Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Manuscript Category

Sequence Analysis

ClearCNV: CNV calling from NGS panel data in the presence of ambiguity and noise

Vinzenz May[1], Leonard Koch[2], Björn Fischer-Zirnsak[2,3], Denise Horn[2], Petra Gehle[4,5], Uwe Kornak[3,6], Dieter Beule[1] and Manuel Holtgrewe[1,*]

[1] Berlin Institute of Health at Charité – Universitätsmedizin Berlin, Core Unit Bioinformatics (CUBI), Charitéplatz 1, 10117 Berlin, Germany
[2] Charité – Universitätsmedizin Berlin, corporate member of Freie Universität Berlin and Humboldt-Universität zu Berlin, Institute of Medical Genetics and Human Genetics, Augustenburger Platz 1, 13353 Berlin, Germany
[3] Max-Planck-Institut für Molekulare Genetik, FG Development & Disease, 14195 Berlin, Germany,
[4] Charité-Universitätsmedizin Berlin, corporate member of Freie Universität Berlin and Humboldt-Universität zu Berlin, Department of Internal Medicine - Cardiology, Berlin, Germany
[5] DZHK (German Center for Cardiovascular Research), partner site Berlin, Germany
[6] Institute of Human Genetics, University Medical Center Göttingen, Göttingen, Germany.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: While the identification of small variants in panel sequencing data can be considered a solved problem, the identification of larger, multi-exon copy number variants (CNVs) still poses a considerable challenge. Thus, CNV calling has not been established in all laboratories performing panel sequencing. At the same time such laboratories have accumulated large data sets and thus have the need to identify copy number variants on their data to close the diagnostic gap.

Results: In this manuscript we present our method clearCNV that addresses this need in two ways. First, it helps laboratories to properly assign data sets to enrichment kits. Based on homogeneous subsets of data, clearCNV identifies CNVs affecting the targeted regions. Using real-world data sets and validation, we show that our method is highly competitive with previous methods and preferable in terms of specificity.

Availability: The software is available for free under a permissible license at {{https://github.com/bihealth/clear-cnv}}

Contact: manuel.holtgrewe@bih-charite.de

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

Hybrid capture methods (Ng et al., 2009) allow for targeted sequencing ranging from whole exome sequencing to panel sequencing of few known disease genes. They have thus made high throughput sequencing affordable for clinical applications by strongly reducing the required sequencing data. From the perspective of bioinformatics there are few differences in analyzing small panels, whole exome (WES), or whole genome (WGS) sequencing data for single nucleotide variants (SNVs), and small insertions and deletions.

However, the detection (commonly also referred to as calling) of copy number variants (CNVs) is considerably harder because of structured but very inhomogeneous variances in depth of coverage that are typical for hybrid capture methods. Reasons for such variance include GC content of the targeted regions, biochemical properties of the used enrichment kits,

and batch effects in producing the enrichment reagents (Daniel et al. 2011, Benjamini 2012).

In this manuscript we present our software package *clearCNV* that (1) contains a program that helps users to properly assign panel sequence data

