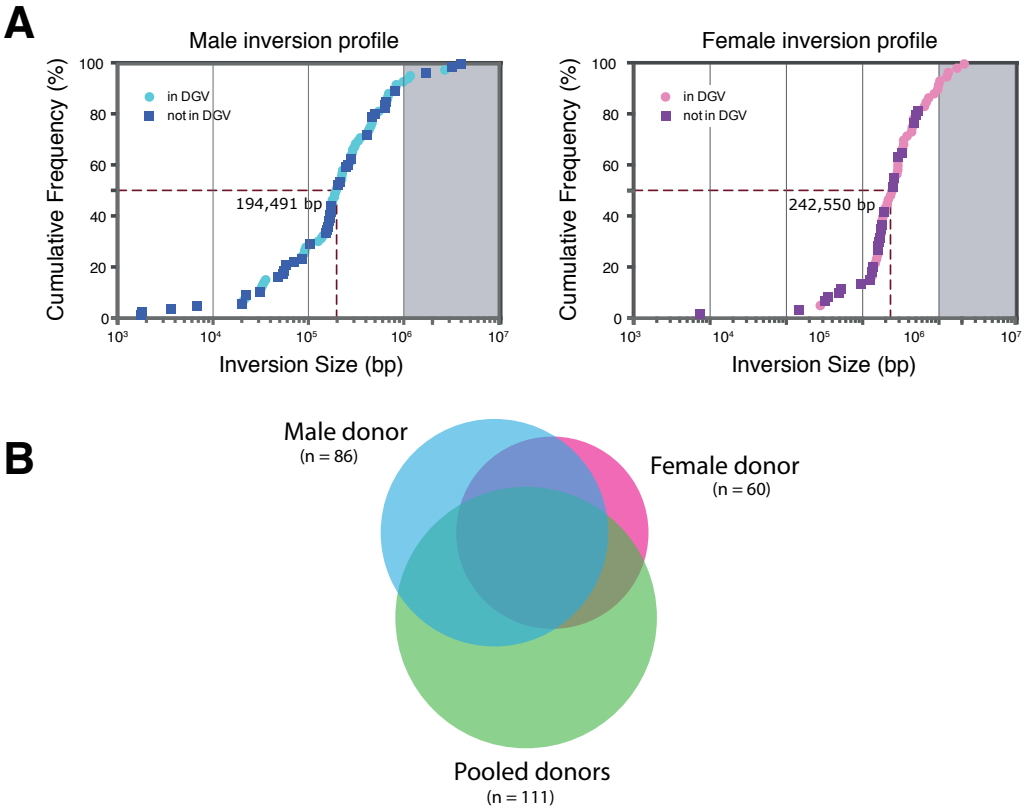
sion frequency possible in this study was limited to 0.04, based on the requirement of detecting the event in a minimum of two individual cells. Inversions with an allelic frequency > 0.5 (red dotted line) represent alleles commonly inverted in the sampled population. For additional information of the ROIs, including their genomic coordinates, see Supplemental Tables S4.



Supplemental Figure S10: Generating directional composite files from multiple Strand-seq libraries

We generated composite files from all single cell Strand-seq libraries generated for the individual donors, as illustrated for Chr 8 from the male donor. **A)** To generate the composite file, the chromosomes inherited as either WW or CC were selected and merged into two files for the chromosome. Then, the reads from the WW-file were

