Supplemental Material

Improving detection of copy number variation by simultaneous bias correction and read-depth segmentation

Szatkiewicz et al.

Table of Contents

Supplemental Tables

Table S1: Fraction of mappable bases in human and mouse genomes

Organism	Genome Size[1] (Gb)	Read Length (nt) used to compute mappable base[2]	Mappable Sequence[1] Size (Gb)	Mappable Sequence[1] Percentage
Mus musculus	2.655	54	2.284	86.0%
Mus musculus	2.655	100	2.389	90.0%
Homo sapiens	3.095	36	2.490	80.4%
Homo sapiens	3.095	100	2.765	89.3%

Note:

1. See ***Materials and Methods*** for description of the reference genomes. To generate the results in this table, sequences from chrN_random and chrUn were excluded.
2. See ***Materials and Methods*** for details of our algorithm used to compute mappable base. The fractions reported here are similar to other studies in the literature. For example, Rozowsky et al (55) used an indexing algorithm and found the fractions of mappable bases based on 30bp read is 81% in the mouse genome and 79.6% in the human genome (Table 1 of Rozowsky et al (55)).

Table S2: Evaluation of the effects of modeling techniques using NA12878

Model		Deletions			Duplications		
Step	***Technique***	***Total Number***	***Total Mbp***	***Sensitivity***	***Total Number***	***Total Mbp***	***Sensitivity***
1	+ GC Content	32402	215.8	0.59(362/610)	1350	65.7	0.12(32/261)
2	+ Mappability	11949	62.5	0.72(438/610)	8471	94.3	0.23(60/261)
3	+ Auto Regression	7815	60.8	0.71(434/610)	3051	79.8	0.20(53/261)
4	+ Random uniform distribution	9364	62.7	0.73(447/610)	3529	80.8	0.20(53/261)
5 (1)	+ CNV quality control: size filter	7734	61.8	0.73(447/610)	2009	80.1	0.19(50/261)
(2)	+ CNV quality control: RDA filter	5370	48.8	0.73(446/610)	1577	68.1	0.17(45/261)

We incorporated various techniques for bias correction, i.e. to account for GC content and mappability, to model the auto-regression in the read-depth data computed from overlapping windows, to account for additional noises by fitting a mixture of negative binomial and uniform distributions, and to apply quality control to prioritize the CNV calls. As these techniques incorporated incrementally, there are five partial versions of our algorithm as shown in Table S2. In the first step the partial algorithm only corrects for GC content. In the next step, the partial algorithm corrects for GC content and mappability together. Similarly, the auto-regression and the mixture modeling are added incrementally. The algorithm becomes our full algorithm after the CNV quality control step is applied where we (1) removed predicted CNVs that are shorter than 800bps (i.e. those that appear in only one window), and (2) additionally removed predicted CNVs with RDA value ranging between 0.75-1.25. To evaluate the effect of different techniques on CNV inference, we used the high confidence CNVs reported in Mills et al (1) (610 deletions and 462 duplications) to examine the performance of different partial versions of our algorithm. Sensitivity is computed as the total number of high-confidence CNVs that were overlapped by the GENSENG predicted calls (>50% overlapping) divided by the total number of high-confidence CNVs. We then used the total number and total base pairs of the GENSENG predicted calls as a surrogate measure of specificity. As shown in Table S2, correcting for GC content alone resulted in low sensitivity and specificity. After various techniques were added

step-wisely, both the sensitivity and the specificity were improved. Correcting for mappability resulted in the most substantial improvement in both sensitivity and specificity. Quality control of the predicted CNVs substantially improved specificity with minimal loss in sensitivity. The full algorithm demonstrated the best performance and thus was used as the release of our GENSENG software.

Table S3: Sensitivity to detect deletions in CEU trio data using deletions reported in Handsaker et al

Genome	Total high-confidence Deletions[1]	Deletions total number calls[2] (total Mbp)		Sensitivity[3] >1bp overlap		Sensitivity[3] > 50% overlap	
	Handsaker et al(25)	GENSENG	CNVnator[4]	GENSENG	CNVnator	GENSENG	CNVnator
NA12878	2301	5370 (48.8)	4105 (142.1)	0.49	0.39	0.49	0.38
NA12891	2200	4765 (88.1)	2656 (131.3)	0.50	0.38	0.50	0.37
NA12892	2055	4295 (45.0)	2268 (128.0)	0.49	0.34	0.49	0.34
Average	-	-	-	0.49	0.37	0.49	0.36

Note:

1. The high-confidence dataset used in this comparison was generated by Handsaker et al (25) and was downloaded from ftp://ftp.broadinstitute.org/pub/svtoolkit/misc/1kg/NGPaper/. The coordinates of reported deletions were translated from NCBI36 to NCBI37 using liftOver.
2. Default filters were applied. CNVnator filter: the default q0 filters removes any predicted calls that have >50% reads with zero-valued MAPQ (i.e. reads with multiple mapping locations). GENSENG filter removes any predicted calls that have RDA values lower than 0.75.
3. Sensitivity was calculated by dividing the number of the overlapping events (>1bp or >50% overlap with the high-confidence CNVs) by the total number of high-confidence CNVs.
4. The total numbers of deletions reported by the authors of CNVnator are 4100, 2223, 2352 for NA12878, NA12891, NA12892 respectively based on the same detection (i.e.100bp bin size) and quality control (i.e. default q0 filter) parameters as used in ***Table S3***. These replicable results suggest that our usage of CNVnator is accurate.

Table S4: Summary GENSENG results from the mouse dataset and comparison to Yalcin et al

	DELETIONS				DUPLICATIONS			
Inbred Mouse Strain	**#GENSENG CNV calls (total Mbp)**	**#Yalcin CNV calls**	**#Yalcin CNVs overlapped (>1 bp) by GENSENG calls (percent)**	**#Yalcin CNVs overlapped (>50% overlap) by GENSENG calls (percent)**	**#GENSENG CNV calls (total Mbp)**	**#Yalcin CNV calls**	**#Yalcin CNVs overlapped (>1 bp) by GENSENG calls (percent)**	**#Yalcin CNVs overlapped (>50% overlap) by GENSENG calls (percent)**
129S1SvlmJ	4387(24.8)	6262	1487(24%)	1474(24%)	428(4.7)	64	28(44%)	22(34%)
A/J	4931(24.9)	5674	1379(24%)	1366(24%)	297(3.6)	64	23(36%)	20(31%)
AKR/J	5961(34.7)	6007	2135(36%)	2123(35%)	1348(4.9)	78	36(46%)	33(42%)
BALB/cJ	4861(32.6)	5509	1450(26%)	1435(26%)	474(3.2)	73	26(36%)	22(30%)
C3H/HeJ	4827(29.4)	6035	1870(31%)	1849(31%)	1509(4.3)	84	26(31%)	22(26%)
CAST/EiJ	10081(55.3)	8995	679(8%)	637(7%)	351(8.8)	77	17(22%)	12(16%)
CBA/J	4579(28.2)	17544	876(5%)	797(5%)	575(3.4)	348	22(6%)	11(3%)
DBA/2J	5957(34.2)	6268	1270(20%)	1244(20%)	889(3.3)	63	19(30%)	14(22%)
LP/J	6551(34.2)	6513	1147(18%)	1128(17%)	736(5.0)	60	22(37%)	20(33%)
NOD/LtJ	11484(58.8)	6371	992(16%)	970(15%)	165(2.3)	60	19(32%)	11(18%)
NZO/HILtJ	6191(30.4)	5953	800(13%)	773(13%)	491(4.5)	45	12(27%)	11(24%)
PWK/PhJ	15780(73.4)	17901	5409(30%)	5390(30%)	348(4.7)	90	37(41%)	31(34%)
WSB/EiJ	6120(32.6)	5741	618(11%)	595(10%)	526(5.5)	56	16(29%)	13(23%)

Note:

1. The released structural variation (SV) calls (4) were downloaded from ftp://ftp-mouse.sanger.ac.uk/current_svs/. These SVs have been classified into several categories based on specific paired-end mapping patterns (4). We extracted 2 categories, including deletions and copy number gains (GAINS and TANDEMDUP), to compare to the GENSENG predicted calls.
2. This comparison focuses on the 19 autosomes.
3. GENSENG filter removes any predicted calls that have RDA values ranging between 0.5-2

Table S5: Method differences between CNVnator and GENSENG

	GENSENG	CNVnator
Features of the method	• A flexible hidden Markov model to integrate correction for multiple biases and inference of CNV in a single analysis	• Mean-shift technique with multiple bandwidth partitioning
Bias correction approach	• 1-step approach. • Correct for GC content, mappability, and additional noises in the data. • Use cubic spline smoothing to describe the GC-content vs read-depth relationship for each specific dataset (fitting the data most closely).	• 2-step approach. • Correct for GC-content only. • Use a simple linear regression to describe the GC-content vs read-depth relationship (in most cases not appropriate).
Input preparation	• Use confidently aligned reads (e.g. MAPQ>10) • Use overlapping windows (e.g. 500bp in length with 200bp overlap).	• Use all reads • Use non-overlapping windows (e.g. 100bp in length)
Post-processing	• Segment merging • Remove CNVs calls with low signal-to-noise-ratio based on RDA statistics	• Segment merging • Remove CNV calls for which the fraction of reads that aligned to multiple locations is >50%

Table S6: Performance on datasets with varying sequence coverage: simulation

COVERAGE	DELETIONS				DUPLICATIONS			
	Total simulated TRUE CNV	**Total predicted CNV calls**	**Sensitivity[1]**	**FDR[2]**	**Total simulated TRUE CNV**	**Total predicted CNV calls**	**Sensitivity[1]**	**FDR[2]**
5x	43	25	0.58	0	21	11	0.52	0
10x	43	32	0.70	0.06	21	13	0.57	0
20x	43	34	0.72	0.09	21	15	0.57	0
30x	43	35	0.72	0.11	21	15	0.57	0
40x	43	35	0.72	0.11	21	17	0.62	0

To identify the lower bound that GENSENG can handle, we applied GENSENG to data with varying sequencing coverage generated by simulation and compared the performance to that based on the native coverage (40x) using the same evaluation metrics (as was done in ***Table 1*** of the main texts). In ***Table S6***, we repeated the simulation process as described earlier with the targeted coverage been set as 5x, 10x, 20x, 30x, and 40x. To test the consistency of our simulation, we simulated data at 40x coverage for 10 times and observed replicable results (data not shown). As in ***Table 1***, stringent GENSENG filter removes any predicted calls that span only one window, have RDA values ranging between 0.5-1.25 or mappablility < 0.3. The results demonstrate that: (1) higher sequencing coverage improves CNV detection power; (2) the lower bound of sequencing coverage that yield reasonably good performance of GENSENG is 10x (70% sensitivity and 6% FDR for deletions; 57% sensitivity and 0 FDR for duplications); (3) GENSENG could potentially work on data sequenced to as low as 5x but with much reduced sensitivity (fell by 14% for deletions and fell by 10% for duplications).

