## Supplementary Methods

### Data acquisition and alignment

TCGA whole genome sequence datasets generated by Illumina paired-end sequencing were downloaded from dbGAP in the form of BAM alignment files. The BAM files were generated by the TCGA consortium by aligning raw paired-end reads (readpairs) to the reference genome (hg18 or hg19) using either MAQ (Li et al. 2008) or BWA (Li and Durbin 2009). Since most TCGA datasets were generated using multiple sequencing libraries, which have distinct insert size distributions, data processing was conducted on a per read-group basis. To gather read-group statistics, 50 million reads were extracted from each BAM file, and for each read-group the following statistics were collected: number of reads present, average read length, edit distance, insert size and insert size standard deviation (**Supplemental Table 1**). These statistics were used in subsequent alignment and processing steps, which were performed on the read-groups separately.

For each read group, concordant readpairs were defined as those with an insert size not more than 6 standard deviations from the mean insert size. This was done regardless of readpair orientation, and thus the resolution for detecting tandem duplications and inversions is the same as for deletions (roughly 400bp). Discordant readpairs for which both reads possessed an alignment to the reference genome were extracted from the original BAM file using SAMTools (Li et al. 2009) and converted to FASTQ using the HYDRA utility *bamToFastq (*http://code.google.com/p/hydra-sv/). FASTQ files were checked to ensure that they contained two entries for each read ID, in the correct order.

Discordant readpairs were aligned to the reference genome (1000 Genomes Version of NCBI Build 37: ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/human_g1k_v37.fasta.gz) with Novoalign v.2.07.10 (C. Hercus, unpublished; http://www.novocraft.com), using an index of word size 14 and a step of 1 (-k 14 –s 1). Repetitive alignments were resolved using random mode and the read-group statistics were used to obtain the approximate fragment lengths and standard deviations (-r Random -i <mean> <standard deviation>). This more sensitive alignment step helps to eliminate false positives by removing concordant mappings missed by the original BWA or MAQ alignment.

### Pairing, duplicate removal and library classification

Discordant mappings were converted to BEDPE format (Quinlan and Hall 2010) using custom scripts. The BEDPE read-group files from each dataset were concatenated into a single file for duplicate

removal. Duplicates were removed using the custom script, *dedupDiscordantsMultiPass.py*, which is available as part of the HYDRA suite (http://code.google.com/p/hydra-sv/). This utility allows for removal of duplicates with inexact coordinates, which is not possible with Picard or SAMTools. We considered readpairs to be duplicates if both end coordinates were within 3bp (-s 3) of each other. Following duplicate removal, the combined files were then separated back into their previous constituent read-group files.

Since multi-sample variant detection with HYDRA-MULTI requires knowledge of each sequencing library, the duplicate-free read-group files were then classified into their approximate initial genomic libraries using insert size statistics. If both the insert size mean and standard deviation of two read-groups were within 90% of each other, the two read-groups were merged into a single library. The average and standard deviation of the merged library insert sizes were then calculated with a weighted average. The resulting updated library statistics and library BEDPE files were then iteratively compared against each read-group by decreasing insert size until read-groups were no longer merged. The resulting library BEDPE files and newly generated library statistics were then used as input to HYDRA-MULTI.

**Breakpoint detection and filtering**

Breakpoint detection was performed using HYDRA-MULTI, a new multi-sample version of the HYDRA paired-end mapping algorithm (Quinlan et al. 2010) (manuscript in preparation) that uses a similar population-based breakpoint detection method to our previous method (Quinlan et al. 2011). In essence, all discordant mappings are pooled prior to breakpoint calling, and presence/absence genotypes are calculated retrospectively based on the number of readpairs from each dataset that were incorporated into the breakpoint call. A HYDRA-MULTI configuration file was prepared detailing the insert size distribution of each of the 377 total sequencing libraries from the 129 datasets.