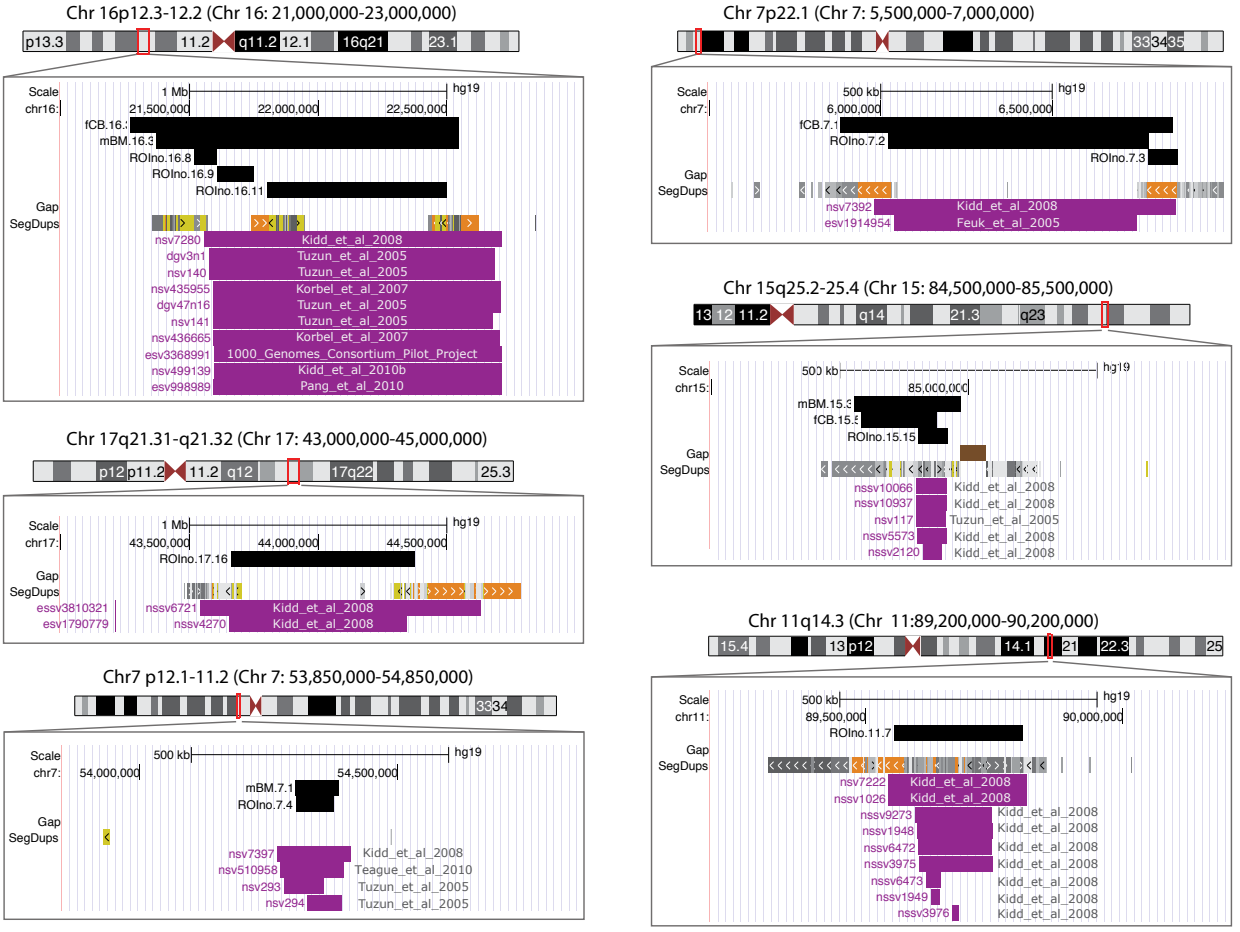
reverse complemented by flipping all '+' reads to '-' reads, and *vice versa*. This reverse-complemented file was then merged with the CC-file to generate a large composite file that has increased read depths while preserving the directionality of the data. The average read depth of the single Strand-seq libraries for Chr 8 was 71.5 reads/mega-base (Mb), whereas the final read depth of the Chr 8 composite file was 2701 reads/Mb, a 38-fold increase. **B)** The fold increases in reads/Mb calculated for the composite files of each chromosome, generated for the male (blue triangles) and female (red squares) donors, as compared to the average reads/Mb seen in the single Strand-seq cells for the corresponding chromosomes. The number of libraries merged together to generate the composite file is indicated in the table below. Note that in generating the composite file, we assume that all the cells derived from a single donor have the same inversion profile, and the composite file represents all the structural variants found in these cells.



Supplemental Figure S11: Size ranges and overlapping inversions found for each invertome.

A) Cumulative frequency of the size range of inversions (in base pairs; bp) identified for the male (left, blue) and female (right, pink) invertome. New inversions that are

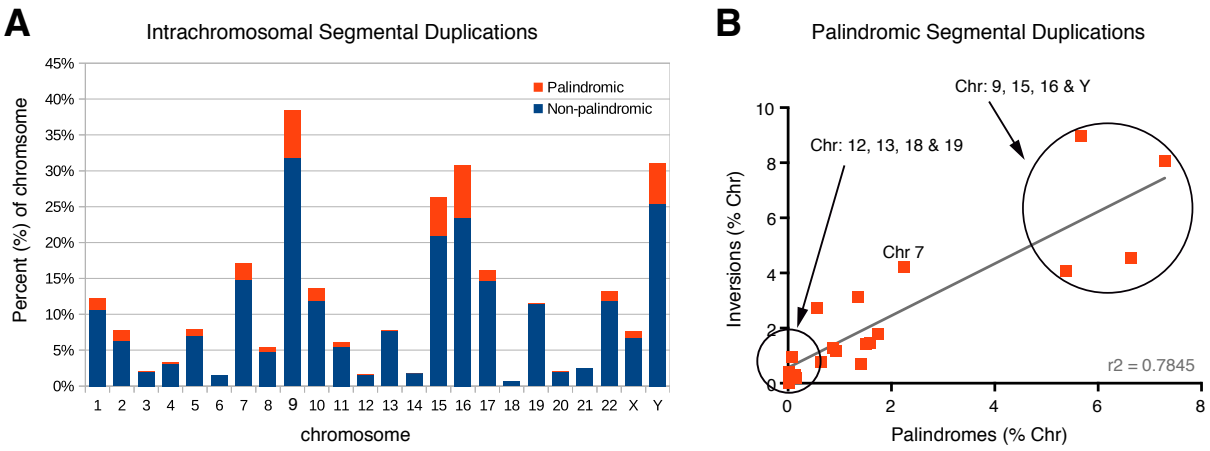
not listed in the Database of Genomic Variants (DGV) are represented by squares (dark blue for male, dark purple for female), whereas those overlapping with known inversions listed in the DGV are represented by circles (light blue for male, and light pink for female). The inversions show an even distribution of sizes, with the median size (red dotted line) indicated. The vast majority of inversions are well below 1 megabases in size (grey box), which marks the limit of detection for cytogenetic techniques commonly used to identify inversions. **B)** The Venn diagram depicts the number of inversions that overlap between the different datasets. The male invertome is shown in blue, the female invertome is shown in pink, and the total number of inversions found in the pooled donor population is shown in green.