Table S7: Performance on datasets with varying sequence coverage: 1000GP NA12878

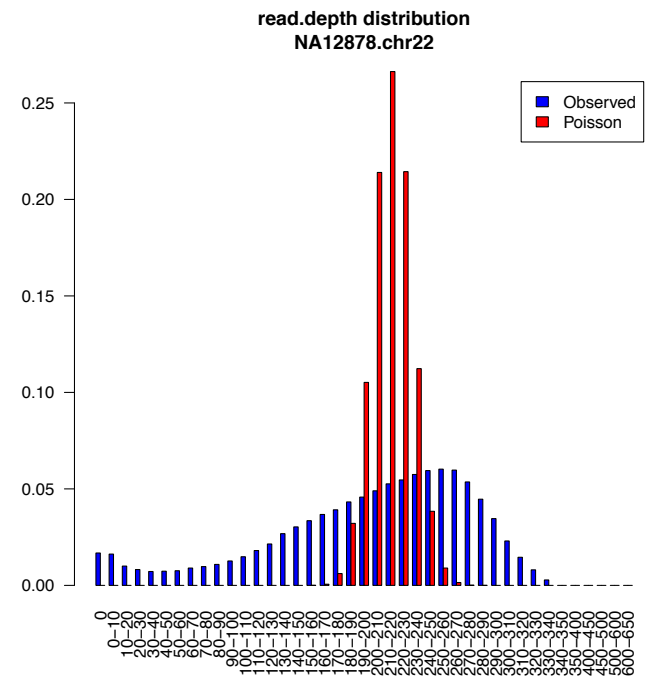
COVERAGE	DELETIONS			DUPLICATIONS		
	Total high-confidence CNVs[1]	**Total predicted CNV calls (Mbp spanned)**	**Sensitivity[2] (Number of high-confidence CNV detected)**	**Total high-confidence CNVs[1]**	**Total predicted CNV calls (Mbp spanned)**	**Sensitivity[2] (Number of high-confidence CNV detected)**
5x	610	745 (5.42)	0.38	261	482 (21.13)	0.12
10x	610	1344 (6.51)	0.53	261	769 (47.16)	0.14
20x	610	1939 (6.76)	0.57	261	941 (55.76)	0.14
30x	610	2488 (7.64)	0.62	261	1042 (30.55)	0.14
40x	610	5071 (11.73)	0.70	261	984 (38.58)	0.15

To identify the lower bound that GENSENG can handle, we applied GENSENG to datasets down-sampled from 1000GP and compared the performance to that based on the native coverage (40x) using the same evaluation metrics (as was done in ***Table 2*** of the main texts). In ***Table S7***, using the DownsampleSam.jar tool from Picard, we down-sampled the high coverage 1000GP data from NA12878 and achieved a series of sequencing coverage of 5x, 10x, 20x, 30x, 40x. As in ***Table 2***, stringent GENSENG filter removes any predicted calls that span only one window, have RDA values ranging between 0.75-1.25 or mappablility < 0.3. The results demonstrate that: (1) higher sequencing coverage improves CNV detection power; (2) the lower bound of sequencing coverage that yield reasonably good performance of GENSENG is 10x (53% sensitivity for deletions; 14% sensitivity for duplications); (3) GENSENG could potentially work on data sequenced to as low as 5x but with much reduced sensitivity (fell by 32% for deletions and fell by 3% for duplications).

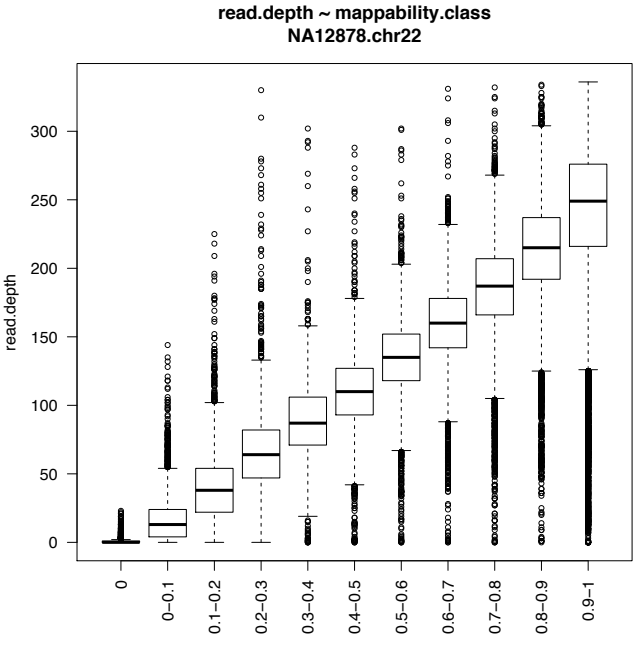
Supplemental Figures

Figure S1: Relationship between read-depth and known confounders: NA12878

S1a



S1b



S1c

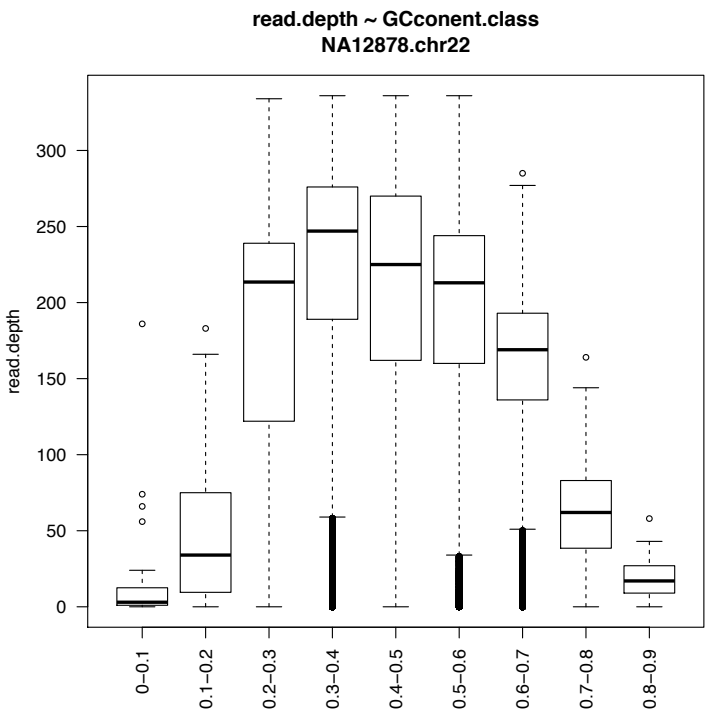


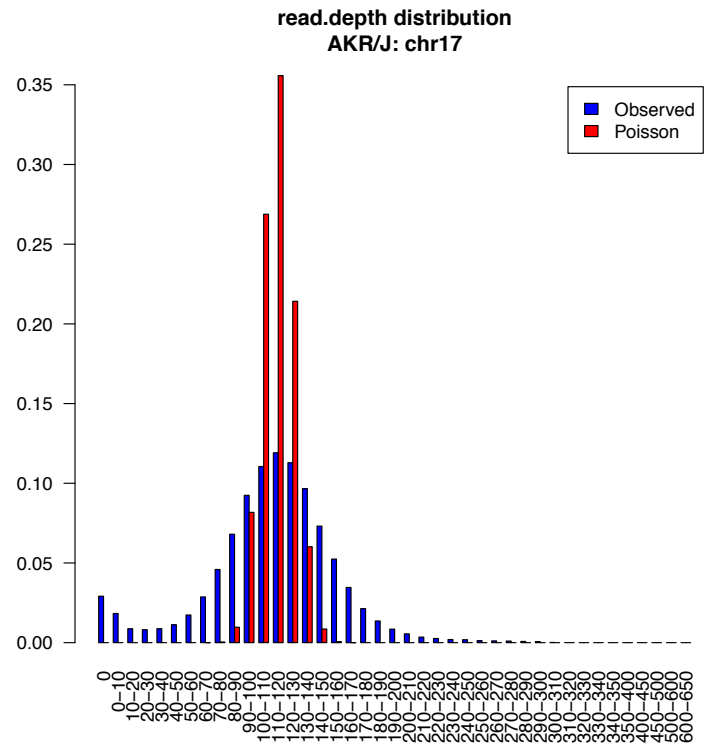
Figure S1a: Read-depth distribution for chromosome 22 of NA12878. Blue: observed distribution. Red: expected Poisson distribution. Read-depth was computed as the number of fragments in each genomic window, a.k.a window count.

Figure S1b: Boxplot of read-depth/window-count by mappability score class. A positive correlation is observed, where low mappability scores tend to have low read-depth and high mappability scores tend to have high read-depth.