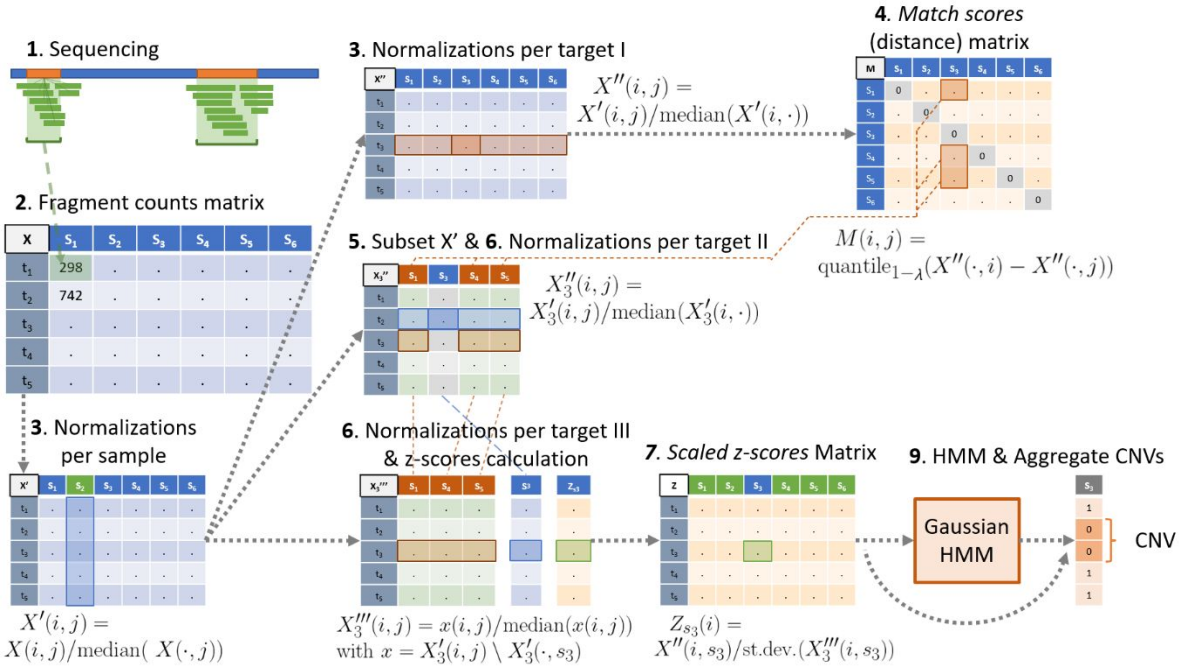

Fig. 1. CNV calling algorithm. This figure is an illustrates the steps described in the main text. In step 3, green indicates a sample that is normalized in that step. In steps 5 & 6 orange indicates the sample group background and blue the CNV calling sample. In steps 6 and 7, green indicates the calculated (and scaled) z-score.

Nevertheless, CNVs are of large interest as they account for 4.5 to 12 percent of genome variation in humans (Sudmant,P.H. et al. 2015, Collins, R.L. et al. 2020) and are implied in many diseases (Nowakowska 2017, Conrad et al. 2010, Zhang et al. 2009). Copy number variants are a subclass of structural variants; the latter is commonly defined as variation with a size of larger than 50bp. The size of CNVs can range from the lower limit to the loss or amplification of whole chromosome arms or chromosomes.

From the authors' experience, CNV calling for panel sequencing data has not been systematically established in many laboratories performing targeted panel sequencing yet despite the wide use of panel sequencing. While many centers are now introducing WES or even WGS into standard care, they have considerable numbers of panel sequencing data already available (Marshall et al. 2020). Obviously, being able to reanalyze this data for CNVs is highly desirable to solve more cases without additional sequencing.

Various tools for the CNV analysis of panel sequencing have recently been published in the literature including CoNVaDING (Johansson et al. 2016) and AtlasCNV (Chiang et al. 2019). Further, tools for the analysis of exome data have been enabled for the analysis of panel data, including panelcn.MOPS (Povysil et al. 2017) or ExomeDepth (Plagnol et al., 2012). Some methods have been developed and evaluated solely for a single panel such as AtlasCNV while others can be used more widely such as panelcn.MOPS or ExomeDepth. Approaches to combine CNV calling tools to achieve the highest possible accuracy can differ in their results by a lot given different datasets (Moreno-Cabrera et al. 2020, Sadedin et al. 2018)

However, centers wishing to analyze their panel data in hindsight often also face unexpected challenges. From the experience of the authors, these also include missing, incomplete or incorrect documentation of which gene panel or gene panel version was used for a particular sample.

to the used panel and to separate sequencing batches, (2) provides a novel method that allows to analyze their data for copy number variations, and (3) provides an easy-to-use visualization of the coverage data and the called CNVs.

2 Methods

2.1 CNV Calling algorithm

The first part of our method is the implementation of a novel algorithm for the identification of CNVs from targeted sequencing data. Some steps are built on the ideas of already existing algorithms. The steps of the algorithm are described below and illustrated in Fig. 1.

1. Target file creation. Overlapping and nearby targets are merged to avoid ambiguities further downstream.

2. Fragment counting. We count the number of fragments (reads or read pairs) per target. Fragments overlapping with multiple targets are assigned to the one closest to the center. The results are tabulated in a matrix x with entries $X(i,j)$ in row i and column j; that is the number of fragments of sample j on target i. Samples with a median fragment count smaller than five are excluded to avoid downstream problems.

3. Data normalization. The matrix is first normalized per sample (per column) by dividing each column's values by the column's median $X'(i,j) := X(i,j) \,/\, \text{median}(X'(i,\cdot))$ and then per target (per row) $X''(i,j) := X'(i,j) \,/\, \text{median}(X'(\cdot,j))$, where "$\cdot$" indicates all elements in that dimension.