Supplemental Figure S12: Concordance between Invert.R-predicted inversions and validated inversions

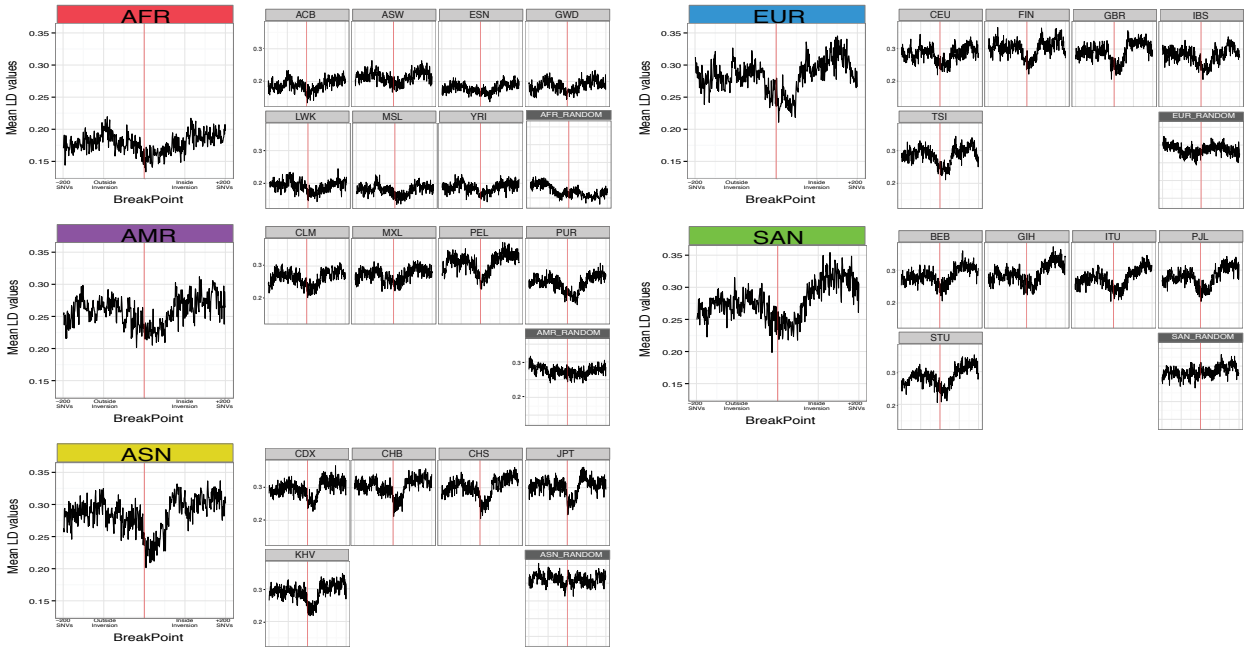
Select examples illustrating the degree of overlap between inversions predicted using Invert.R (black bars) with those published in previous studies (purple bars). The

alternative studies listed utilized combinations of: paired-end mapping, sequencing BAC clones, PCR validation, and/or FISH visualization techniques to map these inversions. Accession numbers for each variant are shown. Invert.R-predicted inversions correspond to those found for the mixed donor sample (ROIno.No; listed in Supplemental Table S4), male donor (mBM.No; Supplemental Table S5) and famale donor (fCB.No; Supplemental Table S6). Segmental duplications (SegDups).



Supplemental Figure S13: Correlation between palindromic segmental duplications and inversions

A) The percent of bases of each chromosome that are segmental duplications was determined by pulling the repeats form the UCSC Genome browser, splitting them into palindromic (inverted orientation; orange) and non-palindromic (direct orientation; blue) and then calculating the repetitive bases compared to total chromosome bases. **B**) A linear regression of the percent of inverted bases per chromosome (of the non-redundant events found in all three (i.e. from the pooled cord blood, male invertome, and female invertome) datasets) as compared to the percent palindromic segmental duplications.



Supplemental Figure S14: Levels of linkage disequilibrium at inversion breakpoints

Mean level of linkage disequilibrium (LD) calculated for neighboring single nucleotide variants (SNVs) summarized for all passing breakpoints, and plotted individually for each population from the 1000 Genomes Consortium. Each plot is centered on the breakpoint (vertical red line) with the histogram to the left representing the LD for the first 200 SNVs outside of the inversion, and the histogram to the right represents LD for the first 200 SNVs inside the inversion (see Methods for details). For comparison, LD was also calculated for 100 randomly-selected loci and plotted for each continental population (dark grey plot).

Population abbreviations: African Ancestry in Southwest US (ASW); African Caribbean in Barbados (ACB); Bengali in Bangladesh (BEB); British in England and Scotland (GBR); Chinese Dai in Xishuangbanna, China (CDX); Colombian in Medellin, Colombia (CLM); Esan in Nigeria (ESN); Finnish in Finland (FIN); Gambian in Western Division, The Gambia (GWD); Gujarati Indian in Houston,TX (GIH); Han Chinese in Bejing, China (CHB); Iberian populations in Spain (IBS); Indian Telugu in the UK (ITU); Japanese in Tokyo, Japan (JPT); Kinh in Ho Chi Minh City, Vietnam (KHV); Luhya in Webuye, Kenya (LWK); Mende in Sierra Leone (MSL); Mexican Ancestry in Los Angeles, California (MXL); Peruvian in Lima, Peru (PEL); Puerto Rican in Puerto Rico (PUR); Punjabi in Lahore, Pakistan (PJL); Southern Han Chinese, China (CHS); Sri Lankan