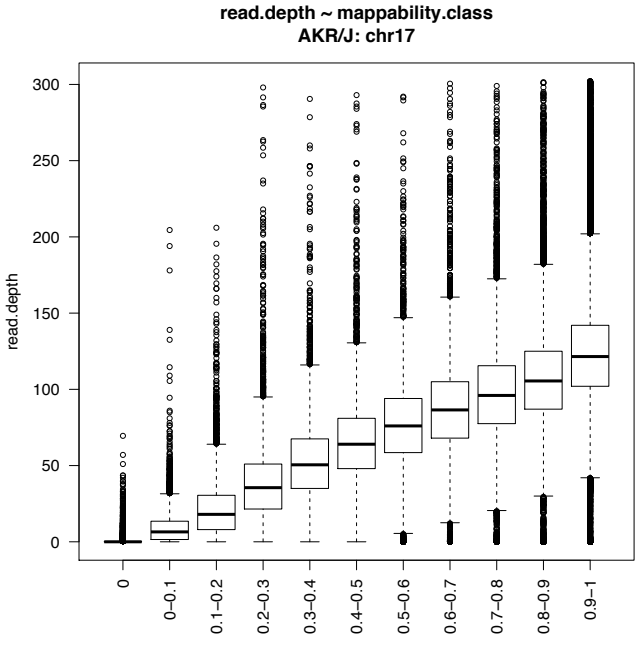
Figure S1c: Boxplot of read-depth/window-count by GC content class. A non-linear relationship between GC content and read-depth is observed, where sequences with extreme GC contents (extremely low or high) tend to have lower read-depth.

Figure S2: Relationship between read-depth and known confounders: AKR/J

S2a



S2b



S2c

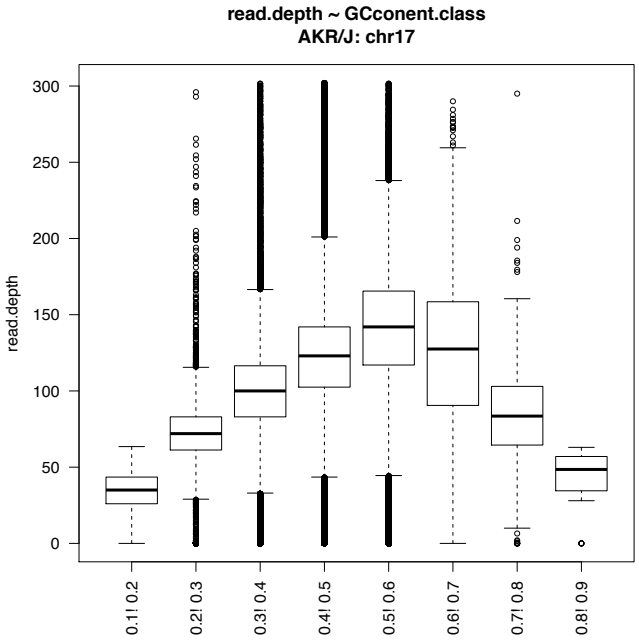
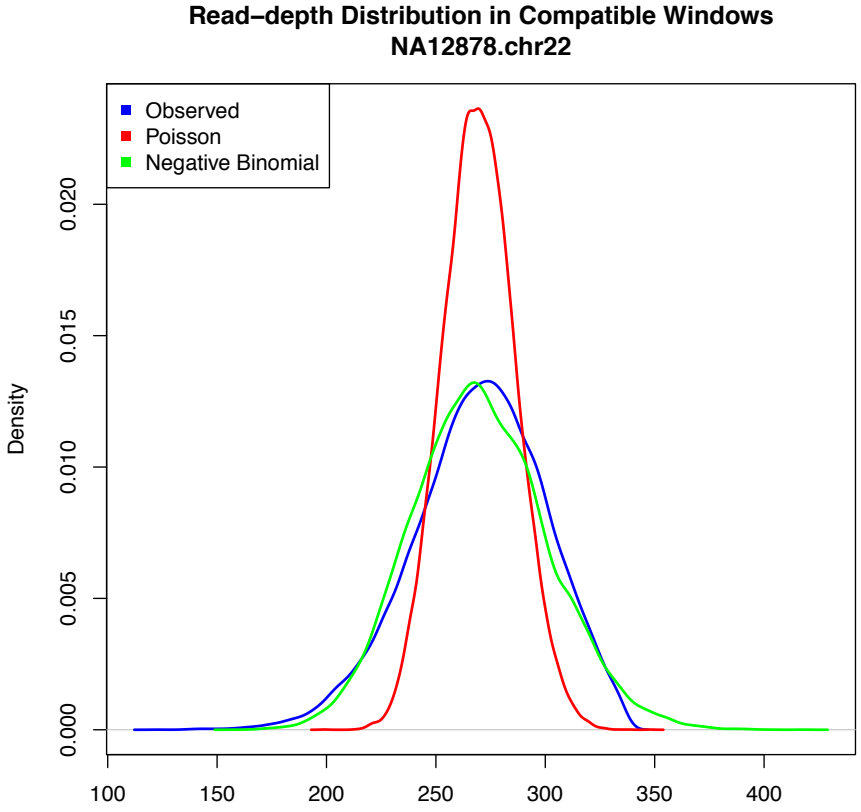


Figure S2a: Read-depth distribution for chromosome 17 of mouse inbred strain AKR/J. Blue: observed distribution. Red: expected Poisson distribution. Read-depth was computed as the number of fragments in each genomic window, a.k.a window count.

Figure S2b: Boxplot of read-depth/window-count by mappability score class. A positive correlation is observed, where low mappability scores tend to have low read-depth and high mappability scores tend to have high read-depth.

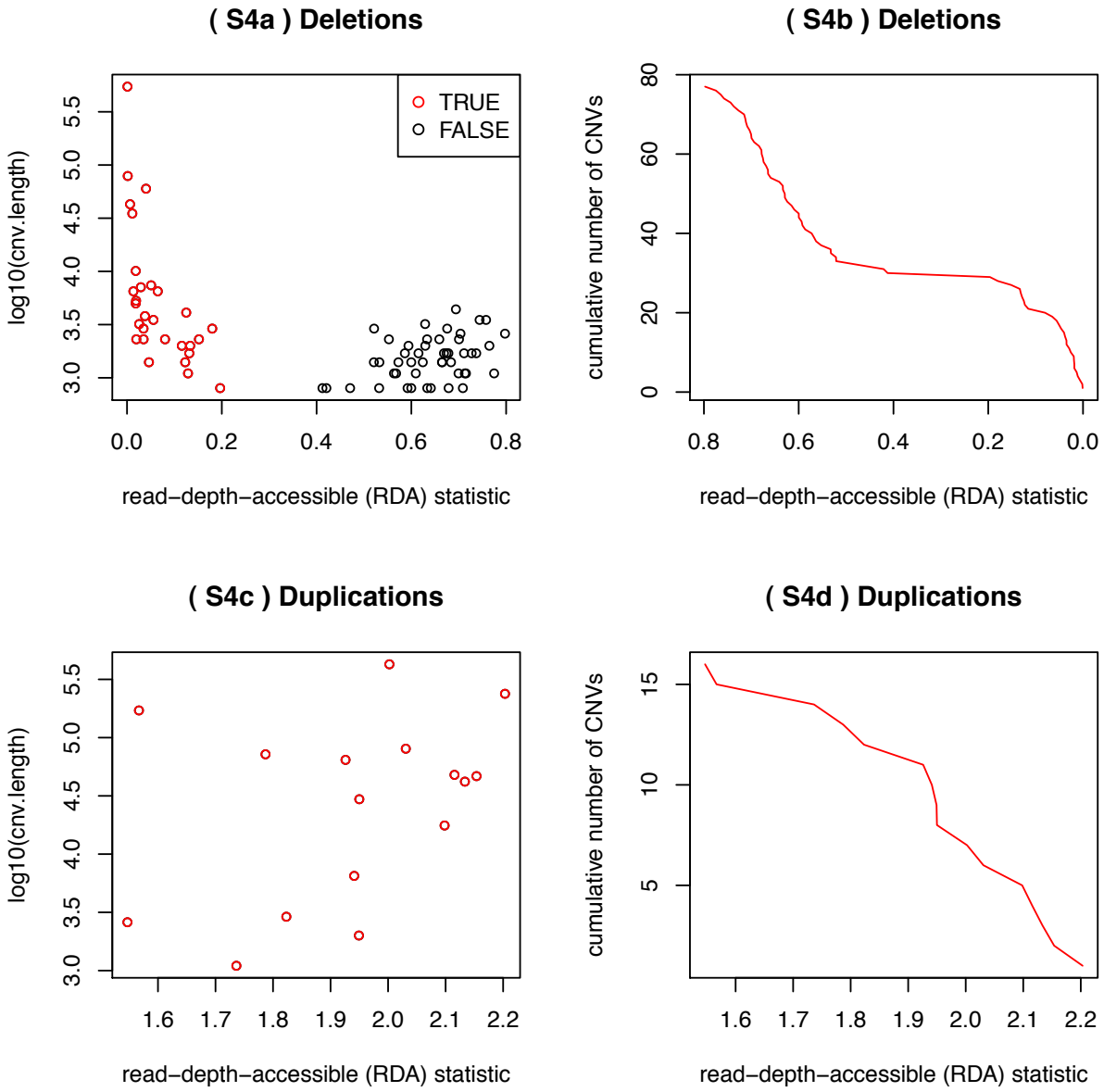
Figure S2c: Boxplot of read-depth/window-count by GC content class. A non-linear relationship between GC content and read-depth is observed, where sequences with extreme GC contents (extremely low or high) tend to have lower read-depth.