A total of 4,686,652 breakpoints were predicted, which is far more than expected. This is largely due to a previously unreported library preparation artifact that produces a profuse number of false small (<10kb) inversion calls. These small inversion calls were judged to be false based upon the following observations: 1) they are found equally among tumor and normal datasets; 2) different datasets vary widely in the number of calls; and 3) hierarchical clustering of samples based on small inversions revealed no genetic relationship between tumor/normal pairs, whereas hierarchical clustering using other breakpoint classes produced tight clustering between tumor/normal pairs. False

inversion calls are not due to incomplete removal of duplicate readpairs, and we confirmed the presence of the discordant readpairs underlying these false inversion calls in the original BAM files downloaded from TCGA. We are unaware of the precise molecular cause of this artifact, but we suspect that it involves systematic formation of chimeric molecules early during library generation, perhaps as a consequence of intra- or inter-molecular annealing between tracts of microhomology. Removal of small inversions resulted in a total of 1,636,145 HYDRA-MULTI calls.

Paired-end mapping is prone to false positives due to read mapping artifacts and reference genome assembly errors, and thus we also required breakpoints calls to fulfill the following criteria: 1) at least 3 readpairs support the call; 2) the reads have a mean mapping quality greater than 30; 3) the reads have a mean number of mappings less than 1.5; 4) the variant call is at least 100bp in size; and 5) neither end of the call overlaps simple or satellite repeats by more than 50% (*bedtools pairtobed -type either -f 0.5*), as defined by a union of the UCSC "simpleRepeat" track and the simple and satellite repeat annotations present in the "RepeatMasker" tracks (combined using *bedtools merge*). These filtering steps resulted in 34,621 high confidence calls, 6179 of which were judged to be somatic by their presence in one tumor sample and none of the remaining 128 samples.

**Assembly and validation**

To enable efficient assembly of somatic breakpoints, we modified the *sga walk* function from the String Graph Assembler (SGA) (Simpson and Durbin 2012) to report all walks from all connected components of the string graph, with advice from the developer (J. Simpson). For each HYDRA-MULTI call, we extracted readpairs that mapped within 500bp of the predicted breakpoints, including readpairs with one unmapped read. We then ran the following assembly pipeline: *sga preprocess* (default), *sga index* (--no-reverse), *sga correct* (-k 13 -x 2 -d 128), *sga index* on the error corrected reads (default), *sga rmdup* (default), *sga overlap* (-m 15), *sga assemble* (-m 15 -d 0 -g 0 -b 0 -l 100), and our modified version of the *sga walk* program (-d 10000 --component-walks).

Resulting contigs were aligned to the reference genome using BWA-SW (v.0.5.9) (Li and Durbin 2010). Split-mappings with at least 25bp of non-overlap with an adjacent mapping on the query sequence were extracted from the BAM file and converted to BEDPE format. We then used *bedtools pairtopair* (-type both) from the BEDTools software suite (Quinlan and Hall 2010) to assess whether HYDRA-MULTI calls were validated by split-mapping contigs. We considered a call to be validated if the

breakpoints predicted by split-mapping fell within the 200bp predicted breakpoint interval defined by HYDRA-MULTI, and if split-mapping and HYDRA-MULTI predicted the same variant class (e.g., deletion).

**Comparison to 1000 Genomes deletions**

To assess the number of known SVs within our dataset, we used a set of deletion calls from the 1000 Genomes project (Mills et al. 2011). This callset is a combination of validated deletion calls identified from the high coverage trios, validated deletion calls from the low coverage individuals, and the set of "gold standard" deletions that were used to tune the 1000 Genomes calling algorithms. There are 60069 entries in this file, but many are redundant calls generated by distinct centers. After merging calls with *mergeBed*, there were 14368 non-redundant deletions. To compare our deletion calls to this high-quality validated set, we used *bedtools intersect* requiring 50% reciprocal overlap (-r -f 50).