4. Match scores are a distance metric to identify samples with similar coverage patterns (similarly used in the context of CNV calling by Johannson et al. (2016). A match score m of two samples (vectors) s and k is defined as the mean difference of two vectors. clearCNV additionally removes the λ greatest differences before computing the mean: $m_{s,k} :=$

$\text{mean}(\text{quantile}_{1-\lambda}(\text{abs}(s-k)))$, where λ (default is 0.02) is a user-adjustable factor that attributes for expected uneven variance. Such a variance includes signals for CNVs as well as forms of noise. The final visualizations (Fig. 2) help to adjust this factor.

5. *Sample group.* Each sample S gets assigned a set of background samples. Any sample x in each sample group satisfies that its match score $m_{x,S}$ is below the median match score of all match scores of all samples multiplied by a user-specifiable constant θ (default 2): $m(x,S) \leq \text{median}(m(\cdot,\cdot)) \cdot \theta$. This way each sample gets assigned an individually sized sample group. We determined the default value for θ empirically by inspecting histograms of all match scores. No CNV calls are generated on a sample that has a sample group size below a user-specifiable threshold γ (default 20), however it may appear in another sample's sample group. θ and γ were determined by choosing a relative optimum between the number of samples and the variance in a sample group. In CNV calling, a subset of fragment counts normalized per sample is chosen according to the selected sample group of sample S. Let this table be X'_S.

6. *Data normalization II & III.* X'_S is normalized per target to get X''_S and the vector $X''_S(\cdot,S)$ in X''_S containing all values of S is extracted. X''_S without S ($X''_S \setminus S$) is again normalized per target to remove any effect of S on the sample group's statistics which yields X'''_S. The 10% columns of X'''_S with greatest variance are then dropped from X'''_S to further reduce variance.

7. *Scaled z-scores*. z-scores are calculated for sample S, which is found in X''_S on each row i: $z(i) := X''_S(i,S) \,/\, \sigma(i)$, where σ is the vector of per row standard deviations of X'''_S. The resulting z-scores are then scaled to reduce the effects of noise in CNV calling: $z'(i) := z(i)^{2-\alpha}$, where z(i) is the z-score of target i of sample S. The value α is the user-provided factor which is 0.65 by default. We determined the default for α in comparison with plotted heatmaps and checked where most CNV calls aligned with our judgement. The resulting vector of z' is saved in a matrix Z which contains all scaled z-scores of all samples.

8. *r-scores* approximate a copy number of a target in a sample. r-scores are created in the previous step on the vector $X''_S(\cdot,S)$. Ideally, a r-score of 1.0 indicates a wild type, while 0.5 indicates a heterozygous deletion, 1.5 indicates a heterozygous duplication and 2.0 indicates a homozygous duplication and so on. These values are saved to a matrix R for each target and sample. This matrix holds all r-scores at the end.

9. *CNV calling*. Two types of CNVs are called: a) multi-exon CNVs and b) single-exon CNVs.

9a) At first, the Viterbi algorithm is used on a Gaussian HMM (Hidden Markov Model). The means of the three states (*deletion, wild type, duplication*) are semi-automatically adjusted. For *deletion*, the mean is calculated as $m_{del} = -3\,\sigma$, for *wild type* it is $m_{wt} = \overline{m}$, and for *duplication* it is $m_{dup} = 4\,\sigma$, with σ the st.dev., and $\overline{m}$ the median of all scaled z-scores. The Viterbi algorithm is then run on the z-scores in Z. The covariances are set to 1.0. The transition probability matrix is created from the user-adjustable transition probability τ (default $\tau = 0.001$):

$$\begin{pmatrix} 1-(\tau*2) & \tau & \tau \\ \tau & 1-(\tau*2) & \tau \\ \tau & \tau & 1-(\tau*2) \end{pmatrix}.$$

The resulting hidden states of each sample are saved in a matrix H that holds all hidden states of all targets and all samples. For each sample S, a consecutive interval T of targets is aggregated to a single CNV if the average ratio score $r = \text{mean}(R(T,S))$ satisfies $r < \mu$ (default is 0.75) or $r > \omega$ (default is 1.35) and all hidden states of $H(T,S)$ are the same and not the wild type. μ and ω were chosen under the assumption that they separate the ratio scores well. The score function c of a CNV is the absolute value of the mean of scaled z-scores of all contributing targets $c(T,S) = \text{mean}(\text{abs}(Z(T,S)))$.