Tamil in the UK (STU); Toscani in Italy (TSI); Utah residents with Northern and Western European ancestry (CEU); Yoruba in Ibadan, Nigeria (YRI). Continental groups: African (AFR) - ACB, ASW, ESN, GWD, LWK, MSL, YRI; American (AMR) - CLM, MXL, PEL, PUR; Asian (ASN) - CHS, CDX, CHB, JPT, KHV; European (EUR) - CEU, FIN, GBR, IBS, TSI; South Asian (SAN) - PJL, GIH, ITU, BEB, STU

Supplemental Discussion

From a single cell, confidence in a predicted inversion depends on the total number of reads that represent the inversion, and the proportion of reads that are W or C. Compared to heterozygous alleles, homozygous inversions are easier to detect as they show a higher magnitude of change in template strand state, and are thus supported by more reads overall. Similarly, larger inversions contain a greater number of supporting reads compared to smaller events, making them easier to confidently call within a Strand-seq library (**Supplemental Fig. 2**). The likelihood of detecting an inversion in a single cell is dependent on: i) the coverage of the Strand-seq library (which impacts the number of reads representing the inversion), ii) the level of spurious background reads in the library (which makes it difficult to locate meaningful changes in template strand state), and iii) whether the chromosome was inherited as WC (as it is impossible to distinguish between homozygous reference and homozygous inversions in WC chromosomes).

In order to distinguish an inversion from sporadic rearrangements (such as sister chromatid exchanges) the template strand state change must recur at the same genomic location in a minimum of two individual cells. Consequently, the minimum number of unrelated cells required to identify an inversions is inversely correlated to the frequency of the variant in the population being studied. For instance, to uncover a very common polymorphism with a 0.1 minor allele frequency a minimum of 20 cells must be analyzed, whereas to uncover a rare inversion in a population, with a frequency < 0.01, at least 200 cells are required. Therefore, the minimal cells required can be estimated by the equation: n/MAF (where n is the minimum number of cells harboring the inversion for inclusion (i.e. 2 in this study), and MAF is the minor allele frequency of the inversion in the population being studied). Ultimately, the level of confidence for any inversion

prediction comes from the frequency of the inversion in the population being studied; each time the inversion is found independently in a single cell the overall support for the predicted variant strengthens for the whole population of cells.

When Strand-seq libraries are derived from the same individual, we expect the inversion will be evident in every cell that inherits the inverted chromosome as WW or CC (recall, homozygous inversions are masked in WC chromosomes). Given that sister chromatid segregation is random and independent, any given chromosome will exhibit a 1:2:1 segregation pattern for WW: WC: CC, making 50% of the chromosomes in a cell informative for inversion analysis. Consequently, we estimate a minimum of 7 cells ($p = 0.992$) are required to have every chromosome represented as either WW or CC at least once, and thus a minimum of 14 Strand-seq libraries will allow every inversion call to be supported by at least two cells. This represents the limit required to build an inversion profile for an individual, with the proviso that additional libraries will improve resolution and confidence, especially for small alleles.

Supplemental Methods

Data alignment:

Sequence data was aligned to GRCh37/hg19, as opposed to GRCh38/hg38, because previous inversion studies were performed on this build of the reference genome or earlier (Antonacci et al. 2009; Donnelly et al. 2010; Martinez-Fundichely et al. 2014) (Bansal et al. 2007) , making it of greatest interest to others in the community. This does not affect the conclusions of this study, as our overall approach to mapping genomic rearrangements in Strand-seq data, and the bioinformatic pipelines developed for this analysis, can be applied to data aligned to other reference assemblies.

Invert.R: a bioinformatic tool to characterize inversions in single cells

Invert.R is a custom, R-based (R Core Team 2013) software package that systematically assesses strand orientation of Strand-seq libraries (Falconer et al. 2012) to characterize any changes in strand state in single cells, and compiles this information to find patterns across multiple cells. The source code of this package is available online at (https://sourceforge.net/projects/strandseq-invertr/), and the execution file is available below, in the 'Invert.R source code, execution file 'InvertRwrap.R' section of the Supple-

mental Information.

Localizing putative inversions in single cells

To first filter chromosomes for analysis, Invert.R determines the read depth and template strand state of a single Strand-seq library at the given genomic location (either an entire chromosome or specified chromosomal locus). To do this, the program accepts a BAM or BED sequence file, and calculates the total reads/Mb to ensure a user-defined minimum read depth (minReads; e.g. 20 reads/Mb) is met at that location. It then determines the chromosome strand state by comparing the total number of Crick (C, forward, '+') reads to the total number of Watson (W, reverse, '-') and assess whether the library is predominately WW, WC or CC at this location. Note: if an ROI location is already known and has been specified then Invert.R will only assess reads outside the ROI to call the chromosome strand state. Selecting only high read depth libraries that are at least 85 % WW or CC, Invert.R starts at the first aligned read within the region and surveys a user-defined number of reads forward (bin) to count the number of W and C reads within the bin. It then calculates the ratio of W and C reads in this bin (between 0.0 – 1.0) and assigns a 'W/C ratio' to that first read. After assigning a W/C ratio, the program steps forward to the next aligned read and repeats the calculation to assign a W/C ratio sequentially to every read until the end of the genomic location is reached. A W/C ratio of 1.0 signifies all reads within the bin are in the same orientation as the chromosome strand state (either 100% W or 100% C), a W/C ratio of 0.0 signifies all the reads are in the opposite orientation to the chromosome strand state, and a W/C ratio of 0.5 means the bin contains equal numbers of W and C reads. Therefore, a change in the W/C ratio values along the genomic location represents a change in template strand orientation.