**Identification of breakpoint clusters**

CGRs were identified using a custom method composed of two steps (**Fig. 2A**). First, somatic breakpoint calls from a single sample that were found within 100kb of each other were merged using *clusterBed* from the BEDTools suite. Second, "chains" were formed from genomic loci sharing breakpoint calls in common. Initially, a chain is formed when two loci are linked to one another by one or more HYDRA-MULTI calls. Loci are chained together through an iterative process by looping through the loci and updating the composition of chain as they grow due to the addition of new loci. At each iteration, the locus is added to a pre-existing chain if that locus shares an SV breakpoint call with any other locus in that chain. Moreover, if that same locus also shares a SV call with one or more additional (previously unlinked) chain, all chains linked by that locus are merged into a single chain. The end result of this simple algorithm is that all loci within a chain are linked to one another by a series of HYDRA-MULTI breakpoint calls, and no two loci from distinct chains are linked by a call. "Breakpoint clusters" were defined as chains comprising 3 or more breakpoints.

We then merged breakpoint clusters that were within 1mb of each other. This last step was added to minimize fragmentation, where subsections of the same apparent rearrangement may be reported separately due to false negative breakpoint calls.

**Simulation of breakpoint cluster identification**

We performed four simulations to test the specificity of breakpoint cluster identification. In all cases we identified clusters exactly as for the real data and report the mean of 100 replicates. For the Monte Carlo simulation, the genomic coordinates of tumor-specific somatic breakpoints were randomly shuffled using *bedtools shuffle*, excluding assembly gaps. To emulate the filtering of HYDRA-MULTI breakpoint calls, breakpoints were only placed within uniquely-mappable regions of the genome, as defined by the UCSC wgEncodeCrgMapabilityAlign100mer track, and were not allowed to overlap with simple or satellite repeats. For the other simulations, a random subset of 1000 Genomes calls, validated germline SVs, and "rare" individuals specific SVs were sampled to match the number of tumor-specific somatic SVs identified in each sample.

**Genome annotation enrichment analysis**

To assess whether breakpoint clusters are enriched in SV hotspots or genomic regions prone to read mapping artifacts, we assessed overlap between our breakpoint calls and segmental duplications (Bailey et al, 2001), known fragile sites in the human genome (Fungtammasan et al. 2012), simple repeats, microsatellites, and assembly gaps. For each comparison, we counted the observed number of overlaps between breakpoints and the above genome annotations. Using pybedtools (Dale et al. 2011), we then conducted a Monte Carlo simulation by shuffling the genomic locations of each breakpoint and each annotation 100 times, and for each iteration we counted the number of overlaps found using the randomized locations. We assessed potential enrichments between breakpoints and genome annotations by measuring the log2 ratio of observed count of intersections versus the median intersection count from the 100 MC simulations.

**Read-depth analysis**

We used *bedtools coverage* from the BEDTools software suite to measure read depth in genomic windows containing 5kb of uniquely mappable sequence, as defined by the wgEncodeCrgMapabilityAlign100mer track. There were a total of 534,957 windows with a mean size of 5787bp. We corrected for GC bias using a normalization procedure that expresses copy number as a Z-score, as described previously (Quinlan et al. 2010). This Z-score is simply the number of standard deviations that the read-depth of a given window differs from the mean read-depth of all other windows that have a similar GC content (defined in 1-3% intervals), as calculated by fitting a normal

distribution. To estimate the approximate copy number of a genomic window we divided the raw read depth of that window by the median raw read-depth of all other windows with a similar GC-content, and multiplied by 2. While this calculation assumes diploidy and tumor homogeneity, which may not be valid for many tumors, the resulting estimates are sufficient to estimate the number of CNA states (if not their true absolute value). To detect CNAs we performed circular binary segmentation (Olshen et al. 2004) of Z-scores using the DNAcopy package in R with the following parameters: undo.splits="sdundo" and undo.SD=2 (http://cran.r-project.org/). We defined CNA change-points as the interval between two adjacent CNA segments (+/- 5kb) whose median Z-score differed from each other by at least one median absolute deviation. We defined somatic change-points as those that were found in a single tumor sample but not in any of the 65 normal sample, requiring 100% reciprocal overlap between change-points (*bedtools intersect* -r -f 1*)* and the same direction of copy number change.