9b) To call single-exon CNVs, two thresholds are applied to Z. A single target t of sample S is called a deletion if $Z(t,S) < -3.5$ and $R(t,S) < 0.75$ or a duplication if $Z(t,S) > 4.5$ and $R(t,S) > 1.35$ and only if it is not contained in an already called multi-exon CNV by the HMM-guided method. All default values for parameters were determined by empirical methods, including comparisons with data visualizations.

10. *Output.* The CNV calls are saved in a tabular file containing the gene names, aberration, size, score, and sample score. Furthermore, the scaled Z-scores matrix Z and the ratio-scores matrix R and a list of samples that failed to have a sufficient sample group size are written to output files.

2.2 Result Visualization

The second part of our method is a web browser—based visualization for the relative copy numbers per target and per sample represented by the ratio scores, as well as the scaled z-scores. This allows the user to visually screen the results of their experiments as well as the results of the CNV calling algorithm.

Scaled z-scores and ratio-scores are both visualized in responsive heatmaps. Each heatmap additionally shows a track of mappability, target size and GC-content at each target. These are calculated from the target file, the reference and a uniqueness-of-reference file. The scaled z-scores

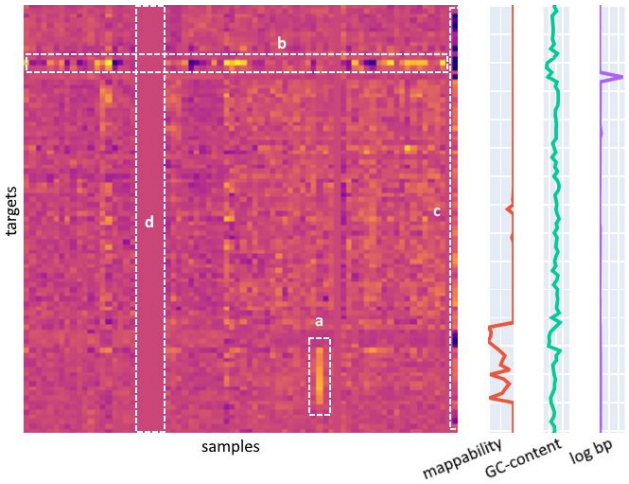

Fig. 2. Example heatmap of r-scores. This is a small example of a heatmap showing the ratio scores of each target (row) for each sample (column). A darker spot indicates a lower r-sore and vice versa. Aligned to the targets are three tracks: 1. Mappability, GC-content and log bp, which is the size of a target in bp then log transformed. Each of these three tracks show a value of 0 if the colored curve is on the left side. The Mappability and GC-content tracks have a value of 1 if the curve is on the right side. Log bp can be any size if on the right side but it scales with the maximum value in the track. Additionally, we marked several phenomena in the heatmap to illustrate its potential. a) shows a possible copy number gain, b) shows a target with high variance (or copy number variability), c) shows a low-quality sample with high variance, d) shows several samples that were too noisy and whose r-scores were set to 1.0 (imputed).

are clipped to the interval [-6,6]. The ratio-scores are clipped to the interval [0,2]. An example of such a heat map is shown in Fig. 2.

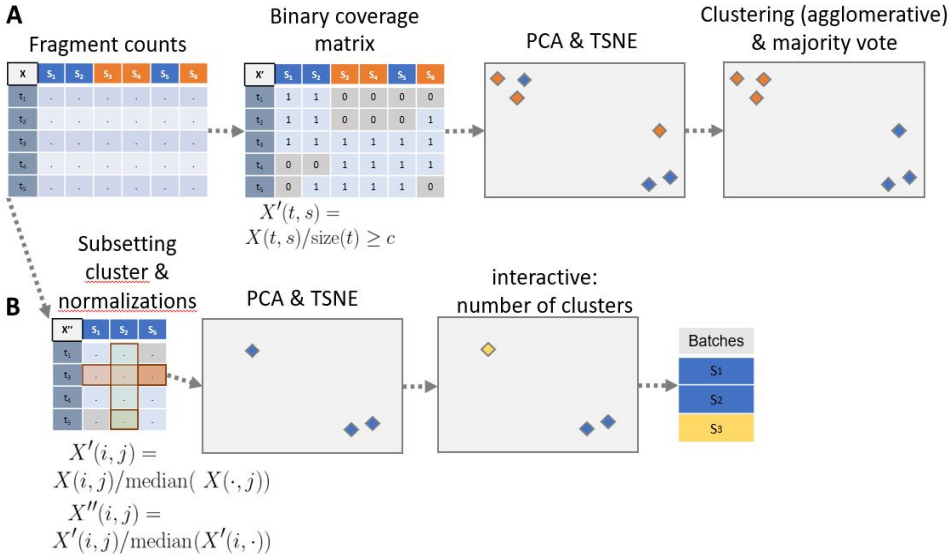

Fig. 3. Sample to Panel re-assignment and batch separation. The steps are numbered according to the main text's steps. Sub figure A illustrates the sample re-assignment and sub figure B illustrates the batch separation. $X, X', and\ X''$ illustrate matrices, where " . " means any numerical entry. Colors blue and orange indicate different clusters (or with yellow different batches). Frames overlaying a matrix indicate the vector that is subject to an operation.