After the W/C ratio of every read is calculated, Invert.R flags putative inversions as genomic regions where W/C ratio values dip below and then return above a dynamic threshold limit that is automatically calculated based on the number of spurious background reads from that specific Strand-seq library. Library background is defined as the average W/C ratio of all reads above a defined baseline (e.g. 0.8). The threshold is then calculated as that average background minus 20%. This ensures only regions where the W/C ratio falls below 20% of the background level are flagged as putative inversions.

Additionally, a user-defined minimum number of sequential reads (e.g. 20) must remain below the threshold in order for a putative inversion to be called at the location (see below for more detail).

To predict the upstream and downstream breakpoints of the putative inversion, the program locates the nearest 5′ and 3′ flanking reads that are above the threshold and marks the outermost limits of the inversion using a modification of the SCE locator in BAIT (Hills et al. 2013). It does this by walking step-wise away from the putative inversion until it locates the nearest read that fulfill two criteria: 10 neighboring reads outside of the inversion are in the direction of the un-inverted chromosome (i.e. the chromosome strand state), and at least 25% of the 20 neighboring reads within the inversion are in the other direction. For instance, to call the 5′ breakpoint in a CC chromosome, Invert.R identifies the first 5′ read that has a W/C ratio below the threshold, and then calculates a W/C ratio for the preceding 10 reads, moving away from the inversion until the ratio is 100% C. It then checks that the ratio of the succeeding 20 reads in the 3’ direction is at least 25% W. If the test fails, it moves to the next 5′ read, and repeats the test until both conditions are met. The start location of the first read meeting both criteria is assigned as the 5′ breakpoint for the putative inversion, which defines the outermost 5′ site that the strand orientation changes. The 3′ breakpoint is identified using the same principles.

To predict the genotype of the putative inversion, the extent of change in strand orientation is determined. Here, Invert.R calculates the average W/C ratio of all reads falling between the predicted breakpoints and subtracts this from the average W/C ratio of all reads falling outside the predicted breakpoints to calculate the change in W/C ratio (Δ W/C ratio) for the putative inversion. If both homologues contain an inversion (i.e. a homozygous inversion) then strand orientation will completely switch and a Δ W/C ratio of ~ 1.0 is expected. Alternatively, if only one homologue contains an inversion (i.e. a heterozygous inversion) then a partial switch will be seen and a Δ W/C ratio of ~ 0.5 is expected. Invert.R only calls a putative inversion if ΔW/C ratio ≥ 0.3, which ensures a sufficient number of reads pass the threshold and reduces false-positive calls. The plotting function of Invert.R allows the user to visually assess the concordance of inversion calls with Strand-seq reads to manually confirm the putative inversion. For every library

interrogated, Invert.R produces a histogram of the calculated W/C ratio values, with Strand-seq reads (color-coded as C in blue, and W in orange) and reference sequence gaps (grey) plotted above, along with predicted inversions depicted below (red). The program also collapses the W/C ratios to generate a BedGraph file that can be uploaded onto UCSC Genome Browser, and writes a table of all putative inversion calls for that library for further analysis.

Finding concordant inversion predictions in multiple cells

Once Invert.R has identified the putative inversions in single Strand-seq libraries, the concordance across multiple libraries can be ascertained, to consider the frequency that a putative inversion is found at the same location in different cells. For this, Invert.R considers the amount of overlap between inversions calls made for each library in a dataset, and generates an ROI (regions of interest) list that summarizes all the cells. First, the number of overlapping inversion predictions in the dataset are found using the genomeCoverageBed function of BEDtools (v2.17.0) (Quinlan and Hall 2010), with the outer-most limits of the overlaps defined using reduce function of GenomicRanges (v2.14)(Lawrence et al. 2013). Invert.R calculates the maximum number of cells with a putative inversion called at each location, allowing users to filter the list of ROIs based on the minimum number of libraries (minLibs) having a putative inversion called in the region. Invert.R then calculates the cumulative base pair coverage of predicted inversions across all the libraries to determine the frequency that the putative inversion was called at the location. This is visualized by overlaying histograms from multiple Strand-seq libraries into a single plot, with the proportion of overlap graphically depicted as a heat map below. Invert.R also refines the inversion breakpoints by looking for consensus between inversion calls, and defines the minimum inverted region as the overlap present in at least 80% of the cells, and the maximum inverted region (which defines the outer limits of the inversion) as the overlap present in at least 20% of the cells. Invert.R outputs this as a list of ROIs, which defines the genomic coordinates of putative inversion present in the dataset.

Genotyping and allelic frequency calculations

To precisely genotype a Strand-seq library at a given ROI, the number of W and C reads in the library is counted at the ROI. If there is a user-defined minimum number of reads

(minReads, e.g. ten) present in the region, three Fisher's exact tests (one for a wildtype, heterozygous, and homozygous state) are performed independently to determine the best fit genotype. For tests of wildtype and homozygous states, a level of background (bg) is introduced when calculating the expected ratio of W and C reads for these genotypes. For example, at an ROI of a WW chromosome with 100 reads, if bg is set to 0.02 (i.e. 2% background) the expected proportion of W and C reads are: 98 W and 2 C for a wildtype state, 50 W and 50 C for a heterozygous state, and 2 W and 98 C for a homozygous state. The Fisher's exact test asks whether the observed ratio of W and C reads at the ROI are significantly different from these expected ratios, and therefore the highest p-value derived from each test is designated the best fit genotype. Significance is assigned to the genotype if the p-values of the other two tests are both below 0.05, indicating the ROI is significantly different from the other two states.