Figure S3 Read-depth distribution after accounting for known confounders



Based on the input data computed for chromosome 22 of NA12878, we extracted compatible windows that have the same GC content class (0.4-0.5), the same mappability score class (0.9-1), and have high likelihood to be copy number normal regions (e.g. having not overlapped any high-confidence CNVs or other candidate CNVs). We then compared the observed read-depth distribution from these compatible windows to its theoretical expectations. Blue: observed distribution; red: Poisson distribution; green: negative binomial distribution. As shown in the figure, the Poisson distribution still does not fit the data; whereas the negative binomial distribution fits the data well.

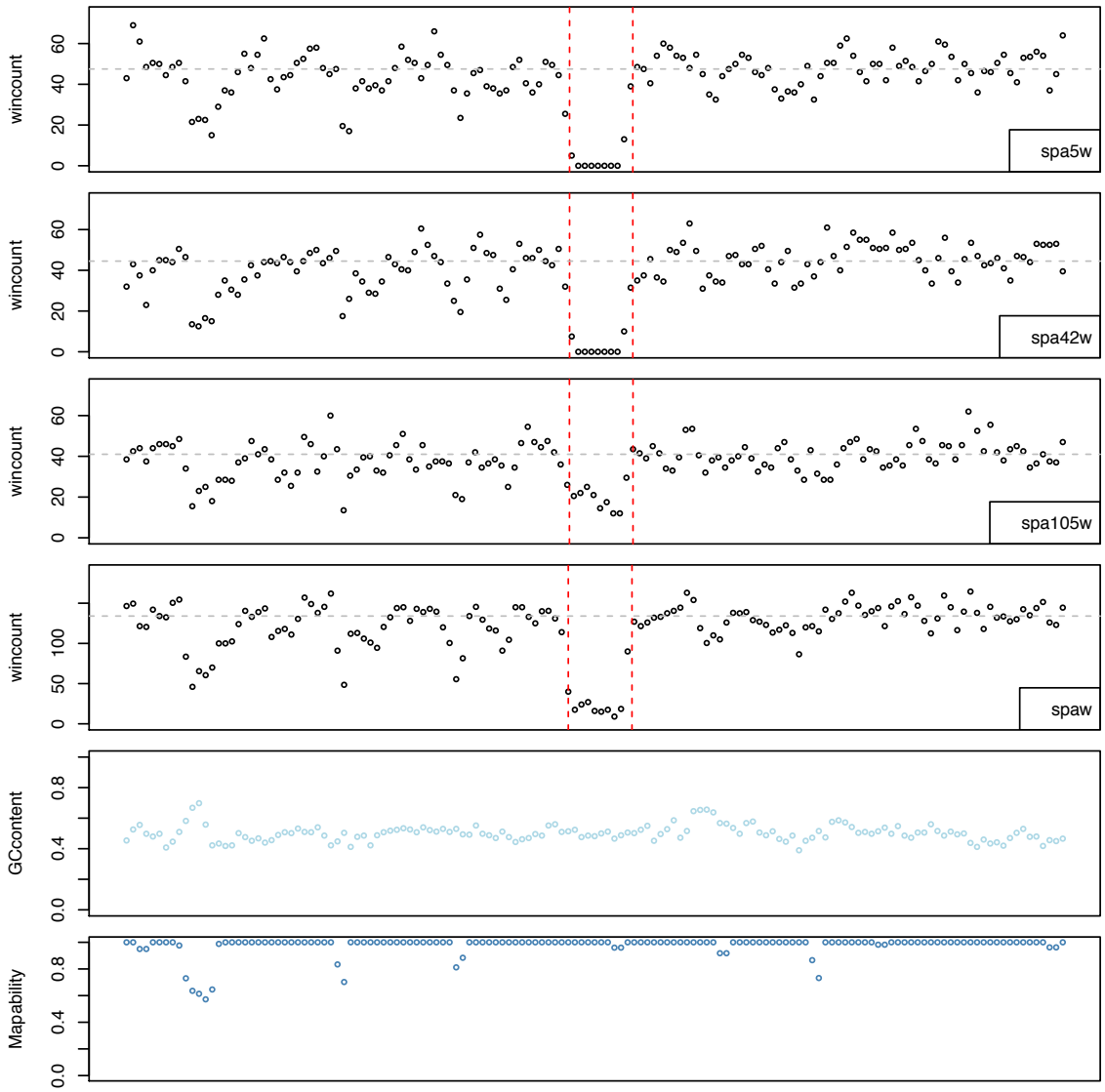
Figure S4 RDA ranking and prioritization of CNV prediction from simulation study



From the 716 predicted GENSENG calls (690 deletions and 26 duplications), a CNV-quality-control filter was applied to remove any GENSENG predicted CNVs with mappability less than 0.3, yielding in 77 predicted deletions and 16 predicted duplications shown in this figure. In ***Figure S4a***, the red dots indicate the true discoveries (29) and black the false discoveries (48). It is evident that ranking the RDA values (the X-axis) correctly prioritized the predictions made by GENSENG. In ***Figure S4c***, all 16 predicted duplications had RDA>1.25 and were all true discoveries.

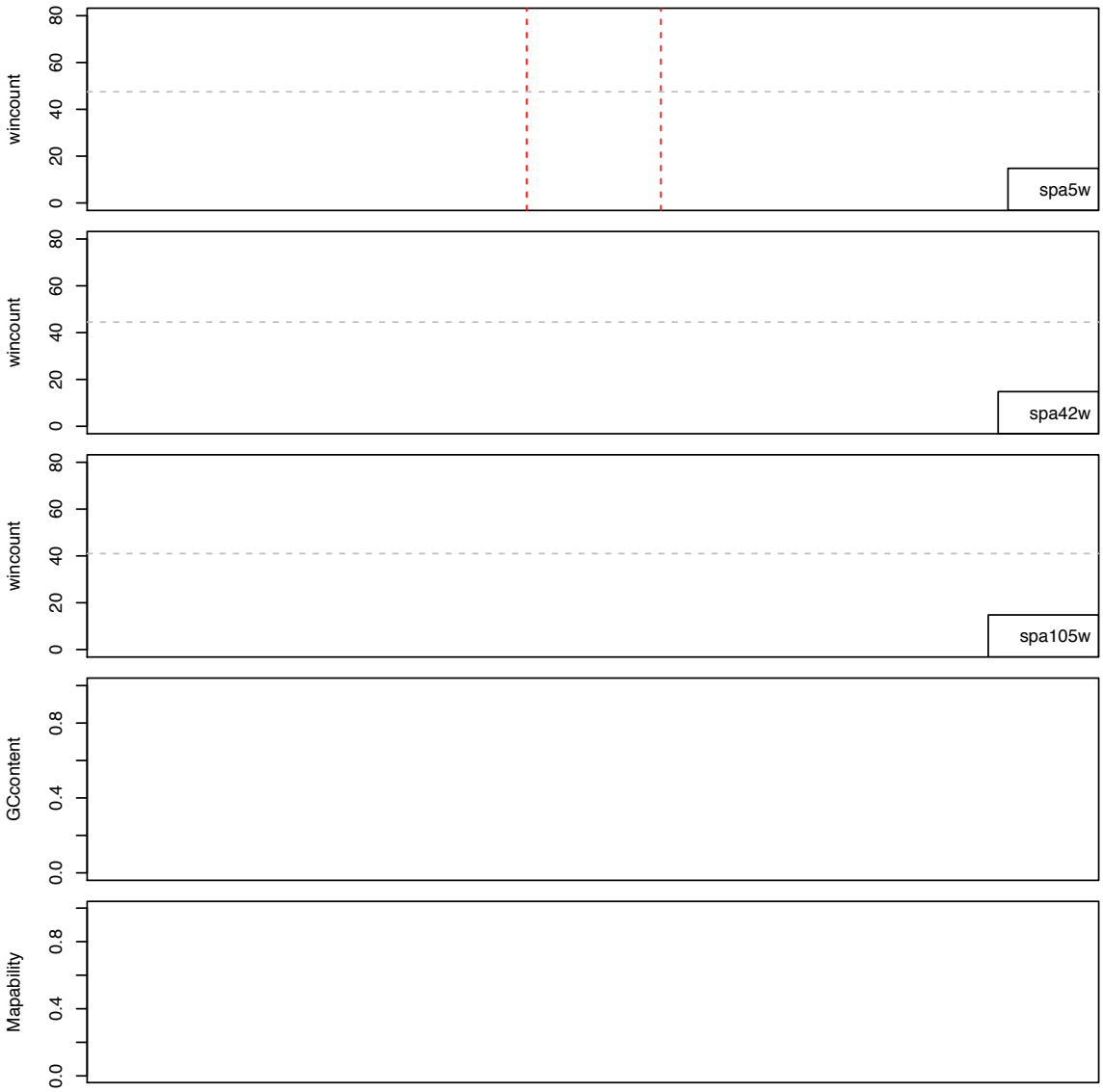
Figure S5 Example CNVs identified from the human HTS dataset

Figure S5a: A shared deletion



There are 6 panels from the top to the bottom. The X-axis of each panel indicates genomic position in base pair but the specific coordinates are not shown. The first panel: read-depth computed from individual spa5w; the second panel: read-depth from individual spa42w; the third panel: read-depth from individual spa105w; the fourth panel: read-depth computed after pooling all the reads together from spa5w, spa42w, and spa105w. The GC content and mappability of the region are plotted in the fifth and the sixth panels respectively. In the first through the fourth panels, the black dots in Y-axis indicate read-depth signal; the red dashed lines are the boundaries from GENSENG prediction; the grey lines are the median read-depth of the chromosome.

Figure S5b: A private duplication



There are 5 panels from the top to the bottom similarly to ***Figure S5a***. The first panel: read-depth from individual spa5w; the second panel: read-depth from individual spa42w; the third panel: read-depth from individual spa105w; the fourth panel: GC content; the fifth panel: mappability. In individual spa5w (first panel), the steel blue dots indicate read-depth signal from a predicted duplication whose boundary is enclosed by the red dashed lines.