2.3 Sample to Panel re-assignment

The third part of our method supports users in the assignment of aligned sequencing results in BAM format to panel target information in BED (Browser Extensible Format) files. This is important for retrospective analyses in the presence of artifacts such as sample swaps or erroneous documentation of the used sequencing kit. A BED file is a text file containing genomic regions, *e.g.*, exons. The following steps are illustrated in Fig. 3. The projection by PCA and TSNE, as well as the clustering can be interactively worked with in a *plotly Dash* app (https://plotly.com/dash/).

When applied with clearCNV, batch separation (see 2.4) should be done before the CNV calling step.

1. *BED file merging*. The panel re-assignment algorithm starts with merging all input target files to one union of target files. This is necessary to make the given samples comparable.

2. *Fragment counting* is done the exact same way as in the CNV calling algorithm. An entry in the resulting matrix is addressed as $X(i,j)$ for the i-th row (target) and j-th column (sample).

3. *Binary matrix*. We are interested only whether a target is covered and not in the depth of coverage. Since we expect off-target effects in the enrichment, we call a target covered only if the per target fragment counts divided by target size is above 1/50. In the case of reads of 100bp size, this, for example, would correspond to a read depth coverage of two. The final matrix X is binary (1 = covered, 0 = uncovered).

4. *PCA and transformation*. A principal component analysis (PCA) transformation is applied to X with δ dimensions (default is 20, adjustable by the user) yielding X'. The default of δ is chosen according to the order of magnitude of the number of targets per panel.

5. *TSNE*. A t-distributed stochastic neighbor embedding (TSNE) projects X' to a latent space (here with two components) which allows fast and simple clustering on the resulting matrix X''. The random process within the TSNE makes it necessary for the user to occasionally re-run the process to arrive at a desired projection and clustering result.

6. *Clustering*. An agglomerative clustering on X'' finds the clusters.

7. *Cluster assignment*. The resulting clustering is mapped to the provided target files. A majority vote is used to assign each cluster to a target file. This implies the constraint that each cluster must have a majority of correctly assigned samples to a target file.

8. *Output*. At this point the data sets are untangled and new lists of bam-files are written to the according output files.

2.4 Batch separation

We observed separable subsets in the data, which were not explained by erroneous sample–panel assignments. We suspect that limited numbers of well plate units may have introduced such batch effects. We suspected even more possible reasons such as the design of custom enrichment kits or flow cell biases.

Batch separation is done for each cluster with its previously assigned panel resulting from the sample re-assignment. The batch-separation algorithm is like the sample re-assignment algorithm. Again, the interactive parts are implemented in a *plotly Dash* app.

1. *Fragment counts sub setting*. The matrix containing the fragment counts per target is subset to the samples found in the given cluster. The targets are subset from the union BED-file to contain only targets that are present in the assigned panel. Differently to the panel re-assignment procedure, the resulting matrix is not reduced to a binary matrix but holds the per sample and per target normalized fragment counts.

2. *PCA and transformation*. Analog to panel re-assignment step 4.

3. *TSNE*. Analog to panel re-assignment step 5.

4. *Clustering*. The interactive interface lets the user control the number of batches on each set of samples for each panel.

5. *Output*. A list of alignment file paths (BAM format) per identified batch is written to an output file for each found cluster. This file can be used in the CNV calling step.

2.5 Evaluation

To evaluate the performance of our algorithm, we compared it to existing approaches to call CNVs on targeted sequencing data. The tool selection was limited to those having a scientific publication and being freely available for research applications. We chose ExomeDepth (Plagnol et al., 2012), CoNVaDING (Johansson et al. 2016), panelcn.MOPS

(Povysil et al. 2017) to evaluate comparatively with clearCNV. We chose not to use the recent tool Atlas-CNV by Chiang et al. (2019) in the evaluation because it was designed to find single Exon CNVs in the *eMERGESeq* panel, which we did not use (an earlier evaluation showed no competitive results on our data set, data not shown). A brief overview of the tools' features can be found in the supplement section S8.