For the analysis of overlap between CNA change-points and HYDRA-MULTI breakpoint calls (**Supplementary Table 7**), we identified overlapping calls using *bedtools intersect*, allowing 10kb of bi-directional slop. This is fairly strict given the relatively low resolution and imprecision of read-depth analysis. To assess enrichment we divided the observed number of breakpoints that overlapped CNA change-points by the mean number of overlaps identified using 100 sets of randomly shuffled breakpoint calls. These were the same shuffled calls used in the Monte Carlo simulation to test the specificity of CGR detection, as described above. For the comparison of CNA change-points and breakpoint clusters (**Supplementary Fig. 4C**), a breakpoint call was judged to overlap a cluster if it was found within 50kb, and breakpoint clusters were randomly shuffled 100 times to obtain enrichment values, as described above for breakpoint calls.

For estimation of copy number states, we extracted all CNA change-points within 100kb of a breakpoint cluster. We used this generous definition in order to compensate for imprecise change-point detection and false negative HYDRA-MULTI breakpoint calls, thus helping to ensure that the number of CNA states at a breakpoint cluster was not underestimated (which could lead to misclassification of stepwise rearrangements as one-off CGRs). To estimate the number of copy number states at breakpoint clusters we used a greedy clustering algorithm that operates on a sorted list of predicted copy numbers taken from change-points. The algorithm merges the values into a group if the smaller value is at least 80% of the larger value, and then recalculates the copy number by taking the mean of the values in the group. The only exception is that the two copy number values for

a given change-point cannot be placed into the same group. When this happens, a new group is initiated and the process is repeated for the remaining values. We chose this greedy algorithm after testing more conventional methods including k-means clustering, hierarchical clustering, and kernel density estimation. These methods routinely underestimated the number of copy number states at a nontrivial fraction of breakpoint clusters, leading to misclassification of stepwise rearrangements as one-off CGRs.

**Monte Carlo simulation of progressive rearrangement**

To assess the likelihood that an extreme CGR was due to one-off rather than stepwise mutation, we performed a simulation based on the method of Stephens et al. (2011). For each observed CGR, we performed a Monte Carlo simulation in which the observed SV breakpoint calls were applied in random order to a progressively mutated synthetic chromosome. We estimated the probability that the observed CGR is caused by stepwise rearrangement (the null hypothesis) by dividing the number of simulation runs that produced the same or fewer copy number states as observed in the real data by the total number of successful simulation runs.

To perform simulations we used a modified version of SVsim (G. Faust, unpublished), a structural variation simulator. We simulate rearrangements on multiple chromosomes, but we do not allow rearrangements between chromosomes. We use a diploid genome for our simulations to more accurately mirror natural conditions and to help mitigate the loss of genomic regions via deletions. This is more conservative than a haploid simulation in that it generally results in fewer CNA states. During the simulation, we take into account the orientation of mutated chromosomal segments when determining the relationship between readpair orientation and event type. To select breakpoint locations within multi-copy regions generated by a prior duplication, we randomly select one of the breakpoint loci, and then select the second locus that is closest to it on the mutated chromosome. If a breakpoint cannot be applied due to a prior deletion, we attempt to apply the rearrangement to the homologous chromosome; if it cannot be applied to the homolog, we abort the simulation run and try again. At the end of each successful run, we count the number of distinct copy number states across the entire mutated genome. As our ability to observe copy number states in actual data is restricted to the resolution of our read-depth analysis, we only count states in our simulations that appear in regions exceeding 10kb in length. We continue this process until 1000 successful simulation runs have completed for each CGR.