Figure S6 Example shared CNVs identified from mouse HTS datasets

Figure S6a: A shared duplication

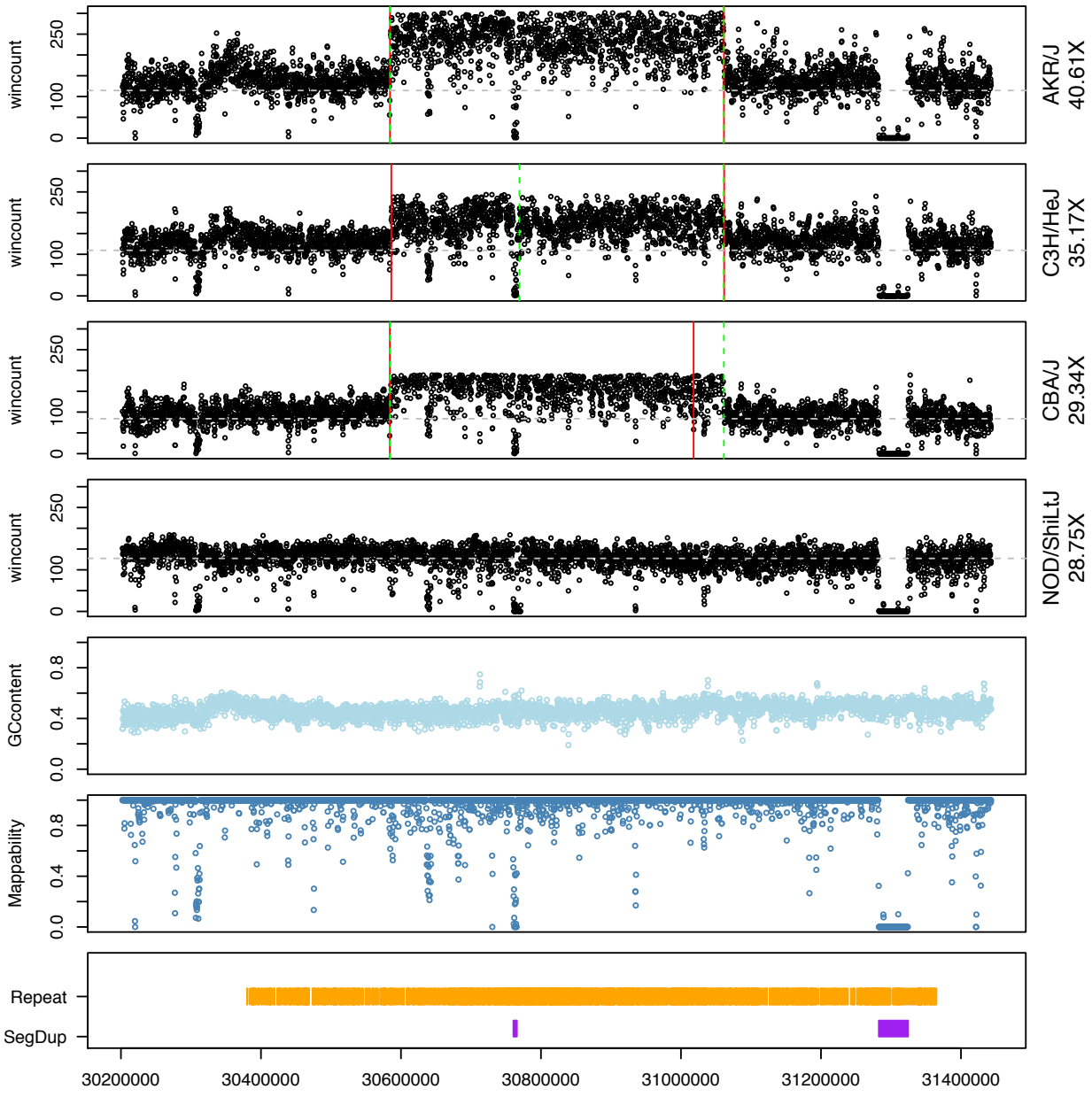


Figure S6a shows a shared duplication from the mouse chromosome 17, also found from microarray studies (e.g. (5)). For this region, AKR/J, C3H/HeJ, CBA/J, 129S1/SvImJ, A/J, DBA/2J, and LP/J belong to the same haplotype (6) that contains this duplication, hence identity by descent. NOD/ShiLtJ and BALB/cJ belong to another haplotype (6) that does not contain this duplication. There are 6 panels from the top to the bottom. The X-axis of each panel indicates genomic position in base pair. In the first through the fourth panels, the black dots in Y-axis

indicate read-depth signal from representative strains of each haplotype with their names and sequencing coverage labeled in the right margin; the red solid lines are the boundaries from GENSENG prediction; the green dashed lines are the boundaries reported by the Mouse Genomes Project (4); the grey lines are the median read-depth of the chromosome. The GC content and mappability of the region are plotted in the fifth and the sixth panels respectively. The last panel shows the locations of segmental duplication (purple) and repeat-mask repetitive DNAs (orange).

Figure S6b: A shared deletion

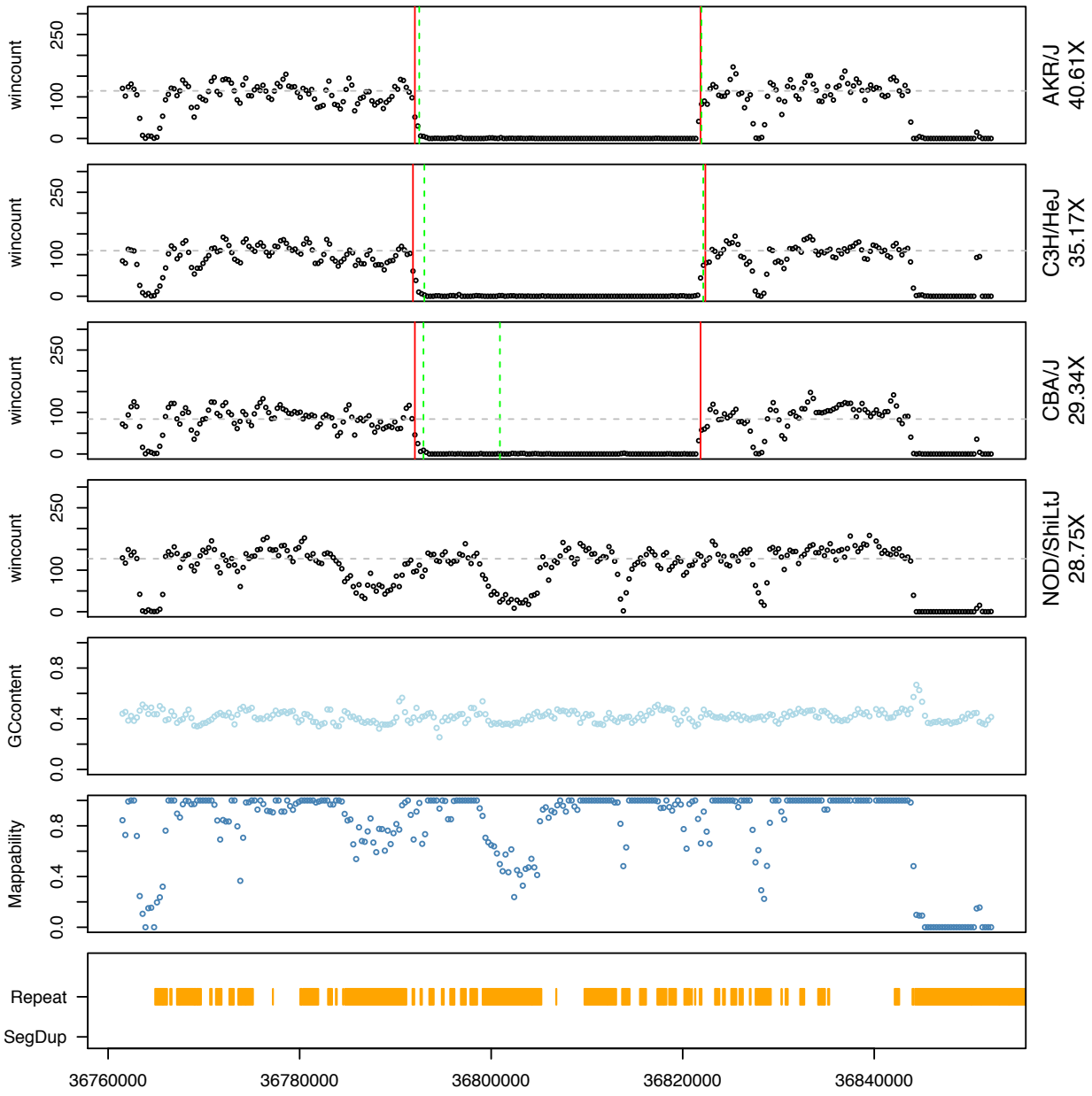


Figure S6b shows a shared deletion from the mouse chromosome 17. Within the compatible interval with no historical recombination, chr17:36775200-36842845 (6), AKR/J, C3H/HeJ, and CBA/J belong to the same haplotype (6) that contains this deletion, hence identity by descent. NOD/ShiLtJ, 129S1/SvImJ, A/J, DBA/2J, BALB/cJ, LP/J belong to another haplotype (6) that does not contain this deletion. There are 6 panels from the top to the bottom. The X-axis of each panel indicates genomic position in base pair. In the first through the fourth panels, the black dots in Y-axis indicate read-depth signal from representative strains of each haplotype with their names and sequencing coverage labeled in the right margin; the red solid lines are the boundaries from GENSENG prediction; the green dashed lines are the boundaries reported by the Mouse Genomes Project (4); the grey lines are the median read-depth of the chromosome. The GC content and mappability of the region are plotted in the fifth and the sixth panels respectively. The last panel shows the locations of segmental duplication (purple) and repeat-mask repetitive DNAs (orange). Note that beginning at approximately 3684100 bp, there appears to be a deletion-like artifact in all strains with zero-valued read-depth. Close inspection suggested that it was caused by zero-valued mappability and was not called as deletion by GENSENG because of its ability to correct biases.