We wrote CNV calls and internal or explicit scorings of clearCNV, CoNVaDING, ExomeDepth and panelcn.MOPS, each to a uniformly formatted file for a comparative evaluation.

We attempted to evaluate all four tools on simulated targeted sequencing data. The data was simulated with CapSim (Cao 2018). A detailed analysis of the generated simulated data showed that important properties and biases that we observe in real-world data are not captured appropriately in the simulation, in particular missing correlation between adjacent exons. Nevertheless, simulated data allows to evaluate tool performance with a known ground truth.

The results on the simulated data can be summarized as follows. ClearCNV is competitive with the other tools on simulated data and showed the highest positive predicted value (PPV) at the cost of some sensitivity. ExomeDepth shows good performance overall, CoNVaDING and panelCN.MPOS showed a very high false positive rate for certain samples (we were unable to determine the root cause for this).

All details regarding the simulation, analysis, and performance results can be found in the Supplementary Sections S6 and S7.

To compare the results of the CNV calling of each tool on real-world data, we used two different approaches. First, we compared the scores given to each single target in each sample for each tool. We did this before we chose a subset of CNV calls to be validated via quantitative PCR (qPCR). The details can be found in section S2 of the supplement. Second, we compared the scores of the aggregated called CNVs after we had the qPCR results. We ranked the called CNVs for each tool to achieve comparability. The ranking is described in detail in the results section. Finally, we visualized these rankings for each tool's results, which can be seen in the results section and in Fig. 5.

Regarding the results, two different parameters were scored. First, each tool scores single targets for each sample. These are the target scores. Second, CNV calls are scored and the *target scores* are the underlying scores. They are aggregated in some way, e.g., by taking the mean of the consecutive targets that form the called CNV. These are the *CNV scores*.

Each tool has a different scoring metric per target and only ExomeDepth and clearCNV aggregate called CNVs. In the case of CoNVaDING we chose to score the targets according to the median "AUTO_ZSCORE" scores which are found in the matching *.totallist files. panelcn.MOPS does also not provide aggregated CNV calls and no target scores. After very helpful correspondence with the authors, we followed their advice to calculate scores per target and used the RC ratio (RC.norm/medRC.norm) to score single targets per sample. To aggregate called CNVs, we merged consecutive targets if they were called the same copy number unequal two. clearCNV generates both the target scores and the CNV scores.

We performed our evaluation on seven custom panels manufactured by Agilent from different genetic rare disease fields. Data was generated in a diagnostic setting and all patients gave informed consent for further research. Four panels have about six thousand targets, the others have about one or two thousand. In total, we had data from 1407 different individual samples. More detailed information about the data can be found in the supplement section S1, Table S1, and Fig. S1.

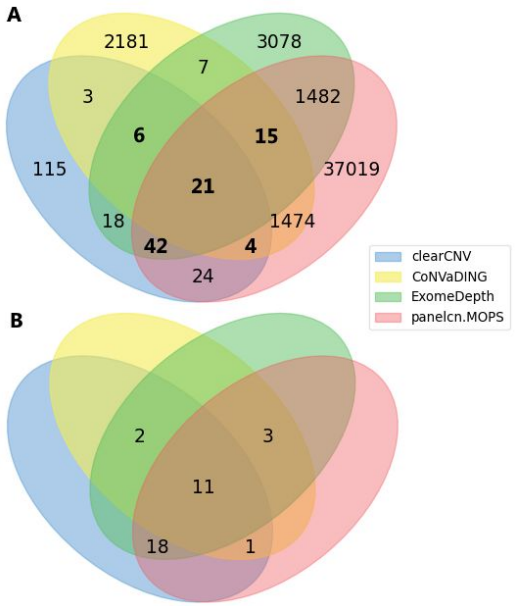

Fig. 4. Venn diagrams of called and confirmed CNVs. Sub figure A shows all CNV calls on all available data. CNVs selected for validation are marked in bold letters. Sub figure B shows only the confirmed (by qPCR) CNVs. Unlabeled subsets have a cardinality of zero.

We chose not to evaluate with simulated data as targeted sequencing data is known to contain a large amount of noise and biases that have not been comprehensively characterized and modeled yet.

3 Results

3.1 Sample to panel re-assignment and batch separation

38 out of 1407 total samples were re-assigned to different panels or panel versions, for which the documentation was not complete anymore. 16 Samples had a fragment-per-target count so low that they were excluded from any further processing by clearCNV. A detailed log of the sample re-assignment and batch separation process can be found in the supplement section S3.