To calculate allelic frequencies in a population of cells, the proportion of genotyped cells with a wildtype, heterozygous or homozygous state are tabulated for each ROI. At diploid alleles (i.e. those on autosomes and female Chr X), frequencies are calculates as $p^2 + 2pq + q^2 = 1$. Therefore the wildtype allele frequency is found as [wtFreq = 2(wt cells) + het cells / 2(total cells)], and the inverted allele frequency is [invFreq = 2(hom cells) + het cells / 2(total cells)]. At monoploid alleles (i.e. those on the sex chromosomes of males) the frequency is calculated as $p + q = 1$. Therefore the wildtype allele frequency is (wtFreq = wt cells / total cells), and the inverted allele frequency is (invFreq = hom cells / total cells). For ROIs present on Chr X, the frequencies of the males and females are combined as $p = 2/3p^{female} + 1/3p^{male}$.

Invert.R source code

```
#' Wrapper function for InvertR
#'
#' This script will move through .bam or .bed files in a folder and perform several steps (see
Details).
#'
#' 1. calculate the WCratio chromosome-by-chromosome
#' 2. Locate the ROIs in chromosomes passing the WCcutoff
#' 3. write a bedgraph file of wcRatios -> can upload on to UCSC Genome browser
#' 4. write an ROI file for each index with all chromosomes included
#'
#' @import collapseStrands.R
#' @import countFreqs.R
#' @import findROILocation.R
#' @import plotgaps.R
#' @import plotROIFrequencies.R
#' @import processBam.R
#' @import processBed.R
#' @param regionTable Genomic coordinates to be analyzed (ROI list or Chr Table)
#' @param dataDirectory Output directory. If non-existent it will be created
#' @param binSize The number of reads in each bin used to calculate wcRatio
#' @param WCcutoff The number of watson or crick reads used to define chrStates
#' @param gapfile Input txt file of gaps in the genome
#' @param type File input type, either 'bed' or 'bam'
#' @param dup If \code{TRUE}, removes duplicate reads
#' @param qual Filter reads based on specified qulaity score
#' @param padding Number of bases to extend beyond genomic coordinates listed in regionTable
#' @param verbose If \code{TRUE} Verbose messages
#' @param strand If \code{TRUE} Plot the crick and watson reads above the histrogram
#' @param png If\code{TRUE} Generates a png figure of each file
#' @param findROIs If\code{TRUE} Runs findROIlocations to locate putative inversions
#' @param ROI If\code{TRUE} Expects ROI list, if\code{FALSE} Expects chromosome table
#' @param minDepth The minimum number of reads/Mb required to analyze the chromosome
#' @param minReads The minimum number of reads within the ROI required for inclusion
#' @author Ashley D. Sanders, Mark Hills
#' @export

runInvertR <- function(regionTable, binSize=50, WCcutoff=0.75, dataDirectory='./InvertR_
analysis/', gapfile=0, type='bed', dup=TRUE, qual=10, padding=0, minDepth=20, minReads=20,
verbose=TRUE, png=TRUE, strand=TRUE, ROI=FALSE, genotype=TRUE, findROIs=T)
{#
  options(warn=-1)
  if(type == 'bam') {library(Rsamtools)}

  dir.create(dataDirectory)
  fileDestination <- dataDirectory

  #for every chromosome...
  for(i in seq(1,nrow(regionTable)))
  {##
    ch <- regionTable[i,1]
    chr <- paste('chr', regionTable[i,1], sep="")
    startLoc <- regionTable[i,2]
    endLoc <- regionTable[i,3]
    if (chr == 'chrY') { WCcutoff = 0}
    ## NOTE change WCcutoff=0 if chrY (b.c. cannot have a WC chr, and may have large inversions
(e.g. cad11) which would be missed if wcCutoff high)

    if (ROI == TRUE)
    {
```

```
      ROIname <- paste('ROINo.', i, sep="")
      dir.create(paste(fileDestination, ROIname, sep=""))
      chrfileDestination <- (paste(fileDestination, ROIname, '/', sep=""))
      dir.create(paste(fileDestination, 'WCLibs', sep=""))
      padding<- round((endLoc-startLoc)*0.33, digits=0)
    }else{
      ROIname <- 'wholeChr'
      dir.create(paste(fileDestination, chr, sep=""))
      chrfileDestination <- (paste(fileDestination, chr, '/', sep=""))
      dir.create(paste(fileDestination, 'WCLibs', sep=""))
      padding<- 0
    }


    pattern <- paste('.', type,'$', sep="")
    fileList <- list.files(path='.', pattern=pattern, full.names=TRUE)
    fileLength <- length(fileList)
    indexCounter <- 0
    options(scipen=20)
    #for reading in multiple files at a particular location
    allFrequencies <- data.frame(index=vector(), rname=vector(), pos=vector(), strand=vector(),
mapq=vector(), WCratio=vector(), chrState=vector())
    #allROIlocationTable <- data.frame(index=vector(), chr=vector(), ROIstart=vector(),
ROIend=vector(), deltaWC=vector(), roiReadDepth=vector())
    allROIlocationTable <- data.frame(index=vector(), chr=vector(), callingTh=vector(),
ROIstart=vector(), ROIend=vector(), ROIsize= vector(), deltaWC=vector(), roiReads=vector())
    wcLibraries <- data.frame(index=vector(), chr=vector(), wcCall=vector())

    #for every filename....
    for(fileName in fileList)
    { ###
      indexCounter <- indexCounter + 1
      if(verbose==T){message(paste('** RUNNING ', fileName, ' [lib.No ', indexCounter,'/',
fileLength, '], ', ' chromosome [', ch, '/', nrow(regionTable), '] **', sep=""))}
      index <- basename(fileName)

      #read in files; either using processBed or processBam
      if(type == 'bed')
      {
        tempFile <- processBed(startLoc, endLoc, chr, fileName, qual=qual, rmdup=dup,
padding=padding, verbose=verbose)
        chrState <- tempFile[[2]]
        if(chrState >= WCcutoff) {chrState<-'ww'}else if(chrState <= -WCcutoff){chrState<-'cc'}
else{chrState<-'wc'}
        # if chrState is Negative chr is CRICK/CRICK
        processFile <- tempFile[[1]]
      }else if(type == 'bam') {
        tempFile <- processBam(startLoc, endLoc, chr, fileName, qual=qual, rmdup=dup,
padding=padding, verbose=verbose)
        chrState <- tempFile[[2]]
        if(chrState >= WCcutoff) {chrState<-'ww'}else if(chrState <= -WCcutoff){chrState<-'cc'}
else{chrState<-'wc'}
        # if chrState is Negative chr is CRICK/CRICK
        processFile <- tempFile[[1]]
        processFile<- cbind(chr, processFile[2:length(processFile)]) # pastes chr instead of ch
to file
        processFile<- processFile[!duplicated(processFile[2]),]
      }
      if(verbose==T){message(paste('-> bedFile generated for ', index, ', chromosome ', ch,
sep=""))}

      if(length(processFile[[1]]) > 1)
      {####
        #Filters out low (< minDepth) read depth libraries. If enough reads are in this library,
proceed...
        if(length(processFile[[1]]) > 1 && nrow(processFile)/((endLoc-startLoc)/1000000) >
minDepth)
```

```
        {