Supplemental Methods

Supplemental Methods for "Improving detection of copy number variation by simultaneous bias correction and read-depth segmentation"

Jin P. Szatkiewicz, WeiBo Wang, Patrick F. Sullivan, Wei Wang, Wei Sun

Contents

1 Overview

GENSENG's analytic protocol comprises four steps:

1. Read quality control;

2. Computation of read-depth and covariate values;

3. HMM inference of copy number while correcting for biases;

4. Post-segmentation processing.

2 Input Data Preparation

2.1 Read quality control

1. Remove any read that fails platform/vendor quality checks, or either a PCR duplicate or an optical duplicate.

2. Extract all single-end reads and properly paired paired-end reads.

3. Extract confidently aligned reads with MAPQ $\geq$ a specified threshold. In this study, we use MAPQ $\geq$10, which was empirically determined.

2.2 Selection of window size

We use a sliding window approach where each window is of 500bp in length and is overlapped by the adjacent window by 200bp. The size of the window and the degree of overlap was empirically determined.

2.3 Computation of read-depth

Each read (e.g. 36-mer or 51-mer from the 1000 Genomes Project data) is represented by its middle base pair. A fragment is counted where read mapping information is available.

1. If two ends of a pair fall in two windows, assign 1/2 to each window where the ends fall;
2. If both ends of a pair fall in the same window, assign 1 to the window;
3. If paired-ended but only one-end present, assign 1/2 to the window where the ends fall;
4. If single-end, always assign 1 to the window where the end falls.

2.4 Computation of covariate values

The set of covariates include GC content and mappability score.

GC content is computed as in the following steps. (1) Calculate the proportion of G or C bases in each window from a given reference genome.(2) Apply a cubic spline smoothing and then transform the GC proportion based on the fitted curve so that the transformed GC proportion and logarithm of the read-depth are linearly correlated. (3) The transformed GC proportion is median-centered and is referred to as GC content hereafter.

Mappability score is calculated a priori in four steps: (1) Identify K-mers where each K-mer consists K consecutive bases starting at each base position from the reference genome. (2) Align the K-mers back to the reference genome using a desired aligner, e.g. BWA (37). Ideally, the aligner and the alignment parameters are chosen to match what was used for generating read alignment files from the sample genomes. (3) Identify mappable base positions where the corresponding K-mers map back to themselves unambiguously (i.e. there is a single best hit and it is the true position of the K-mer). For example, the X0 field produced by BWA (37) relates a K-mer from a specific place in the genome to the number of best hits of that K-mer in the entire genome. If a K-mer has a X0 value of 1, the corresponding base can be identified as a mappable base. (4) Compute mappability score as the proportion of mappable bases in a given window, which measures the uniqueness of specific regions of the reference genome.

In summary, the input data is a triplet for each window represent by $\{O, G, L\} = \{o_1.....o_T, g_1.....g_T, l_1.....l_T\}$, where T is the total number of windows of a chromosome, o_t denotes the read-depth, g_t denotes the GC content, and l_t denotes the mappability score of the t^{th} window.

3 Model Introduction

3.1 HMM setup

We use a time-homogeneous discrete hidden Markov model (HMM) to segment the genome to regions of same copy number. In our HMM, time represents the sliding windows tiled along a chromosome, denoted by t.

The state represents the underlying copy number (CN). The state variable $q_t = CN_t$ is hid-

den and discrete with N possible values, $(0, 1, ..., N-1)$, where N, is derived from the data by K-mean clustering the logarithm of the read-depth. A particular sequence of the states is described by $q = (q_1, ..., q_T)$, where T is the total number of sliding windows of a chromosome. Let π_j be the initial state probability, the probability that the state of the first window is state j. The underlying hidden Markov chain is defined by state transitions $P(q_t|q_{t-1})$ and is represented by a time-independent stochastic transition matrix $A = \{a_{jz}\} = P(q_t = z|q_{t-1} = j)$.

Each copy number state emits an observation, the read-depth. The observation variable, O_t, is a discrete count variable. A particular sequence of the observations is described by $o = (o_1, ..., o_T)$. The emission probability of a particular observation at a particular time t for state j is described by $e(t, j) = P(O_t = o_t|q_t = j)$. For a detailed description of the emission probability, see Section 3.2.

We use the Baum-Welch algorithm (Baum et al 1970) to find the maximum likelihood estimates (MLE) of the HMM parameters. Following Bilmes (1998), we define the complete-date likelihood and solve the Q function in order to find the maximum likelihood estimates (MLE) of the HMM parameters.

A standard HMM assumes the Markov property, $P(q_t|q_{t-1}, q_{t-2}, q_{t-3}, ...q_1) = P(q_t|q_{t-1})$. An additional assumption that is often employed is that the observations are independent given the states, $O_t \perp O_i(i \neq t)|q_t$, which is valid when the windows are non-overlapping. When the windows are overlapping, this assumption is invalid; and instead, the observations are drawn from an autoregression process (Juang and Rabiner 1985). We have implemented an autoregressive HMM to model this feature of the data. For details of the autoregressive HMM, see Section 5.

3.2 Emission probability

The emission probability of the read-depth, $e(t, j) = P(O_t = o_t|q_t = j)$, is modeled as a mixture of a uniform distribution and a negative binomial distribution.

$$e(t, j) = c/R_m + (1 - c)e^{NB}(t, j) \tag{1}$$

where c is the proportion of the random uniform component and is fixed as constant for each state; and R_m is the largest read-depth among all windows and thus $1/R_m$ is the uniform density.

To describe the negative binomially distributed component, $e^{NB}(t, j)$, we first explain the re-

lationship between the Poisson and the negative binomial distributions. The Poisson distribution imposes that the variance equals to the mean. The negative binomial distribution allows overdispersion. Specifically, if O follows a Poisson distribution with mean μ, and μ follows a gamma distribution, the resulting distribution for O is a negative binomial distribution. The variance of negative binomial distribution is $\mu_t + \phi\mu_t^2$, where $\phi\mu_t^2$ is the overdispersion part of the variance. As $\phi \to 0$, $f_{NB}(o_t; \mu_t, \phi)$ reduces to a Poisson distribution with mean μ_t and variance μ_t. $f_P(o_t; \mu_t) = \frac{\exp(-\mu_t)\mu_t^{o_t}}{o_t!}$.

Next, the mean value of the negative binomially distributed component is expressed as a function of a set of covariates to account for confounders.

$$\mu_{tj} \quad = \quad \alpha_0 * (CN_t)^{\beta_1} * (l_t)^{\beta_2} * (g_t)^{\beta_3} \tag{2}$$

where t denotes the t^{th} window, j is the index of the copy number state, j emphasizes the dependency of the mean μ_t on the copy number CN_t, l_t is the mappability score, g_t is the GC content. For computational convenience, we set $CN_t = 0.5$ when $j = 0$, and set $CN_t = j$ when $j > 0$.

We then employ a log link function to acknowledge the fact that $\mu_{tj} > 0$ and obtain:

$$\log(\mu_{tj}) \quad = \quad \beta_0 + \beta_1 * \log(CN_t) + \beta_2 * \log(l_t) + \beta_3 * \log(g_t) \tag{3}$$

$\beta_0, \beta_1, \beta_2, \beta_3$ are the regression coefficients. Specifically, $\beta_0 = \log(\alpha_0)$, is the intercept parameter and is interpreted as the average level of read-depth signal when all covariates are equal to zero. β_1 is the amount of increase of read-depth for every unit increase of copy number, CN. β_2 is the amount of increase of read-depth for every unit increase of the mappability score, l. β_3 is the amount of increase of read-depth for every unit increase of the GC content, g.

Thus, given the above regression model for the mean, the negative binomial probability distribution function is expressed as the following:

$$e^{NB}(t,j) = f_{NB}(o_t; \mu_{tj}, \phi_j) \quad = \quad \frac{\Gamma(o_t + 1/\phi_j)}{o_t!\Gamma(1/\phi_j)} \left(\frac{1}{1+\phi_j\mu_{tj}}\right)^{1/\phi_j} \left(\frac{\phi_j\mu_{tj}}{1+\phi_j\mu_{tj}}\right)^{o_t} \tag{4}$$

The complete emission probability is then expressed as the following:

$$e(t,j) \quad = \quad c/R_m + (1-c)\frac{\Gamma(o_t + 1/\phi_j)}{o_t!\Gamma(1/\phi_j)} \left(\frac{1}{1+\phi_j\mu_{tj}}\right)^{1/\phi_j} \left(\frac{\phi_j\mu_{tj}}{1+\phi_j\mu_{tj}}\right)^{o_t} \tag{5}$$

4 The Program Flow of HMM inference

A time-homogeneous HMM has been implemented in C++.

- HMM input: $\{\mathbf{O}, \mathbf{G}, \mathbf{L}\}$ and Λ_0. Here $\{\mathbf{O}\}$ is the read-depth, $\{\mathbf{G}\}$ is the GC content, and $\{\mathbf{L}\}$ is the mappability score computed for each sliding window. Λ_0 is either the initial values of the HMM parameters or the parameter estimates from the previous iteration. The HMM parameters include the state parameters and the emission parameters.
- HMM output: The estimated HMM parameters Λ_1. The log likelihood from each iteration, $\log(p(\mathbf{O}|\Lambda))$.
- A one-step update of the Baum-Welch algorithm (Baum et al 1970) is illustrated below. The expectation (E-step) and the maximization (M-step) procedures iterate until the convergence criterion (smaller than 10^{-6} change in the log-likelihood) is reached.
- Most of the computations are carried out in log scale to avoid underflow or overflow. A utility function `logsumexp` is used to facilitate the computation. Specifically, it is defined as $\texttt{logsumexp}_j(v) = \log\left(\sum_j \exp(v_j)\right)$, where $v = \{v_j\}$ is a vector.
- For efficient implementation, we estimate the $\log(\mu_{tj})$ directly using the IRLS method. Alternative approach could be estimating the regression coefficients.