3.2 CNV calls

The Venn diagram in Fig. 4.A shows all called CNVs on all data sets by CoNVaDING, ExomeDepth, panelcn.MOPS and clearCNV. As it can be seen, the results show a great discordance in terms of called CNVs. To select a feasible number of variants for validation by qPCR, we limited the results to those CNVs called by three tools or more. We finally had to exclude samples for which no DNA for validation was available. This resulted in a set of 88 CNV calls to be validated. We could confirm 35 CNV calls, of which 15 were duplications and 20 were deletions. One deletion could be confirmed by inspecting the corresponding WGS track in IGV highlighting extended fragment spans (see section 2 and Fig. S2 in the supplement). The other 34 CNVs were confirmed via qPCR, following the protocol detailed in Ott *et al.* (2010).

Tab. 1. CNV calls by all tools. This table shows the total number of CNV calls made by each tool and the according number of validated and confirmed CNV calls. The subsets can be inspected in Fig. 4.

Tool	Total CNV calls	validated calls	confirmed calls
clearCNV	233	73	32
CoNVaDING	3 711	46	17
ExomeDepth	4 669	84	34
panelcn.MOPS	40 081	82	33

Even after several adjustments of the DNA melting temperatures, we were not able to identify the true copy number via qPCR of nine out of the 88 CNV calls. We treated ambiguous results as unconfirmed calls (same as wild type). We analyzed mappability and GC-content of the CNV calls and our whole data. We found that for about 15% of the targets (data points) a too high GC content rendered qPCR validation infeasible. The details can be found in supplement section S4.

Fig. 4B shows a Venn diagram of the 35 confirmed CNVs. The subsets overlapping only two or one tools are left blank, as these CNV calls were excluded from validation. Eleven CNV calls were made by all four tools and were successfully validated. Three CNV calls that were confirmed by qPCR were not called by clearCNV. CoNVaDING missed 18, panelCN.MOPS missed two and ExomeDepth missed one. As can be seen in Figure 4A, the overall number of CNV calls varies greatly between the tools. Besides the total number of true positives, one must consider the rank of the true within the false positives within each tool.

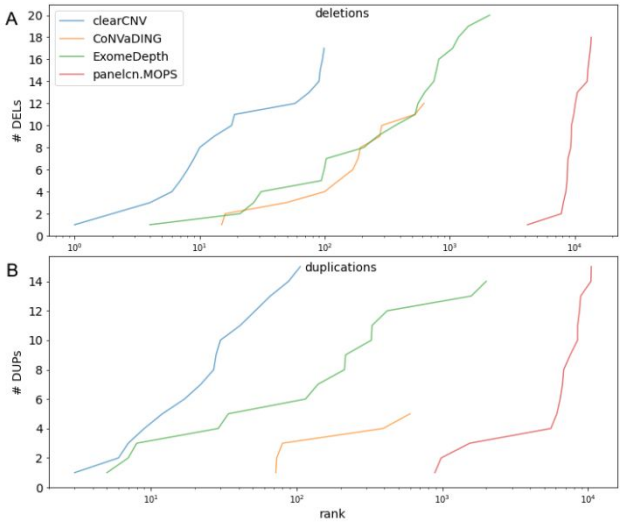

Fig. 5. CNV calling score ranks vs. cumulated number of confirmed CNVs. All CNV calls are ranked for each tool and separately for deletions (sub fig. A) and duplications (sub fig. B). Each tool's results with standard parameters are represented by a solid line.

Fig. 5 shows receiver/operator characteristics-like plots for each tool. Each tool attaches scores to its resulting CNV calls. These scores can be understood as a transformed approximation of confidence. Ranking these scores (sort, then enumerate) allows to compare each tool's results directly. The horizontal axis shows the (log10-scaled) rank of the curve and the vertical axis shows the cumulative number of positively validated CNVs. The curve ends for each tool where the lowest ranking CNV call was confirmed.

It can be seen that ExomeDepth and clearCNV both created some true positive CNV calls among the highest ranks. Also, this approximation of specificity shows a difference in calling either deletions or duplications on our data set. ExomeDepth starts similarly specifically as clearCNV. The curve flattens and reaches into the 1000th rank to find all 20 deletions. The CNV calls by CoNVaDING and panelcn.MOPS start with lower ranks and end with very low ranks. But in the case of duplications, CoNVaDING's results show even a slightly higher specificity than ExomeDepth's. Overall, clearCNV misses three CNV calls but shows superior specificity when compared to other tools. A detailed analysis of the three missed CNVs can be found in Supplement section S5. All results can be downloaded from the supplementary repository (https://github.com/bihealth/clear-cnv-supplementary).