          #calculate the ratio of _ to + reads (i.e. the wcCall)
          if (ROIname == 'wholeChr'){
            wcCall <- round((table(processFile$strand)[2]-table(processFile$strand)[1])/
nrow(processFile), digits=3)
            #wcCall <- chrState
          }else{
            tempFile <- processFile[which(processFile$pos < startLoc),]
            tempFile <- rbind(tempFile, processFile[which(processFile$pos > endLoc),])
            #filters reads that flank the ROI to calculate the wcCall of these surrounding reads
(since an inversion at the ROI will impact the wcCall)
            wcCall <- round(( table(tempFile$strand)[2]-table(tempFile$strand)[1] ) /
nrow(tempFile), digits=3)
          }

          if( is.na(wcCall)) {wcCall <- 1}

          ######
          # wcCall can become NA if 100% of reads are + or -
          if(wcCall != 'NaN')
          {
            if(wcCall <= -WCcutoff | wcCall >= WCcutoff)
            { ##This is a pure (WW or CC) library...

              # calculates A WCratio value FOR EACH READ based on the proportion of W and C in
(binSize #) succeeding reads
              fileFrequencies <- countFreqs(processFile, checkNum=binSize, verbose=FALSE)
              if(verbose==T){message(paste('-> fileFrequencies counted for ', index, ' file,
chromosome ', chr, sep=""))}
              if(length(fileFrequencies) > 1 && nrow(fileFrequencies) > 2)
              {
                # reduces the table size by identifying only the locations where the WCratio
values change
                outputFile <- collapseStrands(fileFrequencies, index, asBedgraph=TRUE)

                if(verbose==T){message(paste('-> strands collapsed for ', index, ' file,
chromosome ', chr, sep=""))}

                #find the location of ROIs that dip below threshold level
                fileFrequencies <- cbind(fileName, fileFrequencies)
                fileFrequencies<- cbind(fileFrequencies, chrState)
                allFrequencies <- rbind(allFrequencies, fileFrequencies)

                if(findROIs==T){ # if true then run findROILocation script, else ROIlocationTable
=1
                  #minReads specifies the minimum number of reads within the roi that are required
to include it in the ROIlocationTable list
                  locationFile <- findROILocation(outputFile, fileFrequencies, chrState=chrState,
verbose=verbose, baselineThreshold=0.8, minReads=minReads)
                  ROIlocationTable <- locationFile[[1]]
                  # ROIlocationTable <- data.frame(index=vector(), chr=vector(),
callingThreshold=vector(), ROIstart=vector(), ROIend=vector(), ROIsize= vector(),
deltaWC=vector(), roiReads=vector())
                  Th <- locationFile[[2]]
                }else{ROIlocationTable = 1
                      Th<- 1}

                if(length(ROIlocationTable) != 1)
                {
                  if(verbose==T){message(paste('-> Total of: ', nrow(ROIlocationTable), ' ROIs
found for ', index, ' file, chromosome ', chr, sep=""))}
                  allROIlocationTable <- rbind(allROIlocationTable, ROIlocationTable)
                  deltaWC<-  ROIlocationTable[1,7]

                } else { deltaWC <- 0 }
                #if(verbose==T){message(paste('-> NO ROIs found for ', index, ' file, chromosome
', chr, sep=""))}
```

```
                ### generates a png  of the single Ss library with gaps and ROI locations
highlighted, also calculates the number of reads in the region -> table
                plotROIFrequencies(chrfileDestination, index, fileFrequencies, chr, binSize,
startLoc, endLoc, ROIname=ROIname, gapfile=gapfile, callingThreshold=Th, ROI=ROIlocationTable,
padding=padding, strand=strand, png=png)

                ###### write a bedfile of the reads, and then append the bedgraph
                bedfile<- cbind(chr,  processFile$pos, processFile$pos+100, index,
processFile$mapq, as.character(processFile$strand))
                head<- paste('track name=', index, '_reads_', chr, ' visibility=1
colorByStrand="103,139,139 243,165,97"', sep="")
                write.table(head, file=paste(chrfileDestination, index, '_', chr, '_(b=', binSize,
', t=', Th, ').bedgraph', sep=""), row.names=FALSE, col.names=F, quote=F, append=F)
                write.table(bedfile, file=paste(chrfileDestination, index, '_', chr, '_(b=',
binSize, ', t=', Th, ').bedgraph', sep=""), row.names=FALSE, col.names=F, quote=F, append=T)

                ####  bedgraph of the collapsed WCratios for the single Ss library -> can be
uploaded onto the ucsc genome browser
                write.table(outputFile, file=paste(chrfileDestination, index, '_', chr, '_(b=',
binSize, ', t=', Th, ').bedgraph', sep=""), row.names=FALSE, quote=FALSE, col.names=FALSE,
append=T)
              }
            } else {
              message(paste('~>  ', index, ' is WC for ', chr, ' - moving on to next lib',
sep=""))
              wcLibrary <- cbind(index, chr, round(wcCall, digits=3))
              wcLibraries <- rbind(wcLibraries, wcLibrary)
            }
          } else { if(verbose==T){message(paste('~>  ', index, ' wcCall = NaN for ', chr, ' -
moving on to next lib', sep="")) } }
          ######
        } else { if(verbose==T){message(paste('~~>  ', index,  'read count below minReads, moving
on to next lib', sep="")) } }
        ####
      } else { if(verbose==T){message(paste('~~>  ', index, ' has no reads in the region - moving
on to next lib', sep="")) } }
    }###

    if(findROIs==T){
    write.table(allROIlocationTable, file=paste(fileDestination, 'ROI_locations_Table_b', binSize,
'_', chr, '.txt', sep=""), row.names=FALSE, quote=FALSE, append=FALSE)  }

    write.table(allFrequencies, file=paste(fileDestination, 'allFrequencies_b', binSize, '_', chr,
'.txt', sep=""), row.names=FALSE, quote=FALSE, append=FALSE)
    write.table(wcLibraries, file=paste(fileDestination, '/WCLibs/wcLibraries_', chr, '.txt',
sep=""), row.names=FALSE, quote=FALSE, append=FALSE)

    # calculate Stats for the plots:
    if (nrow(allROIlocationTable) != 0){
      AvTh <- round(mean(allROIlocationTable[,3]), digits=2)
    }else{
      AvTh <-0
      allROIlocationTable<-0}

    if(nrow(allFrequencies) < 1){ allFrequencies <- data.frame("fileName", chr, startLoc, endLoc,
'+', 0, 1, 'cc' )}
    plotROIFrequencies(fileDestination, 'overlay', allFrequencies, chr, binSize, startLoc, endLoc,
ROIname=ROIname, gapfile=gapfile, callingThreshold=AvTh, ROI=allROIlocationTable, padding=padding,
strand=FALSE, png=png)

    if(verbose==T){message(paste(' ~~ Overlaid plot generated for ', chr, ' *YIPEE!* moving on to
next chromosome...', sep=""))}
  }##

}#
```