4.1 Model Initialization:

(a) The number of states:

N is found from the data by finding the number of k-mean clusters of $\log(O)$. Here we assume N=7, for CN = 0,1,2,3,4,5,6+.

(b) Initial state probability, π_j:

For state CN = 2: 0.9995; for other states: (1-0.9995)/(N-1).

(c) Initial state transition probability, a_{jz}:

Self-transition probability: for state CN = 2: 0.9995; for other states: 0.995;

Transition probability to other states, i.e. a_{jz} when $z \neq j$:

Transiting from CN = 2 to $CN < 2$: (1-0.9995)/3; from CN = 2 to $CN > 2$: (1-0.9995)/12;

Transiting from $CN < 2$ to CN = 2: (1-0.995)/9; from $CN < 2$ to $CN < 2$: (1-0.995)/90; from $CN < 2$ to $CN > 2$: (1-0.995)/400;

Transiting from $CN > 2$ to $CN < 2$: (1-0.995)/40; from $CN > 2$ to CN = 2: (1-0.995)/1.25; from $CN > 2$ to $CN > 2$: (1-0.995)/20;

(d) Initial mean values of the negative binomially distributed component, $\log(\mu_{tj})$:

Assume normal copy number (CN=2) for all windows.

Set $\beta_3 = 0.5$. β_3 is the coefficient for GC content. 0.5 is the empirically determined value.

intercept=$\log(\text{median}(O)) - \log(2) - \text{median}(\log(L)) - \beta_3 \text{median}(G)$.

for $j = 0$, offset = $\log(0.5) + \log(l_t)$.

for $j = 1..N-1$, offset = $\log(j) + \log(l_t)$.

$\log(\mu_{tj}) = \text{intercept} + \text{offset} + \beta_3 g_t$.

(e) Initial overdispersion parameters, ϕ_j:

In this study, we have one overdispersion parameter ϕ for different states jointly through setting $\phi_j = \phi$. And set $\phi = 1$ for initialization.

(f) Initial mixing probability, c:

$c = 0.01$. The mixing probability is the same for the normal state and the other states.

(g) Initial parameter for the uniform distribution, R_m:

R_m=max($\mathbf{O}$).

4.2 E-step

Given the current parameter estimates Λ_0, we efficiently compute the desired quantities.

4.2.1 The Emission Probability

$$e(t,j) \quad = \quad c/R_m + (1-c)\frac{\Gamma(o_t + 1/\phi_j)}{o_t!\Gamma(1/\phi_j)} \left(\frac{1}{1+\phi_j\mu_{tj}}\right)^{1/\phi_z} \left(\frac{\phi_j\mu_{tj}}{1+\phi_j\mu_{tj}}\right)^{o_t} \tag{6}$$

4.2.2 The Forward Probability

$$f(t,j) \quad = \quad P(o_1, o_2, ...o_t, q_t = j \texttt{ ends at } t|\Lambda_0) \tag{7}$$

Algorithm

1. Initialization:

$$f(1,j) \quad = \quad \pi_z e(1,j) \tag{8}$$

$$\log(f(1,j)) \quad = \quad \log(\pi_j) + \log(e(1,j)) \tag{9}$$

2. Recursion, for $t \in (2:T)$ and for $j \in (1:N)$,

$$f(t,j) \quad = \quad e(t,j) \sum_j f(t-1,j) a_t(z,j) \tag{10}$$

$$\log(f(t,j)) \quad = \quad \log(e(t,j)) + \texttt{logsumexp}_j \left[\log(f(t-1,j)) + \log(a_t(j,z))\right] \tag{11}$$

3. Termination: computation of the overall likelihood $\log(p(\mathbf{O}|\Lambda_0))$

$$p(\mathbf{O}|\Lambda_0) \quad = \quad \sum_z f(T,z) \tag{12}$$

$$\log(p(\mathbf{O}|\Lambda_0)) \quad = \quad \texttt{logsumexp}_z \log(f(T,z)) \tag{13}$$

4.2.3 The Backward Probability

$$b(t,z) \quad = \quad P(o_{t+1}, o_{t+2}, ...o_T|q_t = z \texttt{ ends at } t|\Lambda_0) \tag{14}$$

Algorithm

1. Initialization:

$$b(T,z) \quad = \quad 1 \tag{15}$$

$$\log(b(T,z)) \quad = \quad 0 \tag{16}$$

2. Recursion, for $t \in (T:2)$ and for $z \in (1:N)$,

$$b(t-1,z) = \sum_j [a_t(z,j)e(t,j)b(t,j)] \quad (17)$$

$$\log(b(t-1,z)) = \texttt{logsumexp}\,[\log(a_t(z,j)) + \log(e(t,j)) + \log(b(t,j))] \quad (18)$$

4.2.4 The Posterior Probability

$$\gamma(t,j) = P(q_t = j|\mathbf{O}, \Lambda_0) \quad (19)$$

Algorithm

$$\gamma(t,j) = \frac{f(t,j)b(t,j)}{p(\mathbf{O}|\Lambda_0)} \quad (20)$$

$$\log(\gamma(t,j)) = \log(f(t,j)) + \log(b(t,j)) - \log(p(\mathbf{O}|\Lambda_0)) \quad (21)$$

4.3 M-step

4.3.1 Estimate the initial state probability π_j

The initial probability π_j is simply the posterior probability of being state j at position 1, therefore the new estimate of π_j, denoted by $\overline{\pi}_j$, is computed as the following:

$$\overline{\pi}_j = \frac{f(1,j)b(1,j)}{p(\mathbf{O}|\Lambda_0)}, \quad (22)$$

$$\log(\overline{\pi}_j) = \log(f(1,j)) + \log(b(1,j)) - \log(p(\mathbf{O}|\Lambda_0)). \quad (23)$$

4.3.2 Estimate the transition probability a_{jz}

The estimated a_{jz} is denoted by $\overline{a}_{jz}$, for $j \neq z$, and is computed as the following:

$$\zeta(t,j,z) = f(t,j)e(t+1,z)b(t+1,z) \quad (24)$$

$$\log(\zeta(t,j,z)) = \log(f(t,j)) + \log(e(t+1,z)) + \log(b(t+1,z)) \quad (25)$$

$$\overline{a}_{jz} = \frac{\sum_{t=1}^{T-1} \zeta(t,j,z)}{\sum_{t=1}^{T-1} \gamma(t,j)} \tag{26}$$

4.3.3 Estimate the emission parameters, an overview:

Because we fix c and R_m as constant, parameter estimation will only concern the negative binomially distributed component.

To estimate the negative binomial parameters, a weighted GLM function is implemented in C++. The argument of this function include "family", "observation", "covariate", "offset", and "prior". The argument "family" means either Poisson or negative binomial. The argument "prior" means the probability that each observation belongs to the negative binomially distributed component. For the t^{th} window and state j, the "prior" is denoted by $p_{t,j}$ and is computed as the following:

$$p_{t,j} = \frac{(1-c)e^{NB}(t,j)\gamma(t,j)}{c/R_m + (1-c)e^{NB}(t,j)} \tag{27}$$

Following the implementation function `MASS/glm.nb` in R (Venables and Ripley, 2002), we use an alternating iterative estimation procedure to obtain the new estimate of $\log(\mu_{tj})$, denoted by $\log(\overline{\mu}_{tj})$, and the new estimate of ϕ, denoted by $\overline{\phi}$.

- First, we fix $\overline{\phi}$ and compute $\log(\overline{\mu}_{tj})$ by fitting weighted GLM using the iteratively reweighted least squares (IRLS) method. For details, see Section 4.3.4.
- Then, we fix $\log(\overline{\mu}_{tj})$ and compute $\overline{\phi}$ using the Newton-Raphson method with weight. For details, see Section 4.3.5.
- The above two steps alternated until convergence.

4.3.4 Estimation of $\log(\mu_{tj})$ using the IRLS method

4.3.4.1. Define the necessary variables for estimating $\log(\mu_{tj})$, where $t = 1...T$, $CN_t = q_t = 0...j..(N-1)$.

- "Prior"
  $p_t = \{p_{t,0}, ..., p_{t,j}, ..., p_{t,N-1}\}$
  $p = \{p_1, ...p_t, .., p_T\}$

- "Observation"

  $y_t = \{o_t, ...o_t, ...o_t\}$ (o_t repeats for N times)

  $y = \{y_1, ...y_t, .., y_T\}$

- "Covariates"

  Let $\texttt{cov}$ denote covariates and let M denote the number of covariates.