4 Discussion

clearCNV showed competitive sensitivity and excellent specificity on different real-world data sets, which were partially very heterogeneous in the underlying batches and unknown variances (more details in Supp. section S3). High specificity is important in clinical applications as it reduces the number of false positive and effort for validation.

Differences in specificity of the different tools can be attributed to different design decisions by their authors. panelcn.MOPS was not designed to operate at a high specificity, which is an intentional choice by the authors who worked with very high quality data (see discussion on GitHub: https://github.com/bioinf-jku/panelcn.mops/issues/19). CoNVaDING and ExomeDepth were designed to handle noise and more difficult data, but both rely on the user to discard low-quality samples or to isolate reference samples which have low noise to be used as models to fit their models on. clearCNV works without such preparatory steps by clustering the data beforehand and then filtering out low quality samples.

clearCNV also allows the user to visualize the results of the clustering and filtering steps to validate the parameter choices and allow to adjust parameters for fine-tuning when necessary.

Adding preprocessing steps, such as clustering, and batch separation allowed clearCNV to compensate for greater structural difficulties observable in the data. Other tools take similar but not as far-reaching approaches by finding subsets of samples that form a common statistical background for any single sample. clearCNV does that in addition to the two previous steps of panel-re-assignment and batch separation, which are also embedded in a user-friendly interactive *Dash* interface.

Acknowledgements

The Authors thank Dr. January Weiner for the helpful discussion.

Funding

The authors have no funding to declare beyond their organisation.

Conflict of Interest: none declared.

References

Benjamini,Y. and Speed,T.P. (2012) Summarizing and correcting the GC content bias in high-throughput sequencing. Nucleic Acids Res., 40, 1–14.

Cao,M.D. et al. (2018) Simulating the dynamics of targeted capture sequencing with CapSim. Bioinformatics, 34, 873–874.

Chiang,T. et al. (2019) Atlas-CNV: a validated approach to call single-exon CNVs in the eMERGESeq gene panel. Genet. Med., **0**, 1–10.

Collins,R.L. et al. (2020) A structural variation reference for medical and population genetics. Nature, 581, 444–451.

Conrad,D.F. et al. (2010) Origins and functional impact of copy number variation in the human genome. Nature, **464**, 704–712.

Daniel et al. (2011) Analyzing and minimizing PCR amplification bias in Illumina sequencing libraries. Genome Biol., 12.

Johansson,L.F. et al. (2016) CoNVaDING: Single Exon Variation Detection in Targeted NGS Data. Hum. Mutat., **37**, 457–464.

Marshall,C.R. et al. (2020) The Medical Genome Initiative: moving whole-genome sequencing for rare disease diagnosis to the clinic. Genome Med., 12, 48.

Moreno-Cabrera,J.M. et al. (2020) Evaluation of CNV detection tools for NGS panel data in genetic diagnostics. Eur. J. Hum. Genet., 1645–1655.

Ng,S.B. et al. (2009) Targeted capture and massively parallel sequencing of 12 human exomes. Nature, **461**, 272–276.

Nowakowska,B. (2017) Clinical interpretation of copy number variants in the human genome. J. Appl. Genet., **58**, 449–457.

Ott,C.E. et al. (2010) Deletions of the RUNX2 gene are present in about 10% of individuals with cleidocranial dysplasia. Hum. Mutat., **31**, E1587–E1593.

Povysil,G. et al. (2017) panelcn.MOPS: Copy-number detection in targeted NGS panel data for clinical diagnostics. Hum. Mutat., **38**, 889–897.

Sadedin,S.P. et al. (2018) Ximmer: A system for improving accuracy and consistency of CNV calling from exome data. Gigascience, 7, 1–11.

Sismani,C. et al. (2015) Copy number variation in human health, disease and evolution. Genomic Elem. Heal. Dis. Evol. Junk DNA, 129–154.

Sudmant,P.H. et al. (2015) An integrated map of structural variation in 2,504 human genomes. Nature, 526, 75–81.

Zarrei,M. et al. (2015) A copy number variation map of the human genome. Nat. Rev. Genet., 16, 172–183.

Zhang,F. et al. (2009) Copy number variation in human health, disease, and evolution. Annu. Rev. Genomics Hum. Genet., **10**, 451–481.