Supplemental References

Antonacci F, Kidd JM, Marques-Bonet T, Ventura M, Siswara P, Jiang Z, Eichler EE. 2009. Characterization of six human disease-associated inversion polymorphisms. Human molecular genetics 18(14): 2555-2566.

Bansal V, Bashir A, Bafna V. 2007. Evidence for large inversion polymorphisms in the human genome from HapMap data. Genome research 17(2): 219-230.

Donnelly MP, Paschou P, Grigorenko E, Gurwitz D, Mehdi SQ, Kajuna SL, Barta C, Kungulilo S, Karoma NJ, Lu RB et al. 2010. The distribution and most recent common ancestor of the 17q21 inversion in humans. American journal of human genetics 86(2): 161-171.

Falconer E, Hills M, Naumann U, Poon SS, Chavez EA, Sanders AD, Zhao Y, Hirst M, Lansdorp PM. 2012. DNA template strand sequencing of single-cells maps genomic rearrangements at high resolution. Nature methods 9(11): 1107-1112.

Hills M, O'Neill K, Falconer E, Brinkman R, Lansdorp PM. 2013. BAIT: Organizing genomes and mapping rearrangements in single cells. Genome medicine 5(9): 82.

Lawrence M, Huber W, Pages H, Aboyoun P, Carlson M, Gentleman R, Morgan MT, Carey VJ. 2013. Software for computing and annotating genomic ranges. PLoS computational biology 9(8): e1003118.

Martinez-Fundichely A, Casillas S, Egea R, Ramia M, Barbadilla A, Pantano L, Puig M, Caceres M. 2014. InvFEST, a database integrating information of polymorphic inversions in the human genome. Nucleic acids research 42(Database issue): D1027-1032.

Quinlan AR, Hall IM. 2010. BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics 26(6): 841-842.

R Core Team. 2013. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.