**Identification of templated insertion events**

To identify templated insertion events and small-scale rearrangements at SV breakpoints, we examined contigs whose split-alignments were found to validate HYDRA-MULTI calls, but that contained at least 20bp of unaligned sequence directly at the breakpoint, as determined by BWA-SW. We then aligned these contigs to the reference genome with YAHA (Faust and Hall 2012), an algorithm that allows for highly sensitive determination of split alignments (options: kmer size 15, -M 15 -P 0.8 -H 5000). Alignments were visualized with a modified version of PARASIGHT (J. Bailey and E. Eichler, unpublished: http://eichlerlab.gs.washington.edu/jeff/parasight) and scrutinized for insertions derived from elsewhere in the genome, as well as for small-scale rearrangements directly at the breakpoint.

**Estimating breakpoint allele frequencies**

To measure the intra-tumor allele frequency of breakpoint, we aligned all the raw reads from each dataset to the assembled breakpoint-containing contigs. For each contig we extracted the 200bp of sequence flanking the breakpoint and aligned raw reads to the 200bp fragments using BWA (default options). To consider an alignment as positively genotyping the presence of the variant allele, we required that it spanned the breakpoint with at least 20bp on both sides. To genotype the reference allele, we extracted the 200bp flanking each of the two breakpoint positions in the reference genome, and performed alignment as above. The intra-tumor breakpoint allele frequency is given by the number of reads that align to the variant junction (breakReadCount) divided by the average number of reads aligning to the respective reference junctions (refReadCount1 and refReadCount2): breakReadCount / (breakReadCount + ((refReadCount1 + refReadCount2) / 2)). In order to consider allele frequency measurements as sufficiently precise for comparing simple and complex variants, we required that at least 3 reads identified the variant allele.

## References

Dale, R.K., Pedersen, B.S., and Quinlan, A.R. 2011. Pybedtools: a flexible Python library for manipulating genomic datasets and annotations. *Bioinformatics* **27**(24): 3423-3424.

Faust, G.G. and Hall, I.M. 2012. YAHA: fast and flexible long-read alignment with optimal breakpoint detection. *Bioinformatics*.

Fungtammasan, A., Walsh, E., Chiaromonte, F., Eckert, K.A., and Makova, K.D. 2012. A genome-wide analysis of common fragile sites: what features determine chromosomal instability in the human genome? *Genome Research* **22**(6): 993-1005.

Li, H. and Durbin, R. 2009. Fast and Accurate Short Read Alignment with Burrows-Wheeler Transform. *Bioinformatics*.

-. 2010. Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics* **26**(5): 589-595.

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., and Durbin, R. 2009. The Sequence Alignment/Map format and SAMtools. *Bioinformatics* **25**(16): 2078-2079.

Li, H., Ruan, J., and Durbin, R. 2008. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res* **18**(11): 1851-1858.

Mills, R.E., Walter, K., Stewart, C., Handsaker, R.E., Chen, K., Alkan, C., Abyzov, A., Yoon, S.C., Ye, K., Cheetham, R.K. et al. 2011. Mapping copy number variation by population-scale genome sequencing. *Nature* **470**(7332): 59-65.

Olshen, A.B., Venkatraman, E.S., Lucito, R., and Wigler, M. 2004. Circular binary segmentation for the analysis of array-based DNA copy number data. *Biostatistics* **5**(4): 557-572.

Quinlan, A.R., Boland, M.J., Leibowitz, M.L., Shumilina, S., Pehrson, S.M., Baldwin, K.K., and Hall, I.M. 2011. Genome sequencing of mouse induced pluripotent stem cells reveals retroelement stability and infrequent DNA rearrangement during reprogramming. *Cell Stem Cell* **9**(4): 366-373.

Quinlan, A.R., Clark, R.A., Sokolova, S., Leibowitz, M.L., Zhang, Y., Hurles, M.E., Mell, J.C., and Hall, I.M. 2010. Genome-wide mapping and assembly of structural variant breakpoints in the mouse genome. *Genome Research* **20**(5): 623-635.

Quinlan, A.R. and Hall, I.M. 2010. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics* **26**(6): 841-842.

Simpson, J.T. and Durbin, R. 2012. Efficient de novo assembly of large genomes using compressed data structures. *Genome Research* **22**(3): 549-556.