  If GC content (G) is the only covariate, $M = 1$ and define the covariate vector as:

  $x_t = \{g_t, ...g_t, .., g_t\}$ (g_t repeats for N times)

  $x = \{x_1, ...x_t, .., x_T\}$

  If $M > 1$, each covariate will be inserted into x like G

  $\texttt{cov}_t = \{\texttt{cov}_{t,0}, \texttt{cov}_{t,1}, ..., \texttt{cov}_{t,N-1}\}$

  $\texttt{cov} = \{\texttt{cov}_1, ...\texttt{cov}_t, .., \texttt{cov}_T\}$

  $x = \{\texttt{cov}^1, ..., \texttt{cov}^M\}$

- "Offset"

  $\text{offset}_t = \{[\log(CN_t = 0.5)+\log(l_t)], [\log(CN_t = 1)+\log(l_t)], ...[\log(CN_t = j)+\log(l_t)], .., [\log(CN_t = N-1)+\log(l_t)]\}$

  $\text{offset} = \{\text{offset}_1, ...\text{offset}_t, .., \text{offset}_T\}$

- "The weighted log-likelihood function"

$$Lm = \sum_{t=1}^{T}\sum_{j=0}^{N}\left[\log\left(\Gamma(o_t + 1/\phi)\right) - \left(\frac{1}{\phi} + o_t\right)\log(\frac{1}{\phi} + \mu_{tj}) + \log(o_t + 1.0) + o_t\log(\mu_{tj})\right]p_{tj} \quad (28)$$

4.3.4.2. Fit a weighted Poisson regression model using the IRLS procedure.

4.3.4.3. Perform a score test

The score test (Dean 1992) is used to test whether the overdispersion parameter, ϕ, is significantly greater than 0. If the score test is significant, we

- Estimate ϕ using the Newton-Raphson method as described in Section 4.3.5.
- Proceed to Section 4.3.4.4.

4.3.4.4. Fit a weighted negative binomial regression model using the IRLS method.

4.3.5 Estimation of overdispersion using the Newton-Raphson method

Given $\log(\mu_{tj})$, we use the Newton-Raphson method to estimate the overdispersion parameter, ϕ. In this study, we estimate one overdispersion parameter ϕ jointly for all states, and set $\phi_j = \phi$ for $j = 0..N-1$.

The following weighted log-likelihood and its first, second derivatives are used in the Newton-Raphson method to estimate ϕ. The weighted log-likelihood is the same as Equation (28).

$$Lm = \sum_{t=1}^{T}\sum_{j=0}^{N}\left[\log\left(\Gamma(o_t + 1/\phi)\right) - \left(\frac{1}{\phi} + o_t\right)\log(\frac{1}{\phi} + \mu_{tj}) + \log(o_t + 1.0) + o_t\log(\mu_{tj})\right]p_{tj}$$

It is computationally slightly easier to estimate $\varphi = 1/\phi$. Then,

$$Lm = \sum_{t=1}^{T}\sum_{j=0}^{N}\left[\log\left(\Gamma(o_t + \varphi)\right) - (\varphi + o_t)\log(\varphi + \mu_{tj}) + \log(o_t + 1.0) + o_t\log(\mu_{tj})\right]p_{tj}$$

Thus the score function is

$$Score(\varphi) = \frac{\partial Lm}{\partial \varphi} = \sum_{t=1}^{T}\sum_{j=0}^{N-1}\left[\Psi(o_t + \varphi) - \Psi(\varphi) - \frac{\varphi + o_t}{\varphi + \mu_{tj}} - \log(\varphi + \mu_{tj}) + 1 + \log(\varphi)\right]p_{tj}$$

where $\Psi(x) = \partial \log\Gamma(x)/\partial x$, the digamma function. The observed Fisher information is

$$Info(\varphi) = -\frac{\partial^2 Lm}{\partial \varphi^2} = \sum_{t=1}^{T}\sum_{j=0}^{N-1}\left[-\psi(o_t + \varphi) + \psi(\varphi) + \frac{\mu_{tj} - o_t}{(\varphi + \mu_{tj})^2} + \frac{1}{\varphi + \mu_{tj}} - \frac{1}{\varphi}\right]p_{tj}$$

where $\psi(x) = \partial^2 \log\Gamma(x)/\partial x^2$, the trigamma function.

We use the Newton-Raphson method given the score function and the fisher information.
Initialize $\varphi = \frac{\sum_{t=1}^{T}\sum_{j=0}^{N-1} p_{tj}}{\sum_{t=1}^{T}\sum_{j=0}^{N-1} p_{tj}(o_t - \mu_{tj})^2}$
WHILE (ABS(Dev) > Toleration) {
Dev = $\frac{Score(\varphi)}{Info(\varphi)}$

$\varphi = \varphi$+Dev

}

5 Autoregressive HMM

When overlapping windows are used, the observed read-depth is drawn from an autoregressive process. We implemented an autoregressive HMM. Specifically, a residual term is included as an additional predictor in the negative binomial regression model assuming first order autoregression. Given the notations defined in Section 4.3.4, we obtain:

- $\log(\overline{\mu}_{tj})$: estimated using IRLS method. The number of covariate $M = 1$, $x = \{\texttt{cov}^1\}$, $\texttt{cov}^1 = \{\texttt{cov}^1_1, ..., \texttt{cov}^1_T\}$, $\texttt{cov}^1_t = \{g_t, ...g_t, .., g_t\}$ (g_t repeats for N times).

- $\log(\overline{\mu}^{'}_{tj})$: estimated using IRLS method. The number of covariate $M = 2$, $x = \{\texttt{cov}^1, \texttt{cov}^2\}$, $\texttt{cov}^1 = \{\texttt{cov}^1_1, ..., \texttt{cov}^1_T\}$,$\texttt{cov}^1_1 = \{0, ..., 0\}$ (0 repeats for N times),$\texttt{cov}^1_t = \{\log(o_{t-1}) - \log(\mu_{t-1,0}), \log(o_{t-1}) - \log(\mu_{t-1,1}), ..., \log(o_{t-1}) - \log(\mu_{t-1,N-1})\}, 1 \leq t \leq T$,$\texttt{cov}^2 = \{\texttt{cov}^2_1, ..., \texttt{cov}^2_T\}$, $\texttt{cov}^2_t = \{g_t, ...g_t, .., g_t\}$ (g_t repeats for N times).

The fitting process is implemented as the following:

Step 1: Fit initial weighted GLM to obtain $\log(\overline{\mu}_{tj})$ as in Section 4.3.4.

Step 2: Compute residual $r_{t-1,j} = \log(o_{t-1}) - \log(\overline{\mu}_{t-1,j})$ for $t > 1$ and let $r_{1,j} = 0$.

Step 3: Refit weighted glm to obtain $\log(\overline{\mu}'_{tj})$ including r_{t-1} as covariate $\texttt{cov}^1$.

Step 4: Compute emission probability using $\log(\overline{\mu}'_{tj})$.

Step 5: Compute Forward, Backward, Posterior probability as before.

Step 6: Parameter update for transition probability as before.

Step 7: Refit weighted glm to obtain $\log(\overline{\mu}_{tj})$ as in Section 4.3.4 using updated posterior probability.

Step 8: Update $r_{t-1} = \log(o_{t-1}) - \log(\overline{\mu}_{t-1,j})$ for $t > 1$ and let $r_{1,j} = 0$. where $\log(\overline{\mu}_{tj})$ is from the Step7.

Step 9: Refit weighted glm to obtain $\log(\overline{\mu}'_{tj})$ including r_{t-1} as covariate $\texttt{cov}^1$, using updated posterior probability.

Step 10: Iterate steps 4-9 until E-M converges.

6 Post-segmentation Processing

A predicted CNV region cnv is a succession of sliding windows with the same state CN predicted by HMM. cnv can be described by a tuple $(cnv_{type}, cnv_{left}, cnv_{right})$, which gives the type cnv_{type}, and the boundaries cnv_{left}, cnv_{right}. cnv_{left} is the start point of the first window in cnv, and cnv_{right} is the end point of the last window in cnv.

Given the predicted CNV regions $CNV = \{cnv_1.....cnv_S\}$, where S is the number of the predicted CNV regions, a two steps post-segmentation processing will be carried out to refine CNV. The first step is to remove the small CNV regions, and the second step is to merge the CNV regions with their nearby similar regions.

Step1. Remove every small CNV region from the predicted CNV regions CNV. A predicted CNV region cnv is a small CNV region if $cnv_{right} - cnv_{left} \leq$ threshold value (e.g. 1). After the removal, those remaining regions are denoted as CNV_{large}.

Step2. Merge the regions in CNV_{large}. CNV regions in CNV_{large} will be scanned from left to right in the order of the position of their boundaries. A CNV region cnv^i and its next right CNV region cnv^j can be merged if they satisfy the following two conditions.

$$\begin{aligned} cnv^j_{right} - cnv^i_{left} &\leq 0.5 * (cnv^i_{right} - cnv^i_{left} + cnv^j_{right} - cnv^j_{left}) \\ cnv^j_{right} - cnv^i_{left} &\leq min(cnv^i_{right} - cnv^i_{left}, cnv^j_{right} - cnv^j_{left}) \end{aligned}$$

The merged CNV region $cnv^{'}$ is described by the tuple $(cnv_{type}, cnv^i_{left}, cnv^j_{right})$. cnv_{type} is decided by the majority state of the windows between cnv^i_{left},cnv^j_{right}. And cnv^i and cnv^j will be replaced from CNV_{large} by $cnv^{'}$. The scanning will continue from $cnv^{'}$ and its next right CNV region. The scanning stops when it reaches the right-most CNV region.

7 Reference

Rozowsky, J., Euskirchen, G., Auerbach, R.K., Zhang, Z.D., Gibson, T., Bjornson, R., Carriero, N., Snyder, M. and Gerstein, M.B. (2009) PeakSeq enables systematic scoring of ChIP-seq experiments relative to controls. *Nat Biotechnol*, **27**, 66-75.

Baum L., Petrie T., Soules G., and Weiss N. (1970) A maximization technique occurring in the

statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Statist.* **41**: 164-172.

Bilmes J. (1998) A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. International Computer Science Institute Berkeley CA, 94704.

Dean, C. (1992) Testing for overdispersion in Poisson and binomial regression models. *Journal of the American Statistical Association*, **87**(418):451-457.

Juang, B.H. and Rabiner, L.R. (1985) Mixture autoregressive hidden Markov models for speech signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing.* **Vol. ASSP-33**, No.6 , p.p. 1404-1413.

Rabiner, L.R. (1989) A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, **77** (2): 257-286.

Venables, W. and Ripley, B. (2002) Modern applied statistics with S. *Springer*.